



**altio**  
S E E • N O W

# Markets Tutorial

---

## *Summary*

---

An exercise for developers new to AltioLive to build their first application

**Level:** Basic  
**Applies to:** AltioLive version 5.4  
**Date:** December 2009

### **Integra SP**

88 Wood Street London  
EC2V 7RS  
United Kingdom

[www.altio.com](http://www.altio.com)

tel: +44 (0) 20 8528 1045

# Contents

---

Contents .....	2
Introduction.....	3
Welcome .....	3
Create an Application.....	3
Log into the AltioLive Studio .....	3
AltioLive Studio: Creating an Application .....	4
Adding the MAR Application.....	4
Designer: Building the Markets Application View .....	5
Setting Font Styles.....	6
Setting Window Styles .....	8
Creating a window .....	9
Adding controls to a window .....	10
Setting up data for the application.....	15
Application Manager: Adding Service Functions and Datapools .....	17
Setting up the Service functions .....	17
Setting up the Datapool .....	19
Setting up Datakeys .....	20
Designer: Setting up Data for Controls.....	22
Refreshing the Application Information .....	22
Setting Initial Data.....	22
Getting Datatypes .....	23
Relationships between data elements.....	24
Associating data with controls .....	25
Seeing the window with data.....	28
Linking controls together .....	29
AltioDB: Extending the application to accept data updates.....	32
Altio Login Manager: Using the Login Servlet .....	33
Add User.....	33
Running the Application outside the Designer.....	34
Where to go next.....	35
Troubleshooting .....	35
General.....	35
Markets application-specific .....	35
Other sources of help.....	36
Appendix: example solutions for Markets application.....	37

## Introduction

---

### Welcome

As a new AltioLive developer you should read this document to help you:

- Work through an example application
- Start developing your own applications

You should have already read the Getting Started Manual and completed the Build a new ROS view example exercise.

### Create an Application

We will build a completely new application. The purpose of this application is to show sample data (stock prices) from three stock markets. The user can select one market to see extra detail on it. It will show you some standard methods used to build applications.

First we will build the View look and feel, without putting any data in. At this stage you won't see data updates. Then we will extend the View to show live data updates. This approach is useful to show a visual mock-up of an application with relatively little development time.

We will be using all of the Altio tools. There are many other features and options available which are outside the scope of this exercise.

If you have any problems, check the Troubleshooting section at the end of the manual.

### What you will need

You will need a basic understanding of web applications, XML, and/or HTML.

This tutorial assumes that you are using an AltioLive Studio edition.

### Should you need help

See the Troubleshooting chapter at the end of the document. The online help (available from the AltioLive Studio) gives detailed instructions on all features of AltioLive and includes a detailed troubleshooting section.

### Log into the AltioLive Studio

1. From your Start Menu, select **Programs | AltioLive ... | AltioLive Studio**.
2. Log in with a Login ID of **admin** and a Password of **admin**.

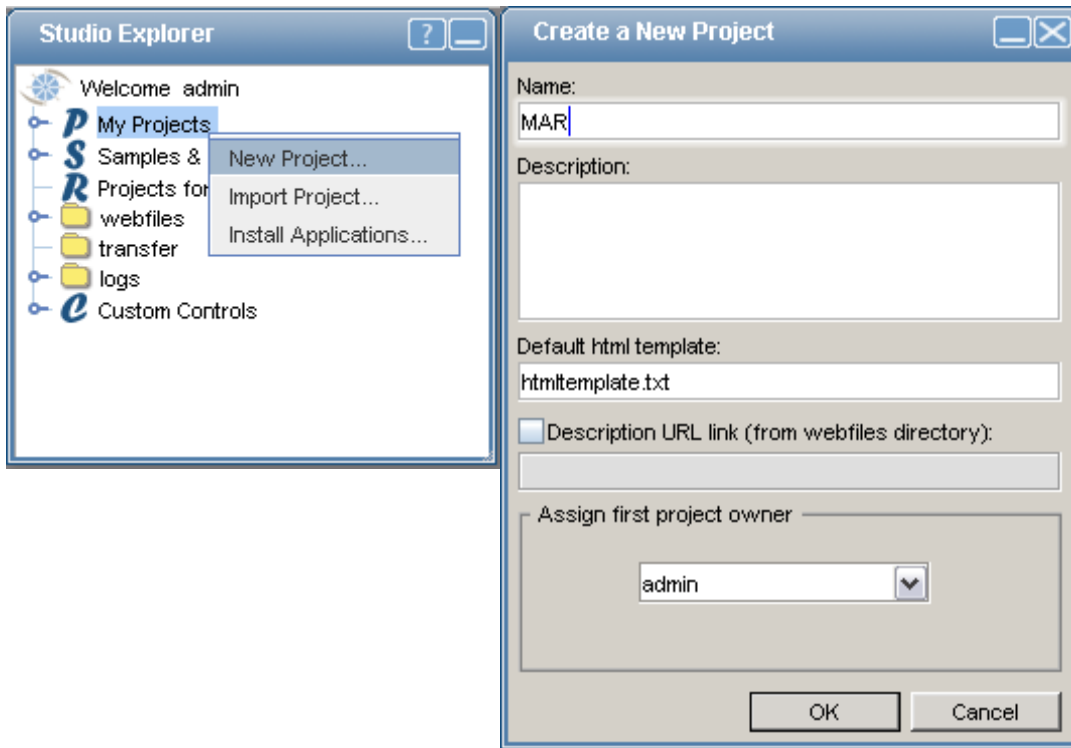
## AltioLive Studio: Creating an Application

---

This tool is the entry point of AltioLive and allows you to access all system tools, register new applications, run, manage and remove existing applications, upload files, and more. In this exercise, we will register a new application project.

### Adding the MAR Application

1. From the Studio Explorer window:
  - Right-click on the **My Project** node, and select **New Project**.
  - In the **Create a new Project** window, Enter MAR in the **Name** field, then click **OK**.

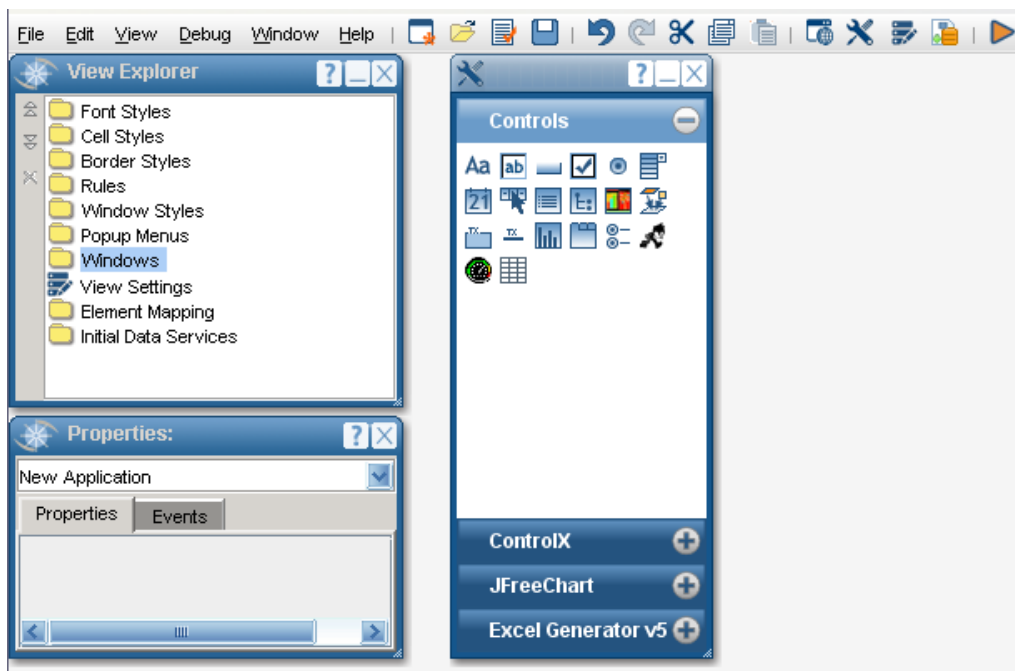




## Designer: Building the Markets Application View

We will use the Designer again to build the Markets Application View. More information on the Designer is in the online help.

1. Expand the **My Projects** node by double-clicking on it.
2. You will see a node for the project that you have just created, MAR. Right-click, and select the **Designer** option.

The Designer will load, with the **View Explorer** visible.



- The buttons on the top right are used to build, save and test applications. The Run button:  switches between design and test mode.
- The narrow window (in the middle of the screen) with a series of icons on it, is the **Controls** window, from where you drag controls onto windows.
- The file and View are displayed to the right of the Toolbar: `Application/View: MAR/newview.xml`. The View is initially called **view**; you can rename it when it is saved.
- The **View Explorer** window contains information specific to the currently open view.
- The **Properties** window, displayed by clicking on the **Properties** icon , is used to set all the components of the view.

In this section we will set up default font styles and window styles. While it is possible to build the application using the standard font styles and windows and switch to using custom styles later, it is more efficient to set up custom styles before building the interface.

# Setting Font Styles

Font styles enhance the appearance of the application and can give a consistent appearance to multiple controls, which is especially relevant to control labels.

