



AltioLive Developer Training

Version 5.4

2. The First Steps to Build your Own Application

Summary

This follows on from the introductory module by explaining how to start creating an application with AltioLive.

Integra SP

88 Wood Street London
EC2V 7RS
United Kingdom

www.altio.com

tel: +44 (0) 20 8528 1045

Contents

Introduction	3
The Design flow.....	3
How the data will be displayed.....	3
The ways the user need to alter the data	3
The requirements the users have for data accuracy	3
Build it!.....	3
Stock Prices Application: Presentation of the application we will create	5
Application description	5
Method	5
Building the Stock Prices Application using the Developer Tools.....	6
Creating a new project using AltioLive Studio	6
Creating a Data Source using the Altio DB Manager	9
Stocks XML: an introduction to XML.....	13
Configuring back end applications using the Application manager.....	14
Introduction to the Application Manager.....	14
Launching the Application Manager	14
Creating and setting a new HTTP Service Function	15
Testing the HTTP Service Function and creating Datakeys	16
Setting the Datakey properties	17
Testing the Application with the Datakeys	18
Building a View for the application using the Designer	19
Introduction to the Designer tool	19
Launching the Designer tool	20
Creating and Setting Font Styles for the application	21
Creating and setting a Window Style for the application	22
Displaying and Setting the default window properties	23
Adding controls to the main window.....	24
Saving your application	25
Displaying and Setting the Control properties.....	25
Running the view	26
Saving the View.....	26
Importing data into the Designer and associating data with windows and controls	27

Associating data with a view.....	27
Associating data with controls.....	29

Introduction

This session provides an understanding of the complete AltioLive development process. During the session, we will create a simple application and explore the three main Developer tools:

- the **Application Manager**,
- the **Designer**,
- the **Altio DB Manager**.

The Design flow

The application design strategy is the same as for any software. It is essential to scope the user requirements and expectations; here are some suggestions:

- Consider how the data will be displayed to the user,
- Consider in what ways the user will need to alter the data (for example: delete, add, update),
- Consider what requirements the users have for the accuracy of the data displayed to them.

How the data will be displayed

This can affect how the XML data is structured; so it's important to balance the view of the data with the efficiency of the most used service requests (the common service requests should be as efficient as possible). There is another session on XML and data that includes an overview of data structure.

The ways the user need to alter the data

This affects directly the service requests, datapools and datakeys you create. Often the user will require the basic functions of adding, updating and deleting data; however there may be additional requirements where special java servlets need to be written to enable integration with the Back end.

Any data changed in the Back end application, and then subsequently at the client, will need unique Datakey attributes defined. This is covered in the session on Service Requests.

The requirements the users have for data accuracy

Different 'types' of data can be updated to the client at different rates.

There is a general compromise to be made between the rate of polling for data and the number of simultaneous client sessions that can be supported. It can be advantageous to build several datapools with different poll rates, where only the critical data has a high poll rate, rather than having a single datapool where all the data is updated to the clients at a high rate.

Build it!

The RAD/JAD aspects of the AltioLive Developer Edition can entice us to go directly into prototyping an application. The initial definition and scope of the application must be completed before the **Application Manager** is used to create service requests, datakeys and datapool, and the **Designer** is used to build the look and feel of the application.

When using the **Designer** to create windows and controls consider how many windows the user will require open at any one time and be aware that it is easy to make a small browser window cluttered.

Of course, the whole process is much easier where the task is to emulate an existing application. The appearance is already defined and the facilities required by the users are known.

Stock Prices Application: Presentation of the application we will create

This Altio application is based on a database of stock prices. The complete application will allow a user to view and manipulate data; add entries, delete entries and alter the stocks view by applying filter parameters. In this session we will build the application to a basic level so that a user may see a complete list of stocks from the database. Extra functionality will be added in further sessions.

The training uses the Altio Prototyping Kit. This allows us to use static XML files with **Altio DB Manager**, an XML database that can be used to simulate a Back end application. With the prototyping kit we can also simulate live updates arriving at the client from other users.

Application description

The application will consist of two windows:

- The **Main** window will contain a summary list of all the stocks. There will be an option to filter the list to show a more specific selection of stocks. Buttons will allow the user to add a new stock or delete an existing stock.
- A **New Stocks** window will be used for data entry to add new stocks. The window will allow entry of required data, via a range of controls, to allow a new stock to be added to the stocks database.

Method

The application is created in three steps using the main components of the Altio Developer Edition:

1. Creating a new project using AltioLive Studio.
2. Defining the service request interfaces to the back-end applications by using the **Application Manager**.
3. Create the application views, which comprise windows and controls, through use of the **Designer**.

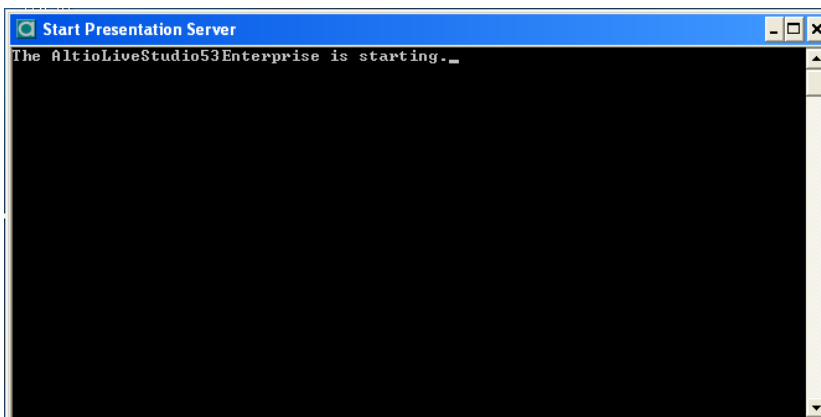
Building the Stock Prices Application using the Developer Tools

Creating a new project using AltioLive Studio

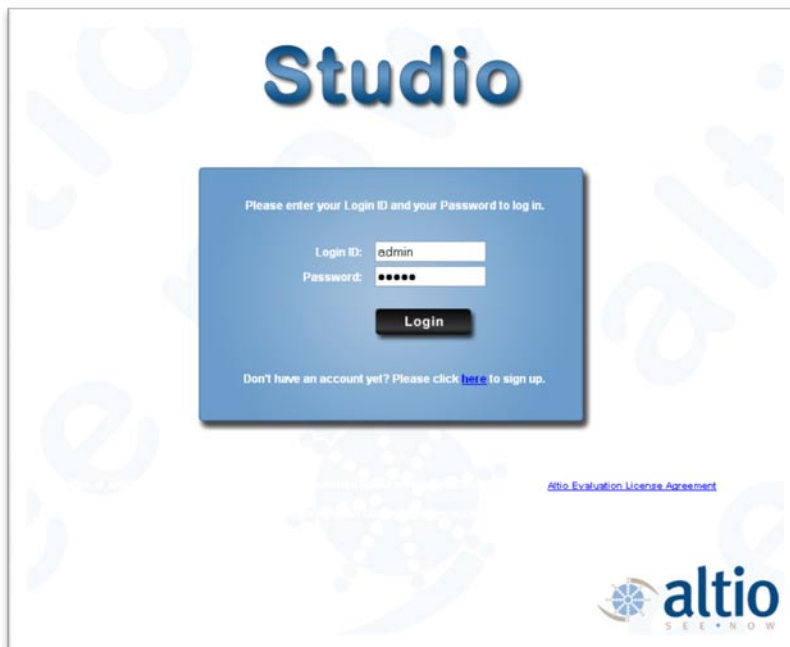
Please note: we are running the server and client parts of the software on your local machine.


