Reference Guide
D301480X412
April 2012

OpenEnterprise Browser Control Reference Guide (V2.83)

Remote Automation Solutions

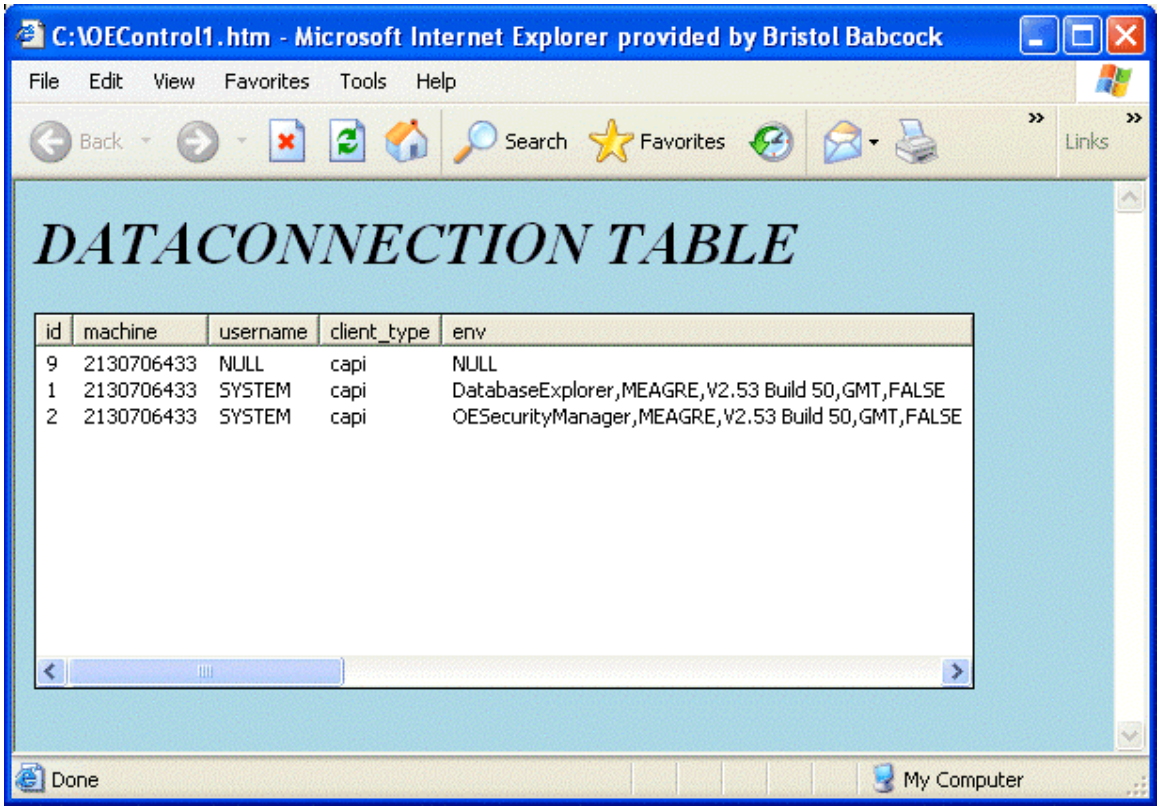
Website: www.EmersonProcess.com/Remote

Contents

1 Browser Control

1.1 Overview

The OpenEnterprise Browser Control is an ActiveX control implementing a Table View populated by data sourced from an OpenEnterprise database. This ActiveX control is fully configurable using HTML, and can be embedded as an object within a Web Browser to display data accessible over a Windows network . Active and static SQL queries are supported as well as transactional updates.



Browser Control Functionality

1.2 Browser Control Functionality and Requirements

The control exhibits the following technologies and functionality:

- Thin Client - downloadable by Web Browser.
- Fully configurable using HTML and Script (JavaScript & VBScript).
- Generic control - all the schema knowledge is within the HTML.
- Columns and rows are built using the query results.
- Low bandwidth database client:
    - No schema knowledge – the control has no knowledge of the tables and attributes in the database it is connected to.

  - Active query support - so no periodic polling or refresh required.
  - Delta deadband support to further reduce bandwidth use.
  - Max Rows support.