1. At the **View Explorer** window, select the **Font Styles** folder icon.
2. Click the **Edit** menu and select **New Font Style** to create a new Font Style. The new font style will be added to the folder and the default Font Style properties opened.  
*Please Note: to open an existing font style, expand the **Font Styles** folder, right-click on the font style you want to set then select **Properties** in the context menu.*
3. Set its properties as follows. Press Return after entering each value:

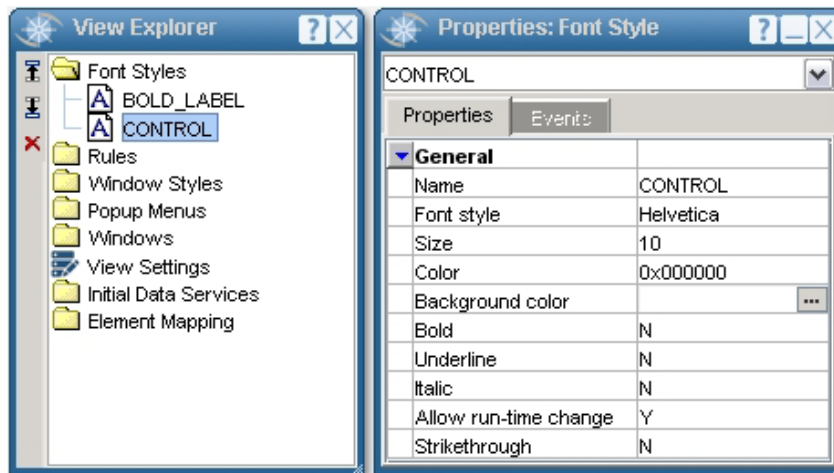
General Property	
<b>Name</b>	BOLD_LABEL
<b>Font Style</b>	Helvetica
<b>Size</b>	14
<b>Bold</b>	Y


4. Select the **Font Styles** folder icon again, and create second font style. Because the Properties window is already open, it automatically switches to show the properties of the newly selected font style.
5. Set its properties as follows:

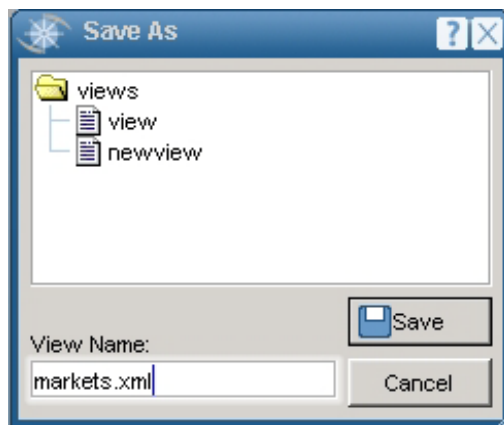
General Property	
<b>Name</b>	CONTROL
<b>Font</b>	Helvetica
<b>Size</b>	10

## Markets Tutorial

The screen will show as below.



6. Select **BOLD\_LABEL**, right-click, select clone font style and rename it **BOLD\_LABEL\_BLUE**.
7. Change the color of the new Fontstyle to blue.
8. Save the changes by clicking **Save**  on the toolbar. In the **View Name** field, enter **markets.xml**.



9. Click **Save**.

**Please Note:** Although you are regularly prompted to save your work while following this manual, you should try to save every 20 minutes or so. (AltioLive saves automatically when you run the application and when you exit the browser.) Many application servers will time-out after 30 minutes, which can result in changes being lost. To lengthen this time-out period, refer to the application server documentation.

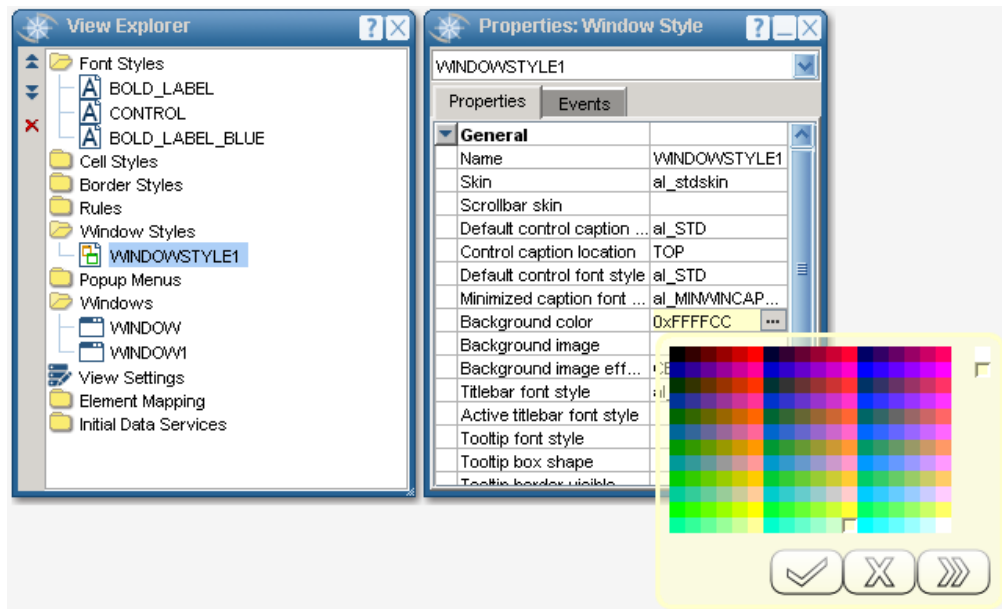
## Setting Window Styles

The look and feel of the Altio windows is defined by the window styles. These comprise a graphical image for the border, called a skin, plus settings for font styles, background colors, toolbars, and so forth.

We will use only one window style for the Markets application.

1. Select the **Window Styles** folder.
2. Click the **Edit | New Window Style** menu.  
A new window style (WINDOWSTYLE1) is added to the **Window Styles** folder; the **Properties** window for the new style opens.
3. Use the following property settings for the new window style:

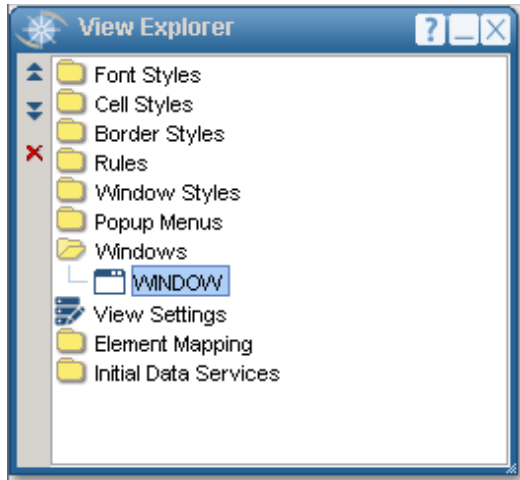
General Category	
<b>Name</b>	WINDOWSTYLE1
<b>Skin</b>	al_stdskin
<b>Default control font style</b>	CONTROL
<b>Background color</b>	0xFFFFCC (pale yellow – this is on the bottom row, 7 <sup>th</sup> )
<b>Titlebar font style</b>	BOLD_LABEL
<b>Active titlebar font style</b>	BOLD_LABEL_BLUE




## Creating a window

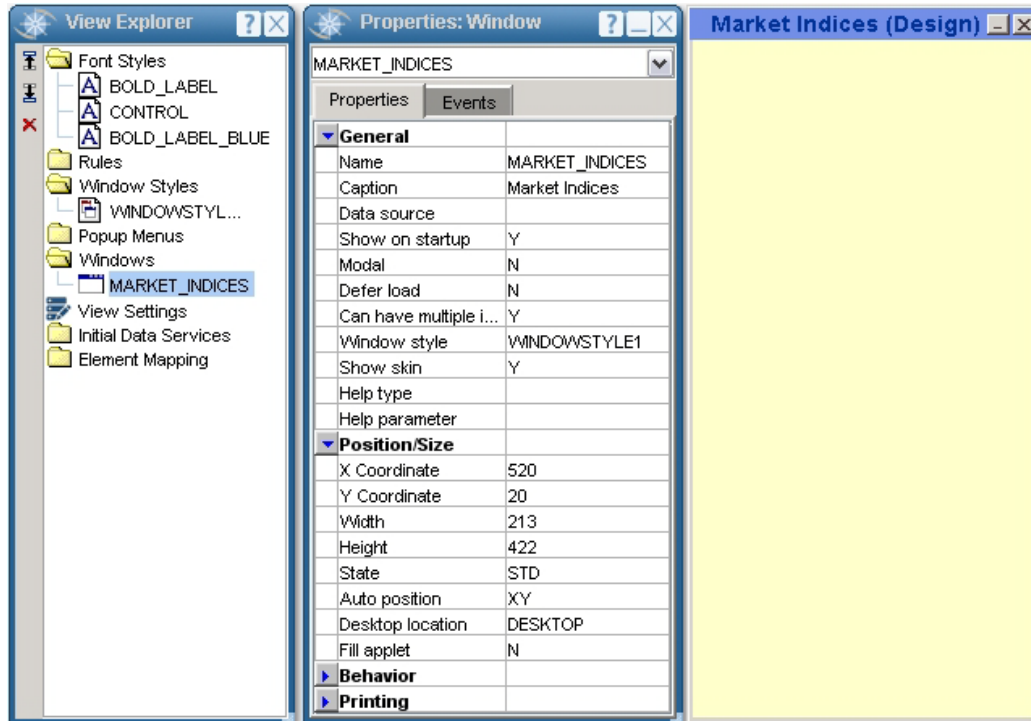
As part of a new View, a window is created automatically.



1. From the **View Explorer**, double click on the **Windows** folder then select the window displayed under it.




2. If the **Properties** window is not displayed, click the **Show Properties Window** icon .
3. At the **Properties** window, adjust the properties for the Window using the following settings:

GENERAL CATEGORY	
<b>Name</b>	MARKET_INDICES
<b>Caption</b>	Market Indices
<b>Window Style</b>	WINDOWSTYLE1
POSITION/SIZE CATEGORY	
<b>X Coordinate</b>	520
<b>Y Coordinate</b>	20
<b>Width</b>	300
<b>Height</b>	400



The **Run** button  puts the Designer in test mode. While in test mode, you can click the **Stop** button  to return to Design mode. This allows you to switch from the designer into test mode and back again.

**Please Note:** that the window can be dragged to any position in design mode. When the application is run it will use the position specified in the Window properties.


4. Click the **Run** button to view the window in test mode. You'll see an empty window. It can be moved around or minimized.
5. Click the **Stop** button to revert back to design mode.
6. Click the **Save** icon  to save your work.

## Adding controls to a window

Now we will add some controls. We will use a dropdown list to select which stock market to display data from, and a separate label to display the selected market.

While it is possible to drag and resize controls, it is difficult to get them at precisely the right position and size this way. Instead, we will manually alter position co-ordinates.

## Adding a dropdown list (Select Control)

1. If the **Control Palette** is not displayed, click on the **Control Palette** icon .
2. From the Controls window on the left, drag the drop-down list (Select) control onto the **MARKET\_INDICES** window at the position shown here.
3. If the Properties window is not displayed, right-click on the control and select **Properties** in the context menu.



