

# How to

## Save and Load Filters

**Tools:** AltioLive Studio  
AltioLive Application Manager  
AltioLive Designer

**Skill Level:** Medium

**Summary:** This document explains how to save and load filters applied to an AltioLive control using JDBC Service Functions.

### Integra SP

88 Wood Street London  
EC2V 7RS  
United Kingdom

[www.altio.com](http://www.altio.com)  
tel: +44 (0) 20 8528 1045

## Contents

Introduction .....	3
Creating a simple application that allows users to set filters at runtime .....	6
Launching the Designer .....	6
Setting Static Data .....	6
Designing the User Interface .....	7
Adding and setting a List.....	7
Adding a button to display the Filters window.....	10
Adding a Label to display the list filters converted in XML.....	12
Adding a Refresh button.....	13
Running a preview of the application.....	15
Preparing the database and binding it to the Application.....	17
Preparing the SQL database .....	17
Using the Application Manager to Define Service Functions .....	18
Defining a GET service function .....	19

Defining the Datakeys.....	24
Defining a SAVE service function .....	25
Importing data into the Designer and carry on building the User Interface .....	29
Creating and setting the Initial Data Service .....	30
Saving the filters .....	32
Running a preview of the application.....	36
Loading the filters .....	38
Running a preview of the application.....	42
Appendix.....	44

## Introduction

Since the version 5.4 of AltioLive, users can create filters at runtime to filter the data displayed in an application.

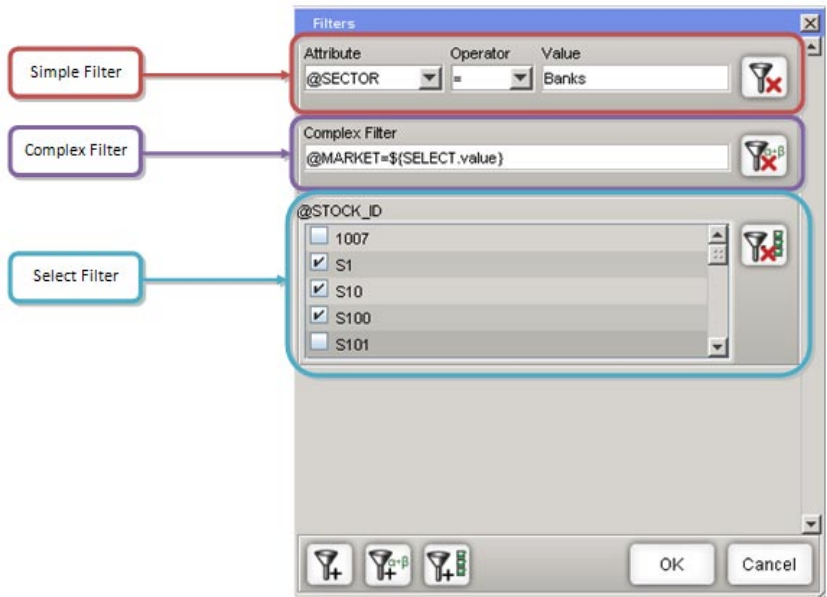
The filters functionality provides a highly flexible mechanism to save and load filter details, some example scenarios are:

- **Security profiles:** a backend system does not provide security restrictions on data but there is a requirement to restrict data made available to the end user. A solution developer can apply both context based routing and filters to ensure only data allowed for users is displayed. This is achieved by binding all controls to a single control (the data source) and then applying a filter to the data source. A solution developer would retrieve user data and load the appropriate filter for the user profile. In this scenario it would not be possible for a user to modify their filters.
- **User Preferences for filter:** sometimes users may wish to have saved versions of filters that they can manage and apply to a user interface. A screen can be implemented so that users can save and load filters from it. Extending this concept a solution developer could implement shared filters which groups of people can use.

This guide explains step by step how to:

- Create a simple application allowing users to create filters at runtime.
- Save these filters to an SQL database using a JDBC Service Function.
- Load the saved filters back, in order to apply them to the application's controls.

*The Filter window:*



*The application:*

The screenshot shows a web application window titled "Financial Data". It contains a table with the following data:

NAME	MARKET	SECTOR	OFFER
Egg Wl	NIKKEI	Banks	155
Halifax Grp	HANGSENG	Banks	771.5
HSBC Holdings	DJONES	Banks	877
Lloyds-TSB	NIKKEI	Banks	709
Nat West Bank	HANGSENG	Banks	1360
Northern Rock	AMEX	Banks	513
Royal Bank of Sc...	NIKKEI	Banks	1613
Stand.Chart.	HANGSENG	Banks	942
Ald.Domecq	NASDAQ	Beverages	432
Barr (AG)	NIKKEI	Beverages	490
Bulmer (HP) Hdq	HANGSENG	Beverages	390

Below the table, there are three main sections:

- Save:** A text input field labeled "Enter a name for the filter" with a "Save" button next to it.
- Load:** A dropdown menu labeled "Load Filter" with "TestName" selected, and a "Load" button next to it.
- Buttons:** A "Show Filter" button is positioned to the right of the "Save" section, and a "Refresh" button is located at the bottom right of the application window.

On the right side of the table, there is a text label "Filter applied to the list" followed by a placeholder "<FILTERS/>".

## Creating a simple application that allows users to set filters at runtime

The **Designer** provides a suite of tools to create the interface an end-user will see. We will add a list to a window as well as a few other controls to allow a user to apply a filter to that List.



### Launching the Designer

1. From **AltioLive Studio**, create a new project.
2. Right-click on the project node and select **View Designer** in the context menu.

The **Designer** is started in a new browser window.

### Setting Static Data

We need some data to populate the list. As it is only data example, you can enter some XML manually directly from the **Designer**. To do so:

1. Copy the XML that can be found in the [Appendix](#) at the end of this document.
2. From the **Designer**, click on the **File | Edit Static Data** menu item.
3. From the **Enter XML window**, paste the XML.
4. Click on **the Show Datatypes Window**  icon in the menu bar.
5. Click on **the Import from current data...** icon  located on the left side of the **Datatypes** window.



6. The **Datatypes** window should now display the **STOCKS** element under the **DATA** node.
7. Expand the **STOCKS** and **STOCK** nodes to display the available data attributes.

Now the link between the data and the user interface is fully completed. Using Static Data allows integrating sample data to an application quickly without having to use and set Service Functions.


## Designing the User Interface

To create and set the user interface, you will need the **View Explorer** and the **Properties** windows. To display them, simply click on **View | Explorer** and **View | Properties** menu items.

