**Laboratory work No. 5. Displaying text information**

1. Displaying text information in the Label control

To display text information, the Label and TextBox controls are most often used.

Note that not only text can be placed in the control, but, which is not in Delphi and Borland C ++ Builder, and pictures, it has a set of properties (Image, ImageAlign, ImageList).

Displaying text in a control does not cause any difficulties, for this control has a Text property (it is of the string type, to display variables of other types, you need to call the ToString () method):

label1.Text = "My Text";

2. Displaying text information in the TextBox controlTextBox control

We have so far used thefor input and output of single-line text, but it is more intended specifically for input and output of multi-line text. To do this, it has aproperty **Multiline**, which must be set to **True**.

The **AcceptsTab** and **AcceptsReturn properties** determine the action when the Tab and Enter keys are pressed, with False (by default) and pressing Tab or Enter, the focus moves to the next control by TabIndex, with True - the keys perform their main purpose - inserting a tab break and line feed, respectively ...

Theproperty **AutoSize** determines whether the control should be automatically resized to fit its contents.

Theproperty **CharacterCasing** determines the case in which the text will be displayed in the control.

property **It isHideSelection** advisable to set theto False - otherwise the programmatically selected text will not be displayed.

Theproperty **Locked** determines whether the user can change the properties of the TextBox programmatically.

Theproperty **ScrollBars** shows the position of the scrollbars - with Both, the scrollbar appears when the text goes beyond the corresponding border.

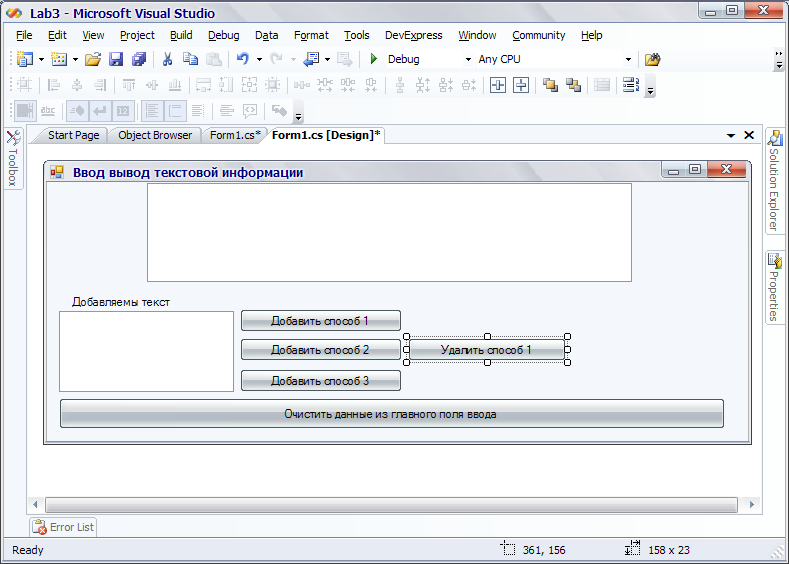
Theproperty **PasswordChar** (works only if Multiline = false) does not allow viewing the entered characters, replacing them with the usual asterisks for entering a password or another character specified in this field.

Theproperty **WordWrap** if True allows text to wrap to a new line when the text line is wider than the width of the window.

Displaying text in a control does not cause any difficulties, for this control has a Text property (has the string type, to display variables of other types, you must call the ToString () method):

textBox1.Text = "Text";

Create a shape



Create a Click event on the buttons:

Button Add method 1:

textBox1.Text = textBox1.Text + "" + textBox2.Text;

in this line, we assign the concatenated string.

Button Add method 2:

textBox1.AppendText (textBox2.Text); // The function adds text to the end of the line textBox1 without spaces and other separating characters change textBox2.Text + ""

Button Add method 3:

textBox1.Lines = new string [] {" Line 1 "," Line 2 "};

Adds an array of strings. The Lines property contains an array of strings.

Button Clear data from the main field input:

textBox1.Text = ""; // in this line we assign an empty line.

Button Delete method 1

textBox1.SelectAll (); // select all text

textBox1.Cut () ; // cut out what was selected

3. Displaying text information in the RichTextBox control

**Main properties of RichTextBox**

Theproperty **AcceptButtonyou** allowsto fix the button, which will be clicked when you press Enter when the RichTextBox is in focus (for example, after typing).