4. Set the following properties for the drop-down list:

GENERAL CATEGORY	
<b>Name</b>	SELECT_MARKET
<b>Caption</b>	Select Market
<b>Default value</b>	DJIA
POSITION/SIZE CATEGORY	
<b>X Coordinate</b>	15
<b>Y Coordinate</b>	20
<b>Width</b>	150
<b>Height</b>	20
<b>List height</b>	120

## Adding a label control

1. Add a label to the window. The **Properties** window will change to show the label properties.
2. Set the following properties:

GENERAL CATEGORY	
Name	MARKET_NAME
Caption	
POSITION/SIZE CATEGORY	
X Coordinate	15
Y Coordinate	50
Width	220
Height	20
APPEARANCE CATEGORY	
Caption Font Style	BOLD_LABEL

The controls should now be placed as shown the following illustration:



## Adding a Chart control

Next we will add a chart to graph the stock values.

1. Select the Chart icon from the Controls Palette and drag it onto the **MARKET\_INDICES** window.

- Set the following properties of the chart:

General Category	
Name	MARKET_CHART
Caption	Market Chart
Position/Size Category	
X Coordinate	15
Y Coordinate	100
Width	250
Height	150
Appearance Category	
Caption Font style	CONTROL


## Adding a list control

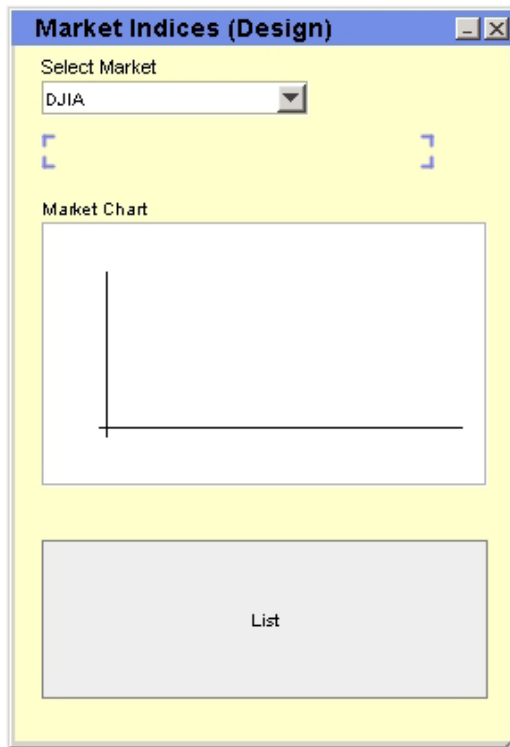
The list control enables tabular data to be displayed. We will use it to display the stock values for each market. To add a list control to the window, follow the steps below.


- Select the list icon from the Controls palette and drag it near the bottom of the **MARKET\_INDICES** window, under the chart.
- Set the following properties of the list:

General Category	
Name	MARKET_LIST
Caption	
Position/Size Category	
X Coordinate	15
Y Coordinate	280
Width	250
Height	90
Appearance Category	
Row height	20
Selected row color	0xFFFFCC (pale yellow)
Behavior Category	
Multi row select	N

## Markets Tutorial

3. Click **Run** to view the window . It will now contain controls with no data. The dropdown list will have the default value we entered earlier, but no other data.



4. Click the **Stop** button  to revert back to design mode.
5. Save your work, and then close the Designer window.

## Setting up data for the application

---

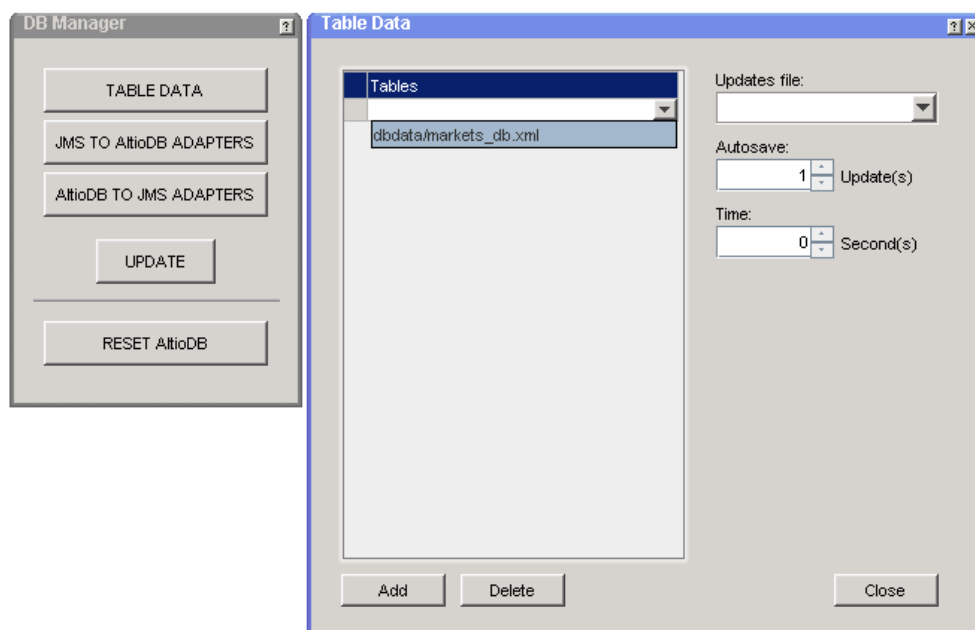
In order to associate data with controls, we must configure some files outside the Designer.

At this stage we will also integrate the application with AltioDB. AltioDB is a database supplied with AltioLive which enables us to integrate with flat files in order to simulate a working back-end application. AltioDB must be configured with the location of the flat files used as database tables, as well as instructions for saving data from the client (however, we won't save any data in this exercise).

1. In the **tomcat\webapps\altio54\docs\example** directory, locate the files **markets\_db.xml** and **updates\_markets\_db.xml**.
2. Copy these to the **tomcat\webapps\altio54\WEB-INF\classes\apps\MAR\dbdata** folder. If the **dbdata** folder does not exist, create it.

These files contain sample initial data and update data for the Markets. We won't incorporate these files into the application at this stage; we are copying them to avoid repetition later. The other files in the folder are used in a different set of exercises.

3. Return to the AltioLive Studio.
4. Right-click on the **MAR** project and select **AltioDB**. The Altio DB manager will load in a new window.
5. Click on the **Table Data** button.
6. In the **Table Data** window, click on the **Add** button.
7. Click in the first row. A drop-down menu will appear. Click on the dropdown menu and select the entry **dbdata/markets\_db.xml**.



## Markets Tutorial

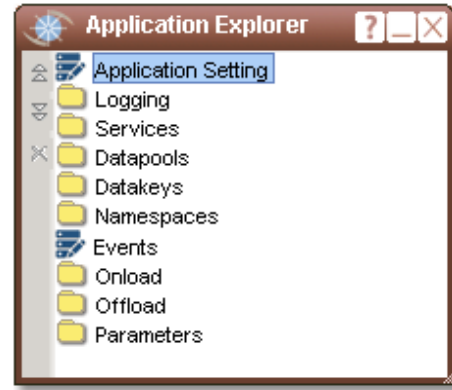
8. Click on the **Update** button. Click **Yes** in response to the prompt.
9. The Results window displays. Click on the **Close** button.
10. Click on the **Reset AltioDB** button to re-initialize the Altio database. Click **Yes** in response to the prompt.
11. Close the **Altio DB** manager.

## Application Manager: Adding Service Functions and Datapools

This tool allows you to administer any project. This includes integrating with the back-end web application and setting application-specific logging methods. In this exercise we will add service functions and datapools to provide data to the View. These read and/or update the underlying data used for the View.


For more information, see the online help.

1. From the AltioLive Studio, right-click on **MAR** and select **Application Manager**. It will load in a new window.



## Setting up the Service functions

First we will set up a service function to get initial data for the View. Service functions allow us to interact with the back-end web application (in this case, AltioDB). The Service Function **GET\_MARKETS** will reference the data contained in the **markets\_db.xml** file.

1. Click the **HTTP** Toolbar button  to add a new HTTP Service Function.
2. Enter the following data for the Service Function:

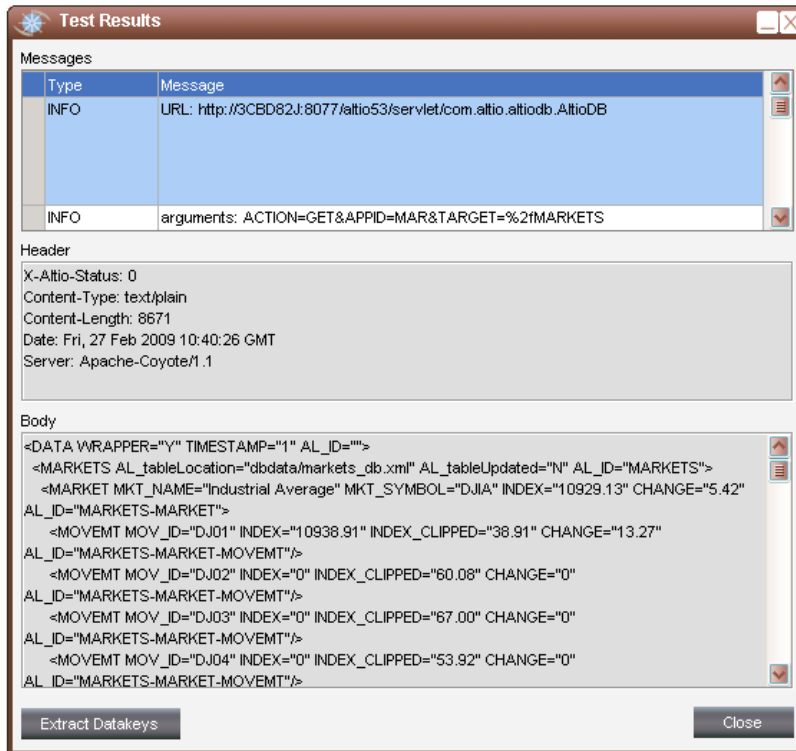
Request Tab		
<b>Service Name</b>	GET_MARKETS	
<b>URL</b>	http://\${http.server.name}:\${http.server.port}/\${http.server.path}com.altio.altiodb.AltioDB	
<b>Parameter Mappings</b>	ACTION	GET
	TARGET	/MARKETS
	APPID	MAR
Response Tab		
<b>Document Type</b>	XML	
Acknowledgement Tab		
<b>Source</b>	NONE	


# Markets Tutorial

The Request Tab:



3. Select the **Test** Tab, then click the **Test** button to test the service, and verify that data comes through in the message body. This will be shown in the **Test Results** window:



4. Click **Close** to close the **Test Results** window.
5. Click **OK** to save the new Service Function.
6. Click the **Save** Toolbar icon: .
7. The **Confirm Save** window pops up, click the **OK** button.
8. The **Validation Results** window displays. Check that there are no errors in the **Messages** table and click on the **Close** button.