1. From the **Start** menu select: **All Programs | AltioLive Studio 5.4 Professional | Start Presentation Server.**

Tomcat will start in a window:

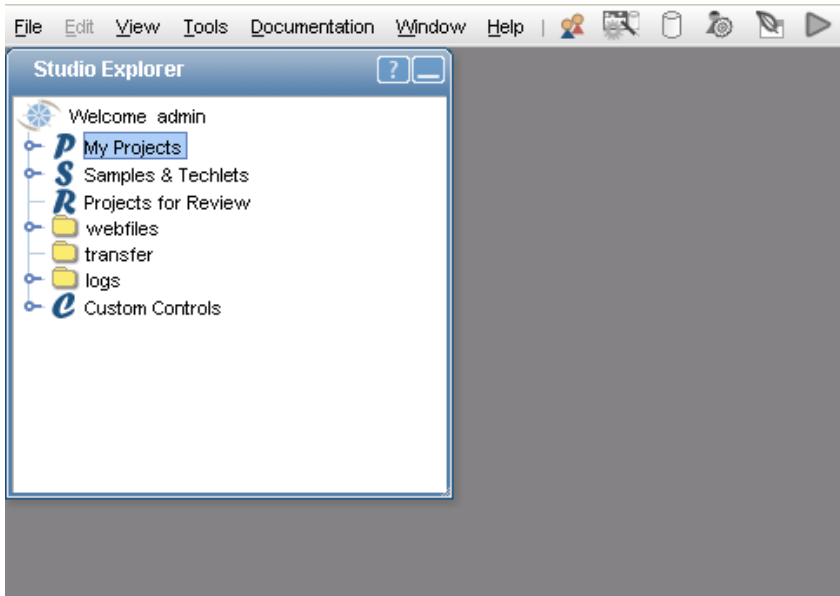


2. From the **Start** menu select: **All Programs | AltioLive Studio 5.4 Professional | AltioLive Studio 5.4.** This will launch your default browser which will display the Login page.

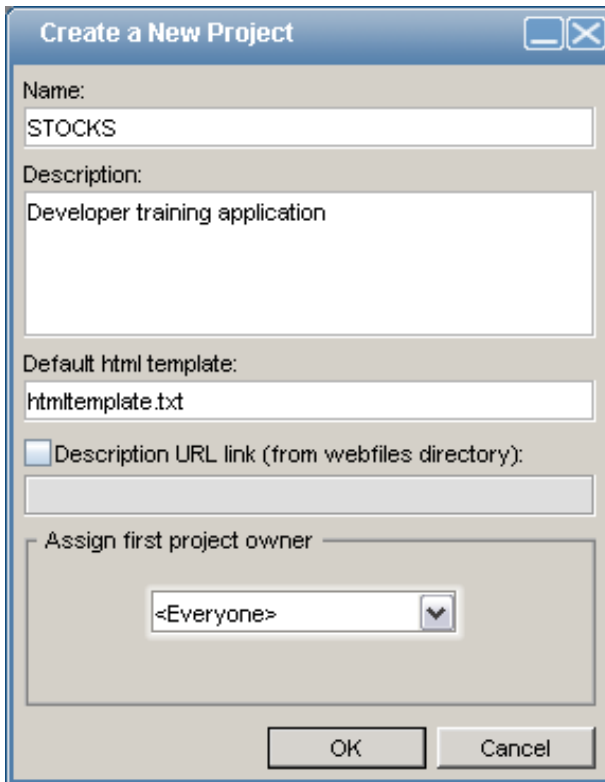


3. Log in by entering your Login ID and Password. By default both **Login ID** and **Password** are **admin**.
4. Click on the Login button .

Once logged in, you are presented with the Altio Developer Studio:

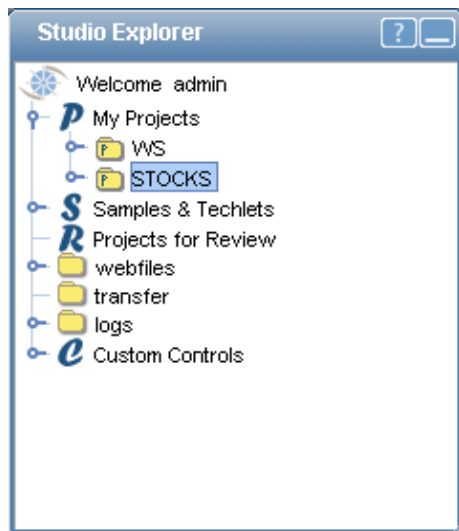


5. From the Studio menu, do one of the following:
 - Select **File | New Project...**
 - Right-click on the **My Projects** node and select the **New Project...** option in the context menu.
6. Enter the following details into the **Create a New Project** window:



7. Click on **OK**.

- From the Studio Explorer window, expand the **My Projects** node and you should see the new **STOCKS** project listed:



Creating a Data Source using the Altio DB Manager

AltioLive Studio includes the **Altio DB** servlet, a simple XML database that is useful for prototyping. For this practical exercise we are using data from static XML files. Before we configure service requests in the **Application Manager**, we will configure the **Altio DB Manager** to specify where the XML data files are located.

A prepared file containing prototype data is located with the training installation, or from your trainer. It is called **stocks.xml**. You can also download it from the Altio website:

<http://www.altio.com/AltioLive/Online-Documentation.aspx>.

altio
SEE NOW

About AltioLive Developer Centre News Customers P

Downloads Demos Online Documentation

Altio Home > AltioLive > Online Documentation

Online Documentation

Welcome to the AltioLive online help. AltioLive offers extensive documentation. This page provides the details of the documentation and how best to make use of it.

Getting Started

The first few sections of the [Online Help](#) and the [Getting Started Guide](#) provide a quick introduction to using and deploying AltioLive. It is then recommended that you follow the [Markets Tutorial](#) which walks you through building an application using AltioLive.

Other support documentation:

1. [Admin Tools](#)
2. [Application Manager](#)
3. [Web Services](#)

Blogs, Forums and Developer Center

[Developer Blog](#), [Forums](#), and [Developer Center](#) provide additional sources of information for using AltioLive.

Training Guides

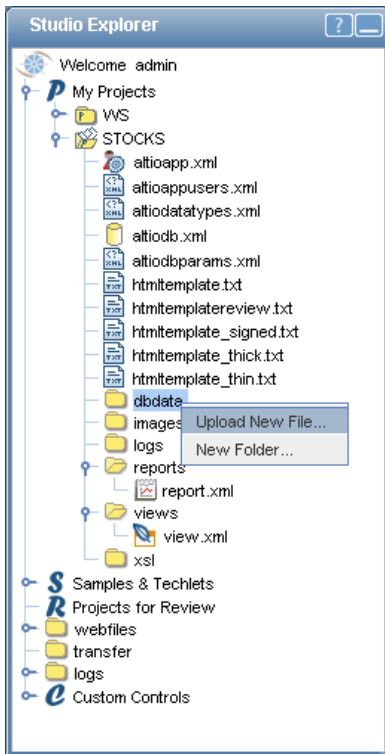
Training guides are provided online to assist with getting started and advanced topics. Should you wish to have consultant based training please contact our sales team on sales@altio.com.

1. [Introduction to AltioLive Training](#)
2. [First Steps](#)
3. [Referencing Data](#)
4. [Events and Actions](#)
5. [Introduction to Datapools](#)
6. [JDBC and SOAP Service Functions](#)
7. [Prototyping Wizard](#)
8. [Debugging Applications](#)
9. [Further Exercises](#)

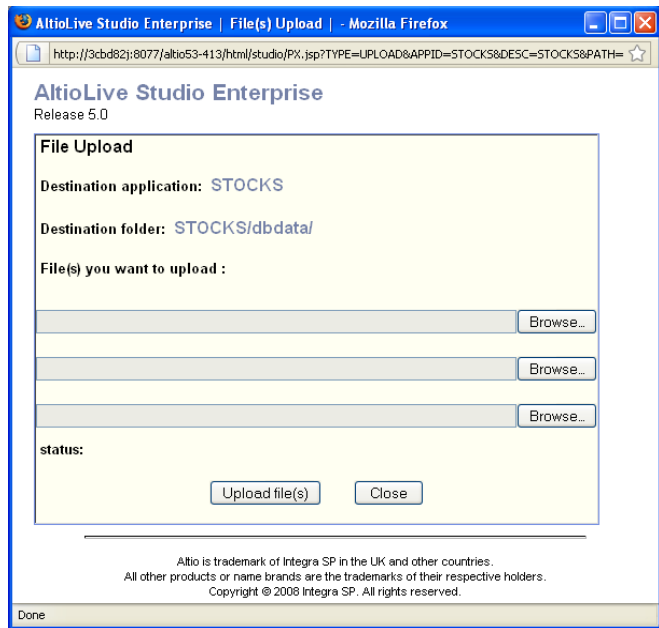
Support files for the training are [stocks.xml](#) and [stocks_updates.xml](#)

1. From the **Studio Explorer** window, expand the **STOCKS** node.
2. Right-click on the **dbdata** node under **STOCKS** and select **Upload new File...** from the context menu. The **File(s) Upload** window opens.