- Database security logon.
- Transactional updates for active queries (Bool, Float, Integer & Char).
- Free format SQL transactions.
- Active or static SQL queries.
- Queries can be changed at runtime.
- Coupled queries.
- No OpenEnterprise dependencies.
- Read Only or updateable views.
- Export to CSV or HTML table.
- Scripts have access to row and column selections.
- Active sorting, ascending or descending, for any named column.
- Security logon and logoff.
- Manual column sorting.
- Configurable database reconnection regimes.
- Timestamps automatically converted to browsers locale.
- Optional status window.
- Paged query facility.

1.2.1 Requirements

OpenEnteprise Browser Controls is designed to run within the Microsoft Internet Explorer browser. The minimum requirement for OpenEnteprise Browser Controls is Internet Explorer Version 5 or later.

Creating a Browser Control

1.3 Creating an OpenEnteprise Browser Controls Instance

The following shows an example of the control being used within a HTML page. In this configuration an active query will be set up to monitor the OpenEnterprise database dataconnection table.

Example: -

```
<html>

<head>

<style>
```

```
body {background color:lightblue}

</style>

<title></title>

</head>

<body>

<h1 id="pageHeader"><i>DESCRIPTION</i></h1>

<form method="POST" name="frmDefault">

      <object classid="clsid:959349BA-A392-4D18-A8B1-3E5878AD816A"
      id="myControl" width="500" height="200">

            <PARAM NAME ="_cx" value="5080">

            <PARAM NAME ="_cy" value="5080">

            <PARAM NAME ="SQL" value="select * from dataconnection">

            <PARAM NAME ="DataService" value="rtrdb1">

            <PARAM NAME ="User" value>

            <PARAM NAME ="Password" value>

            <PARAM NAME ="Active" value="1">

            <PARAM NAME ="FTEnable" value="0">

            <PARAM NAME ="ReadOnly" value="1">

            <PARAM NAME ="MaxRows" value="-1">

            <PARAM NAME ="StatusWindow" value="0">

      </object>

</form>

</body>

<SCRIPT>

</SCRIPT>

</html>
```

In the body of the HTML page, the user creates a form to insert the object (OpenEnteprise Browser Control). The object needs to be defined as OpenEnteprise Browser Control. OpenEnteprise Browser Control is defined by its classid "clsid:959349BA-A392-4D18-A8B1-3E5878AD816A".

The user definable *id* value of *myControl* will be used by the scripting language to reference the controls properties and methods.

The OpenEnteprise Browser Control needs to be set-up with the users specific configuration. Parameters are used to setup the specific configuration. The user uses PARAM NAME and specifies a parameter, with a value, in the object. The parameter can be set during the initiation (as above) or during runtime.

1.4 Parameters

The following parameters can be specified by HTML PARAM tags. Note that the exact case must be used when specifying parameter names. Most properties can also be specified and changed at runtime using a scripting language like VBScript or JavaScript. The user will need to call either the Submit() or Refresh() methods for any runtime property changes to take effect.

Each property described below includes a 'key' section that denotes the following:

- **R** - Property can be read.
- **W** - Property can be updated.
- **I** - Initialization property only (cannot be updated by script at runtime).

Active

ActiveSortAscending

ActiveSortColumn

AutoSize

Cell

ConnectionService

ConnectionStatus

ConnectRetries

ConnectRetryInterval

CoupledSQL

DataService

FirstRow

FTEnable

FTHeartbeatInterval

FTHeartbeatTimeout

LastRow

MaxRows

MaxRowsPerPage

MinIntervalSecs

Password

ReadOnly

RowCount

SelectedRow

SQL

StatusWindow

Title

User

Examples

1.4.1 Active - (RW)

Specifies whether the initial query should be active ("1") or one-shot ("0"). The control defaults to one-shot. Active means the data is always updated in accordance to the changes in the database. Active also provides the user with the option of updating the data on the OpenEnteprise Browser Control by double clicking on a cell.