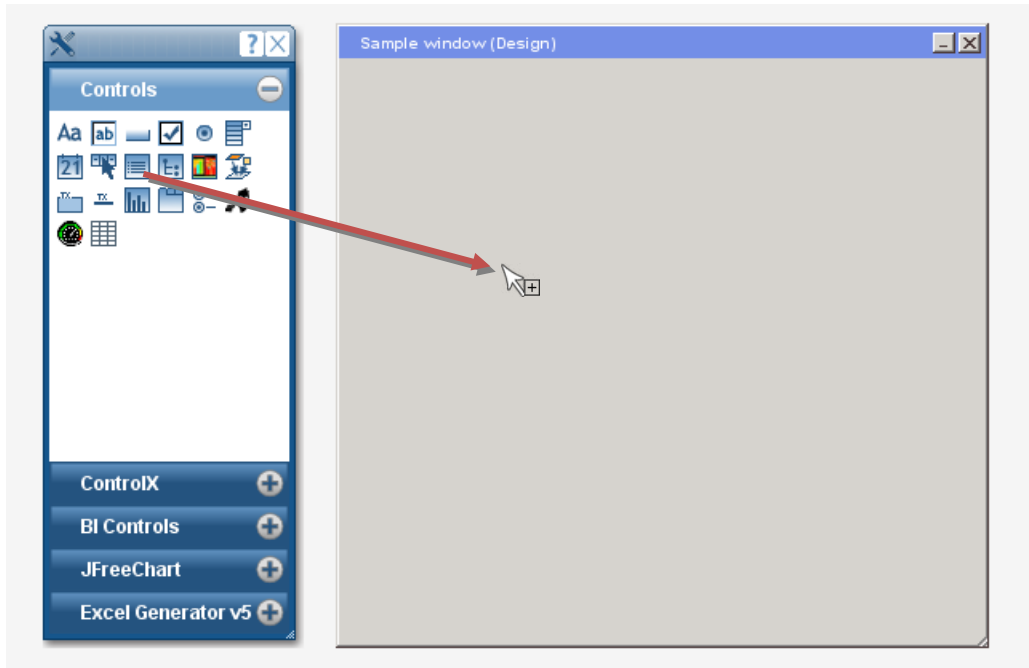
You can also display them with their icons in the menu bar:


-  , to display the **View Explorer** window,
-  , to display the **Properties** window.

## Adding and setting a List

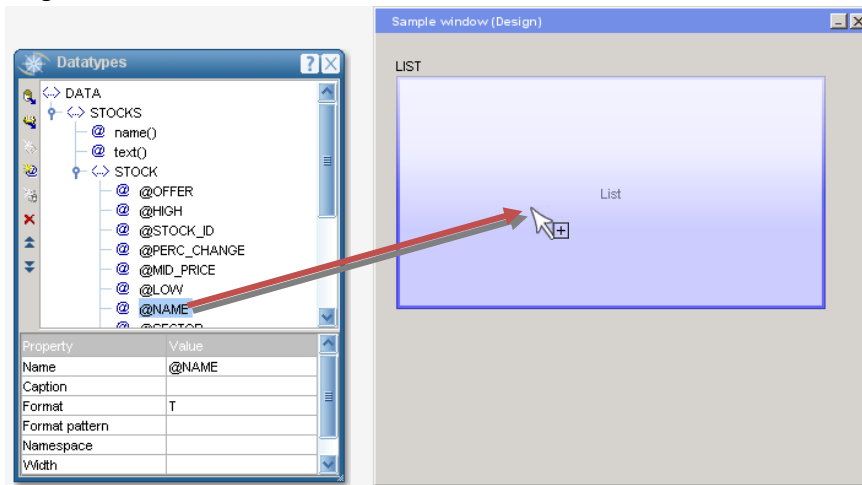
1. From the **View explorer**, expand the **Windows** node. By default, a window is already created.
2. To display a preview of this window, double-click on it.
3. We now need to show the **Control Palette** in order to add a list to this window. To do so, click on the **Show Control Palette** icon .

4. From the **Control Palette**, drag the **List** icon  and drop it onto the window:



5. We now need to associate data to the List. Click on the **Show Datatypes Window** icon .


- From the **Datatypes** window, expand the **STOCKS** and **STOCK** nodes.
- Drag the attribute **@NAME** onto the List.



You will see that the column-heading **NAME** appears in the list control.

- In the same way, drag and drop the attributes **@SECTOR**, **@MARKET** and **@MID\_PRICE** onto the list. The names of the columns appear on the list as the attributes are dragged across.
- Resize the window and the List so that all the columns are displayed.  
We now need to add a button to show the Filters window. This window allows a user to filter the list data at runtime.


### Adding a button to display the Filters window

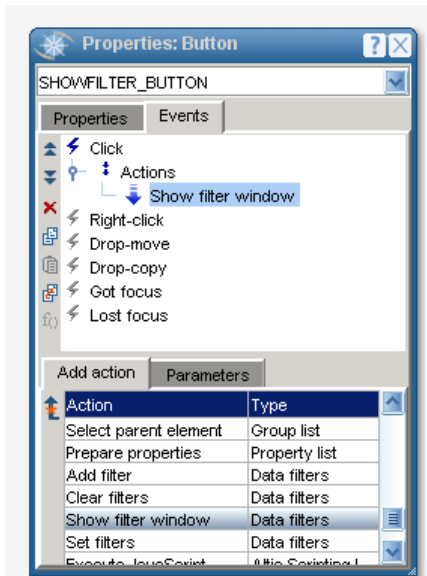
1. From the **Control Palette**, drag the **Button** icon  and drop it onto the window.
2. From the **View Explorer** Window, select the **Button**.
3. From the **Properties** window, set its properties as follows:

Property	Value	Comments
▼ General		
Name	SHOWFILTER_BUTTON	
Caption	Show Filters	

The **Show Filters** button needs further settings as we need to add a **Show filter window** action to be triggered when the user clicks on the button. To do so:

4. Still with the button selected, from the **Properties** window, go to the **Events** tab.  
All the available events for this control are displayed.
5. Select the **Click** event.
6. From the **Add action** tab, a list displays all the available events for this control. Select the **Show filter window** action.


7. Click on the **Add action** icon . You should get a window similar to the one below:




8. Now click on the **Parameters** tab and set the **Control** property to the **LIST**. This specifies on which control the filter(s) will be applied to.

### Adding a Label to display the list filters converted in XML


We can now add a label that will display an XML element containing all the filters set for the List. It is this XML element, and its children elements, that we are going to send to a database to save the filters.

1. From the **Control Palette**, drag the **Label** icon  and drop it onto the window.
2. From the **View Explorer**, select the label and set its properties as follows:


Property	Value	Comments
▼ <b>General</b>		
<b>Name</b>	FILTER_VALUE	
<b>Caption</b>	Xml value of the Filters applied to the list	
<b>Data field</b>	\${LIST.filters}	This is set to display the value of the filters in XML. The label uses one of the runtime properties of the list ( <b>filters</b> ). All the available runtime properties can be shown in the Syntax Builder. The latter can be accessed by clicking on the ellipsis button  in the <b>Data field</b> property.
<b>Wrap text</b>	Y	

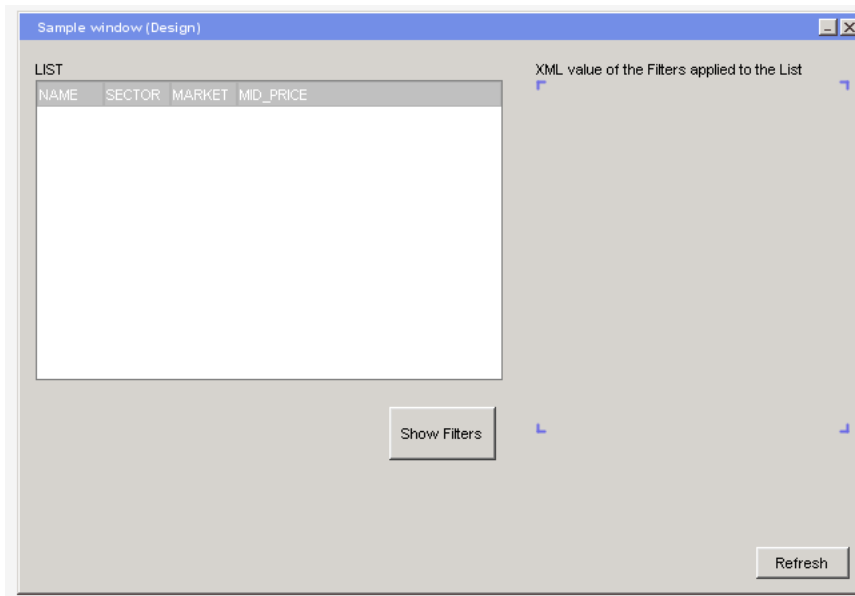
We need some further settings as the content of the **FILTERS** XML element is going to change every time a user is going to add or edit a filter for the list. Therefore the label needs to be refreshed to display the latest content of the **FILTERS** XML element.

