FREE eBook

LEARNING devexpress

Free unaffiliated eBook created from
Stack Overflow contributors.

#devexpres

S

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: devexpress

It is an unofficial and free devexpress ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official devexpress.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with devexpress

Remarks

This section provides an overview of what devexpress is, and why a developer might want to use it.

It should also mention any large subjects within devexpress, and link out to the related topics. Since the Documentation for devexpress is new, you may need to create initial versions of those related topics.

Examples

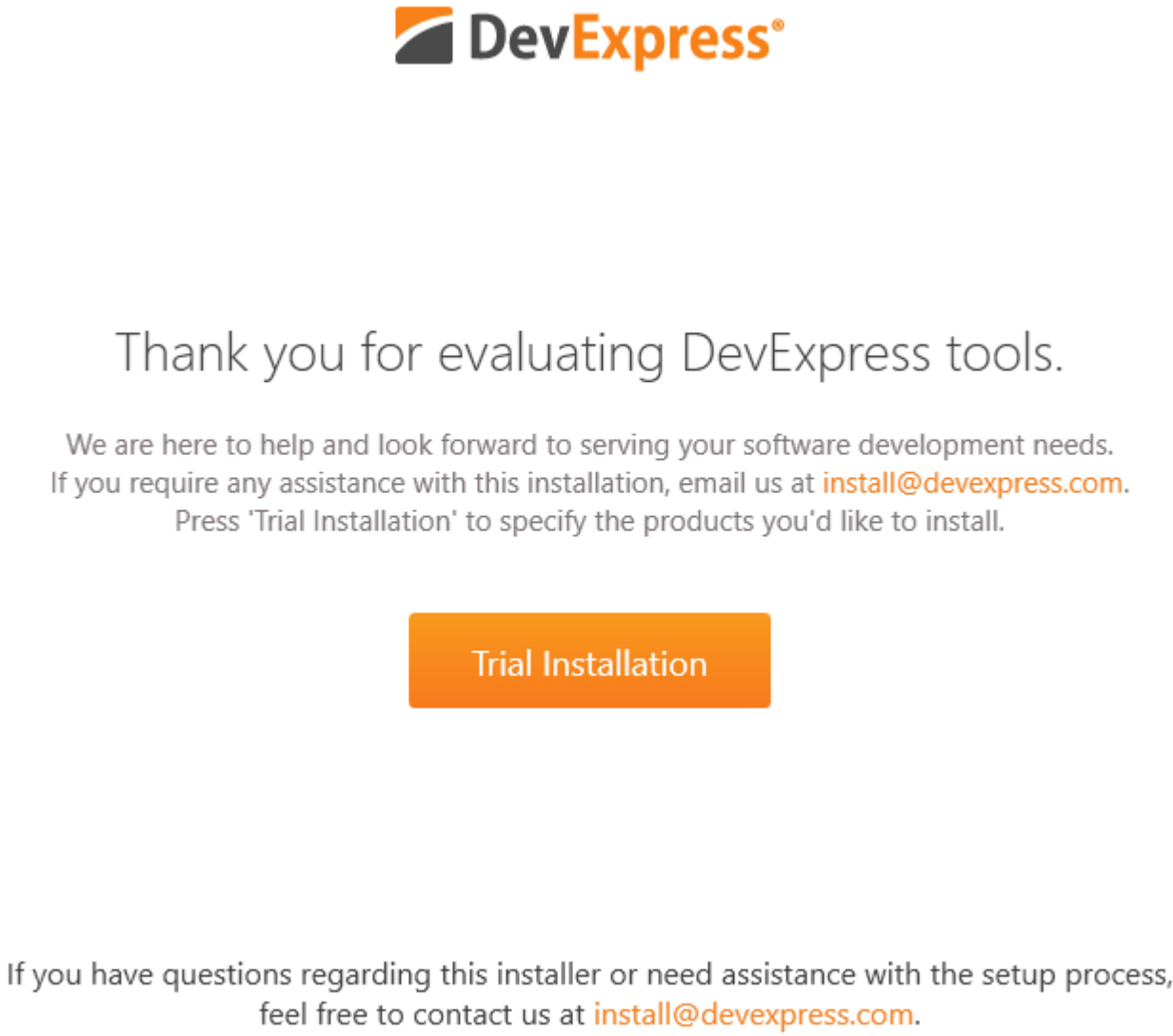
Downloading and installing the DevExpress .NET Products

To install DevExpress .NET products on your development machine, it is first necessary to obtain the **DevExpress .NET Products Installer**.
You can download it from the http://www.devexpress.com website in one of the following ways, based on the status of your current account.