## Setting up the Datapool

Next we will set up the datapool, **MARKETS\_POOL**, which distributes new data updates to the Client. Datapools 'sit on top' of service functions, and use them to collect updates either by polling the back-end application at a regular interval or by having updates pushed to them. We start by copying the **GET\_MARKETS** Service Function, and then modify the copy to be used by the Datapool.

1. At the **Application Explorer** select the **GET\_MARKETS** Service Function.
2. Click the **Edit** menu and select the **Clone GET\_MARKETS** option. An HTTP service will be added, using the same values as the **GET\_MARKETS** service.
3. Amend the title to **GET\_MARKETS\_DP**.
4. Under the **Parameter Mapping** area, click the **Add** button
5. Enter an additional parameter mapping:

Name	Value
TIMESTAMP	\${datapool.timestamp}

The parameter **TIMESTAMP** is used by AltioDB: only those elements added, deleted or updated after the timestamp are returned.

The value **\${datapool.timestamp}** (the timestamp value of the Datapool) is used by the Back-end application (in this case AltioDB) to filter the data so as only to provide data updates that are more recent than the data held at the Datapool.

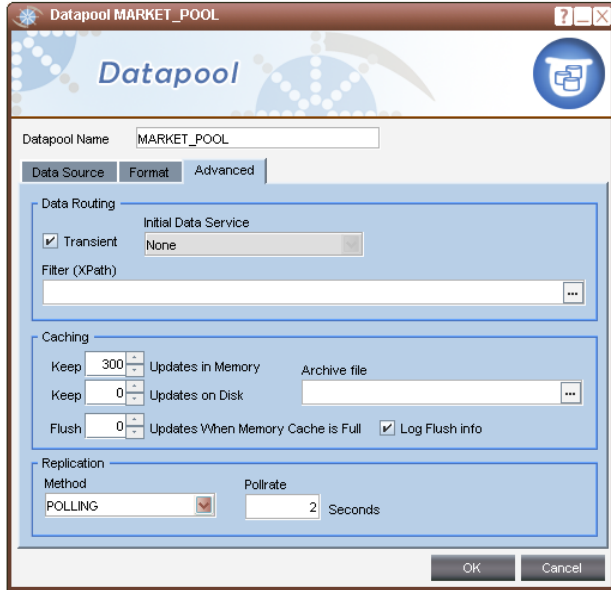
6. Click the **OK** button.
7. Save the changes.

With the Service Function defined, we can configure the Datapool.

1. Right-click the **Datapools** folder, and select the **New Datapool** option. A Datapool configuration window will display.
2. Enter the following data for the datapool:

Data Source Tab	
Datapool Name	MARKETS_POOL
Service (HTTP, JDBC...)	GET_MARKETS_DP
Pollrate	1

Advanced Tab	
Keep Updates in Memory	300



3. Click **OK** to close the datapool window.

## Setting up Datakeys

Next we will set up data keys to help the Synchronization Engine identify saved data. These enable quicker indexing of data.

You can add data keys manually or you can automatically extract data keys from the data returned from a service function.

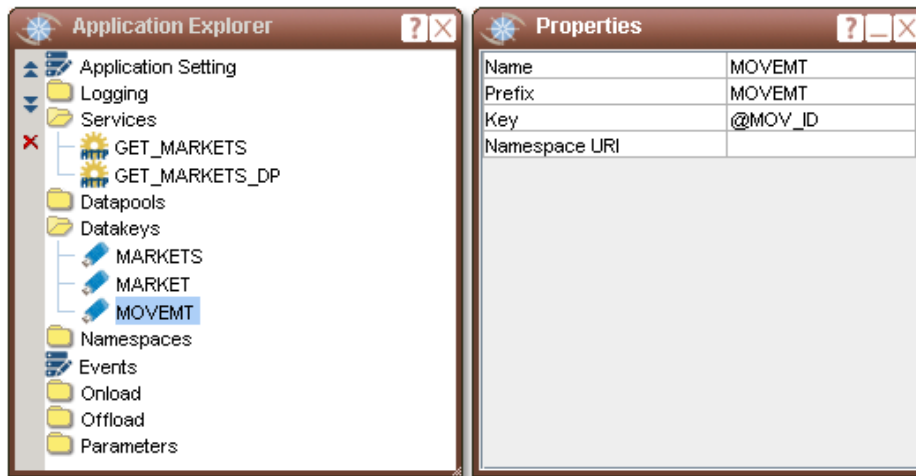
1. Double click the **GET\_MARKETS** Service Function.
2. Click on the **Test** button on the **Test** tab.
3. When the data is returned, click on the **Extract Datakeys** on the **Test Results** window.
4. On the Application Explorer window, select the **Datakeys** folder icon.
5. Notice that three new data keys have been created for **MARKETS**, **MARKET** and **MOVEMT**.
6. Enter the following data for the **MARKET** data key:

<b>Name</b>	MARKET
<b>Prefix</b>	MKT
<b>Key</b>	@MKT_SYMBOL



## Markets Tutorial

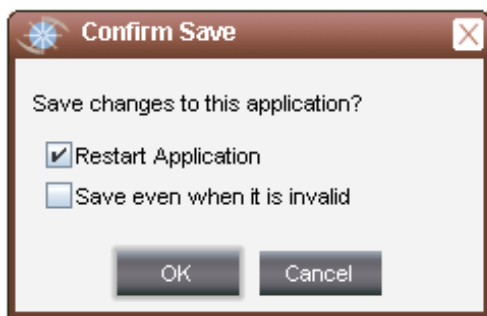
7. And for the **MOVEMENT** data key:

<b>Name</b>	MOVEMENT
<b>Prefix</b>	MOVEMENT
<b>Key</b>	@MOV_ID



Now we will save our changes to the application configuration.

8. At the toolbar, click the **Validate** button  to check your changes.
9. If there are no problems reported in the Results window, click **Save** .



10. Tick the **Restart Application** option before clicking on the **OK** button, to ensure that changes will take effect.

## Designer: Setting up Data for Controls

---

Launch the Designer as previously described. If the View **markets** is not automatically opened, click the **File** menu, select the **Open View** option and select **markets** from the list of Views available to open.


### Refreshing the Application Information

Before we set up the data for the View, the Application information needs to be refreshed in order to show the changes made using the Application Manager.

1. Click the **File** menu.
2. Select the **Refresh Application Information** option.

### Setting Initial Data

First we must set the Service Function used to initialize the View, and the datapool which will provide data updates.

1. Ensure that the Properties window is displayed by clicking the **Show Properties Explorer**  icon on the toolbar.
2. Select the **Initial Data Services** folder icon.
3. At the **Edit** menu, select the **New Initial Data** option.  
A node named **INCLUDE** will be added to the **Initial Data Services** folder.
4. Expand the **General** category of the **Properties** window.
5. From the **Server Command** drop-down menu, select **GET\_MARKETS**.  
This will get data from the service function.
6. Select the **Subscribe** option **MARKETS\_POOL** from the dropdown field.  
This will subscribe to the datapool in order to get updates to the initial data.



## Getting Datatypes

Datatypes are used as sample data to help build a View. This representation of the data makes it easier to link data to controls.

1. At the Toolbar, click on the **View | Datatypes** menu item.

The **Datatypes** window displays. To see a representation of the data available from the Initial Data, the Datatypes must first be loaded:

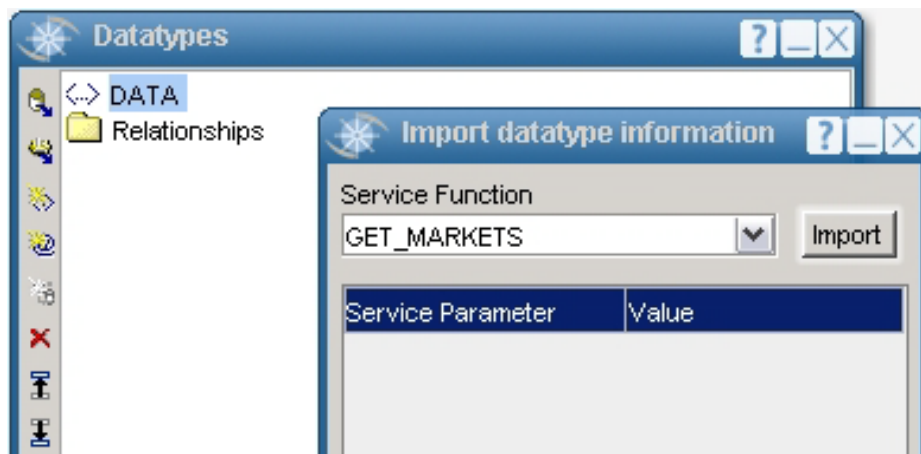
2. Click the **Import From Service...** button .

The **Import datatype information** window is displayed. This allows you to import data from a Service Function and convert it into datatypes.

3. Select the Service Function **GET\_MARKETS**. Click on the **Import** button to load the datatypes.

This will access the **markets\_db.xml** data (using the AltioDB Manager configuration information we added earlier). The Diagram below shows the **Import datatype information** pop-up window.

**Please note:** the data elements displayed in the tree hierarchy are available once the Import Datatype selection is confirmed using the Import button.