### Adding a Refresh button

1. From the **Control Palette**, drag the **Button** icon  and drop it onto the window.
2. From the **View Explorer**, select the button and set its properties as follows:

	Property	Value	Comments
▼	<b>General</b>		
	<b>Name</b>	REFRESH_BUTTON	
	<b>Caption</b>	Refresh	



3. Still with the button selected, from the **Properties** window, go to the **Events** tab.  
All the available events for this control are displayed.
4. Select the **Click** event.
5. From the **Add action** tab, a list displays all the available actions for this control. Select the **Refresh** action.
6. Click on the **Add action** icon .
7. Now click on the **Parameters** tab and set the **Control** property to the **FILTER\_VALUE** label. This specifies which control to refresh.
8. Reposition and resize the controls so that the window looks like the window shown below:



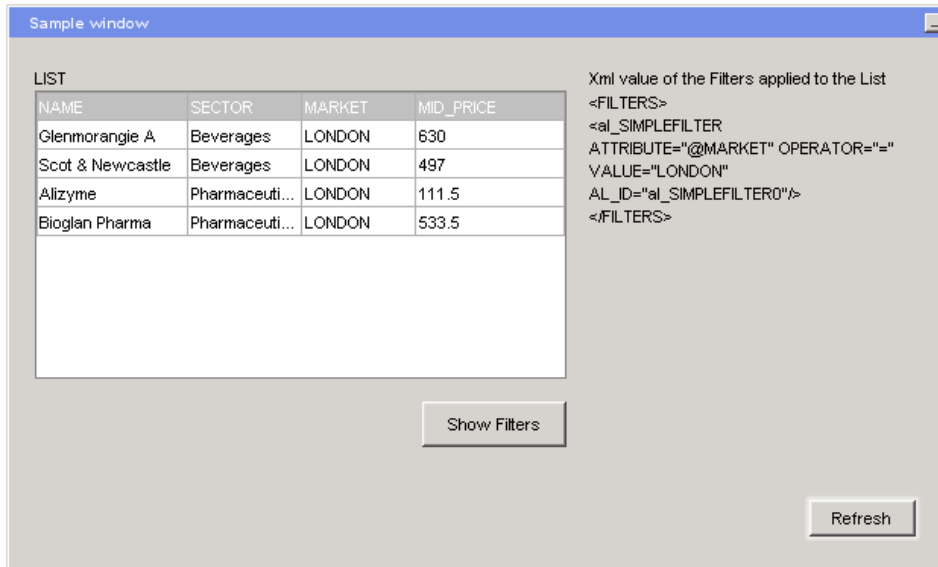
9. Save  the Application.

## Running a preview of the application

You can now click on the **Run** button to see a preview of your application and ensure that it works properly.

1. Click on the **Run** icon  displayed in the menu bar.
2. The list should be populated with data. It is a simple list of stocks that displays for each stock its **name**, to which **sector** and **market** it belongs to and its **middle price**.
3. Click on the **Show Filters** button. The **Filters** window should open.
4. From the **Filters** window, click on the **Add simple filter** button .
5. From the **Attribute** drop-down menu, select the **@MARKET** data attribute.
6. From the **Operator** drop-down menu, select **=**.
7. From the **Value** text field enter **LONDON**.
8. Click on the **OK** button.
9. The List should now only display the stocks of the LONDON market.
10. Click on the **Refresh** button.

11. The label should now display a **FILTERS** element holding the filter we have just set:



The screenshot shows a window titled "Sample window" with a table of stock data and a text area displaying XML. The table has columns for NAME, SECTOR, MARKET, and MID\_PRICE. The text area shows the XML structure for the filters applied to the list.

NAME	SECTOR	MARKET	MID_PRICE
Glenmorangie A	Beverages	LONDON	630
Scot & Newcastle	Beverages	LONDON	497
Alizyme	Pharmaceuti...	LONDON	111.5
Bioglan Pharma	Pharmaceuti...	LONDON	533.5

Xml value of the Filters applied to the List

```
<FILTERS>
<a1_SIMPLEFILTER
ATTRIBUTE="@MARKET" OPERATOR="="
VALUE="LONDON"
AL_ID="a1_SIMPLEFILTER0"/>
</FILTERS>
```

Buttons: Show Filters, Refresh

Currently when the application is stopped, the created filter will be deleted. The next time the user runs the application, the list will display all the stocks by default. If the user wants to filter the data, he will need to recreate filters.

We are going to link this simple application to an SQL database in order to send the XML **FILTERS** element and its contents to the database to be able to store it and then retrieve it when needed by the user.

## Preparing the database and binding it to the Application

In this part we will create JDBC service functions to interact with an SQL database at the Back end. Any data returned from an SQL query has to be converted to XML before processing by the Presentation Server. A template is defined in the Service Function to map SQL columns to XML attributes.

***Please note:** be aware that not all of the features of SQL databases can be invoked using JDBC and the AltiioLive Service Request interface. However, more complex statements are possible than the examples that follow.*

### Preparing the SQL database

For this exercise we will work with a very simple database.

1. Create a simple database in mySQL. Call the schema **filter\_schema** and the table **filter\_table**.
2. Create three columns and set them as follows:

Column Name	Datatypes	NOT NULL	AUTO INC	Flags	Default Value	
FILTER_ID	INTEGER	✓	✓	✓ UNSIGNED	ZEROFILL	NULL
FILTER_NAME	VARCHAR(45)	✓		BINARY		NULL
FILTER_VALUE	BLOB	✓				NULL

- **FILTER\_ID:** will contain a unique number value that is generated by the database. Every time a row is inserted, an incremented number will automatically be put in this column. Most databases provide incrementing number columns. The column should be a primary key.
- **FILTER\_NAME:** will contain the name that the user specifies to describe the filter he created.
- **FILTER\_VALUE:** is a column of type 'BLOB' (Binary Large Object). This will hold the XML **FILTERS** element containing the value of the filters.

3. Enter some example values in the table for example:

FILTER_ID	FILTER_NAME	FILTER_VALUE
1	TestName	<FILTERS></FILTERS>

## Using the Application Manager to Define Service Functions


In this part we will configure two JDBC service functions:

- one to get the data from the database ,
- another one to save the filters created in the back end.

