4DAF Video Player

By Joe Resuello, Technical Support Engineer, 4D Inc.

Technical Note 07-24

Abstract

This Technical Note shows how to embed 4D Ajax Framework (4DAF) objects into an HTML page resulting in a highly customizable user interface. A custom HTML page is created that plays videos from 4D Tech Tips and 4D Web 2.0 Pack demos. This Technical Note provides the 4D developer with instructions on how to construct custom HTML pages embedded with 4DAF objects using minimal Javascript and HTML code.

A sample database is included.

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).

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:

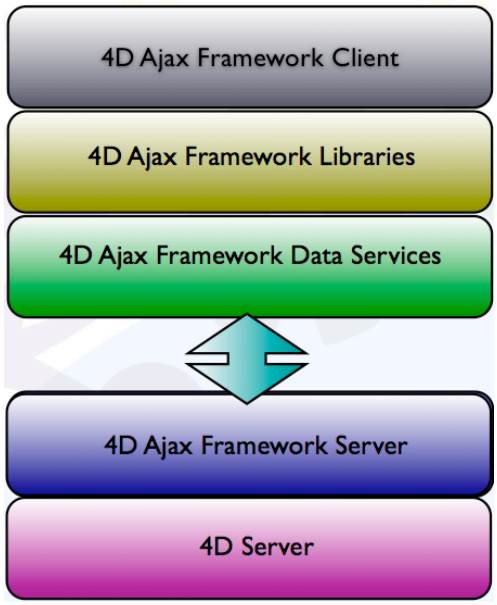
http://www.4d.com/products/4dweb20pack.html

Introduction

Note: This Technical Note includes the merged 4DAF Video Player database 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 Video Player application.

The interface created with this Technical Note is a custom implementation based on the 4D Ajax Framework Libraries. The 4D Ajax Framework Client is only used for configuration purposes. Take a look at the layers of the 4DAF:



Notice that the 4DAF Client is itself a custom implementation of the Libraries as well.

This Technical Note describes how to build the final database in step-by-step fashion. Here is an overview of the steps that need to be performed (these will be covered in detail later):

- Setup (install the 4DAF, create necessary files, etc.)
- Embed an ImageMatrix object in the custom HTML page
- Add a DataFiller to the customer HTML page
- Use an IFrame to view movies (the URLs to the movies come from the database)
- Finishing touches

Setting Up

4D Ajax Framework Installation

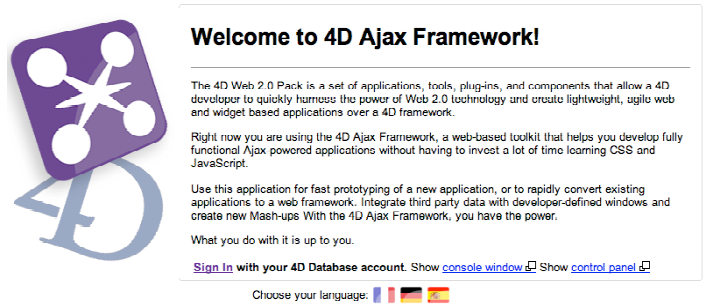
Install the 4DAF to the source database; the structure file is named "Movies.4DB". The "4D Ajax Framework Installation Guide" documentation can be downloaded from here:

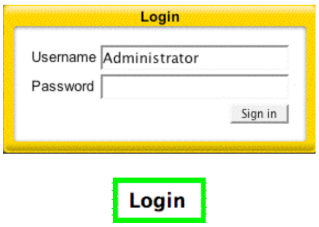
http://www.4d.com/support/documentation.html

Note: The Database and Project methods mentioned in the installation guide have already been implemented in the source database. You do not need to make these changes.

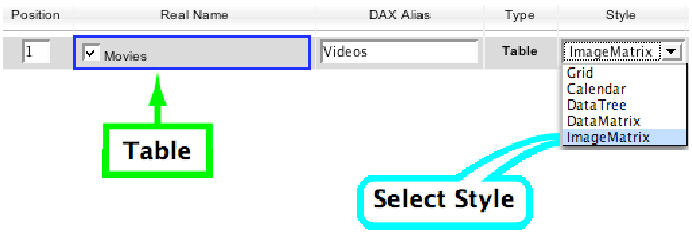
Create a Selection for the ImageMatrix

With the 4DAF installed, go to the 4DAF index.html splash page and log in as "Administrator".





In the 4DAF Client environment go the Control Panel then the Access Control tab. Choose the Selection (in this case a 4D Table) to be embedded into the HTML page and set the style to ImageMatrix.