4. Expand the DATA node, then 'drill-down' through the tree. Select the **MARKETS** datatype. A list of all the attributes within the Markets datatype is displayed to the right.
5. Save the View

## Relationships between data elements

The **Datatypes** window describes the data available for the AltioLive application. The structure of the data for this AltioLive application looks like:

```
<MARKETS>
  <MARKET MKT_NAME="Industrial Average" MKT_SYMBOL="DJIA" INDEX="10929.13" CHANGE="5.42">
    <MOVEMT MOV_ID="DJ01" INDEX="10938.91" INDEX_CLIPPED="38.91" CHANGE="13.27"/>
    <MOVEMT MOV_ID="DJ02" INDEX="0" INDEX_CLIPPED="60.08" CHANGE="0"/>
    ...
  </MARKET>
  <MARKET MKT_NAME="NASDAQ Composite Index" MKT_SYMBOL="NASDAQ" INDEX="3375.42" CHANGE="2.60">
    <MOVEMT MOV_ID="NAS01" INDEX="3375.42" INDEX_CLIPPED="75.42" CHANGE="2.60"/>
    <MOVEMT MOV_ID="NAS02" INDEX="0" INDEX_CLIPPED="26.09" CHANGE="0"/>
    ...
  </MARKET>
  <MARKET MKT_NAME="Standard and Poors 500" MKT_SYMBOL="S & P 500" INDEX="1426.53" CHANGE="4.87">
    <MOVEMT MOV_ID="SP01" INDEX="1426.53" INDEX_CLIPPED="6.53" CHANGE="4.87"/>
    <MOVEMT MOV_ID="SP02" INDEX="0" INDEX_CLIPPED="30.75" CHANGE="0"/>
    ...
  </MARKET>
</MARKETS>
```

Within each **MARKET** element are **MOVEMT** elements. These hold details of changes in market values. The **INDEX\_CLIPPED** attribute values will be plotted on a chart for a selected **MARKET**.

Looking at the Market Indices window, the dropdown field will be used to select a market. The label and chart must be driven by this selection; they should show data for the selected market. In order to display the name and **INDEX\_CLIPPED** values for a particular market, there must be a relationship between **MARKET** elements and **MOVEMT** elements. The Designer automatically assumes this, as these nodes are all branches of the same tree. If they were on separate trees you would have to set up an XPath relationship to specify a connection between elements.

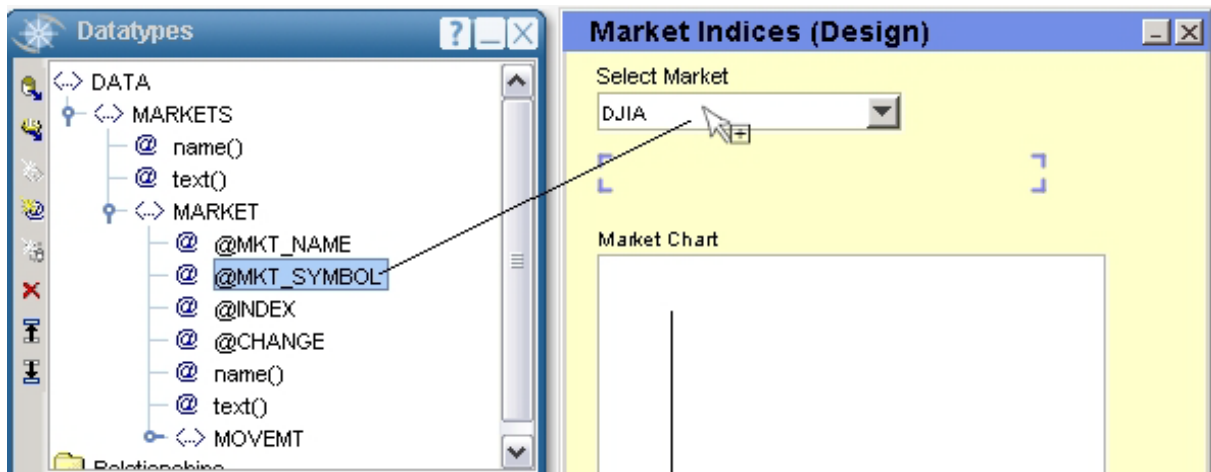
## Associating data with controls

Now the controls have been placed on the window they can have data assigned. This can be done manually by editing the properties or by dragging and dropping data attributes from the Datatypes to each control.

In this example, data will be assigned by drag and drop. It is worth looking at the properties of the controls to see the changes that are made.

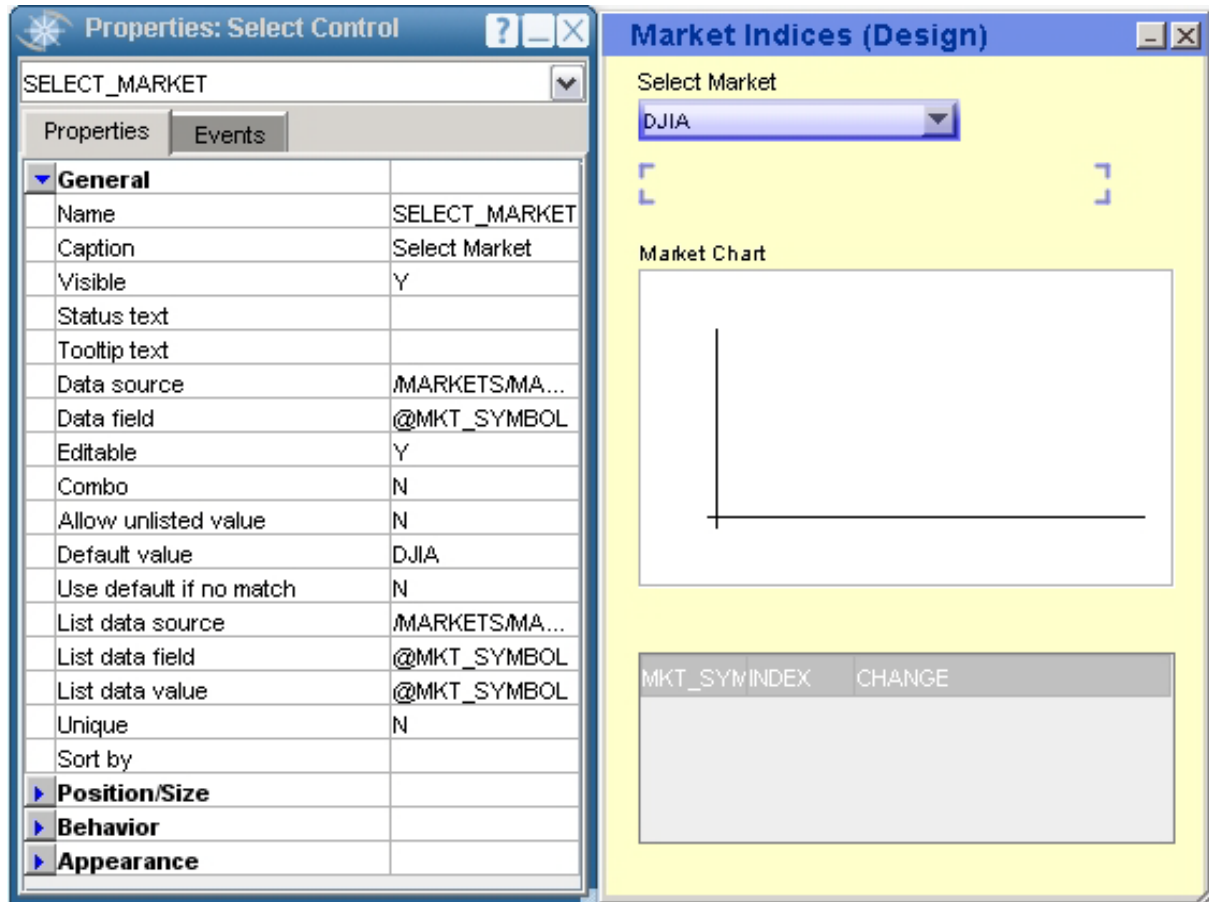
### Select Control (Dropdown list)

1. On the **View Explorer** window, expand the **Windows** node.
2. Double-click on the window **MARKET\_INDICES** to open it.
3. At the toolbar, click the **Show Datatypes Window** button. (When the Datatypes window opens, adjust the position of the windows so that they do not overlap).
4. At the Datatypes window, expand the **DATA** tree and select the **MARKET** node.
5. Click on the **MKT\_SYMBOL** attribute. Drag and drop the **MKT\_SYMBOL** attribute onto the dropdown list.



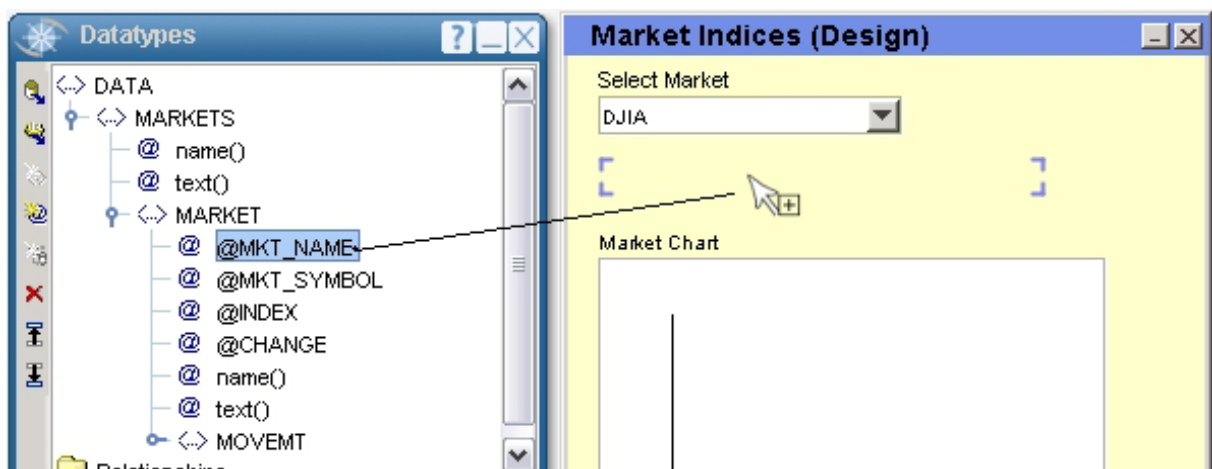
This action results in the **Data source**, **Data field**, **List data source**, **List data field** and **List data value** fields being populated in the control's properties. Right-click on the dropdown list, and select **Properties** to view the **Properties** window to check this. The **Data source** will show an XPath statement, targeting the **MARKET** element. XPath is a language used to interrogate an XML data file and retrieve specific elements or attributes.