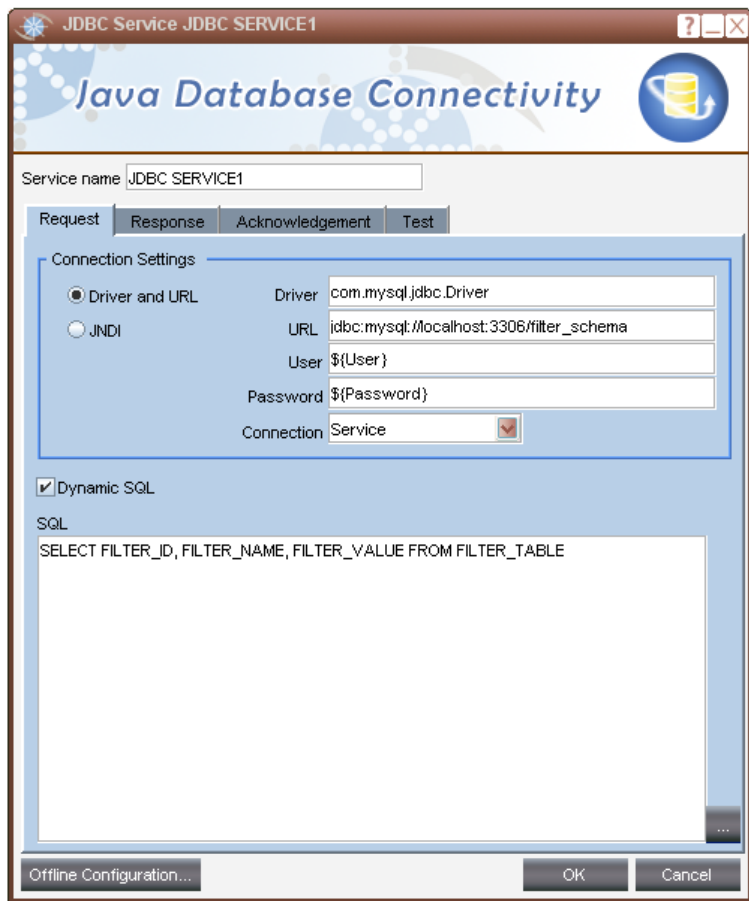
We will start by creating a JDBC Service function that will be used to get the Initial Data in the application.

### Defining a GET service function

When creating a GET service function, it's important to have the correct SQL query. Unlike the HTTP service function, we cannot test the function and datapool until we return to the Designer; so it is advisable to check the SQL using a utility associated with the database before we start. As an example, for **MySQL** we could use the **MySqlManager** to test queries and check results before implementing the command in the Service Function.

1. Before creating the Service Function, you need to copy your MySQL driver jar(s)(**mysql-connector-java-x.x.x-bin.jar**) in the **lib** folder under **altio54\WEB-INF\lib**.
2. Restart AltioLive server.
3. From **AltioLive Studio**, right-click on your project node and select **Application Manager** from the contextual menu.
4. From the **Application manager**, press the **JDBC** button  to create a new JDBC service function.
5. Set the Service name to **GET\_FILTERS**.
6. From the **Request** tab, set the parameters as follow:

Service name	GET_FILTERS	
REQUEST TAB		
Parameter	Value	Description
Radio Button: Driver and URL		
Driver	com.mysql.jdbc.Driver	In the driver's <b>README</b> file
URL	jdbc:mysql://localhost:3306/filter_schema	Set the <b>URL</b> field to the address of the database. Include the schema name but not the database name.
User	Your Sql user, for example: root	You can also set the user as a parameter (by right-clicking on the <b>Parameters</b> folder in the <b>Application Explorer</b> ) and then referring to it with the variable <b>\${User}</b>
Password	Your password	You can also set the password as a parameter (by right-clicking on the <b>Parameters</b> folder in the <b>Application Explorer</b> ) and then referring to it with the variable <b>\${Password}</b>
Connection	Service	
Dynamic SQL	Tick this option	
SQL	SELECT FILTER_ID, FILTER_NAME, FILTER_VALUE FROM FILTER_TABLE	Name of each column in the table ( <b>FILTER_ID</b> , <b>FILTER_NAME</b> ,...) and the name of the table (in this example: <b>FILTER_TABLE</b> ).



7. From the **Response** tab, enter the following in **Template**:

```
<GET_FILTERS>
<MYFILTER NAME='${response.FILTER_NAME}'>
${response.FILTER_VALUE}:xml
</MYFILTER>
</GET_FILTERS>
```

The template is used to format received SQL data into hierarchical XML format. In this example, from the SQL data we are going to create:

<GET_FILTERS>	a <b>GET_FILTERS</b> element
<MYFILTER NAME='\${response.FILTER_NAME}'>	a <b>MYFILTER</b> child element with a <b>NAME</b> attribute. The <b>\${}</b> symbols mean the value of the <b>NAME</b> attribute will be evaluated at runtime.
\${response.FILTER_VALUE}:xml	A child element for the <b>MYFILTER</b> element. The <b>\${}</b> symbols mean the value of this element will be evaluated at runtime. <b>:xml</b> specifies that this value will already have the XML format.
</MYFILTER>	
</GET_FILTERS>	

8. From the **Test** tab, click on the **Test Service** button. The **Test Results** window displays:


The screenshot shows the 'Java Database Connectivity' window with the 'Test Results' tab selected. A red circle highlights the 'Messages' section, and a red arrow points from it to a larger, detailed view of the messages table. The messages table has the following content:

Type	Message
WARNING	No Datakey is defined for element GET_FILTERS
WARNING	No Datakey is defined for element MYFILTER
WARNING	No Datakey is defined for element FILTERS
INFO	Creating new connection with url:jdbc:mysql://localhost:3306/filter_schema;
INFO	connection created in 16 msec.
INFO	statement prepared in 0 msec.

Below the messages table, the 'Header' section is empty. The 'Body' section contains the following XML data:

```
<GET_FILTERS TIMESTAMP="0" AL_ID="GET_FILTERS">  
<MYFILTER NAME="TestName" AL_ID="GET_FILTERS-MYFILTER">  
  <FILTERS AL_ID="GET_FILTERS-MYFILTER-FILTERS">  
  
</MYFILTER>  
</GET_FILTERS>
```

Datakey warnings appear in the test output. Press the **Extract Datakeys** button to create the required keys automatically.

9. A window displays the list of the created datakeys: **GET\_FILTERS**, **MYFILTER** and **FILTERS**. Click on the **OK** button.
10. Save  the Application.
11. Click **OK** in the new window to confirm that the configuration should be updated and the application restarted. A **Validation Results** window will report that the server has been updated.

***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.*

### Defining the Datakeys

Datakeys provide unique identifiers to the Presentation Server.

1. From the **Application Explorer** window, expand the **Datakeys** node.
2. Display the **Properties** window by clicking on **View | Properties** menu item.
3. Select the **MYFILTER** datakey and from the **Properties** window:
  - Enter **MF-** in the **Prefix** field.
  - Select **@NAME** from the **Key** drop-down menu.

This means that for each **MYFILTER** element a unique identifier will be created, it will start with **MF-** followed by the value of the **NAME** attribute.

4. Select the **FILTERS** datakey and from the **Properties** window:
  - Enter **F-** in the **Prefix** field.
  - Enter **parent::MYFILTER/@NAME** in the **Key** field.

This means that for each **FILTERS** element a unique identifier will be created, it will start with **F-** followed by the value of the **NAME** attribute of the parent element **MYFILTER**.

***Please note:** The **GET\_FILTERS** datakey can be ignored as there will be no multiple instances of this element.*

### Defining a **SAVE** service function

Since we are going to the same URL and settings, the easiest way to create the Service Function is to copy the existing one:

1. From the **Application Explorer** window, expand the **Services** node and right-click on the **GET\_FILTERS** Service Function.
2. Select **Clone GET\_FILTERS** in the context menu.
3. Open the new service function, called **GET\_FILTERS1**, by double-clicking on it.
4. In the **Service name** field, enter the name of the new Service Function: **SAVE\_FILTERS**.