Note: Active queries must have the primary key include in the SQL statement otherwise an error will appear “Query Status 12”.

```
function onActive()

      frmDefault.myControl.Active = 1
      // Set the OpenEnteprise Browser Control to be active.

      frmDefault.myControl.submit();
```

Parameters

1.4.2 ActiveSortAscending - (RW)

If configured, specifies whether the active sort, is sorted ascending or descending. The control defaults to ascending order. If ActiveSortAscending is set to 1 it means ascending, else descending. ActiveSortAcending only works if the query type is active and ActiveSortcolumn is not set. ActiveSortAscending and Column does apply with Coupled SQL. It is undefined behaviour if the user tries and sorts using Coupled SQL.

```
function onActive()

      frmDefault.myControl.Active = 1;
                              // Output will change dynamically

      frmDefault.myControl.ActiveSortColumn = ‘name’;

      // Output will be sorted by name

      frmDefault.myControl.ActiveSortAscending =
      1;                          // Sort in ascending order

      frmDefault.myControl.submit();
```

Parameters

1.4.3 ActiveSortColumn - (RW)

Specifies the name of a column to actively sort on. The name must be the name of a column returned in the query results. It also must be an active query to enable the column to be sorted. If the user requires the control to be sorted, when not an active query, the control needs to be configured using the ‘order by’ in the SQL string.

To disable ActiveSortColumn, pass an empty string (no column name) into the OpenEnteprise Browser Control. When ActiveSortColumn is disabled (no column name), Manual Sort is enabled. Manual Sort is sorting by clicking on the column heading and the OpenEnteprise Browser Control will sort on that column.

```
function onActive()

      frmDefault.myControl.Active =
      1;                                                    // Output will
      change dynamically

      frmDefault.myControl.ActiveSortColumn = 'name';

      // Output will be sorted by name

      frmDefault.myControl.ActiveSortAscending =
      1;                          // Sort in ascending order

      frmDefault.myControl.submit();
```

Parameters

1.4.4 AutoSize - (RW)

Specifies whether the displayed columns are automatically resized for active query updates. The control default is false for AutoSize. AutoSize works when the cell is updated and is too long for the column width. The column will change size to fit the cell information. AutoSize is only used for Active query. When the user submits a query (Active/Static) it will always start with an autosize for each column.

```
function onSize()

      frmDefault.myControl.AutoSize = 1;                                  //
Autosize set to active

      frmDefault.myControl.Refresh();
```

Parameters

1.4.5 Cell(*row*, *column*) - (R)

Retrieves data from a specific cell in the List View. The ***row*** argument specifies the row in the control and ***column*** specifies the column within that row. Note the row values start from 0 and the column values start from 1.

```
function onCellRead()

               var cellValue;

               var nRow = 5;

               var nColumn = 8;

               cellValue = frmDefault.myControl.Cell(nRow, nColumn
);

 // Ask for the value in row 5, column 8
```

Parameters

1.4.6 ConnectionService - (R)

This is the actual service string that the control is currently connected to. If the user is connected to a fault tolerant database it will indicate which one of the databases it is connected to e.g. oeserv1:rtrdb1

```
function getServiceStatus()

      var dataservice;

      var Status;

      dataservice = frmDefault.myControl.ConnectionService;             //
Get the dataservice connected to

      status =
frmDefault.myControl.ConnectionStatus;                          // Find out
if connected to database

      if (status == 1)

     {

          alert(“Connected to database” +dataservice);
```

Parameters

1.4.7 ConnectionStatus - (R)

Returns a boolean value indicating whether the control is currently connected to a database. 1 means OpenEnteprise Browser Controls is connected to a database.

```
function getServiceStatus()

      var dataservice;

      var Status;

      dataservice = frmDefault.myControl.ConnectionService;             //
Get the dataservice connected to

      status =
frmDefault.myControl.ConnectionStatus;                          // Find out
if connected to database

      if (status == 1)

     {

        alert(“Connected to database” +dataservice);
```

Parameters

1.4.8 ConnectRetries - (I)

The number of times to retry a database connection attempt. When retries have been exhausted, no further connection attempts will be performed. The control defaults the connect retries to 3. Specify -1 in the control for infinite retries.

```
<PARAM NAME="ConnectRetries" value="-1">
```

Parameters

1.4.9 ConnectRetryInterval - (I)

The number of seconds between successive database connection attempts. The control defaults the interval to 10 seconds.

```
<PARAM NAME="ConnectRetryInterval" value="30">
```

