What's new in EhLib.VCL 9.5

(New features of your applications)

Brief description of new features in this version:

* SearchPanel - search and filter by location
* Advanced drop-down filtering in the TDBComboboxEh and TDBLookupComboboxEh components
* Other changes and additions.

# ***SearchPanel – Inplace search and filter***

A new search and filtering mode has been added to SearchPanel – Inplace search and filter (SearchPanel.Location = splCellInplaceEh).

In this mode, the search bar does not appear above the grid. Search is activated using the key combination Ctrl + F. Search and filter text is entered directly in the grid cell. As you enter text, the grid immediately searches for the first fragment found and, if the FilterOnTyping property is set, filtering is performed to display only the rows found.



This search mode is an advanced version of the OptionsEh.dghIncSearch search.

To enable inplace search and filter:

Set property SearchPanel.Enabled = True

Set the property SearchPanel.FilterOnTyping = True

To enable automatic filtering, set the SearchPanel.FilterOnTyping = True property.

If SearchPanel.PreferSearchToEdit property is set to True, then when user press the alphanumeric keys, the grid will immediately enter the search mode, and not open the cell text editor. The text editor can be opened by F2 or Enter.

In run-time, to enable the search mode, press the key combination - Ctrl + F.
To move to the next / previous found cell, use the Down / Up keys.
To exit the search mode, use the ESCAPE, ENTER or F2 keys.

# ***Advanced drop-down filtering in the TDBComboboxEh and TDBLookupComboboxEh components***

**Ability to filter the drop-down list when entering text in component В TDBComboboxEh**

TDBComboboxEh added the ability to filter a drop-down list when entering text. Similar filtering is already present in the TDBLookupComboboxEh component.



Properties added to TDBComboboxEh.DropDownBox:

**property** AutoFilter: Boolean;

Determines whether data filtering is active when entering text in the TDBComboboxEh editor.

**property** AutoFilterType: TLSAutoFilterTypeEh;

Specifies the type of filtering. lsftBeginsWithEh - filtering by the beginning of the cell text, lsftContainsEh - filtering from any position in the cell text.

Filtering works in the TDBComboboxEh component, as well as in DBGridEh columns when the PickList or PickList / KeyList properties are filled.

Use the TColumnEh.DropDownBox.ListSourceAutoFilter and TColumnEh.DropDownBox.ListSourceAutoFilterType properties to configure the filter in the grid columns.

**Advanced drop-down filtering when entering text in the TDBLookupComboboxEh component**

Added TDBLookupComboboxEh property TDBLookupComboboxEh.DropDownBox.ListSourceAutoFilterAllColumns of type Boolean.

The property allows you to filter data by all columns of the drop-down list.



The list of fields is usually specified by the TDBLookupComboboxEh.ListField property.

 ListField = 'Company;Addr1;City;Country'

You may also need the ability to enter text in TDBLookupComboboxEh without restricting the values you enter from only the list of values in the DisplayField field. To do this, set the Style = csDropDownEh property.

# ***Other changes and additions***

* **Event TPrintDBGridEh.OnPrintDataCell**

Added event OnPrintDataCell to TPrintDBGridEh. The event can be used to additionally draw the contents of the cell when printing the grid.

In the following example, the event handler draws a ProgressBar icon when printing a cell in the 'Population' field.

**procedure** TForm1.PrintDBGridEh1PrintDataCell(Sender: TPrintDBGridEh;

 VPrinter: TVirtualPrinter; Column: TColumnEh; **const** ARect: TRect;

 **var** Params: TPrintColCellParamsEh; **var** Processed: Boolean);

**begin**

 **if** (Column.FieldName = 'ProgressBar') **then**

 **begin**

 DrawProgressBarEh(MemTableEh1.FieldByName('Population').AsFloat, 0, 50000000,

 VPrinter.Canvas, ARect, clLtGray, cl3DDkShadow, clNone);

 Processed := True;

 **end**;

**end**;

* **Event TMemTableEh.OnRecordsViewTreeNodeCollapsed**

Added the OnRecordsViewTreeNodeCollapsed event to TMemTableEh. The event is triggered when a tree branch is collapsed in TreeVIew mode.

* **Encoding during export using TDBGridEhExportAsText.**

Added WriteBOM and Encoding properties to TDBGridEhExportAsText class.

 **property** WriteBOM: Boolean;

Specifies the need to write the Byte Sequence Marker (BOM) attribute when exporting to a stream.

 **property** Encoding: TEncoding;

Specifies the encoding of text information when importing a grid into a stream.

The following example shows code for exporting the contents of DBGridEh1 to a UTF8 file and writing a BOM token.