1)

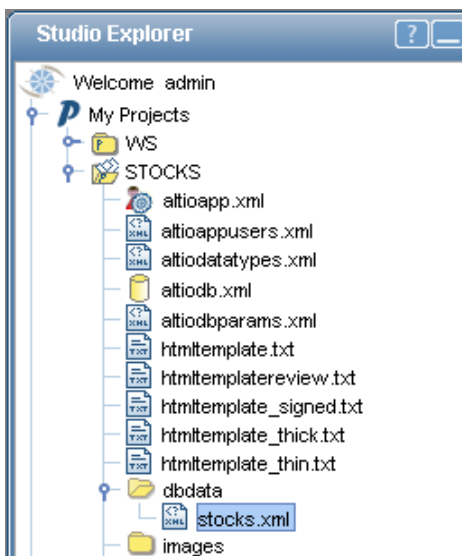


2)




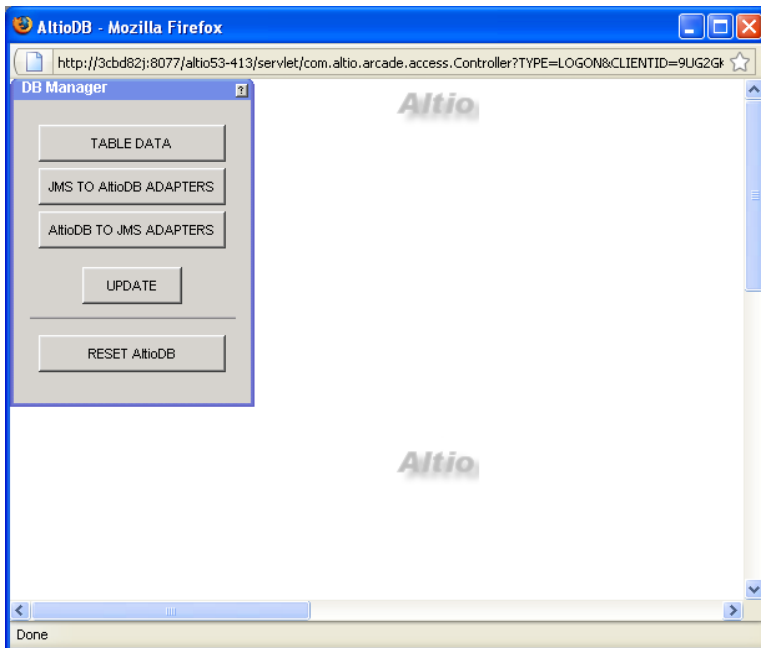
3. Click on the **Browse** button and browse for the **stocks.xml** file.
4. Click **Upload file(s)** to copy the file to the project's **dbdata** directory.
5. Click on the **Close** button to close the dialog.

You should now see the file listed in the **Studio Explorer** window, under your **dbdata** directory:

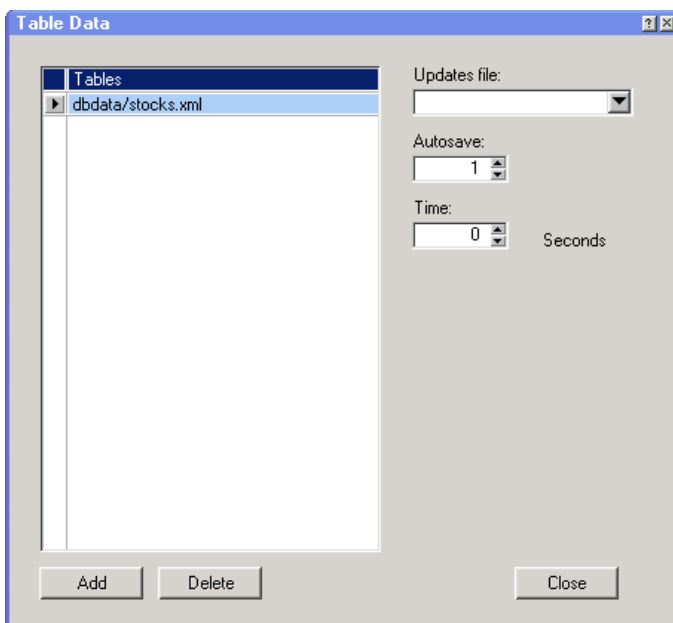


Now we can use **AltioDB Manager** to integrate this file into the **STOCKS** application.

6. From the AltioLive Studio, select the **STOCKS** application folder.
7. Click on the **AltioDB Manager** toolbar icon . This initial screen is displayed:



8. Click on the **TABLE DATA** button.
9. Click on the **Add** button.
10. In the **Tables** panel, an empty row should now be displayed. Click in this row to display a drop-down menu.
11. From the drop-down menu, select the **stocks.xml** file.



12. Click on the **Close** button.
13. From the **AltioDB Manager** window, click on the **UPDATE** button.

14. An **Information** window will display, click on the **Yes** button.
15. A **Results** window shows that the server has been updated. Click on the **Close** button.
16. Click on the **Reset AltioDB** button to apply the changes to the configuration.
17. An **Information** window displays, click on **Yes**.

The data file **stocks.XML** has now been integrated into the application.

Stocks XML: an introduction to XML

XML is used for configuring Altiolive to the look and functionality desired by the user. More importantly, Altiolive uses XML for exchanging data between the Client and the Synchronization Engine.

The stocks XML file is a database of stocks information. Information held for each stock includes the stock name and identifier (**EPIC**) and market values such as high and low prices.

Here is an extract from the **stocks.XML** file:

```
<STOCKS>
<STOCK NAME="Abbey National" MARKET="LONDON" LOW="1170" HIGH="1192" STOCK_ID="S1">
  <HISTORY DATE="20020531" OPEN="70.7" CLOSE="77.7" HIGH="77.7" LOW="69.7" VOL="3122000.0"/>
  <HISTORY DATE="20020601" OPEN="77.7" CLOSE="78.8" HIGH="91.2" LOW="75.0" VOL="3147091.4"/>
</STOCK>
<STOCK NAME="Alliance & Leic" ...
```

Please note: a stock may hold history price data, which includes trading volumes.

When discussing XML data, we use the following conventions:

CONCEPT	EXPLANATION	EXAMPLES
Data Element	An XML element containing data in the form of attributes and/or child elements.	<pre><STOCKS> <STOCK ... > <HISTORY ... /> <HISTORY ... /> </STOCK> </STOCKS></pre>
Data Field or Attribute	An attribute of the Data Element that contains data.	<pre>STOCK_ID="S1" NAME="Abbey National" SECTOR="Banks"</pre>
Element name	The name of the element in this example is STOCK. Referenced as name() .	<pre><STOCK ... /></pre>
Element text	The text of the element in this example is Test . Referenced as text()	<pre><STOCK>Test</STOCK></pre>

As shown on first row, the **STOCKS** element does not contain any data fields, but it has a set of child data elements (**STOCK** and **HISTORY** elements). The **HISTORY** element does not have any child data elements but it has a set of data fields.

As shown on the second row, a **STOCK** element has also a set of data fields (**STOCK_ID**, **NAME** and **SECTOR**).