Parameters

1.4.10 CoupledSQL - (RW).

This is an optional second SQL whose query results will be appended to the results from the query specified by the SQL parameter. This functionality will only work if the select column lists of the two queries are identical. The coupled SQL only works if the same attributes are in both sql statements. It will not work if the user uses 'select * from'.

```
function onSubmitCSQL

      frmDefault.myControl.SQL = "Select name, value from realanalog";

      frmDefault.myControl.CoupledSQL = "Select name, value from digital";

      frmDefault.myControl.submit();
```

Parameters

1.4.11 DataService - (I)

The dataservice that the OpenEnteprise Browser Control will try and connect to. Valid dataservices include standalone and/or redundant dataservices. The control defaults to the dataservice to *rtrdb1*.

```
<PARAM NAME ="FTEnable" value="1">

<PARAM NAME ="DataService" value="oeserv1:rtrdb1,oeserv2:rtrdb1">
```

Parameters

1.4.12 FirstRow(*column*) -(R)

This is used to retrieve the current column value for the first row in the control. The ***column*** argument specifies the column number to retrieve. Columns are ordered from left to right with column values starting from 1. This is very useful for constructing a paged query application.

```
function onFirstRow()

       var cellValue;

       var nColumn = 4;

       cellValue = frmDefault.myControl.FirstRow(nColumn
);

      // Ask for value from first row column 4
```

Parameters

1.4.13 FTEnable - (I)

This specifies whether the database connection should be fault tolerant ("1") or standalone ("0"). Note that the control defaults to standalone. If FTEnable is set to standalone and a fault tolerant database is specified, the control will try and connect to the first database in the dataservice string i.e. oeserv1:rtrdb1

```
<PARAM NAME ="FTEnable" value="1">

<PARAM NAME ="DataService" value="oeserv1:rtrdb1,oeserv2:rtrdb1">
```

Parameters

1.4.14 FTHeartbeatInterval - (I)

This specifies the heartbeat interval used for fault tolerant database connections. The control defaults the FTHeartbeatInterval to 20000 milliseconds.

```
<PARAM NAME =" FTHeartbeatInterval " value="50000">

<PARAM NAME =" FTHeartbeatTimeout " value="60000">
```

Parameters

1.4.15 FTHeartbeatTimeout - (I)

This specifies the heartbeat time-out used for fault tolerant database connections. The control defaults the FTHeartbeatTimeout to 30000 milliseconds.

```
<PARAM NAME =" FTHeartbeatInterval " value="50000">

<PARAM NAME =" FTHeartbeatTimeout " value="60000">
```

Parameters

1.4.16 LastRow(*column*) - (R)

This is used to retrieve the current column value for the last row in the control. The ***column*** argument specifies the column number to retrieve. Columns are ordered from left to right with column values starting from 1. This is very useful for constructing a paged query application.

```
function onLastRow()

     var cellValue;

     Var nColumn = 4;

     cellValue = frmDefault.myControl.LastRow(nColumn
);                                  // Ask for value from last row column 4
```

Parameters

1.4.17 MaxRows - (RW)

Specifies the maximum number of records to be returned by the query. The control defaults MaxRows to 20. Specify -1 in the control for infinite number of rows (any minus number specified changes it to infinite no rows).

```
function onMRows()

      frmDefault.myControl.MaxRows = -
1;                                                    // sets max rows to
infinite

     frmDefault.myControl.submit();
```

Parameters

1.4.18 MaxRowsPerPage - (R)

This Retrieves the maximum number of rows that can be displayed in the control at its current height. This is very useful for constructing a paged query application.

```
function onRowPerPage()

      var rowPage;

      rowPage = frmDefault.myControl.MaxRowsPerPage;                    //
Ask for no of Max rows per page

}
```

Parameters

1.4.19 MinIntervalSecs - (RW)

This specifies the minimum number of seconds between active query updates. It is very useful for low bandwidth situations to minimize the bandwidth usage. The control defaults MinIntervalSecs to two.

```
function onMinInt()

      frmDefault.myControl.MinIntervalSecs = 20;                  // set
min Interval to 20

     frmDefault.myControl.submit();
```