- Download the Trial Version
  If you have not yet purchased DevExpress .NET products, but would like to try them, you are free to download the trial version.

- Download the Registered Version
  If you already have a license for DevExpress .NET products, you can download the registered version from the Client Center.

The **DevExpress .NET Products Installer** installs .NET assemblies, which contain the DevExpress visual controls and components, and integrates them into your Visual Studio IDE. In addition, this tool installs demo applications that illustrate the functionality provided by DevExpress .NET products.



You can run the installer in one of two modes.

- GUI Install Mode
  This is the graphical user interface (GUI) mode, in which you can go through the steps of the installation wizard and manually specify all install settings visually.

- Silent Install Mode
  This is the silent installation mode, which allows you to run the installer from a command line as a console application and specify all necessary parameters.

Read Getting started with devexpress online: https://riptutorial.com/devexpress/topic/5580/getting-started-with-devexpress

Chapter 2: Implementing conditional selection of Grid rows in DevExpress GridView

Remarks

Hope the sample illustration above helps someone who struggled like me to make rows conditionally select, as default functionality of DevExpress selects all rows irrespective whether it has a 'checkbox' or not (when you use either ASPxGridView1.SelectAllRowsOnPage() or ASPxGridView1.SelectRows(), because of the way 'Checkbox' column is implemented). Though there are various examples on DevExpress support site, I couldn't spot a working sample or scenario that explains this requirement.

Examples

Sample illustration using ASP.Net Web Forms

GridView definition on ASPX page

As shown below, first column of grid is defined as a checkbox column, which is conditionally cleared as shown in further examples below (the header checkbox is only for selecting/un-selecting all rows on current page, but same can be extended for all on grid easily):

```
<dx:ASPxGridView ID="ASPxGridView1" runat="server" AutoGenerateColumns="True"
ClientInstanceName="ASPxGridView1" KeyFieldName="CustomerID" Width="100%"
        OnHtmlRowCreated="ASPxGridView1_HtmlRowCreated"
OnCustomJSProperties="ASPxGridView1_CustomJSProperties">
    <SettingsPager PageSize="30" />
    <Settings VerticalScrollBarMode="Visible" VerticalScrollableHeight="350" />
    <Paddings Padding="0px" />
    <Border BorderWidth="0px" />
    <BorderBottom BorderWidth="1px" />
    <Columns>
        <dx:GridViewDataTextColumn Caption="#" VisibleIndex="0">
            <DataItemTemplate>
                <dx:ASPxCheckBox ID="cbCheck" runat="server" AutoPostBack="false"
CssClass="chkSelDgProdRow" OnLoad="ASPxGridView1_cbCheck_Load" />
            </DataItemTemplate>
            <HeaderTemplate>
                <dx:ASPxCheckBox ID="cbPageSelectAll" runat="server" ToolTip="Select/Unselect
all rows on the page" ClientInstanceName="cbPageSelectAll"
                    ClientSideEvents-CheckedChanged="function(s, e) {
checkUncheckSelectableRowsOnPage(s.GetChecked()); }"
OnLoad="ASPxGridView1_cbPageSelectAll_Load" />
            </HeaderTemplate>
        </dx:GridViewDataTextColumn>
        <dx:GridViewDataTextColumn FieldName="ContactName" VisibleIndex="2">
        </dx:GridViewDataTextColumn>
```

```
        <dx:GridViewDataTextColumn FieldName="CompanyName" VisibleIndex="1">
        </dx:GridViewDataTextColumn>
        <dx:GridViewDataTextColumn FieldName="ContactTitle" VisibleIndex="3">
        </dx:GridViewDataTextColumn>
        <dx:GridViewDataTextColumn FieldName="City" VisibleIndex="5">
        </dx:GridViewDataTextColumn>
        <dx:GridViewDataTextColumn FieldName="Country" VisibleIndex="6">
        </dx:GridViewDataTextColumn>
        <dx:GridViewDataTextColumn FieldName="Phone" VisibleIndex="9">
        </dx:GridViewDataTextColumn>
    </Columns>
    <ClientSideEvents BeginCallback="OnGridCallBackBegin" />
</dx:ASPxGridView>
```