Configuring back end applications using the Application manager

Introduction to the Application Manager

The **Application Manager** tool is used to configure the integration of AltioLive with a back end application. The **Application Manager** is used to define and set:


- **Service Requests:** which send and/or request data from the back end application,
- **Datapools:** used to distribute updates and live data to multiple clients,
- **Datakeys:** the definition of unique key fields in the data,
- **Logging:** for tracing and debugging.

The output of the **Application Manager** is stored in the **altioapp.xml** file. For clustered configurations, the **Application Manager** tool maintains this file and distributes it to all the Presentation Servers in the cluster.

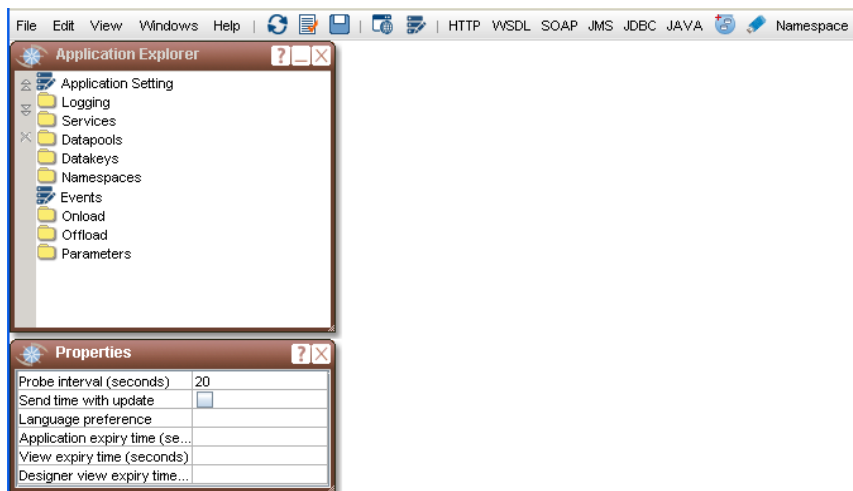
This practical exercise introduces the **Application Manager**. We will add a simple service request to retrieve the stocks data so that it is available when we need it in the **Designer**.

Datapools are used where asynchronous updates to the client data are required. In this example we are using synchronous client/server communication, so we will leave definition of Datapools until a later stage.

Launching the Application Manager


1. From the **Studio Explorer** window, expand the **My projects** node to display the **STOCKS** project.
2. Do one of the following:
 - Right-click on the **STOCKS** project and select **Application Manager** in the context menu.
 - Select the **STOCKS** project and click on the **Application Manager** icon .
 - Expand the **STOCKS** node and double-click the **altioapp.xml** item.

This opens the **Application Manager** in a new browser window:



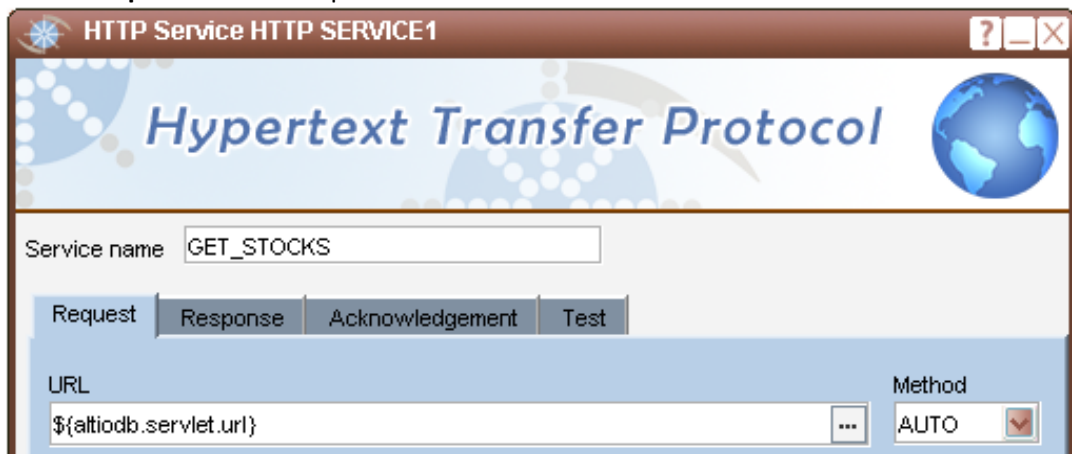
Creating and setting a new HTTP Service Function

1. Do one of the following:

- Click on the **New HTTP service** button  in the toolbar.
- Right-click on the **Services** node in the **Application Explorer** window and select **New HTTP Service** from the context menu.

A new **HTTP Service** window will open with four tabs: **Request**, **Response**, **Acknowledgement** and **Test**.

2. On the **Request** tab set the parameters as shown below:



- In the **Service name** field, enter **GET_STOCKS**.
- In the **URL** field, enter **\${altiodb.servlet.url}**. It is a parameter that is dynamically replaced with the URL of the **Altio DB** servlet, which is in reality: **http://localhost:8077/altio54/servlet/com.altio.altiodb.AltioDB**. This and other such parameters make it easy to transfer an application from one machine to another. These parameters can be configured in either the **Presentation Server Administration** Tool for server-wide access (such as this one), or within the **Application Manager** under the parameters node for application-specific use.

3. Still on the **Request** tab, add the following parameter mappings:

Parameter mapping	
Name	Value
APPID	\${application.id}
ACTION	GET
TARGET	/STOCKS

Buttons: Add, Delete

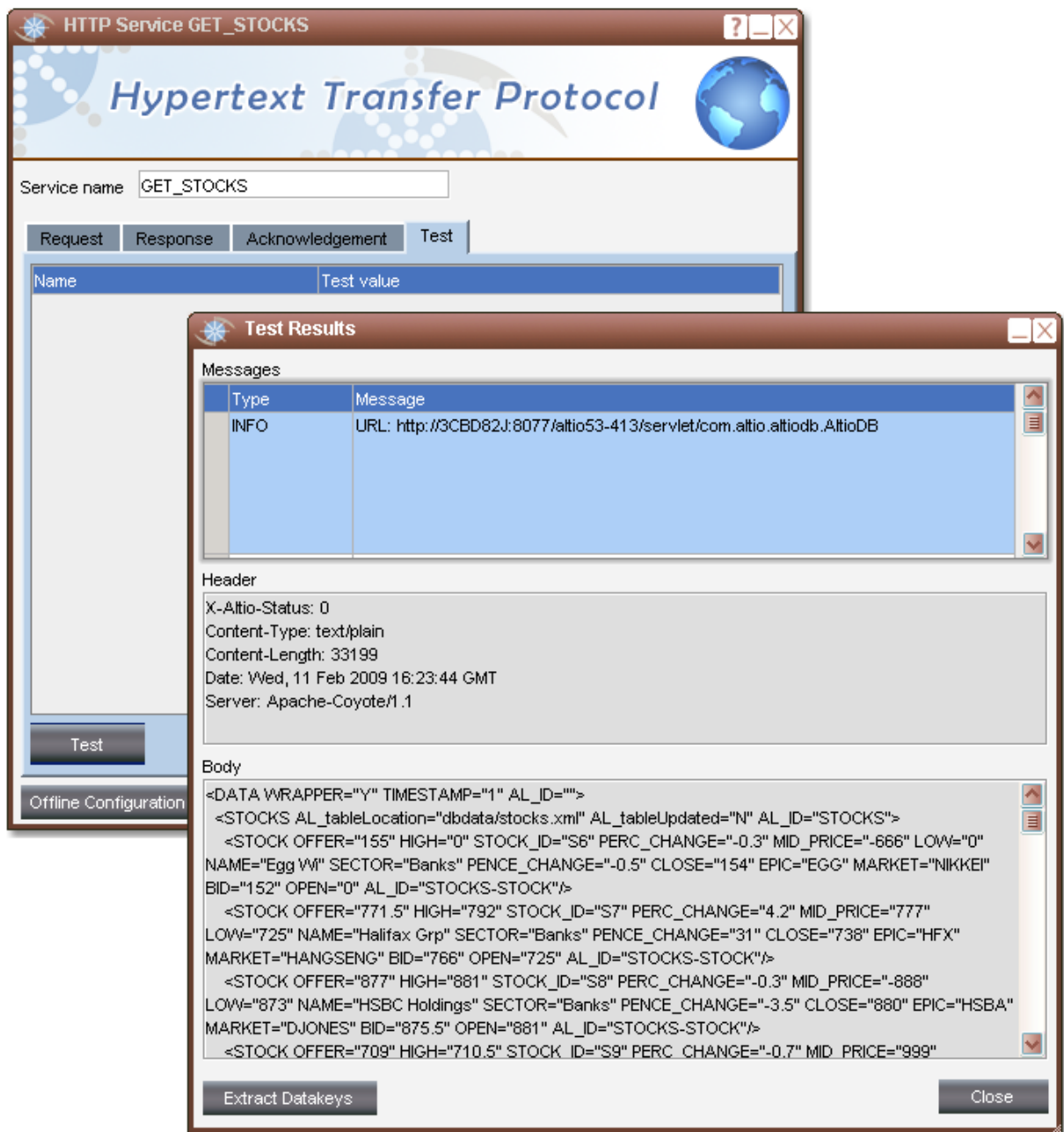
- **APPID** with the value **\${application.id}** is a parameter that is dynamically replaced with the APPID from the session.
- **ACTION** with the value **GET** means that this service call gets data from the server (as opposed to updating or deleting). It is a fix string required by **Altio DB Manager**.