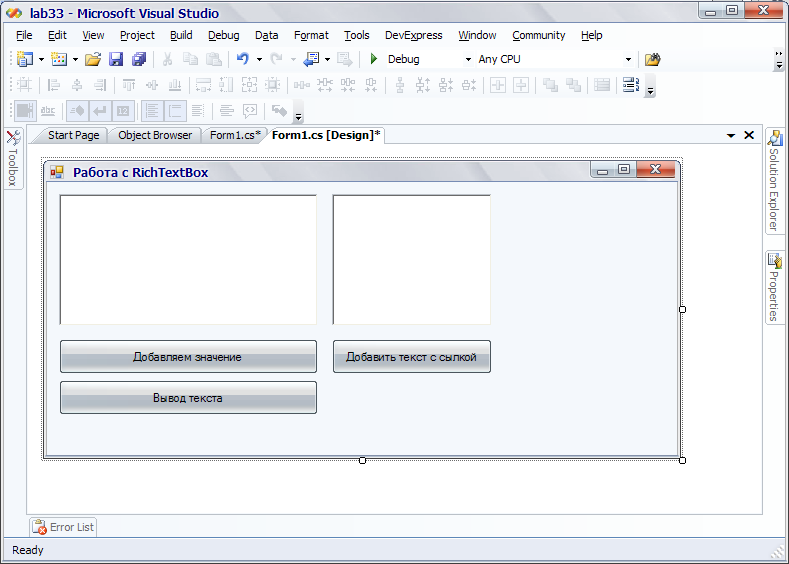
**WordSelection** if the property is true, the user can double-click on any part of the word in the control to make it selected.

**BulletIndent and SelectionBullet** - BulletIndent defines the indent in pixels of the text of enumerations (list items) from the icon corresponding to the list item when the SelectionBullet property is set to true.

The Text property is used to display the text.

Theproperty **DetectUrlsyou** allowsto recognize and highlight Internet links and launch a browser using this link if there is a LinkClicked event handler in the solution.

Example of work



Button Add value

// Clear all

richTextBox1.Clear ();

richTextBox1.SelectionBullet = true;

// Set indentation in pixels for list items

richTextBox1.BulletIndent = 50;

richTextBox1.SelectionFont = new Font ("Arial", 16);

// Display text that is not affected by the set

// indentation (SelectionBullet = false)

richTextBox1.SelectedText = "Below is a list \ n";

richTextBox1.SelectionFont = new Font ("Arial", 12);

// Next is a list

richTextBox1.SelectionBullet = true;

richTextBox1.SelectionColor = Color.Red;

richTextBox1.SelectedText = "Apple" + "\ n";

richTextBox1.SelectionFont = new Font ("Arial", 12);

richTextBox1.SelectionColor = Color.Orange;

richTextBox1.SelectedText = "Orange" + "\ n";

richTextBox1.SelectionFont = new Font ("Arial", 12);

richTextBox1.SelectionColor = Color.Purple;

richTextBox1.SelectedText = "Grapes" + "\ n";

richTextBox1.SelectionBullet = false;

richTextBox1.SelectionFont = new Font ("Verdana", 10);

richTextBox1.SelectedText = "The list is over \ n";

Button Text output

int viNom = 0;

// Text output in one line

richTextBox1.Text + = "Text" + (Convert.ToInt32 (viNom)). ToString ();

viNom ++;

//// The first way to display multi-line text

richTextBox1.Text + = "String" + (Convert.ToInt32 (viNom)). ToString () + "\ r \ n";

viNom ++;

richTextBox1.Text + = "String" + (Convert.ToInt32 (viNom)). ToString () + "\ r \ n";

//// The second way to display multi-line text

richTextBox1.AppendText ("String" + Convert.ToString (viNom) + "\ r \ n");

viNom ++;

richTextBox1.AppendText ("String" + Convert.ToString (viNom) + "\ r \ n");

//// the third way to display multi-line text

// richTextBox1.Lines = new string [] {"Line 1", "Line 2"};

Button Add text with link

RichTextBox2.Text = "My site http: \\ CSarp.kg";

Link click event:

private void richTextBox2\_LinkClicked (object sender, LinkClickedEventArgs e)

{

MessageBox.Show ("Clicked on links");

System.Diagnostics.Process.Start (e.LinkText); // line for the transition

}

System.Diagnostics.Process.Start - we start the process, you can specify the path in the parameters and start the file.