## Markets Tutorial

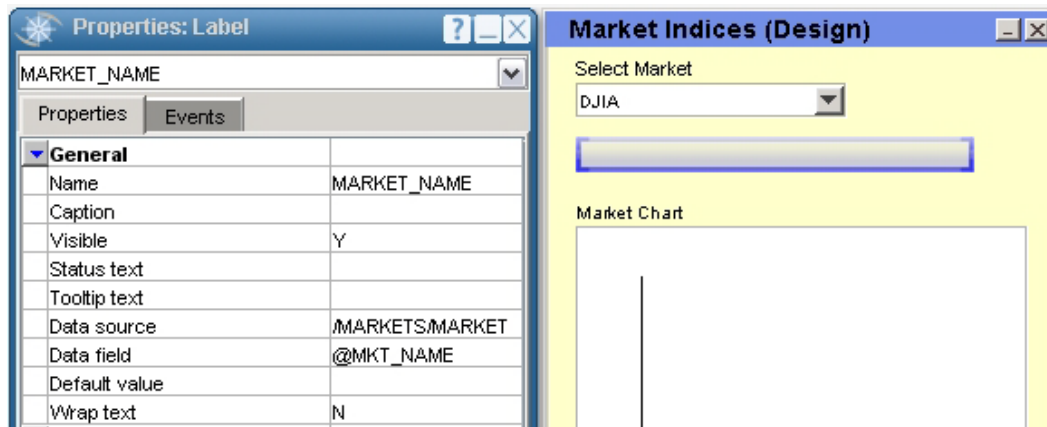


## Label

1. Drag and drop the **MKT\_NAME** attribute onto the label using the same method as above.

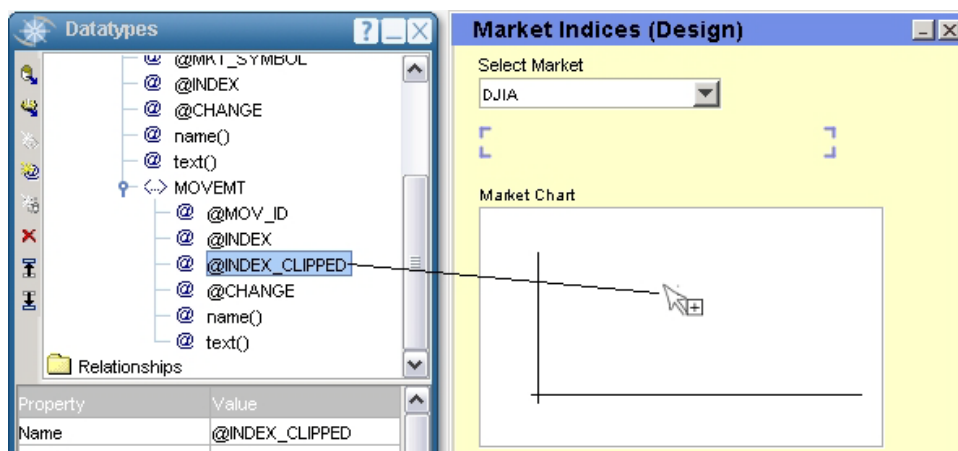


2. Select **Properties** from the Shortcut Menu for the control. Again, the action has affected the properties. This time it has populated the **Data source** and **Data field** attributes.



## Chart

1. To provide data for the chart open the **MOVEMT** data type and drag and drop the **INDEX\_CLIPPED** attribute across to the chart.



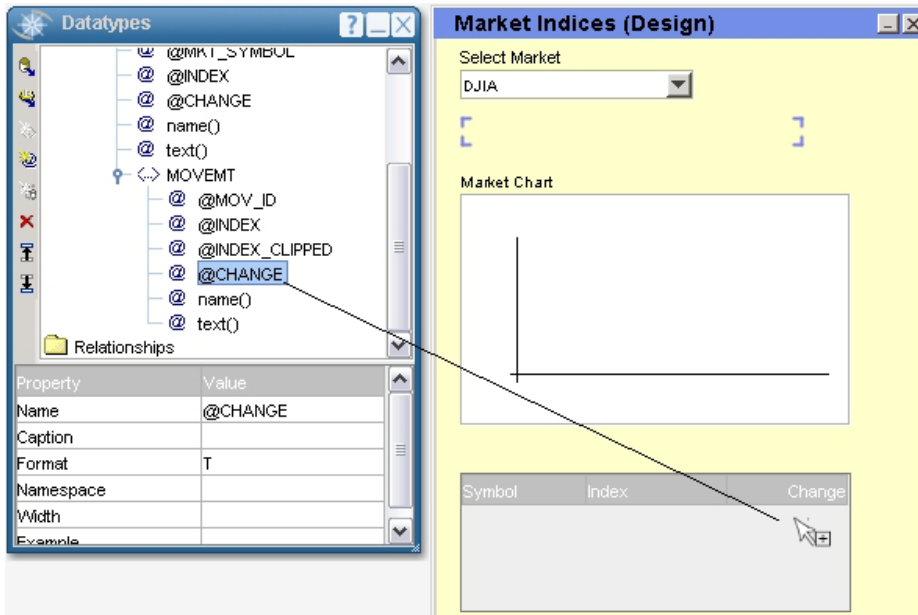
This adds a plot to the Chart and sets its **Data source** and **Y Attribute** to be displayed on the Y axis.

**Please note:** If no plot appears under the chart, do the following:

1. From the **View Explorer**, select the chart, right-click on it and select the **New Plot** option in the context menu.
2. Select the Plot and from the **Properties** window set the following parameters:
  - **Data source:** /MARKETS/MARKET/MOVEMT
  - **Y Attribute:** @INDEX\_CLIPPED

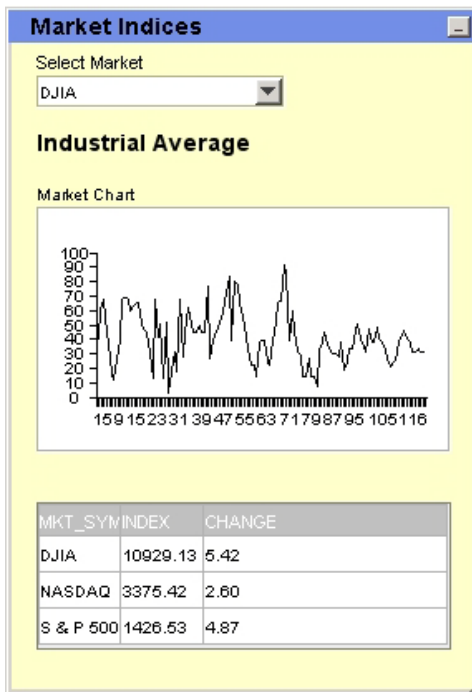
## List Control

To add columns to the list select the **MARKET** data type again and drag and drop the attributes **MKT\_SYMBOL**, **INDEX** and **CHANGE** onto the list. You will see the column headings appear on the control.



This has added data to each of the controls.

## Seeing the window with data



1. Click Run to see the window with the data.

Currently, choosing a different market in the select box does not affect the data shown in the other controls. To achieve this, the next step is to link the chart and label controls to the select box so that their content is related to the market selected.

2. Return to the Designer and save the View again.

## Linking controls together

Now we need to drag and drop *between* controls to create the links. This will give us an XPath statement linking the controls together. To stop the control from moving while dragging, hold down the Shift key.

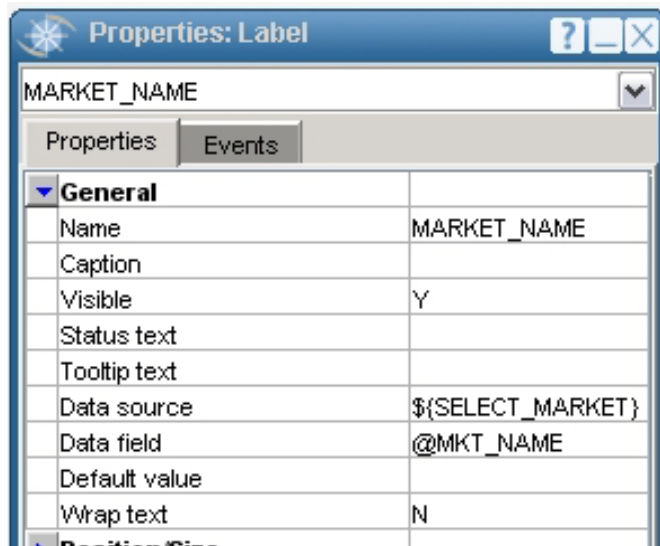
### Linking the Dropdown List to the Label

1. Click on the dropdown list control, then hold down the Shift key and use the mouse to drag to the label.



2. View the properties for the label.

The value for the **Data source** has changed so that it uses the value in the dropdown list. Since the dropdown list has a data source, the data source for the label is inherited from it.



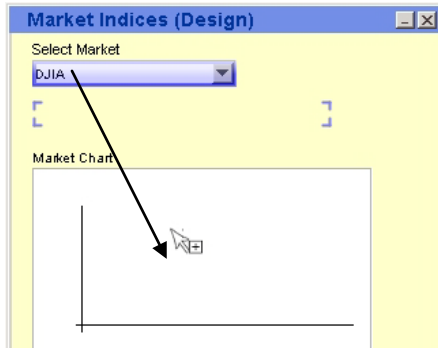
3. Click Run. In Test mode the label content changes when the select box value changes.

## Linking the Dropdown List to the Chart

The next step is to have the chart to show the index values for the selected market (currently it is showing all the markets together and it is hard to tell which values belong to which). To achieve this, the chart must be linked to the select box.

1. Select the dropdown list again.
2. Hold down the Shift key and drag and drop from the dropdown list to the chart to create a link between the two.