After completing these steps you can view the default implementation of an ImageMatrix by clicking on the "Movies" table in the portal. Here is a look at the ImageMatrix when opened from the Portal in the 4DAF Client.

In this Technical Note a custom ImageMatrix will be created later.

Create the HTML File

The next step is to create the HTML file (use any HTML text editor of your choice). For the included database this HTML file should be named "index_movies.html". Here is the source code of "index_movie.html" when it was first created.

```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
      type="text/css" title="XPress" />

      <script type="text/javascript">
      <!--
      </script>
   </head>
   <body onload="Login('Guest','');" onunload="dax_bridge.logout();">
   </body>
</html>
```

Copy this code to your HTML file. To make things easier make sure it is named "index_movies.html".

The file now has some included libraries and a style sheet from the 4DAF. More guidelines and recommendations on what to include can be found at:

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

For the purposes of this Technical Note, the 4D developer only needs to be concerned with the content that will be added within the <script> and <body> elements.

Be sure to place this HTML file within the default HTML root folder.

Tip: If the 4DAF was installed using the instructions in the install guide, the default HTML root folder is called 'dax' and it sits next to the structure file.

Onload and Onunload Attributes

In the HTML above notice the *onload* and *onunload* attributes in the <body> element. These attributes correspond to page events and are used in order to bypass the 4DAF Login and Logout. They used as follows:

1) Onload: This occurs when the users connect to the web page.
2) Onunload: This occurs when the users exit the web page.

As defined in the code, on 'Onload' users automatically login as user 'Guest.' To make this user valid, go to 4D to project method DAX_DevHook_Login and add the following code (say, at line 62):

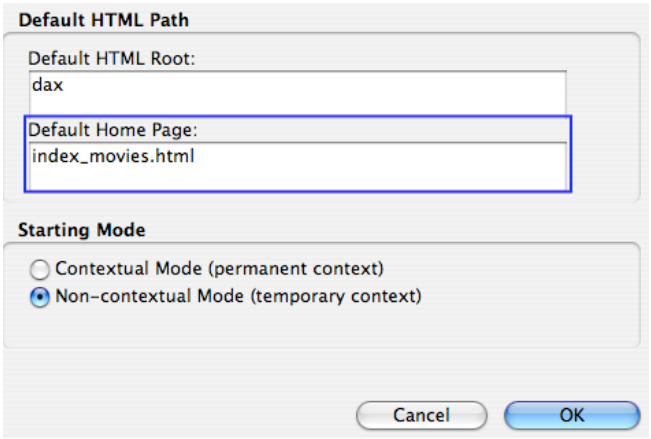
```
$loggedIn_b:=($userName_t="Guest")
```

Now when users connect to the web page they automatically login to the 4DAF as 'Guest'. The code in the Onunload event takes care of logging out of the 4DAF as users exit the page.

Set the HTML Page as Default

Customizing 4DAF objects into an HTML page essentially means the 4D developer is bypassing the need for users to connect to the 4DAF Client via the default "index.html" file. Thus, when users connect to the web server (ie. via http://localhost:8080) it would be best that they get directed to the custom HTML page on connection (in this case "index_movies.html").

To redirect users to the custom page, in 4D go to Preferences -> Web. Enter the name of the HTML page in the Default Home Page area.



Now when users connect to the web server they will automatically be directed to the custom HTML page instead of the default index.html page provided by the 4DAF.

Embed ImageMatrix into HTML page

Code must be added within the <script> and <body> elements in order to embed the ImageMatrix into the page.

Code in the <script> element

Here is the code added to the HTML file in the <script> element (in bold):

```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">

      // After successful login
      onAfterInit = function () {

      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';

      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      };

      </script>
   </head>
   <body onload="Login('Guest','');" onunload="dax_bridge.logout();">
   </body>
</html>
```

This JavaScript code defines the “onAfterInit” function for our custom HTML page. This function is for any actions or events that will occur after the connection has been established.

Also note that the ImageMatrix object is created here. For information on the ImageMatrix object and its parameters go to:

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

Note: Parameter 6 (*zoomContents*) is defined for the ImageMatrix object. However, the *zoomCell* event to call this parameter will not be executed for this sample database. You are encouraged to play around with the *zoomCell* event as well as other events provided by the 4DAF when you are more comfortable with doing so.