5. Set the parameter as follows:

REQUEST TAB		
Parameter	Value	Description
Service name	SAVE_FILTERS	
SQL	INSERT INTO FILTER_TABLE (FILTER_NAME, FILTER_VALUE) VALUES ('\${client.FILTER_NAME}', '\${client.FILTER_VALUE}')	<p>We specify:</p> <ul style="list-style-type: none"> <li>• in which table the values need to be inserted (in this example: <b>FILTER_TABLE</b>)</li> <li>• the name of each column in the table (<b>FILTER_NAME</b> and <b>FILTER_VALUE</b>)*.</li> <li>• then the values to insert in these columns. The <b>{} </b>symbols mean the values will be provided at runtime.</li> </ul>

*\*Note that we do not include the **FILTER\_ID** column as its value is a unique number that is generated by the database.*

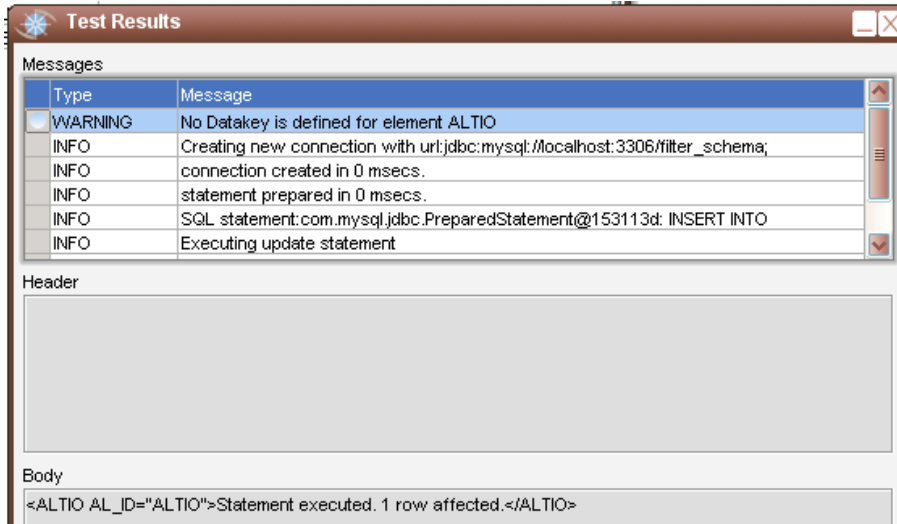
6. From the **Acknowledgement** tab, set the parameters as follow:


ACKNOWLEDGEMENT TAB		
Parameter	Value	Description
On success: DEFINE	Filter Saved Successfully	This is the message that will be sent to the <b>AltioLive</b> client when the service succeeds.
On failure: DEFINE	Failed to save Filter	This is the message that will be sent to the <b>AltioLive</b> client when the service fails.

7. Go to the **Test** tab and enter the following test values:

TEST TAB	
Name	Test Value
client.FILTER_VALUE	<FILTERS><SAMPLEFILTER ID="1"/></FILTERS>
Client.FILTER_NAME	TestName 2

8. Click on the **Test** Service button. The **Test Results** window, should display the following message:

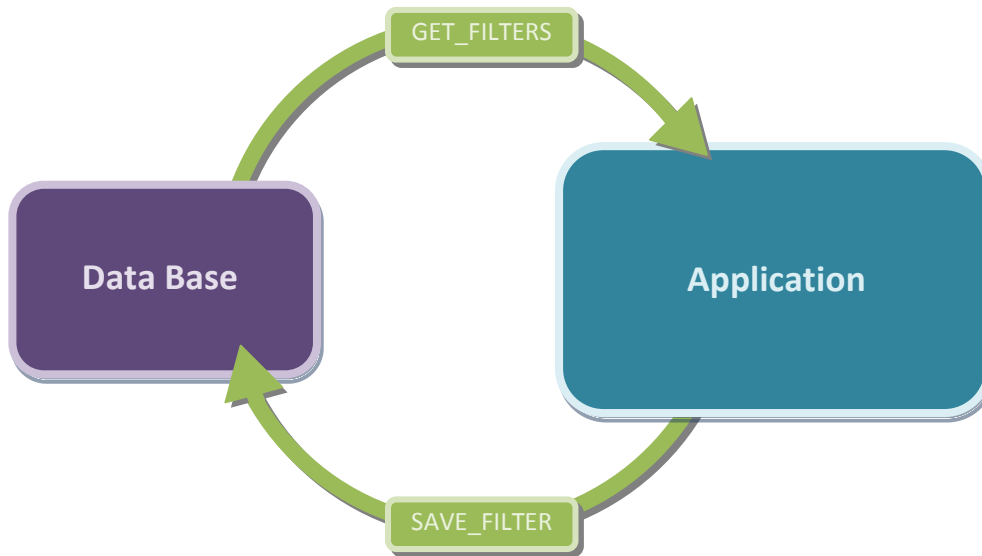


9. Click on the **Save** icon . Click **OK** in the new window to confirm that the configuration should be updated and the application restarted. A **Results** window will report that the server has been updated.
10. Provided the server was updated successfully, close the web browser's window running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

## Importing data into the Designer and carry on building the User Interface

Earlier in the session we created a simple application allowing a user to filter the data of a list.

We then created an SQL table to store the filters, and two service functions to get the data from the database and another one to send the created filters to the database.





To complete the link between the data and the application, we need to add a call to the **GET\_FILTERS** Service Function every time the application is launched. This can be done in the **Designer** with the **Initial Data Service**.

### Creating and setting the Initial Data Service

1. From **AltioLive Studio**, right-click on your project and select **View Designer** in the context menu.  
The **Designer** is started in a new browser window
2. From the **View Explorer** Window, right-click on **Initial Data Services** and select **New Initial Data** in the context menu.
3. A new node, called **INCLUDE**, should be displayed. Select it.
4. From the **Properties** window, expand the **General** Properties.
5. From the **Server Command** drop-down menu, select **GET\_FILTERS**.

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



6. Click the **Show Datatypes** icon  to open the **Datatypes** window.
7. Click on the **Import from Service...** icon  located on the left side of the **Datatypes** window. The **Import datatype information** window is then displayed.
8. From the **Service Function** drop-down menu, select **GET\_FILTERS** and click on the **Import** button.
9. The **Datatypes** window should now display the **STOCKS** and **GET\_FILTERS** elements under the **DATA** node.



10. Expand the **GET\_FILTERS** and **MYFILTER** nodes to display the available attributes.

Now the link between the data and the user interface is fully completed.