- **TARGET** with the value **/STOCKS** indicates the root of required data. It means that all of the data under the **STOCKS** data element will be returned. Other back-end applications will require different parameters, since these ones are specific to **Altio DB**.

4. The default settings in the **Response** and **Acknowledgement** tabs are fine for this simple service function.

Testing the HTTP Service Function and creating Datakeys

1. Go to the **Test** tab and click **Test** to see what the service function will return. A window will indicate that the data is loading. Shortly afterwards a new window will open with a panel displaying a summary of the service function details and a panel below displaying the XML data returned.



Please note: the first line of **INFO** shows the URL with the **\${}** syntax resolved and the second line shows the resolved URL arguments (with http encoding). You could have simply typed the resolved URL arguments into the field on the **Request** tab; however the method we just used is much clearer.


Please note: The **WARNING** entries in **Messages** are displayed because no **datakeys** have been defined for the elements in the data returned. **Datakeys** are used to uniquely identify each data element within AltioLive. A datakey is defined for an element name and optionally a namespace. The datakey is then used by AltioLive to create the special attribute **AL_ID**. In the data above you will see that each element has been given a default datakey which is not unique.

2. Click the **Extract Datakeys** button to create datakey definitions for the **STOCKS**, **STOCK** and **HISTORY** elements.
3. When the **Extract Datakeys** window displays, click **OK**.



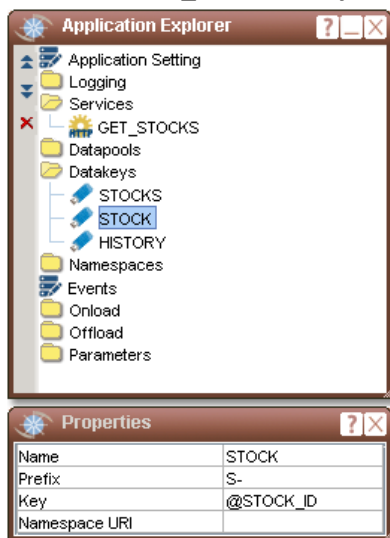
4. Close the **Test Results** window.

Setting the Datakey properties

1. From the **Application Explorer** window, expand the **Datakeys** node.
2. Do one of the following to display the **Properties** window:
 - Click on the **View Properties** icon  in the toolbar.
 - Click on **View | Properties** menu item.

Please note: The **STOCKS** datakey can be ignored, because there are not multiple instances of it.

3. Select the **STOCK** datakey and from the **Properties** window:
 - Enter **S-** in the **Prefix** field.
 - Enter **@STOCK_ID** in the **Key** field.



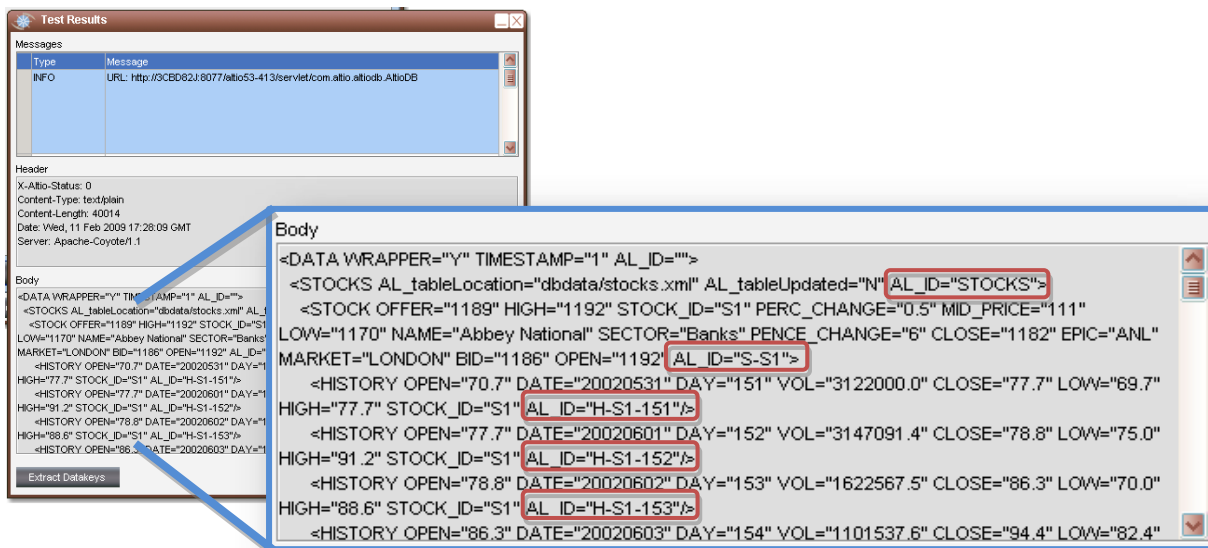
4. Select the **HISTORY** datakey and from the **Properties** window:


- Enter **H-** in the **Prefix** field.
- Enter **concat(@STOCK_ID,'-',@DAY)** in the **Key** field.

Testing the Application with the Datakeys

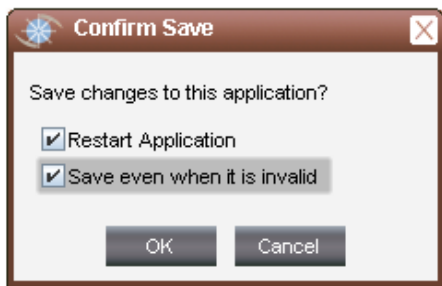
1. Return to the **GET_STOCKS** service window: in the **Application Explorer** window, expand the **Services** folder and double click the **GET_STOCKS** Service.
2. Click the **Test** tab and click the **Test** button again.

Please note: You should see that each element in the results now has a unique **AL_ID** attribute and there should be no warnings generated.



3. Close the **Results** window and click **Save** . Click **OK** in the new window to confirm that the configuration should be updated and the application restarted. A Results window will report that the server has been updated. Provided the server was updated successfully, close the web browser running the Application Manager.

Please note: If your application contains errors, it will not be saved unless you tick the **Save even when it is invalid** option from the **Confirm Save** window:




Building a View for the application using the Designer

Introduction to the Designer tool

The **Designer** provides a suite of tools to allow us to create one or more views of our AltioLive application. A view is the interface an end-user will see. With an application view, you can vary the styles and amount of data displayed, depending on who the view is going to be used by. A common scenario is to have a user view and an administrator view; the user view allows a subset of the functions available to the administrator.