Parameters

1.4.20 Password - (I)

The initial logon password. The control defaults to no password.

```
<PARAM NAME ="USER" value="SYSTEM">

<PARAM NAME ="PASSWORD" value="SYSTEM">
```

Parameters

1.4.21 ReadOnly - (RW)

This specifies whether the control will be read only or updateable. Updateable controls allow the user to double-click on a cell to update the value. Note that a control is only capable of being updateable when running an active query. The control defaults ReadOnly to updateable.

```
function onReadOnly()

      frmDefault.myControl.ReadOnly = 1;
                                              // Sets read-only to true

      frmDefault.myControl.Refresh();
```

Parameters

1.4.22 RowCount - (R)

This returns the current number of rows in the control. For a coupled SQL it will also return the number of rows in the control.

```
function readRowCount()

               var numRows;

               numRows = frmDefault.myControl.RowCount;                //
Ask for no of rows in the control
```

Parameters

1.4.23 SelectedRow(*column*) - (R)

This is used to retrieve the current column value for the currently selected row in the control. The ***column*** argument specifies the column number to retrieve. Columns are ordered from left to right with column values starting from 1.

```
function readSelRow()

     var valueCell;

     valueCell = frmDefault.myControl.SelectRow(5);                  // Ask
for the value of the select row on the OpenEnterprise

// Browser Control column number 5

}
```

Parameters

1.4.24 SQL - (RW)

The initial SQL query. This can be left blank for no initial query. Default value is "`select id, username, client_type, clientid, connectiontime, workstation from dataconnection`". The SQL query parameter supports all Polyhedra SQL standards like operators (LIKE, order by etc).

i.e.

```
Select name, value from realanalog where name = 'CFE1:ATEST.001';

Select * from digital where name LIKE 'CFE%'

Select name, value from realanalog where order by name asc.
```

Note: For an Active query the user must have the primary key in the statement otherwise an error of status query 12 will appear.

Parameters

function onSubmitSQL

frmDefault.myControl.SQL = "Select * from realanalog";

frmDefault.myControl.submit();

1.4.25 StatusWindow - (RW)

Specifies whether the control will display the optional Status Window. The Status Window will show various events including database connectivity, query and transaction updates. The control defaults to no Status Window. Status Window will appear if set the parameter is set to 1. When performing a coupled SQL, the amount of rows given in the Status window is the number of rows produced in the second query (not the amount for both queries).

```
function onStatusWindow()

      frmDefault.myControl.StatusWindow = 1;                    //set the
status window to be on

      frmDefault.myControl.Refresh();
```

Parameters

1.4.26 Title - (RW)

This specifies whether the control will contain a Title bar. The control defaults to no title. If the user has set the OpenEnteprise Browser Control to no title in the initialization then the title bar cannot be turned on. If the title is on, the user cannot turn the title bar off but the title bar can be an empty title bar.

```
Function buttonSelected()

      frmDefault.myControl.Title = "alarmsummary";

      frmDefault.myControl.Submit();
```

Parameters

1.4.27 User - (I)

The initial logon username. This must be a valid database user otherwise the logon will fail. The control defaults to no username.

```
<PARAM NAME ="USER" value="SYSTEM">

<PARAM NAME ="PASSWORD" value="SYSTEM">
```

If the user tries to update any of the initialization parameters in runtime, the control provides no indication of whether it is successful. If the user reads the value of that parameter from the control, the control will give the new value, however the control will still be using the initial value. E.g. Data Service will always be connected to the first service.

Parameters

1.5 Examples

An example of changing parameters at runtime.

```
Function buttonSelected()

      Var mRows;

      mRows = 50;

      frmDefault.myControl.SQL = “select name, description , value from
      alarmsummary”;

      frmDefault.myControl.Title = “alarmsummary”;

      frmDefault.myControl.MaxRows = mRows;

      frmDefault.myControl.Submit();
```

An example of the Refresh function

```
function onRefresh()

       window.document.frmDefault.myControl.ReadOnly =1;

       window.document.frmDefault.myControl.StatusWindow = 1;

       window.document.frmDefault.myControl.AutoSize =1;

       window.document.frmDefault.myControl.Refresh();
```

An example of read parameter tags

```
Function onRead()

      var numRows;

     var cellValue;

     var nRow = 5;
```

```
var nColumn = 8;
 numRows = frmDefault.myControl.RowCount;
 cellValue = frmDefault.myControl.Cell(nRow , nColumn );
```

1.6 Methods

The following methods can be invoked at runtime. All the code extracts are JavaScript.