Code in the <body> element

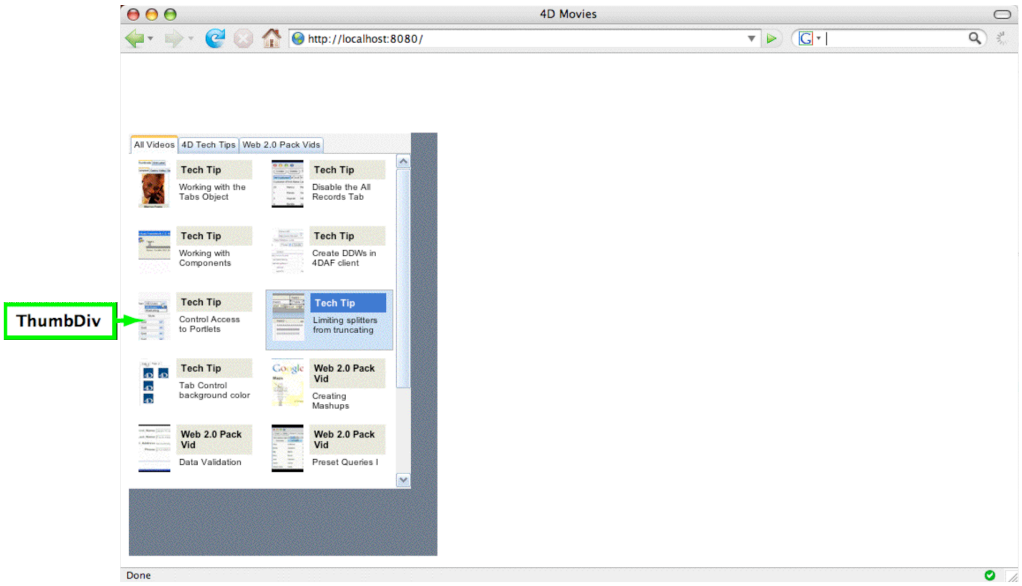
Here is the code added to the HTML file in the <body> element (in bold):

```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">
      // After successful login
      onAfterInit = function () {

      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';

      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      };
      </script>
   </head>
   <body onload="Login('Guest','');" onunload="dax_bridge.logout();">

   <div id="ThumbDiv" style="position:absolute; top:10px; left:10px; width:350px;
height:480px; "></div>

   </body>
</html>
```

A is a block of content on a web page. Here, a named 'ThumbDiv' is created. Take a look back in the <script> block to where the ImageMatrix was created. The first parameter determines where the ImageMatrix is inserted. In the sample database, the ImageMatrix is inserted in the 'ThumbDiv' area.

Results

Connect to the web server (via http://localhost:8080). Here is a screenshot of what should be displayed.



Note: Never mind the gray background for now. The color will supplement other <divs> added to the page later on.

Use the Data Filler for Divs

The Data Filler allows developers to populate areas on a webpage with the contents of a record. For more information about the Data Filler go to:

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

For this Technical Note, the Data Filler will be used to fill a with a description of the videos triggered by a click event.

Code in the <script> element

Code added to the <script> element (in bold):

```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">
      // After successful login
      onAfterInit = function () {

      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';
```

```
      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      ImageMatrix.onCellClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
         };

      };
      </script>
   </head>
   <body onload="Login('Guest','');" onunload="dax_bridge.logout();">
   <div id="ThumbDiv" style="position:absolute; top:10px; left:10px; width:350px;
height:480px; "></div>
   </body>
</html>
```

The onCellClick event is triggered when the user clicks an ImageMatrix record.

Code in the <body> element

Code added to the <body> element (in bold):