The **Designer** uses the same **Explorer** window and **Properties** window as we saw in the **Application Manager** tool.

To show the **Properties** window, you can:




- Double-click an item in the **Explorer** window.
- Select an item in the **Explorer** window and select **View | Properties** from the main menu.
- Select an item in the **Explorer** window and click the **view properties** icon 

Online help can be accessed by clicking the  symbol on any window.

The following buttons are available in the main toolbar:

-  Create a new window
-  Open an existing view
-  Validate current view
-  Save current view

The General settings for the current view are shown on the **Properties** window when **View Settings** is selected on the **View Explorer** window.

New objects can be added by right-clicking on the **View Explorer** window or from the **Edit** menu. Objects are deleted by pressing the **Delete** key, clicking the  button or from the **Edit** menu. Objects can be reordered with the up  and down  buttons.


In the process of creating the application, we will look at the Explorer tree objects. Definitions and more help about each of the object types are available within the online help.

Please note: You can access the Online Help by clicking on the **Help | Help topics** menu item or by following this link:

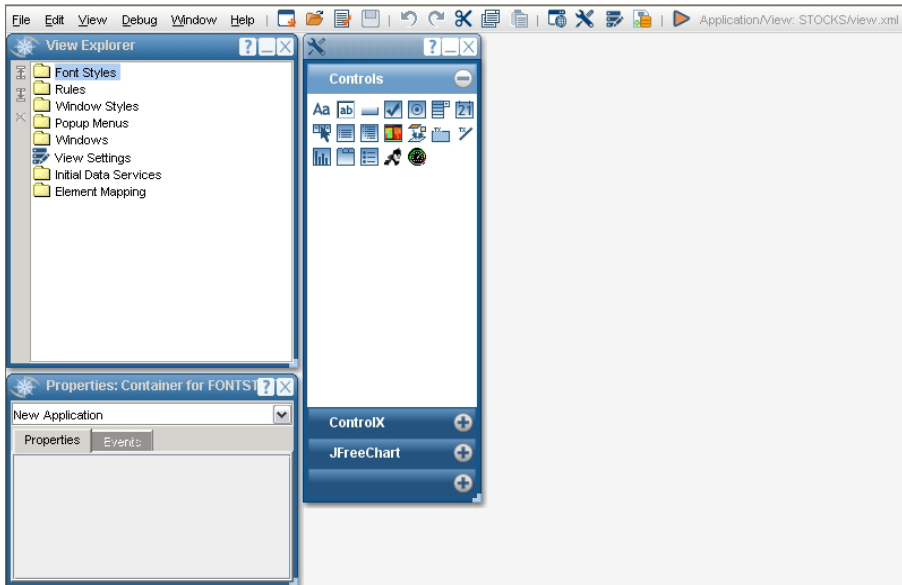
http://www.altio.com/download/documentation/Online_Help/AltioLive_Online_Help.htm

In this practical exercise we will use the **Designer** to create a view with a simple window. We will improve the appearance of the window by configuring window and font styles and then we will add controls to the window and set their properties.

Launching the Designer tool

1. From AltioLive Studio, expand the **STOCKS** project in the **Studio Explorer** window.
2. From the **Studio Explorer** window, do one of the following:
 - Right-click on the **STOCKS** project and select **Designer** in the context menu.
 - Select the **STOCKS** project and click on the **Designer** toolbar icon .
 - Expand the **views** node and double-click the **view.xml** file.

The **Designer** is started in a new browser window. The initial **Designer** screen is shown:

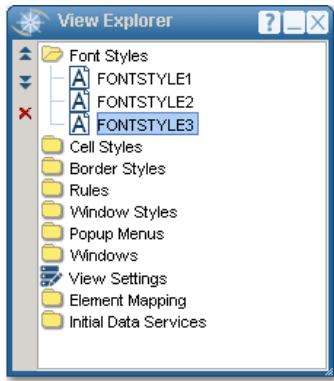



3. Ensure the application has been opened with the **view.xml** (shown on the menu bar).




Creating and Setting Font Styles for the application

- From the **View Explorer** window, create a new font style by doing one of the following:
 - Right-click the **Font Styles** folder and select **New Font Style** in the context menu.
 - Select the **Font Styles** folder and click on **Edit | New Font Style** menu item.
- Repeat this step until you have three different Font Styles:



- From the **View Explorer** window, select the first font style called **FONTSTYLE1** and do one of the following to display its properties:
 - Right-click on **FONTSTYLE1** and select **Properties** in the context menu.
 - Click the **Show Properties Window** button .
 - Click on the **View | Properties** menu item.

Please note:

- Properties are grouped into **General, Position/Size, Behavior and Printing**. Expand the groups by clicking the  buttons. Some attributes contain drop-down lists or check boxes so that you can select valid options.*
- Throughout this document, the following table is used to show changes to the properties of objects. We will only show properties that need to be changed from their default values.*

- Set the **FONTSTYLE1** properties as follows:

▼ General	
Name	BOLD_LABEL
Size	20
Color	0x000000
Bold	Y

5. Display the **FONTSTYLE2** properties and set them as follows:

▼ General	
Name	CONTROL
Size	10
Color	0x0033FF
Bold	N


6. Display the **FONTSTYLE3** properties and set them as follows:

▼ General	
Name	TITLEBAR
Size	14
Color	0xE0E0E0
Bold	Y

Please note: colors can be selected from the drop-down color picker or typed into the edit box. Also notice that the drop-down list at the top of the **Properties** Window contains all the ‘siblings’ of the selected object, so it will currently contain all three Font Styles. Selecting from the list changes the Properties shown and the selection in the **View Explorer** Window.


Creating and setting a Window Style for the application

Creating a Window Style

- From the **View Explorer** window, create a new window style by doing one of the following:
 - Right-click the **Window Styles** folder and select **New Window Style** in the context menu.
 - Select the **Window Styles** folder and click on **Edit | New Window Style** menu item.
- From the **View Explorer** window, select the new window style called **WINDOWSTYLE1** and do one of the following to display its properties:
 - Right-click on **WINDOWSTYLE1** and select **Properties** in the context menu.
 - Click the **Show Properties Window** button ,
 - Click on the **View | Properties** menu item.
- Set the window style properties as follows:




▼ General	
Name	STYLE
Default control font style	CONTROL
Titlebar font style	TITLEBAR

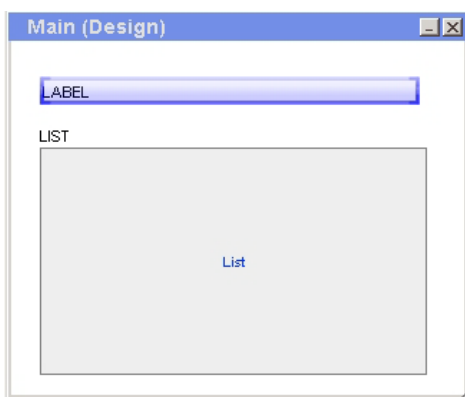
Displaying and Setting the default window properties

- From the **View Explorer** window, expand the **Windows** folder by double-clicking on it.
- Open the window by expanding the **Windows** node and double-clicking on **WINDOW** (the default window's name).
- Show the properties of the window by doing one of the following:
 - Click the **Show Properties Window** button ,
 - From the **View Explorer** window, right-click on **WINDOW** and select **Properties** in the context menu.
 - From the **View Explorer** window, select **WINDOW** and click on the **View | Properties** menu item.
- Now change the Window properties as follows:

Property	Value	Comments
▼ General		
Name	MAIN_WIN	
Caption	Main	<i>Notice how the name displayed on the window changes</i>
Can have multiple instance	N	<i>We only ever want one of these</i>
Window style	STYLE	<i>The Window Style we defined earlier</i>

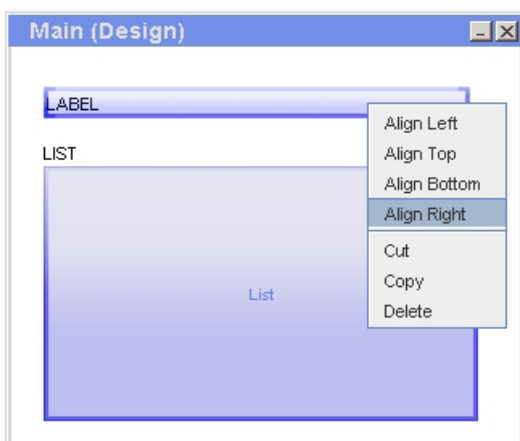
Adding controls to the main window

1. From the **View Explorer** window, double-click on the **Windows** folder.
2. Double-click on **MAIN_WIN**. A preview of the **MAIN_WIN** window displays.
3. If the **Control Palette** is not displayed, click on the **Control Palette** icon .
4. Drag a list  and a label control , from the **Control Palette** onto the window.
5. Reposition controls so that the window looks like the window shown below. To reposition control do one of the following:
 - Select a control and use the cursor keys of your keyboard,
 - Select a control and enter the new size and position into the **Properties window**, under the **Position/Size** properties.