## Saving the filters

In order to save the filters, we need the following controls:

- a text field  (where the user will enter the name he wants to give to the filter)
- a button  (which will be clicked by the user to save the filter)


1. From the **View Explorer** window, expand the **Windows** node and double-click on **WINDOW** to display a preview of it.
2. From the **Control Palette** , drag the text field icon  and drop it onto the window.
3. Set the text field as follows:

Property	Value	Comments
▼ General		
Name	FILTER_NAME	As this text field is meant to provide the value of the <b>FILTER_NAME</b> parameter then it has to have the same name as the parameter.
Caption	Enter a name for the filter	

4. From the **Control Palette** , drag the button icon  and drop it onto the window.
5. Set the button as follows:



	Property	Value	Comments
▼	<b>General</b>		
	<b>Name</b>	SAVE_BUTTON	
	<b>Caption</b>	Save	

The **Save** button needs further settings as we need to call the **SAVE\_FILTER** Service Function when the user clicks on the button. To do so:

1. Still with the button selected, from the **Properties** window, go to the **Events** tab.  
All the available events for this control are displayed.
2. Select the **Click** event.
3. From the **Add action** tab, a list displays all the available events for this control. Select the **Server request** action.
4. Click on the **Add action** icon .
5. Now click on the **Parameters** tab and set the parameters as follows:

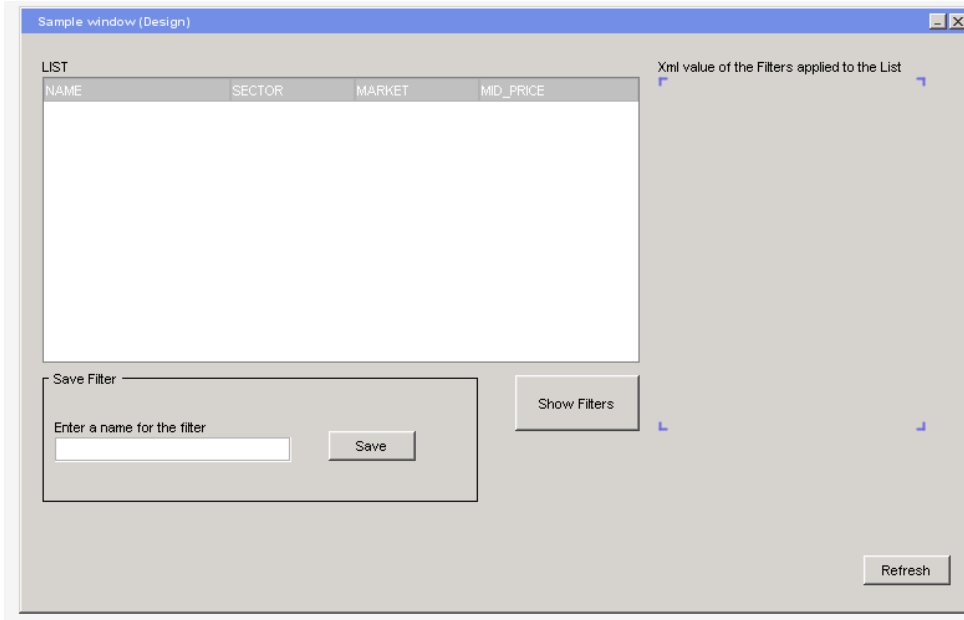
	Parameter	Value	Comments
▼	<b>Parameter</b>	<b>Value</b>	
	<b>Service Function</b>	SAVE_FILTERS	<i>The Service Function we created earlier.</i>

Additionally, we can add a frame around the text field and the save button, to have visual indications that these two controls are meant to be used together to save the filters.

6. From the **Control Palette** , drag the rectangle icon  and drop it onto the window.
7. Set the rectangle as follows:



Property	Value
▼ <b>General</b>	
Name	SAVE_RECTANGLE
Caption	Save Filter

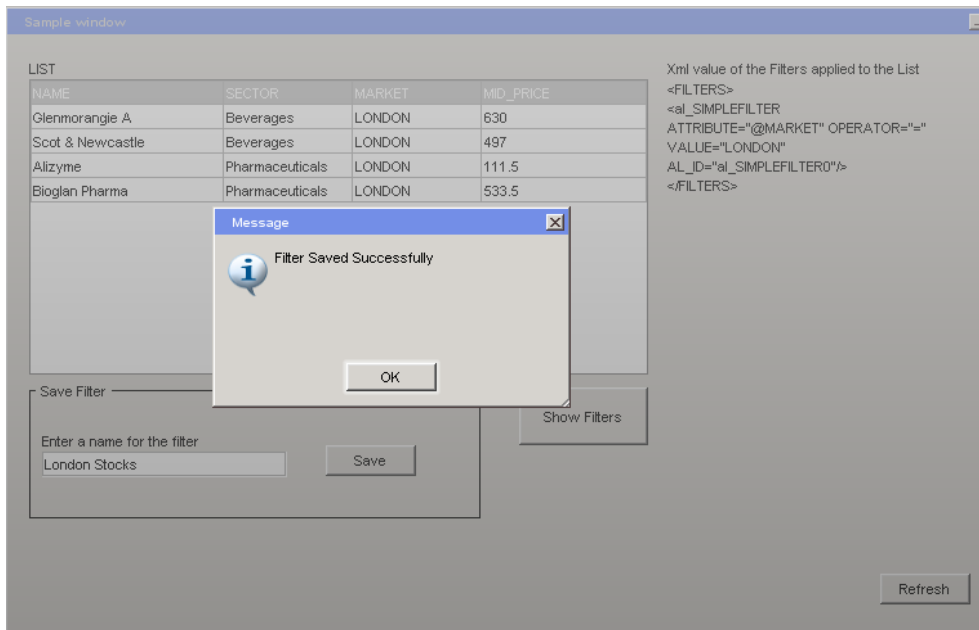
8. Reposition and resize the controls so that the window looks like the window shown below.



## Running a preview of the application

You can now click on the **Run** button to see a preview of your application and ensure that it works properly.

1. Click on the **Run** icon  displayed in the menu bar. The list should be populated with data.
2. Click on the **Show Filters** button. The **Filters** window should open.
3. From the **Filters** window, click on the **Add simple filter** button .
4. From the **Attribute** drop-down menu, select the **@MARKET** data attribute.
5. From the **Operator** drop-down menu, select **=**.
6. From the **Value** text field, enter **LONDON**.
7. Click on the **OK** button.
8. The List should now display only the stocks from the **LONDON** market.
9. Click on the **Refresh** button.
10. The label should now display a **FILTERS** element holding the filter we have just set.
11. In the text field, enter a name for the filter, for example **London Stocks**.
12. Click on the **Save** button.
13. A message should pop up reading “Filter Saved Successfully”.





14. Press the **OK** button.

15. Click on the **Stop**  button.



We have now saved our first filter to the database. To see whether it works properly and whether we can re-use it, we now need to set the application to allow the user to load back the saved filters and apply them to the List control.

## Loading the filters


We now need to add a drop-down menu set to display all the available filters. And a button to load the filter selected in the drop-down menu. To do so:

1. From the **View Explorer** window, expand the **Windows** node and double-click on **WINDOW**.
2. From the **Control Palette** , drag the drop-down menu icon  and drop it onto the window, under the **Save** rectangle.
3. Select the drop-down menu and set its properties as follows:

Property	Value	Comments
▼ General		
Name	FILTER_SELECT	
Caption	Select the filter to load	
List data source	/GET_FILTERS/MYFILTER	Xpath to the element holding the saved filters.
List data field	@NAME	This will use the values held in the NAME attributes to create a list of all the filters available.
List data value	@NAME	This will use the values held in the NAME attributes to create a list of all the filters available.

- From the **Control Palette** , drag the button icon  and drop it onto the window, under the **Save** rectangle.
- Set the button as follows:



Property	Value	Comments
▼ <b>General</b>		
<b>Name</b>	LOAD_BUTTON	
<b>Caption</b>	Load	

- We need further settings for this button as when clicked at runtime, it should set the filter of the list to the one selected in the drop-down menu. To do so:
- Still with the button selected, from the **Properties** window, go to the **Events** tab.
- Select the **Click** event.
- From the **Add action** tab, a list displays all the available events for this control. Select the **Set filters** action.
- Click on the **Add action** icon .