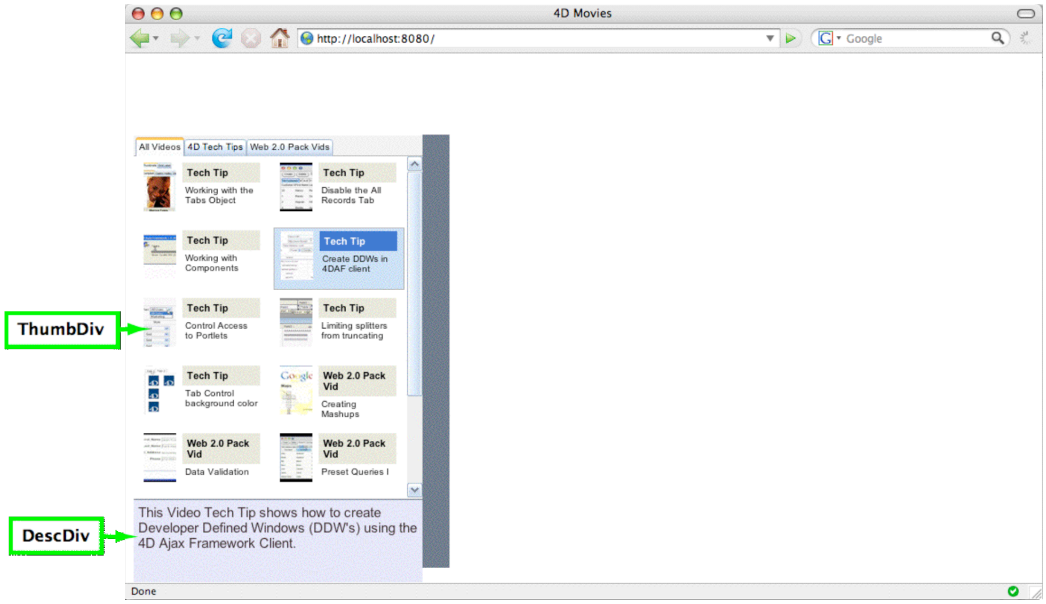
```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">
      // After successful login
      onAfterInit = function () {

      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';

      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      ImageMatrix.onCellClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
```

```
        };

      };
      </script>
   </head>
   <body id="main" onload="Login('Guest','');" onunload="dax_bridge.logout();">
   <div id="ThumbDiv" style="position:absolute; top:10px; left:10px; width:350px;
height:480px; "></div>

   <div id="DescDiv" style="position:absolute; top:495px; left:10px; width:310px;
height:81px; background-color: #E3E4FA; font-size:15px; padding-top:5px; padding-
left:5px; padding-right:5px; padding-bottom:5px; color: #302226;">
   <span class="4daf_[Movies]LongDescription"></span>
   </div>

   </body>
</html>
```

As mentioned in the Data Filler documentation on the Daxipedia, an HTML area must be identified (ie. id='main') for the corresponding code in <script> to execute successfully.

Results

When the user clicks an ImageMatrix record, the div 'DescDiv' is populated with information from the 'LongDescription' field of the record. Here is a screenshot:



Load Movies into IFrame

An <IFrame> is a frame in an HTML page where its content will be loaded from another HTML document. In this Technical Note an <IFrame> will be created and loaded with a URL to a web page that will play the videos. The technique to load the <IFrame> is much like the previous example when the Data Filler was used. The subtle difference is that in this scenario the information snatched from the record is

not displayed as text in the – it is used as the URL to load content in the <Iframe>.

Right now is a good time to create the HTML document this <Iframe> will refer to. Call it "st.html" and place it in the default HTML root folder (next to "index_movies.html". Here is the source code:

```
<html>
   <head>
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
   <meta http-equiv="PRAGMA" content="NO-CACHE" />
   <meta http-equiv="EXPIRES" content="-1" />
   <title>Video</title>
   <head>
   <body>

   </body>
</html>
```

Now it is time to add more code to index_movies.html.

Code in the <script> element

Code added to the <script> element (in bold):

```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">
      // After successful login
      onAfterInit = function () {
      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';

      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      ImageMatrix.onCellClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
         };
```

```
      ImageMatrix.onCellDblClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
         $('myIframe').src = $('movpath').value;
         };

      };
      </script>
   </head>
   <body id="main" onload="Login('Guest','');" onunload="dax_bridge.logout();">

   <div id="ThumbDiv" style="position:absolute; top:10px; left:10px; width:350px;
height:480px; "></div>

   <div id="DescDiv" style="position:absolute; top:495px; left:10px; width:310px;
height:81px; background-color: #E3E4FA; font-size:15px; padding-top:5px; padding-
left:5px; padding-right:5px; padding-bottom:5px; color: #302226;">
   <span class="4daf_[Movies]LongDescription"></span>
   </div>

   </body>
</html>
```

The <Iframe> SRC attribute is the source URL it will be loaded with. The 'movpath' object contains the path from the [Movies]Path field in the database.