ExportToCSV(filename, status)

ExportToHTML(filename, errors)

Login(username, password)

Logout()

Refresh()

Submit()

SubmitTrans(sql)

1.6.1 ExportToCSV(*filename*, *status*)

Exports the controls query data in CSV format to the specified ***filename***. The status of the export is returned in the ***status*** argument. A status of zero indicates success.

```
var filename = "c:\temp\data.html";
var status;
myControl.ExportToCSV(filename, status);
if (status == 0) {
// operation succeeded
```

1.6.2 ExportToHTML(*filename*, *errors*)

Exports the controls query data in HTML table format to the specified ***filename***. The status of the export is returned in the ***status*** argument. A status of zero indicates success.

```
var filename = "c:\temp\data.csv";
var status;
myControl.ExportToHTML(filename, status);
if (status == 0) {
  // operation succeeded
```

1.6.3 Login(username, password)

Requests a database logon as the specified ***username*** and ***password***.

```
var user = document.loginForm.txtUser.value;
var password = document.loginForm.txtPassword.value;
```

```
myControl.Login(user, password);
```

1.6.4 Logout()

Requests a database log out.

```
myControl.Logout();
```

1.6.5 Refresh()

Causes the control to apply any property changes.

```
myControl.StatusWindow = true;

myControl.Refresh();
```

1.6.6 Submit()

Requests the SQL query parameters to be applied. This results in the current SQL parameter query being executed in the context of the other query specific parameters.

```
myControl.MaxRows = 50;

myControl.SQL = "select * from eventhistory";

myControl.Submit();
```

1.6.7 SubmitTrans(*sql*)

Requests the specified ***sql*** transaction to be applied to the connected database. The *sql* can be any free format transaction and does not need to be associated with the query that the control is currently executing.

```
var id = myControl.SelectedColumn(1);

var sql = "update alarmsummary set acknowledged = true where id = " + id;

myControl.SubmitTrans(sql);
```

2 Index

DISCLAIMER

Bristol, Inc., Bristol Babcock Ltd, Bristol Canada, BBI SA de CV and the Flow Computer Division , are wholly owned subsidiaries of Emerson Electric Co. doing business as Remote Automation Solutions ("RAS"), a division of Emerson Process Management. ROC, FloBoss, ROCLINK, Bristol, Bristol Babcock, ControlWave, TeleFlow and Helicoid are trademarks of RAS. AMS, PlantWeb and the PlantWeb logo are marks of Emerson Electric Co. The Emerson logo is a trademark and service mark of the Emerson Electric Co. All other marks are property of their respective owners.

The contents of this publication are presented for informational purposes only. While every effort has been made to ensure informational accuracy, they are not to be construed as warranties or guarantees, express or implied, regarding the products or services described herein or their use or applicability. RAS reserves the right to modify or improve the designs or specifications of such products at any time without notice. All sales are governed by RAS' terms and conditions which are available upon request. RAS does not assume responsibility for the selection, use or maintenance of any product. Responsibility for proper selection, use and maintenance of any RAS product remains solely with the purchaser and end-user.

Engineered and supported by:

Remote Automation Solutions,

Blackpole Road, Worcester, WR3 8YB, UK

Registered office: Meridian East, Leicester, LE19 1UX

Registered in England and Wales, Registration No. 00671801

VAT Reg No. GB 705 353 652

Emerson Process Management
Remote Automation Solutions
1100 Buckingham St
Watertown, CT 06795
T 1 (860) 945 2200
F 1 (860) 945 2278
www.EmersonProcess.com/Remote
binfo@EmersonProcess.com

Emerson Process Management
Remote Automation Solutions
Blackpole Road
Worcester, WR3 8YB
T 44 (0) 1905 856848
F 44 (0) 1905 856930
www.EmersonProcess.com/Remote
oedsupport@EmersonProcess.com

© 2001-2012 Remote Automation Solutions, division of Emerson Process Management. All rights reserved.