11. Now click on the **Parameters** tab and set the parameters as follows:

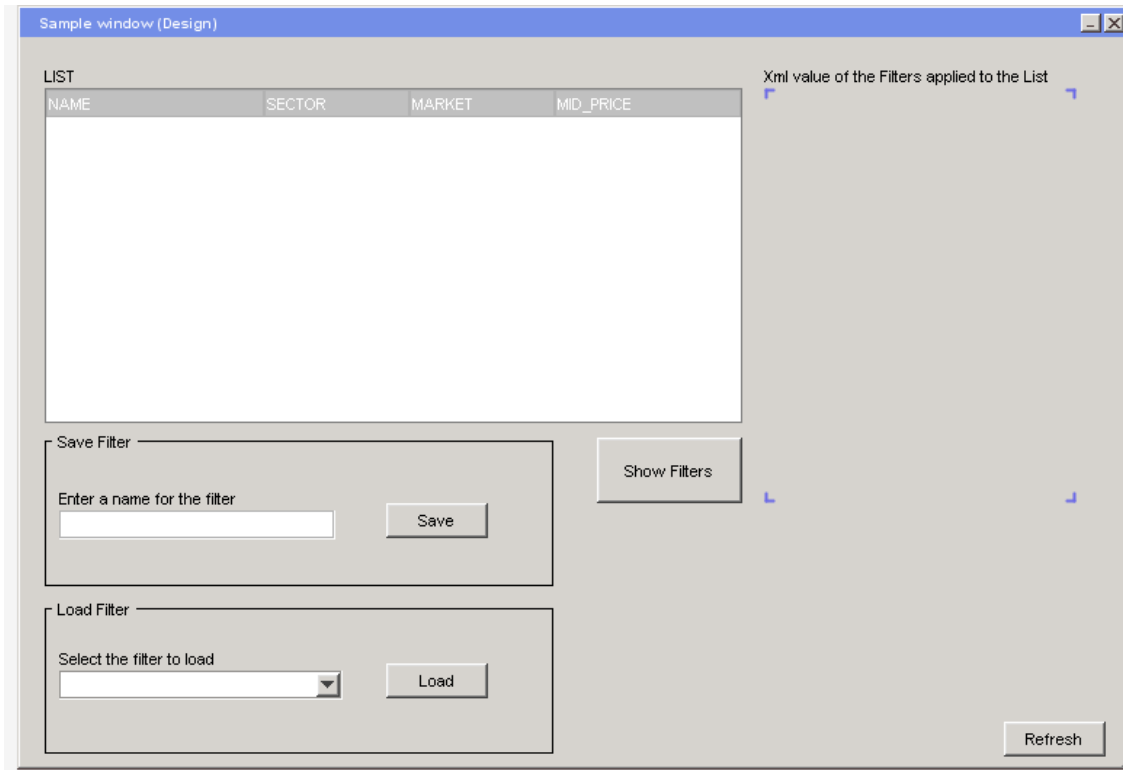
Parameter	Value	Comments
<b>Parameter</b>	<b>Value</b>	
<b>Control</b>	LIST	<i>Specifies the control the Set Filter action should be applied to.</i>
<b>Filters element</b>	/GET_FILTERS/MYFILTER[@NAME=\${FILTER_SELECT.value}]/FILTERS	<i>It specifies the Xpath to the element holding the filter information. Here it is the element <b>FILTERS</b> under the <b>MYFILTER</b> element that has a <b>NAME</b> attribute matching the value selected in the <b>FILTER_SELECT</b> drop-down menu.</i>

Additionally, we can add a frame around the drop-down menu and the **Load** button, to have a visual indication that these two controls are meant to be used together to load the filters.

1. From the **Control Palette** , drag the rectangle icon  and drop it onto the window.
2. Set the rectangle as follows:


Property	Value
<b>General</b>	
<b>Name</b>	LOAD_RECTANGLE
<b>Caption</b>	Load Filter

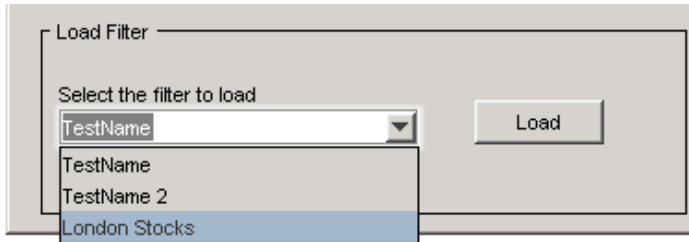
3. Reposition and resize the controls so that the window looks like the window shown below.



## Running a preview of the application

You can now click on the **Run** button to see a preview of your application and ensure that it works properly.

4. Click on the **Run** icon  displayed in the menu bar. The list should be populated with data.
5. Last time we tested the application we saved a filter called **London Stocks**.
6. Expand the drop-down menu, and check the **London Stocks** filter is listed.



7. Select this filter then click on the **Load** button.
8. The filter should now be applied to the list: the list should only display stocks from the LONDON market.

LIST			
NAME	SECTOR	MARKET	MID_PRICE
Glenmorangie A	Beverages	LONDON	630
Scot & Newcastle	Beverages	LONDON	497
Alizyme	Pharmaceuticals	LONDON	111.5
Bioglan Pharma	Pharmaceuticals	LONDON	533.5

9. You can try to create, save and load other filters.

**Please note:**

- *Filters can be applied to any control that has a **data source** property.*
- *More information about filters can be found in the online help that can be accessed from **AltioLive Studio** with the **Help | Help Contents** menu items.*