Code in the <body> element

Code added to the <body> element (in bold):

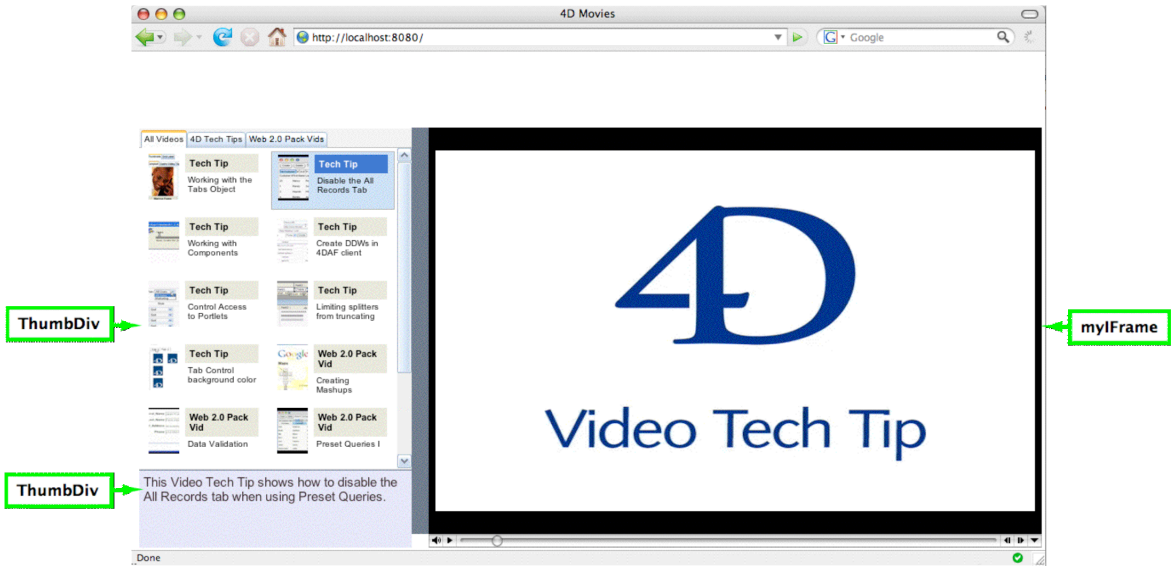
```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">
      // After successful login
      onAfterInit = function () {
      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';

      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      ImageMatrix.onCellClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
         };
```

```
      ImageMatrix.onCellDblClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
         $('myIframe').src = $('movpath').value;
         };
      };
      </script>
   </head>
   <body id="main" onload="Login('Guest','');" onunload="dax_bridge.logout();">

   <div id="ThumbDiv" style="position:absolute; top:10px; left:10px; width:350px;
height:480px; "></div>

   <div id="DescDiv" style="position:absolute; top:495px; left:10px; width:310px;
height:81px; background-color: #E3E4FA; font-size:15px; padding-top:5px; padding-
left:5px; padding-right:5px; padding-bottom:5px; color: #302226;">
   <span class="4daf_[Movies]LongDescription"></span>
   </div>

   <input id="movpath" type="hidden" value="" class="4daf_[Movies]Path"/>
   <iframe src="/st.html" style="position:absolute; top:90px; left:350px; width:
720px; height: 496px; background-color: #E3E4FA;" scrolling="auto" frameborder="0"
id="myIframe">
   </iframe>

   </body>
</html>
```

Here, 'movpath' and the <Iframe> areas are defined. '/st.html' is the blank HTML page the Iframe refers to. This file should be located next to "index_movies.html".