Please note:

- Exact control coordinate values, widths and heights are not important in our exercise; just ensure that the windows and controls are clearly visible and that controls are in a similar position on the window to that shown above.
- Multiple controls may be selected by clicking whilst holding **Ctrl** key or by 'rubber banding' on the window (keep the mouse pressed and drag the selection box to enclose the controls to be selected).
- The selected control is always shown with a blue border.
- Right clicking on multiple selected controls will display a menu to help with alignment of controls:




Saving your application

1. This is a good point at which to save our view by clicking on the **Save** icon .

Displaying and Setting the Control properties

Displaying the Control properties

Show the properties of a control by doing one of the following from the **View Explorer** window:

- Select a control and click the **Show Properties Window** button ,
- Right-click on a control and select **Properties** in the context menu.
- Select a control and click on the **View | Properties** menu item.

Setting the Label control properties

Set the Label control properties as follows:


	Property	Value	Comments
▼	General		
	Name	WIN_TITLE	
	Caption	Summary of Stock	
▼	Appearance		
	Caption font style	BOLD_LABEL	<i>A Font Style we defined earlier</i>

Setting the List control properties


Set the List control properties as follows:

	Property	Value	Comments
▼	General		
	Name	STOCK_LIST	
	Caption	Stock List	
▼	Position/Size		
	Width	540	
	Stretch to fill window	V	As you resize the window, the list will stretch to fill it vertically
	Bottom Margin	5	Leave a 5 pixel border below the list

Running the view

1. Click on the run Button  to launch the application within the **Designer**. The window should now look like this:



2. Click on the Stop Button  to go back to the Designer interface.

Saving the View

1. Save the View by clicking on the **Save** icon .

Importing data into the Designer and associating data with windows and controls

We have created a service function in the **Application Manager** to make the data available to the **Designer**. There are a few more steps involved before the data is accessible:

- Set the initial data source available to the view (the Service Request),
- Get a graphical representation of the datatypes available in the initial data,
- Link selected datatypes to a control (our List Control).

Associating data with a view

This practical exercise will allow us to access the XML data file via a service request. We will add data to a List Control and look at the formatting of the List Control columns.


Earlier in the session we created a back-end application using the **AltioDB Manager**. This accesses a static XML data file. Then, we created a service request (**GET_STOCKS**) in the **Application Manager**, to provide data from the back end application.

To complete the link between the data and our View we need to add a call to the Service Request from this View. This can be done in the **Designer**.

Creating and setting the Initial Data Service

1. From the **Designer**, add a **New Initial Data** by doing one of the following:
 - In the **View Explorer** window, right-click on **Initial Data Services** folder and select **New Initial Data** in the context menu.
 - In the **View Explorer** window, click on the **Initial Data Services** folder and click on **Edit | New Initial Data** menu item.



A new **INCLUDE** node appears.

2. Display the properties of the Initial data by doing one of the following:
 - From the **View Explorer** window, select the **INCLUDE** node and click the **Show Properties Window** button .
 - From the **View Explorer** window, right-click on the **INCLUDE** node and select **Properties** in the context menu.
 - From the **View Explorer** window, select the **INCLUDE** node and click on the **View | Properties** menu item.
3. In the **Properties Window**, expand the **General** Properties and set as follows:

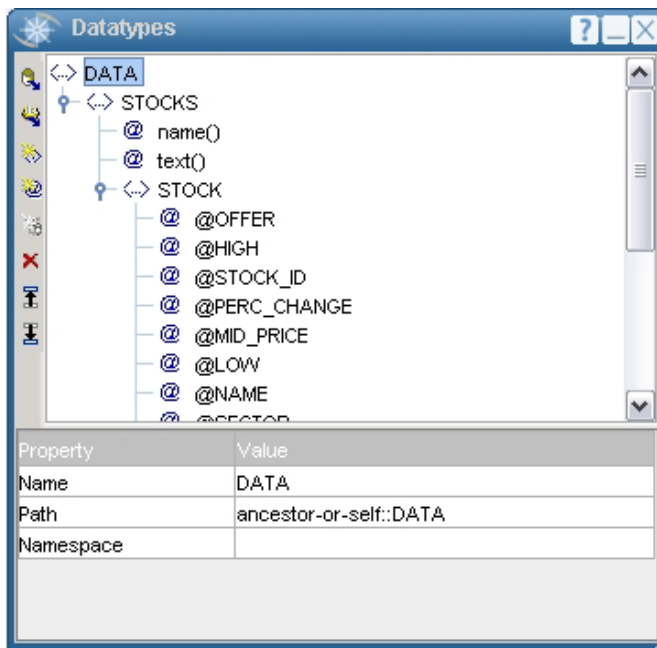
Property	Value	Comments
▼ General		
Server Command	GET_STOCKS	<i>The Service Request we created earlier</i>

When you make the selection, the Service Request is called by the **Designer** and the XML data is made available.

Setting the Datatypes properties

1. Click the **Show Datatypes** icon  to open the **Datatypes** window.
2. Click on the **Import from Service...** icon  located on the left side of the **Datatypes** window. The **Import datatype information** window is then displayed.
3. From the **Service Function** drop-down menu select **GET_STOCKS** and click on the **Import** button.
4. The **Datatypes** window should now display the **STOCKS** element under the **DATA** node.
5. Expand the **STOCKS** and **STOCK** nodes to display the available attributes.

You should also see **HISTORY** as a child of the **STOCK** node:

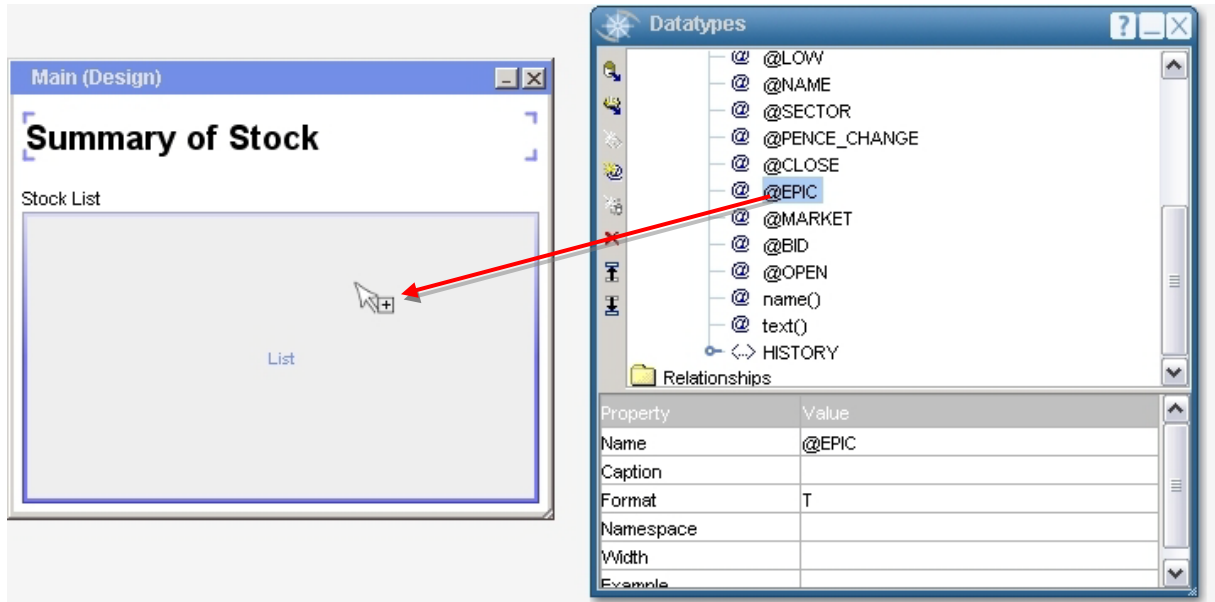


Please note: the data elements are shown as hierarchical, in which case the parent/child relationships between the datatypes is explicitly defined. In cases where the data is not hierarchical (often the case when data is derived from a relational database), then parent/child or master/detail relationships may be defined manually from the **Relationships** node. These are only used for automatically creating XPath statements for master / detail related controls (implemented via shift-dragging between controls).