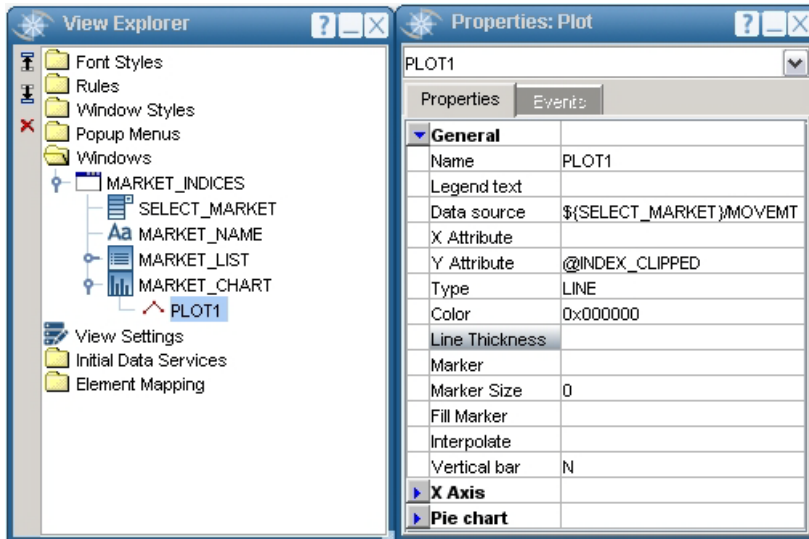
**Please note:** The cursor change to a plus to indicate that you can drop.



3. View the Chart PLOT properties again.

The data source for the plot has changed to specify `$(SELECT_MARKET)/MOVEMENT`

This means that the data source `/MARKETS/MARKET` is inherited from the select box; the `/MOVEMENT` portion of the data source is retained so that the `INDEX_CLIPPED` attributes can be displayed in the chart.



4. Click Run to go to test mode. Notice that when the select market is changed, both the chart and label are automatically refreshed with new data.

5. Return to the Designer by clicking on the Stop button  and save the View .

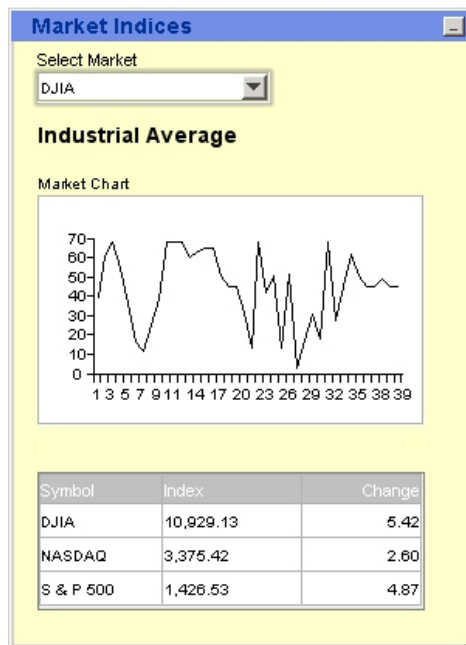
## Settings the Columns


Now that each of our controls has data assigned, all that needs to be done is to adjust the width of the columns in the list. Columns are given a default width of 50 pixels when set up, but you will usually need to adjust the width to match the data displayed in them.

1. At the **View Explorer**, expand the **Windows** node so that the columns for the **MARKET\_LIST** can be accessed.
2. Right-click on the first column (**MKT\_SYMBOL**) and select **Properties** from the context Menu. The Properties window allows you to adjust the setting for the column.
3. Set the properties of the columns as follows.


**Please note:** Alignment is in the Appearance category:

Name	Caption	Width	Format	Alignment
MKT_SYMBOL	Symbol	80	T	
INDEX	Index	90	N	LEFT
CHANGE	Change	60	N	RIGHT



4. Click the Run button  to launch the application within the Designer.  
You have successfully created a new application!

5. Click the Stop button  to return to design mode.

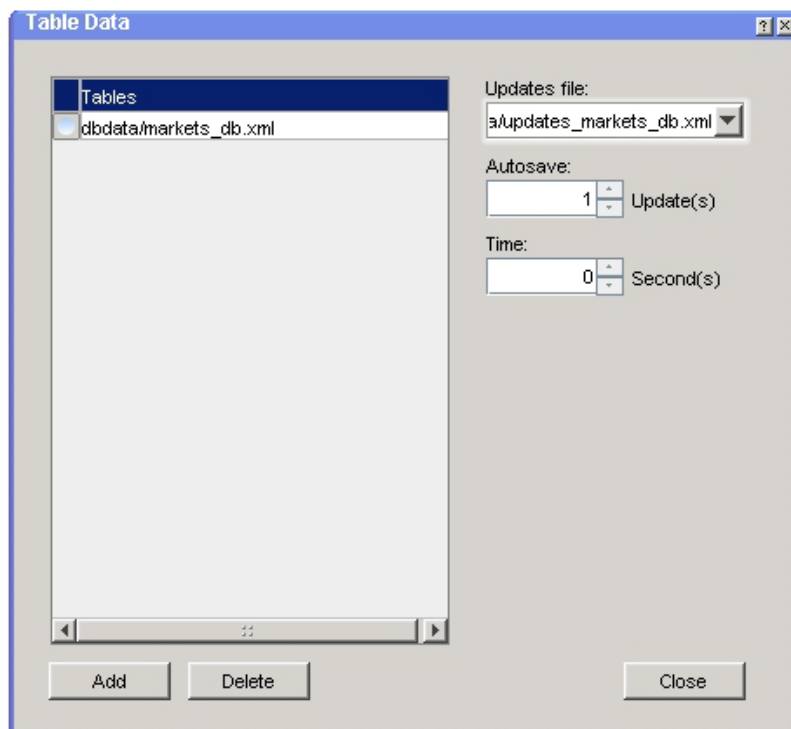
6. Save the View .

## AltioDB: Extending the application to accept data updates

---

In the previous section we created the application configuration files. Now we will incorporate data updates. These updates come from a file **updates\_markets\_db.xml**. In a real world situation the updates would come from the back-end application but we will use a flat file to simulate the effect.

1. Return to the AltioLive Studio.
2. Right-click on the **MAR** project, and select **AltioDB**. It will load in a new browser window.
3. Click on the **TABLE DATA** button.
4. In the **Updates file** field of the Table Data window, click on the dropdown arrow and select the entry **dbdata/updates\_markets\_db.xml**.



5. Click **Update** to save your changes to the file. Click **YES** to answer the question.
6. After you have received the **The machine was updated successfully.** message, click on the **Close** button.
7. Click on the **RESET AltioDB** button. Click **YES** to answer the question .
8. Close the browser window.

9. Now re-launch the Designer and click the **Play** button, to view the application running with the updates.

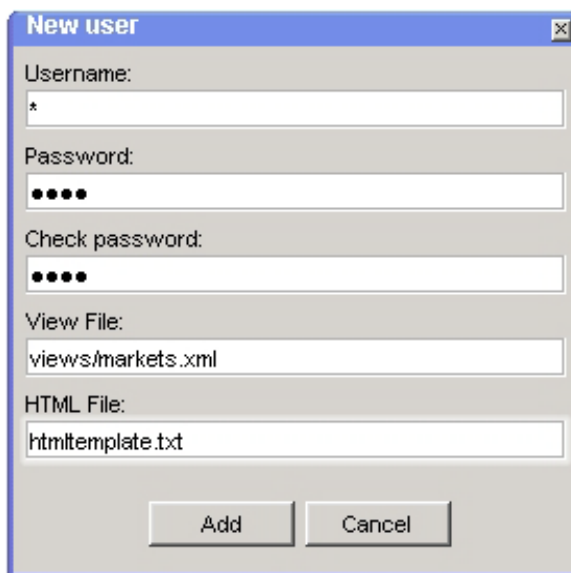
## Altio Login Manager: Using the Login Servlet

---

Now that you have finished developing your application, you can optionally choose to run it without using the Designer by using a login servlet. This example uses the login servlet provided by Altio in the Prototyping Kit.

### Add User

1. Return to the AltioLive Studio.
2. Right-click on the **MAR** project, and select **Altio Login Manager**. It will load in a new browser window.
3. In the User Manager window click on the New... menu item.
4. In the **Username** field, enter \*. The \* means any user can log in, when the right password is used.
5. In the **Password** field enter **demo**. Re-enter **demo** in the **Check password** field.
6. In the **View File** field, enter **views/markets.xml**.
7. In the **HTML File** field, enter **htmltemplate.txt**. The **HTML File** contains the complete HTML page to be loaded, including the Altio desktop. A sample page is provided for your use in the **apps/appname** directory (e.g. markets in this case).



The image shows a 'New user' dialog box with the following fields and values:

- Username: \*
- Password: ●●●●
- Check password: ●●●●
- View File: views/markets.xml
- HTML File: htmltemplate.txt

Buttons: Add, Cancel


8. Click **Add**.
9. When the Service Function completes, close the browser window.

## Running the Application outside the Designer

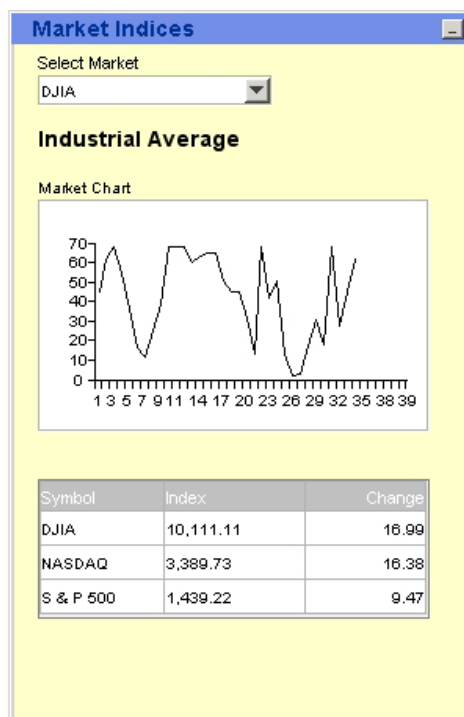
---

The application is ready to run without the Designer. While it is possible to load the application without the Designer in review mode from the AltioLive studio (by double-clicking on the **MAR** project, navigating to the **markets.xml** view file, right-clicking and selecting **Run Review** ), in production mode you will want to use an external Login Handler to access your application, not AltioLive Studio.