## Appendix

<STOCKS>

```
<STOCK OFFER="155" HIGH="0" STOCK_ID="S6" PERC_CHANGE="-0.3" MID_PRICE="666" LOW="0" NAME="Egg Wi" SECTOR="Banks"
PENCE_CHANGE="-0.5" CLOSE="154" EPIC="EGG" MARKET="NIKKEI" BID="152" OPEN="0"/>
<STOCK OFFER="771.5" HIGH="792" STOCK_ID="S7" PERC_CHANGE="4.2" MID_PRICE="-777" LOW="725" NAME="Halifax Grp" SECTOR="Banks"
PENCE_CHANGE="31" CLOSE="738" EPIC="HFX" MARKET="HANGSENG" BID="766" OPEN="725"/>
<STOCK OFFER="877" HIGH="881" STOCK_ID="S8" PERC_CHANGE="-0.3" MID_PRICE="888" LOW="873" NAME="HSBC Holdings" SECTOR="Banks"
PENCE_CHANGE="-3.5" CLOSE="880" EPIC="HSBA" MARKET="DJONES" BID="875.5" OPEN="881"/>
<STOCK OFFER="709" HIGH="710.5" STOCK_ID="S9" PERC_CHANGE="-0.7" MID_PRICE="-999" LOW="704" NAME="Lloyds-TSB" SECTOR="Banks"
PENCE_CHANGE="-5" CLOSE="713.5" EPIC="LLOY" MARKET="NIKKEI" BID="708" OPEN="704"/>
<STOCK OFFER="1360" HIGH="1365" STOCK_ID="S10" PERC_CHANGE="0" MID_PRICE="111" LOW="1310" NAME="Nat West Bank" SECTOR="Banks"
PENCE_CHANGE="0" CLOSE="1344" EPIC="NWB" MARKET="HANGSENG" BID="1348" OPEN="1365"/>
<STOCK OFFER="513" HIGH="519" STOCK_ID="S11" PERC_CHANGE="-1.8" MID_PRICE="-222" LOW="511.5" NAME="Northern Rock" SECTOR="Banks"
PENCE_CHANGE="-9.5" CLOSE="521" EPIC="NRK" MARKET="AMEX" BID="510" OPEN="517.5"/>
<STOCK OFFER="1613" HIGH="1638" STOCK_ID="S12" PERC_CHANGE="-1.8" MID_PRICE="333" LOW="1598" NAME="Royal Bank of Scotland"
SECTOR="Banks" PENCE_CHANGE="-30" CLOSE="1642" EPIC="RBOS" MARKET="NIKKEI" BID="1611" OPEN="1638"/>
<STOCK BID="940" OFFER="942" PENCE_CHANGE="-14" OPEN="931" MID_PRICE="941" NAME="Stand.Chart." SECTOR="Banks" CLOSE="955"
LOW="929" PERC_CHANGE="-1.4" MARKET="HANGSENG" EPIC="STAN" HIGH="952" STOCK_ID="S13"/>
<STOCK BID="430.25" OFFER="432" PENCE_CHANGE="-5.5" OPEN="430" MID_PRICE="431.25" NAME="Ald.Domecq" SECTOR="Beverages"
CLOSE="436.75" LOW="427.75" PERC_CHANGE="-1.2" MARKET="NASDAQ" EPIC="ALLD" HIGH="432" STOCK_ID="S14"/>
<STOCK BID="482" OFFER="490" PENCE_CHANGE="0" OPEN="0" MID_PRICE="486" NAME="Barr (AG)" SECTOR="Beverages" CLOSE="486" LOW="0"
PERC_CHANGE="0" MARKET="NIKKEI" EPIC="BAG" HIGH="0" STOCK_ID="S15"/>
<STOCK BID="382" OFFER="390" PENCE_CHANGE="2.5" OPEN="386" MID_PRICE="386" NAME="Bulmer (HP) Hdg" SECTOR="Beverages"
CLOSE="383.5" LOW="386" PERC_CHANGE="0.6" MARKET="HANGSENG" EPIC="BULM" HIGH="386" STOCK_ID="S16"/>
<STOCK BID="24" OFFER="26" PENCE_CHANGE="0" OPEN="0" MID_PRICE="25" NAME="Burn Steward Dist" SECTOR="Beverages" CLOSE="25"
LOW="0" PERC_CHANGE="0" MARKET="WSTREET" EPIC="BRS" HIGH="0" STOCK_ID="S17"/>
```

```
<STOCK BID="708" OFFER="709.5" PENCE_CHANGE="-7" OPEN="715" MID_PRICE="709" NAME="Diageo" SECTOR="Beverages" CLOSE="716"
LOW="702" PERC_CHANGE="-0.9" MARKET="NIKKEI" EPIC="DGE" HIGH="715" STOCK_ID="S18"/>
<STOCK BID="610" OFFER="650" PENCE_CHANGE="0" OPEN="0" MID_PRICE="630" NAME="Glenmorangie A" SECTOR="Beverages" CLOSE="630"
LOW="0" PERC_CHANGE="0" MARKET="LONDON" EPIC="GMGA" HIGH="0" STOCK_ID="S19"/>
<STOCK BID="32" OFFER="35" PENCE_CHANGE="0" OPEN="0" MID_PRICE="33.5" NAME="Merrydown" SECTOR="Beverages" CLOSE="33.5" LOW="0"
PERC_CHANGE="0" MARKET="DJONES" EPIC="MYW" HIGH="0" STOCK_ID="S20"/>
<STOCK BID="117" OFFER="122" PENCE_CHANGE="0" OPEN="0" MID_PRICE="119.5" NAME="Nichols" SECTOR="Beverages" CLOSE="119.5" LOW="0"
PERC_CHANGE="0" MARKET="NIKKEI" EPIC="NICL" HIGH="0" STOCK_ID="S21"/>
<STOCK BID="495" OFFER="498.5" PENCE_CHANGE="1" OPEN="495" MID_PRICE="497" NAME="Scot & Newcastle" SECTOR="Beverages"
CLOSE="496" LOW="494.5" PERC_CHANGE="0.2" MARKET="LONDON" EPIC="SCTN" HIGH="497" STOCK_ID="S22"/>
<STOCK BID="470.5" OFFER="474.5" PENCE_CHANGE="2.5" OPEN="470" MID_PRICE="472.5" NAME="South African Breweries" SECTOR="Beverages"
CLOSE="470" LOW="469.5" PERC_CHANGE="0.5" MARKET="AMEX" EPIC="SAB" HIGH="473" STOCK_ID="S23"/>
<STOCK BID="122" OFFER="123" PENCE_CHANGE="0" OPEN="0" MID_PRICE="122.5" NAME="Acambis" SECTOR="Pharmaceuticals" CLOSE="122.5"
LOW="0" PERC_CHANGE="0" MARKET="NIKKEI" EPIC="ACM" HIGH="0" STOCK_ID="S24"/>
<STOCK BID="110" OFFER="113" PENCE_CHANGE="0" OPEN="0" MID_PRICE="111.5" NAME="Alizyme" SECTOR="Pharmaceuticals" CLOSE="111.5"
LOW="0" PERC_CHANGE="0" MARKET="LONDON" EPIC="AZM" HIGH="0" STOCK_ID="S25"/>
<STOCK BID="142" OFFER="145" PENCE_CHANGE="0" OPEN="143.5" MID_PRICE="143.5" NAME="Antisoma" SECTOR="Pharmaceuticals"
CLOSE="143.5" LOW="143.5" PERC_CHANGE="0" MARKET="NASDAQ" EPIC="ASM" HIGH="143.5" STOCK_ID="S26"/>
<STOCK BID="3183" OFFER="3195" PENCE_CHANGE="9" OPEN="3175" MID_PRICE="3189" NAME="AstraZeneca" SECTOR="Pharmaceuticals"
CLOSE="3180" LOW="3158" PERC_CHANGE="0.2" MARKET="STOCKHOLM" EPIC="AZN" HIGH="3198" STOCK_ID="S27"/>
<STOCK BID="530" OFFER="537" PENCE_CHANGE="-1.5" OPEN="0" MID_PRICE="533.5" NAME="Bioglan Pharma" SECTOR="Pharmaceuticals"
CLOSE="535" LOW="0" PERC_CHANGE="-0.2" MARKET="LONDON" EPIC="BGP" HIGH="0" STOCK_ID="S28"/>
<STOCK BID="18" OFFER="19" PENCE_CHANGE="-0.5" OPEN="18.5" MID_PRICE="18.5" NAME="Br.Biotech" SECTOR="Pharmaceuticals" CLOSE="19"
LOW="18.5" PERC_CHANGE="-2.6" MARKET="WSTREET" EPIC="BBG" HIGH="18.75" STOCK_ID="S29"/>
<STOCK BID="1925" OFFER="2050" PENCE_CHANGE="-22.5" OPEN="1987.5" MID_PRICE="1987.5" NAME="Cambridge Antibody"
SECTOR="Pharmaceuticals" CLOSE="2010" LOW="1987.5" PERC_CHANGE="-1.1" MARKET="NIKKEI" EPIC="CAT" HIGH="1987.5" STOCK_ID="S30"/>
</STOCKS>
```

## Document Information

**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.