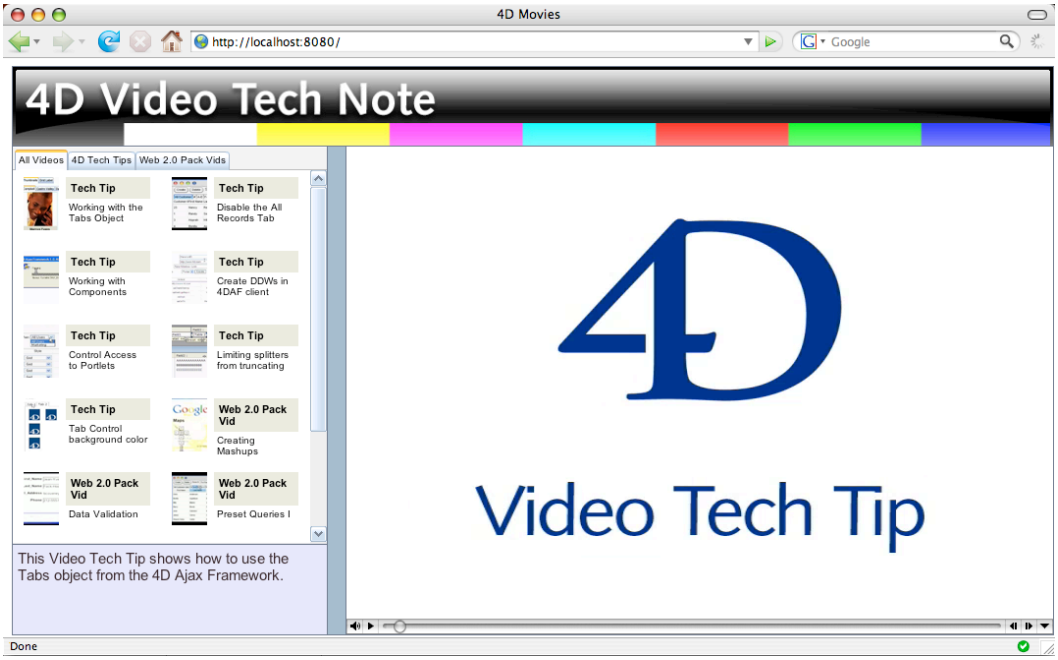
Results

Connect to the web server. Double-click one of the records in the ImageMatrix. It should load the video in the IFrame area to the right.



Final Touches

With additional code for aesthetic purposes this is the final web app:



Here is the final source code for index_movies.html (changes in bold):

```
<html>
   <head>
      <title>4D Movies</title>
      <script language="javascript" type="text/javascript" charset="utf-8"
src="dax/js/localization/resources_en.js"></script>
      <script language="javascript" type="text/javascript" charset="ISO-8859-1"
src="dax/js/compile_gz.js"></script>

      <!-- stylesheets -->
      <link rel="stylesheet" href="/dax/themes/basic/basic_gz.css" media="all"
type="text/css" title="XPress" />

      <script type="text/javascript">
      // After successful login
      onAfterInit = function () {
      var myHeader = '<td style="color: #7D053F; background-color: #FFFFFF;">';
      myHeader = myHeader + '[Movies]Title<\/td>';

      var myContent = '<table><tr><td style="color: #000000; font-size:10px;">';
      myContent = myContent + '[Movies]Description<\/td><\/tr><\/table>';

      ImageMatrix = new dataMatrix ($('ThumbDiv'), 'Movies', myHeader,
                                    '[Movies]Screenshot', myContent,
                                    '[Movies]LongDescription', 'left',
                                    4, 'auto', 'vert', 5, 2, false);

      ImageMatrix.customize(false, false, false, false, true, false, false, false);

      ImageMatrix.onCellClick = function (cellReference) {
```

```
         placeRecordData($('main'), cellReference);
         };

      ImageMatrix.onCellDblClick = function (cellReference) {
         placeRecordData($('main'), cellReference);
         $('myIframe').src = $('movpath').value;
         };
      };
      </script>
   </head>
   <body id="main" onload="Login('Guest','');" onunload="dax_bridge.logout();">

   <table >
   <div id="background" style="position:absolute; top:8; left:9; width:1062px;
height:579px; background-color:#616D7E;" >
   <tr>

   <div id="ThumbDiv" style="position:absolute; top:10px; left:10px; width:350px;
height:480px; "></div>

   <div id="DescDiv" style="position:absolute; top:495px; left:10px; width:310px;
height:81px; background-color: #E3E4FA; font-size:15px; padding-top:5px; padding-
left:5px; padding-right:5px; padding-bottom:5px; color: #302226;">
   <span class="4daf_[Movies]LongDescription"></span>
   </div>

   <input id="movpath" type="hidden" value="" class="4daf_[Movies]Path"/>

   <iframe src="/st.html" style="position:absolute; top:90px; left:350px; width:
720px; height: 496px; background-color: #E3E4FA;" scrolling="auto" frameborder="0"
id="myIframe">
   </iframe>

   <div id="stripe" style="position:absolute; top:90px; left:331px; width:18px;
height:496px; background-color:#98AFC7;" ></div>

   </tr>

   <img src="4DVTN3.jpg" alt="Welcome to demo.4D.com" width="1060" height="80"
style="position:absolute; top:9px; left:10px;" />

   </div>

   </table>

   </body>
</html>
```

For the banner image (“4DVTN3.jpg”), be sure to place it where it is referenced to in the code (next to “index_movies.html”).

Conclusion

This Technical Note guided the 4D developer through the process of constructing custom HTML pages embedded with 4DAF objects using minimal Javascript and HTML code. The developer should now posses a solid familiarity with the Daxipedia resource and is encouraged to delve deeper into the 4D Ajax Frameworks’ features. The developer should also now have introductory knowledge of HTML and they are encouraged to inspect the custom style attributes applied to “index_movies.html” and other HTML pages.