Follow the steps below to launch it independently from the Studio.

1. Load the **AltioLive Studio**.
2. Append **studio-console.jsp** to the URL in your browser URL field just after **.../altio54/**. Load this URL. This is an alternative console to the Studio.
3. In the console window, enter the **User Name** and **Password** you have set up, and an **Application ID** of MAR.
4. Click the **Run**  button.

The Markets application will load in a new browser window. The application screen appears as below. See how the graph and the data in the list (3<sup>rd</sup> column) updates approximately every 3 seconds (depending on your application server).



## Where to go next

---

The manual Integrating AltioLive, available from the start menu, has an overview of methods for integrating with your back-end application.

For more detailed information on using AltioLive, refer to the online help – the contents of this are similar to a reference manual.

## Troubleshooting

---

### General

- **Browser window not showing new changes/Browser window doesn't load new tool:**  
This happens because the browser will cache data if it has not been shut down. Ensure you have closed all your browser windows. Stop and restart your application server. Then restart your browser.
- **You are no longer logged on (error message):**  
This error message occurs when you attempt to save your work after not saving it within the servlet engine's timeout period. Check the servlet engine's documentation for details on changing this period.  
In the meantime, you will have to close all your browser windows, then stop and restart your application server. When you restart, you will return to the point at which you last saved.
- **Error 6: Service Thread Error. Licensed Connections Exceeded.**  
Starting one of the Altio tools more than once, and not restarting your application server in between times, can trigger this error. The evaluation kit permits 5 concurrent sessions, and the Developer edition permits 10. Close all browser windows, restart your application server, and try again.

### Markets application-specific

- **Cannot Import Datatypes**  
*The GET service function is not correct:*  
See Setting up the Service Function in the Online Help.  
*The file name or path referenced in the altiodb configuration is incorrect:*  
See AltioDB: Extending the application to accept data updates.  
*The service function is not correctly referred to in the **Initial Data** for the view:*  
See Setting Initial Data.

- **Incorrect values in Datatypes**

*The View **markets** must be open.*

Open the **File | Open View** menu. Open the **Datatypes** window, delete the existing datatypes. Import datatypes using the **GET\_MARKETS** service.

- **Label and/or Chart details do not change when Market dropdown list is changed**

*The label or chart has a **Data source** attribute specified.*

This can be caused by dragging and dropping an attribute from the datatypes. This cancels the effect of the Context Control attribute. You should Shift+drag from the Context control once again to re-establish the context.

Also see: Designer – Setting up Data for Controls in the Online Help.

- **No data updates**

*The Updates file has not been copied into the dbdata dir:*

See ; **Error! Reference source not found..**

*The Updates file has not been referenced in the altiodb configuration:*

See AltioDB: Extending the application to accept data updates; also check your file against that in the appendix.

*The Updates datapool has not been subscribed to:*

Look on the **View Manager, View** tab. Under **Initial Data**, check that the **GET\_MARKETS** has a Subscribe command set to **MARKETS\_POOL**.

*The Poll rate does not have a value:*

In the Designer check the **View Settings/Connection/Poll rate**, set it to 2.

*Datakeys and Datapools are incorrectly configured:*

See Setting up the Datapool

## Other sources of help

- **Online help**

More troubleshooting help is available in the online help, which can be accessed by clicking the question mark icon on any Altio window, or from the AltioLive Studio Help menu.

- **Developer Center**

<http://www.altio.com/developer/> - a community-gathering place where you'll find the technical resources, knowledge, and peer advice that will assure your success in building and deploying Altio applications.

<http://www.altio.com/forums/> - a wealth of questions and answers from members of the Altio community.

## Appendix: example solutions for Markets application

---

The markets.xml file.

```

<ALTIO DTD='altioview30.dtd' TYPE='VIEW'>
  <APPLICATION APPCONFIG='1.0' ABOUTTEXT='AltioLive New Application' NM='markets_tutorial' MSGWAITTIME='25'
  BGCOL='#FFFFFF' BGIMGEFFECT='CENTER' DIALOGSTYLE='al_STD' PROTOCOL='HTTP-POLL'/>
  <FONTSTYLES>
    <FONTSTYLE NM='BoldCaption' FONT='Helvetica' SIZE='12' BOLD='Y'/>
  </FONTSTYLES>
  <WINDOWS>
    <WINDOW NM='WINDOW' CAPTION='Market Indices' STARTUP='Y' X='418' Y='66' H='410' W='283'>
      <PANEL>
        <LIST NM='Market_LIST' CAPTION='LIST' H='90' W='250' X='15' Y='280' MULTIROWSELECT='N'
        DATA='/MARKETS/MARKET'>
          <COL NM='MKT_SYMBOL' CAPTION='MKT_SYMBOL' W='80' DATAFLD='@MKT_SYMBOL' CTRLTYPE='TEXT'
          CTRLMASK='T' ALIGN='LEFT'/>
          <COL NM='INDEX' CAPTION='INDEX' W='90' DATAFLD='@INDEX' CTRLTYPE='TEXT' ALIGN='RIGHT' CTRLMASK='N'/>
          <COL NM='CHANGE' CAPTION='CHANGE' W='60' DATAFLD='@CHANGE' CTRLTYPE='TEXT' CTRLMASK='N'
          ALIGN='RIGHT'/>
        </LIST>
        <SELECT NM='SELECT_MARKET' CAPTION='Select Market' H='20' W='150' DATA='${windowelement}' X='15' Y='20'
        LISTHEIGHT='120' LISTDATA='/MARKETS/MARKET' LISTDATAFLD='@MKT_SYMBOL' EDITABLE='Y'>
          <ACTIONRULES TRIGGER='ONCHANGE' DESC='Change'>
            <ACTIONRULE>
              <ACTREFRESH CONTROL='CHART'/>
            </ACTIONRULE>
          </ACTIONRULES>
        </SELECT>
        <LABEL NM='MKT_NAME' H='20' W='220' DATA='${SELECT_MARKET}' X='15' Y='50' DATAFLD='@MKT_NAME'
        CTRLFONTSTL='BoldCaption'/>
        <CHART NM='CHART' CAPTION='CHART' H='160' W='250' X='15' Y='100'>
          <AXIS TYPE='Y' SPAN='10' MIN='0'/>
          <PLOT NM='PLOT1' DATA='${SELECT_MARKET}/MOVEMT' DATAYFLD='@INDEX_CLIPPED'/>
        </CHART>
      </PANEL>
    </WINDOW>
  </WINDOWS>
  <IMAGEMAP/>
  <PROPERTIES/>
  <DATA>
    <INCLUDE SVRCMD='GET_MARKETS' SUBSCRIBE='MARKETS_POOL'/>
  </DATA>
</ALTIO>

```

## Markets Tutorial

The matching service function, datapool and datatypes file, altioapp.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--DOCTYPE ALTIO SYSTEM "altioapp30.dtd"-->
<ALTIO DTD='altioapp30.dtd' TYPE='APP'>
  <INIT>
    <PROBEINTERVAL VALUE='20'/>
    <SENDUPDATETIME VALUE='N'/>
  </INIT>
  <LOGGING>
    <APPLOGGER PATH='logs' LEVEL='DEBUG' XML='Y' FILE='log'/>
    <LOGGER PATH='logs' LEVEL='DEBUG' XML='Y' FILE='log' ID='DATAPOOL'/>
    <LOGGER PATH='logs' LEVEL='DEBUG' XML='Y' FILE='log' ID='SERVICE'/>
    <LOGGER PATH='logs' LEVEL='DEBUG' XML='Y' FILE='log' ID='CLIENT'/>
  </LOGGING>
  <SERVICES>
    <SERVICE NM='GET_MARKETS' MULTICALLS='Y'
    URL='http://{http.server.name}:{http.server.port}/{http.server.path}com.altio.altiodb.AltioDB' METHOD='AUTO'
    URLARGS='ACTION=GET&TARGET=/MARKETS&APPID=MAR' BLOCKARG='N' ACKSRC='NONE'>
      <ONSUCCESS MSGSRC='NONE'/>
      <ONFAILURE MSGSRC='NONE'/>
      <OUTPUT DOCTYPE='XML'/>
    </SERVICE>
    <SERVICE NM='GET_MARKETS_DP' MULTICALLS='Y'
    URL='http://{http.server.name}:{http.server.port}/{http.server.path}com.altio.altiodb.AltioDB' METHOD='AUTO'
    URLARGS='ACTION=GET&TARGET=/MARKETS&APPID=MAR' BLOCKARG='N' ACKSRC='NONE'>
      <ONSUCCESS MSGSRC='NONE'/>
      <ONFAILURE MSGSRC='NONE'/>
      <OUTPUT DOCTYPE='XML'/>
      <PARAM NM='TIMESTAMP' VALUE='{datapool.timestamp}'/>
    </SERVICE>
  </SERVICES>
  <DATAPOOLS>
    <DATAPOOL NM='MARKETS_POOL' MEMORYCACHESIZE='100' DISKCACHESIZE='0' TRANSIENT='Y'>
      <MASTER>
        <MASTERPOLLING POLLRATE='2' SERVICE='GET_MARKETS_DP'/>
      </MASTER>
    </DATAPOOL>
  </DATAPOOLS>
  <DATAKEYS>
    <DATAKEY NM='MARKET' PREFIX='MKT' KEY='@MKT_SYMBOL'/>
    <DATAKEY NM='MOVEMT' PREFIX='MOVEMT' KEY='@MOV_ID'/>
  </DATAKEYS>
  <EVENTS/>
</ALTIO>
```

## Document Information

KEYWORDS: DESIGNER, GUI, DESIGN, VIEWS, WINDOWS, GETTING STARTED.

**Integra SP – Altio**

Telephone: +44 (0) 20 8528 1045

Internet: [www.altio.com](http://www.altio.com)

**Copyright © 2010 Integra SP**

Copyright in this document is vested in Integra SP. The contents of the document (wholly or in part) must not be reproduced, distributed, used or disclosed without the prior written permission of Integra SP.

Integra recognizes the trademarks or registered trademarks of any third party product or company name referenced in this document at the time of its publication.