SQL Table Structure

The data is loaded from SQL table 'Customers' whose structure is below (please note the table is standard structure from NorthWind database, having excess 'IsRegistered' column to demonstrate the 'conditional' selection functionality):

```
CREATE TABLE [dbo].[Customers] (
    [CustomerID]   NCHAR (5)     NOT NULL,
    [CompanyName]  NVARCHAR (40) NOT NULL,
    [ContactName]  NVARCHAR (30) NULL,
    [ContactTitle] NVARCHAR (30) NULL,
    [Address]      NVARCHAR (60) NULL,
    [City]         NVARCHAR (15) NULL,
    [Region]       NVARCHAR (15) NULL,
    [PostalCode]   NVARCHAR (10) NULL,
    [Country]      NVARCHAR (15) NULL,
    [Phone]        NVARCHAR (24) NULL,
    [Fax]          NVARCHAR (24) NULL,
    [IsRegistered] BIT           NULL
);
```

Page Load event (.CS Page)

The Page_Load() event & corresponding related code on .CS page. Please note the Custom JS properties for DevExpress needs to have a prefix of 'cp'.

```
    private const string _selectableRowsKey = "cp_SelectableRows";
    private const string _selectedRowsCountKey = "cp_SelectedRowsCount";

    protected void Page_Load(object sender, EventArgs e)
    {
        PopulateGrid();
    }

    private void PopulateGrid()
    {
        using (SqlConnection conn = new SqlConnection())
        {
            conn.ConnectionString =
ConfigurationManager.ConnectionStrings["NWindConnectionString"].ConnectionString;
            conn.Open();

            using (SqlCommand cmd = new SqlCommand("SELECT * FROM [Customers]", conn))
```

```
            {
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(cmd);
                da.Fill(dt);

                ASPxGridView1.DataSource = dt;
                ASPxGridView1.DataBind();
            }

            conn.Close();
        }
    }
```

HtmlRowCreated event (.CS page)

This event is invoked for each Row created on GridView upon data-binding. This is used to conditionally check if 'IsRegistered' is set to 'true', the 'checkbox' on first column is cleared. Also, the same is used to set a Custom JS property for the grid which will have # separated list of Rows that are allowed for selecting/un-selecting on UI.

```
    protected void ASPxGridView1_HtmlRowCreated(object sender, ASPxGridViewTableRowEventArgs
e)
    {
        if (e.RowType == GridViewRowType.Data)
        {
            if (e.GetValue("IsRegistered") != null)
            {
                if (e.GetValue("IsRegistered") != DBNull.Value &&
(bool)e.GetValue("IsRegistered"))
                    e.Row.Cells[0].Controls.Clear();
                else
                {
                    string selectableRows = string.Empty;
                    if (ASPxGridView1.JSProperties.ContainsKey(_selectableRowsKey))
                        selectableRows =
(string)ASPxGridView1.JSProperties[_selectableRowsKey];
                    selectableRows += "#" + e.VisibleIndex.ToString();
                    ASPxGridView1.JSProperties[_selectableRowsKey] = selectableRows;
                }
            }
        }
    }
```

Grid Checkbox Load Event (.CS Page)

This event is invoked for each checkbox when it's 'Load'ed on UI. This is used to set it's state as well as attach JavaScript code to select/un-select the row of the checkbox.

```
    protected void ASPxGridView1_cbCheck_Load(object sender, EventArgs e)
    {
        ASPxCheckBox cb = (ASPxCheckBox)sender;

        GridViewDataItemTemplateContainer container =
(GridViewDataItemTemplateContainer)cb.NamingContainer;
        cb.ClientInstanceName = string.Format("cbCheck{0}", container.VisibleIndex);
        cb.Checked = ASPxGridView1.Selection.IsRowSelected(container.VisibleIndex);
```

```
        cb.ClientSideEvents.CheckedChanged = string.Format("function (s, e) {{
ASPxGridView1.SelectRowOnPage({0}, s.GetChecked()); updateSelectedKeys(s.GetChecked()); }}",
container.VisibleIndex);
    }
```

Select all on Page checkbox on Load event (.CS Page)

This event is used to select/un-select 'Select all on Page' checkbox when navigating pages, based on their selection status.

```
    protected void ASPxGridView1_cbPageSelectAll_Load(object sender, EventArgs e)
    {
        ASPxCheckBox cb = (ASPxCheckBox)sender;
        ASPxGridView grid = (cb.NamingContainer as GridViewHeaderTemplateContainer).Grid;

        bool cbChecked = true;
        int start = grid.VisibleStartIndex;
        int end = grid.VisibleStartIndex + grid.SettingsPager.PageSize;
        end = (end > grid.VisibleRowCount ? grid.VisibleRowCount : end);

        for (int i = start; i < end; i++)
        {
            DataRowView dr = (DataRowView)(grid.GetRow(i));
            if (!grid.Selection.IsRowSelected(i))
            {
                if (dr["IsRegistered"] == DBNull.Value || !(bool)dr["IsRegistered"])
                {
                    cbChecked = false;
                    break;
                }
            }
        }
        cb.Checked = cbChecked;
    }
```