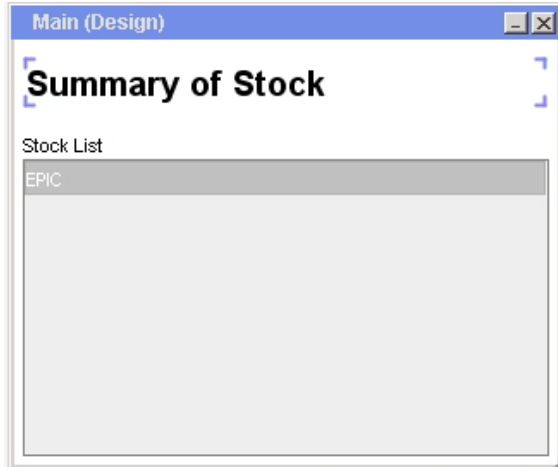
Associating data with controls

Now we will add the data attributes to the list control in our view.

1. Expand the **STOCK** node from the **Datatypes** window.
2. Drag the attribute **@EPIC** onto the **Stocks List** on your **Main** window.

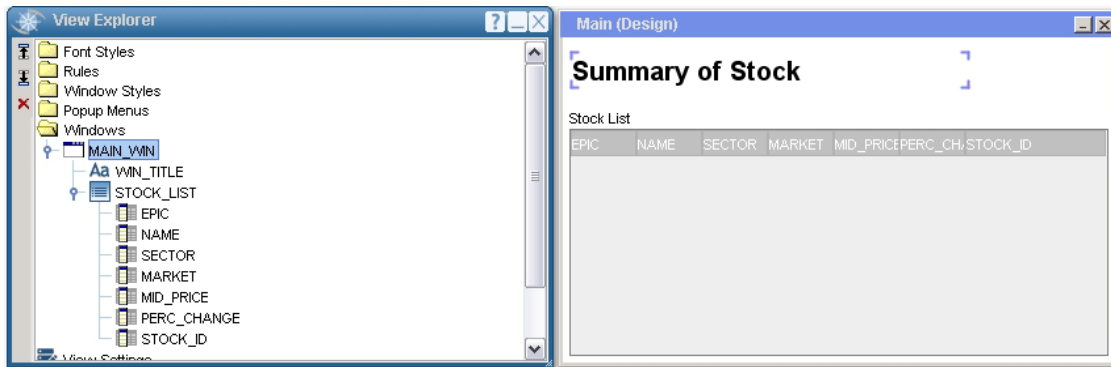


You will see that the column-heading **EPIC** appears in the list control:



3. In the same way, drag and drop the attributes **@NAME**, **@SECTOR**, **@MARKET**, **@MID_PRICE**, **@PERC_CHANGE**, and **@STOCK_ID** onto the list. The names of the columns appear on the list as the attributes are dragged across.
4. Resize the **Main** window so that it displays all the columns you have added to the **Stock List**.

Look at the **View Explorer** Window and expand the **STOCK_LIST**. Notice that the columns have now been added to the **STOCKS_LIST** in the tree:



- From the View Explorer window, click on the **EPIC** column and you will see that the **Properties** window shows the properties of the column.

Please note: The drop-down list at the top of the **Properties** Window contains a list of all the columns contained in the List.

- From the **Properties** window, expand the **General** properties and change the column width as follows:

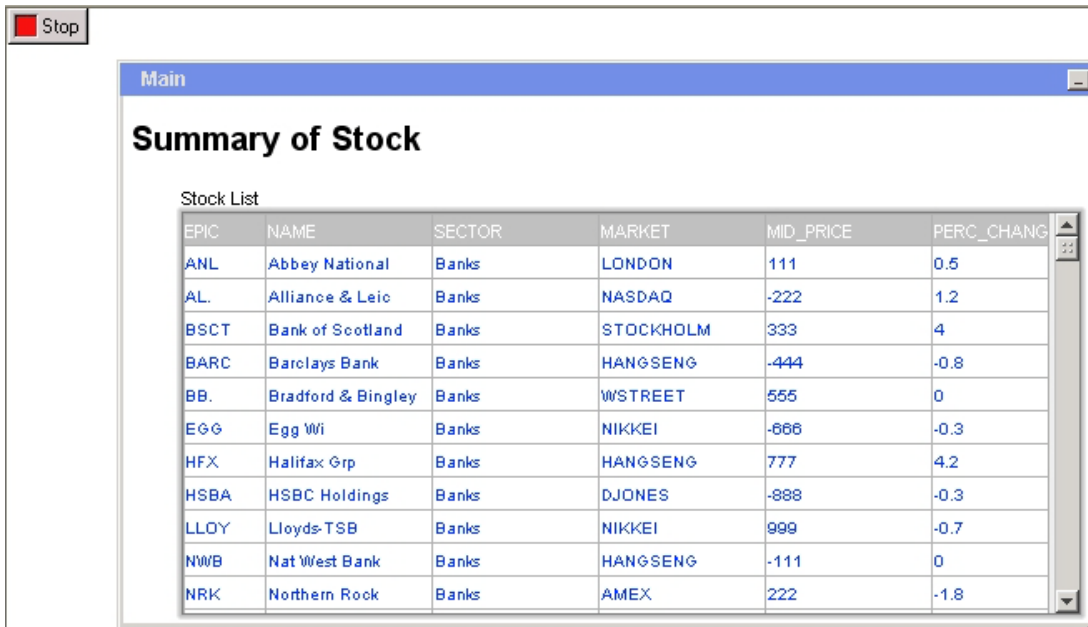
Column	Value
EPIC	50
NAME	100
SECTOR	100
MARKET	100
MID_PRICE	100
PERC_CHANGE	70
STOCK_ID	0

Please note: We need the **STOCK_ID** attribute present to uniquely identify each stock, but it is not necessarily something that we want to display.

7. Click the **Run** button .

We can now view the application with the data presented as the **Designer** is now in Test mode.

You will see that the design windows have gone and the application window is displayed, as it would be if the application were being run normally. Notice that the data has been loaded and is displayed in the list.




8. To return to **Designer** mode, click the **Stop** button in the top left corner .

9. Save the View by clicking on the **Save** icon .

10. Close the **Designer**.

That completes the basic practical sessions to create an application. Later, we will investigate some of the more advanced features.

For help on building an application and the different tools refer also to:

- The general online help available from the **Help** menu when AltioLive is running,
- PDF software manuals also available from the **Documentation** menu in the AltioLive Studio.
- For more specific contextual online help, click the  button located in the top right corner of each window in AltioLive.

Next Step: Referencing data in AltioLive applications

AltioLive Developer Training

Document Information

KEYWORDS: INTRODUCTORY, BUILDING APPLICATION, STARTING.

Integra SP – Altio

Telephone: +44 (0) 20 8528 1045

Internet: 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.