Windows Forms – Introduction, Menus, Events

Contents

[1. Application Class 1](#_Toc448303092)

[2. Form Class and Control Class 1](#_Toc448303093)

[2.1. Partial classes 1](#_Toc448303094)

[2.2. Properties 2](#_Toc448303095)

[3. Menus 2](#_Toc448303096)

[4. Mouse and Keyboard Events 2](#_Toc448303097)

[4.1. Mouse Events 2](#_Toc448303098)

[4.2. Keyboard Events 4](#_Toc448303099)

# Application Class

Assignment

1. Create a new project with the name “MouseEvents”
2. Change the name of the default form to “MainForm”

# Form Class and Control Class



## Partial classes

It is possible to split the definition of a [class](https://msdn.microsoft.com/en-us/library/0b0thckt.aspx) or a [struct](https://msdn.microsoft.com/en-us/library/ah19swz4.aspx), an [interface](https://msdn.microsoft.com/en-us/library/87d83y5b.aspx) or a method over two or more source files. Each source file contains a section of the type or method definition, and all parts are combined when the application is compiled.

Assignment

1. Open the “\*.Designer.cs” file associated to the form

## Properties

Frequently used properties:

* **Design:** Name, etc.
* **Appearance:** Text, ForeColor, BackColor, Font, etc.

# Menus

Assignment

|  |  |
| --- | --- |
|  | Sample code available at <http://online.ase.ro> – “SimpleWebBrowser” Sample |

1. Create the following application



1. Create a setup file for the application, that can be redistributed to other users

|  |
| --- |
| You need to right click on the project in the “Solution Explorer” window and choose the publish option |

# Mouse and Keyboard Events

## Mouse Events

|  |  |  |
| --- | --- | --- |
| Event | Event argument | Description |
| Click | EventArgs | Raised when the control is clicked. |
| DoubleClick | EventArgs | Raised when the control is double-clicked. |
| MouseEnter | EventArgs | Raised when the mouse cursor enters the control. |
| MouseHover | EventArgs | Raised when the mouse cursor hovers over the control. |
| MouseLeave | EventArgs | Raised when the mouse cursor leaves the control. |
| MouseDown | MouseEventArgs | Raised when the mouse cursor is over the control and a mouse button is pressed. |
| MouseMove | MouseEventArgs | Raised when the mouse cursor is moved over the control. |
| MouseWheel | MouseEventArgs | Raised when the control has focus and the mouse wheel is rotated. |
| MouseUp | MouseEventArgs | Raised when the mouse cursor is over the control and a mouse button is released. |

|  |  |
| --- | --- |
|  | [Click](https://msdn.microsoft.com/en-us/library/system.windows.forms.control.click%28v%3Dvs.110%29.aspx) events are logically higher-level events of a control. They are often raised by other actions, such as pressing the ENTER key when the control has focus. |

Depressing a mouse button when the cursor is over a control typically raises the following series of events from the control:

* MouseDown event.
* Click event.
* MouseClick event.
* MouseUp event.

Assignment

|  |  |
| --- | --- |
|  | Sample code available at <http://online.ase.ro> – “MouseEvents” Sample |

1. Create the following UI



1. Add the following code

|  |
| --- |
|  **public** partial class MainForm **:** Form**{** **public** MainForm**()** **{** InitializeComponent**();** **}** **private** void btnReset\_Click**(object** sender**,** EventArgs e**)** **{** lbl**.**Text **=** ""**;** txt**.**Text **=** ""**;** **}** //similar: lbl\_MouseHover, lbl\_MouseLeave, lbl\_Click, lbl\_DoubleClick **private** void lbl\_MouseEnter**(object** sender**,** EventArgs e**)** **{** lbl**.**Text **=** "MouseEnter"**;** TextBoxDraw**(**"###Label MouseEnter"**);** **}** //similar: lbl\_MouseMove, lbl\_MouseUp, lbl\_MouseWheel **private** void lbl\_MouseDown**(object** sender**,** MouseEventArgs e**)** **{** lbl**.**Text **=** "MouseDown"**;** var str **=** "###Label MouseDown"**;** str **+=** Environment**.**NewLine **+** "Button: " **+** e**.**Button**;** str **+=** Environment**.**NewLine **+** "Clicks: " **+** e**.**Clicks**;** str **+=** Environment**.**NewLine **+** "Delta: " **+** e**.**Delta**;** str **+=** Environment**.**NewLine **+** "X: " **+** e**.**X**;** str **+=** Environment**.**NewLine **+** "Y: " **+** e**.**Y**;** TextBoxDraw**(**str**);** **}** //similar: OnMouseHover, OnMouseLeave **protected** **override** void OnMouseEnter**(**EventArgs e**)** **{** **base.**OnMouseEnter**(**e**);** TextBoxDraw**(**"###Form MouseEnter"**);** **}** **private** void TextBoxDraw**(**string str**)** **{** txt**.**AppendText**(**Environment**.**NewLine **+** str**);** **}****}** |

## Keyboard Events

|  |  |  |
| --- | --- | --- |
| **Event** | **Event data** | **Description** |
| KeyDown | KeyEventArgs | Raised when a key is pressed. The KeyDown event occurs prior to the KeyPress event. |
| KeyPress | KeyPressEventArgs | Raised when a character generating key is pressed. The KeyPress event occurs after the KeyDown event and before the KeyUp event. |
| KeyUp | KeyEventArgs | Raised when a key is released. |

Assignment

We want to create a numeric only TextBox that can be used to build a simple calculator application, such as the one bellow.



|  |  |
| --- | --- |
|  | Sample code available at <http://online.ase.ro> – “KeyEvents” Sample |

1. Create a new project with the name “KeyEventsNumericTextBox”
2. Add a TextBox and name it “tbNumericTextBox”
3. Add the following event handler

|  |
| --- |
| **private** void tbNumericTextBox\_KeyPress**(object** sender**,** KeyPressEventArgs e**)****{** OnKeyPress**(**e**);** **if** **(!**char**.**IsDigit**(**e**.**KeyChar**))** **{** // Consume this invalid key e**.**Handled **=** **true;** **}****}** |

1. Change the previous event handler in order to allow the use of the “BackSpace” key.
2. Change the previous event handler in order to allow the digit separator for the current culture.