Grid CustomJS properties setup event (.CS Page)

This event is used to set 'Selected Rows Count' of current Page of GridView. This is used in JavaScript to auto-select header checkbox.

```
    protected void ASPxGridView1_CustomJSProperties(object sender,
ASPxGridViewClientJSPropertiesEventArgs e)
    {
        ASPxGridView grid = sender as ASPxGridView;
        int start = grid.VisibleStartIndex;
        int end = grid.VisibleStartIndex + grid.SettingsPager.PageSize;
        int selectNumbers = 0;

        end = (end > grid.VisibleRowCount ? grid.VisibleRowCount : end);

        for (int i = start; i < end; i++)
            if (grid.Selection.IsRowSelected(i))
                selectNumbers++;

        e.Properties[_selectedRowsCountKey] = selectNumbers;
    }
```

JavaScript code to handle client events and related functionality

This JavaScript function is invoked during 'checkbox' checked change event, as defined in ASPX page:

```
function checkUncheckSelectableRowsOnPage(isChecked) {
    var selectableRowIndexes = ASPxGridView1.cp_SelectableRows;
    var grdStartIndex = ASPxGridView1.visibleStartIndex;
    var grdEndIndex = grdStartIndex + ASPxGridView1.pageRowCount;

    if (selectableRowIndexes != null && selectableRowIndexes != '') {
        var rowIdxes = selectableRowIndexes.split("#");
        var selectedRowsCount = 0;
        if (rowIdxes != null) {
            try {
                for (var i = 0; i < rowIdxes.length; i++) {
                    if (rowIdxes[i] != "") {
                        var rowIndex = parseInt(rowIdxes[i]);
                        if (rowIndex != NaN && rowIndex >= 0 && rowIndex >= grdStartIndex &&
rowIndex < grdEndIndex) {
                            if (ASPxClientControl.GetControlCollection().GetByName("cbCheck" +
rowIdxes[i]) != null) {
                                if (isChecked) {
                                    ASPxGridView1.SelectRowOnPage(rowIdxes[i]);
                                    selectedRowsCount++;
                                }
                                else
                                    ASPxGridView1.UnselectRowOnPage(rowIdxes[i]);
                                ASPxClientControl.GetControlCollection().GetByName("cbCheck" +
rowIdxes[i]).SetChecked(isChecked);
                            }
                        }
                    }
                }
                //updateSelectedKeys();   // Can be used if the selected keys needs to be
saved separately in a Hidden field
                ASPxGridView1.cp_SelectedRowsCount = selectedRowsCount;
                currentSelectedRowsCount = selectedRowsCount;
            }
            finally {
            }
        }
    }
}
```

This JavaScript code is used to 'Select all on Page' checkbox at header, based on selected rows on current Page of GridView.

```
var currentSelectedRowsCount = 0;
function updateSelectedKeys(isChecked) {
    var selKeys = ASPxGridView1.GetSelectedKeysOnPage();
    if (selKeys != null) {
        var cpIDsList = '';
        try {
            for (var i = 0; i < selKeys.length; i++) {
                cpIDsList += selKeys[i] + ',';
            }
        }
```

```
        finally {
        }
        //$("#hdnSelectedCatProdIDs").val(cpIDsList);
        if (isChecked) {
            currentSelectedRowsCount++;
            cbPageSelectAll.SetChecked(selKeys.length == ASPxGridView1.cp_SelectedRowsCount);
        }
        else {
            cbPageSelectAll.SetChecked(selKeys.length == currentSelectedRowsCount);
            currentSelectedRowsCount--;
        }
    }
}
```

This JavaScript client event handler is used to set currently selected rows that are set from Server.

```
function OnGridCallBackBegin(s, e) {
    currentSelectedRowsCount = ASPxGridView1.cp_SelectedRowsCount;
}
```

Read Implementing conditional selection of Grid rows in DevExpress GridView online: https://riptutorial.com/devexpress/topic/8032/implementing-conditional-selection-of-grid-rows-in-devexpress-gridview

Credits

S. No	Chapters	Contributors
1	Getting started with devexpress	Ali.Rashidi, Community, DmitryG
2	Implementing conditional selection of Grid rows in DevExpress GridView	Kris