**procedure** TForm1.Button1Click(Sender: TObject);

**var**

 Path: String;

 gridExp: TDBGridEhExportAsText;

**begin**

 GetDir(0, Path);

 Path := Path + '\ExpFile.Txt';

 gridExp := TDBGridEhExportAsText.Create;

 gridExp.DBGridEh := DBGridEh1;

 gridExp.Encoding := TEncoding.UTF8;

 gridExp.WriteBOM := True;

 **try**

 gridExp.ExportToFile(Path, True);

 **finally**

 gridExp.Free;

 **end**;

**end**;

* **- Fine-tuning the formation of the DBGridEh.STFilter filter string for the DataSet’s through the TDatasetFeaturesEh component.**

TDatasetFeaturesEh added virtual methods for fine tuning when forming a filter string based on the filter set in DBGridEh.STFilter.

To form and apply a filter from DBGridEh.STFilter in the DataSet, the ApplyFilter method is used. In previous versions of the library, the ApplyFilter method called global functions to form a filter string and apply a filter. However, the global functions did not take into account some features of some DataSets.

In the current version, the following virtual functions are added to TDatasetFeaturesEh, which allow you to fine-tune the filter string and the method of applying the filter to the DataSet:

 **procedure** ApplyGridLocalFilter(Grid: TCustomDBGridEh; DataSet: TDataSet; IsReopen: Boolean); virtual;

The method is called when it is necessary to set the local filter in the DataSet (DBGridEh.STFilter.Local = True). Typically, with local filtering, the filter string is assigned to the DataSet.Filter property.

 **function** GetGridFilterAslFilterString(Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method forms and returns a filtering string based on the filter in DBGridEh. The filter is stored in the properties of the TColumnEh.STFilter column.

The IsLocalFilter property determines that it is necessary to form an expression for local filtering.

The method should loop through the columns of the grid and call GetColumnExpressionAsFilterString for each column.

 **function** GetColumnExpressionAsFilterString(Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method generates and returns a filter string for one column of TColumnEh.

 **function** GetBinaryExpressionAsFilterString(FieldName: String; AnOperator: TSTFilterOperatorEh; AnOperand: Variant; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method generates and returns a filter string for one TColumnEh column based on the values of the FieldName, AnOperator, AnOperand parameters.

 **function** GetColumnFilterFieldName(Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns the correct name of the field that will participate in the formation of the filtering expression.

 **function** ColumnOperatorValueToFilterStrValue(AnOperator: TSTFilterOperatorEh; AnOperand: Variant; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns a text expression based on the value of the AnOperator and AnOperand fields.

 **function** IsInOperatorSupported(Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): Boolean; virtual;

The method returns True if the DataSet and filtering method supports the In statement.

 **function** IsLikeOperatorSupported(Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): Boolean; virtual;

The method returns True if the DataSet and filtering method supports the Like operator.

 **function** IsFilterUseFieldOrigin(Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): Boolean; virtual;

The method returns True if Field.Origin should be used as the field name.

 **function** GetNullComparisionFilterString(AnOperator: TSTFilterOperatorEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns a text expression to compare the field with Null. IS NULL or = Null. IS NOT NULL or <> Null

 **function** GetOperatorFilterStrValue(AnOperator: TSTFilterOperatorEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns a textual representation of the comparison operator specified by AnOperator parameters.

 **function** FilterFieldNameToStrValue(FieldName: String; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns the text value of the field name. Some DataSets for local filtering require the field to be enclosed in square brackets - ‘[FieldName]’.

 **function** ExpressionValueToFilterStrValue(AnOperator: TSTFilterOperatorEh; Value: Variant; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns the text value of the expression for the section of the Value expression.

 **function** DateTimeValueToFilterStrValue(AnOperator: TSTFilterOperatorEh; Value: TDateTime; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns the text value of the expression for a constant of type DateTime.

 **function** FloatValueToFilterStrValue(AnOperator: TSTFilterOperatorEh; Value: Extended; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns the text value of the expression for a constant of type Float.

 **function** VarValueToFilterStrValue(AnOperator: TSTFilterOperatorEh; Value: Variant; Column: TColumnEh; Grid: TCustomDBGridEh; DataSet: TDataSet; IsLocalFilter: Boolean): String; virtual;

The method returns the text value of the expression for a constant of type String.

 **procedure** ApplyGridServerFilter(Grid: TCustomDBGridEh; DataSet: TDataSet; IsReopen: Boolean); virtual;

The method is called when it is necessary to set the server filter in the DataSet (DBGridEh.STFilter.Local = False).