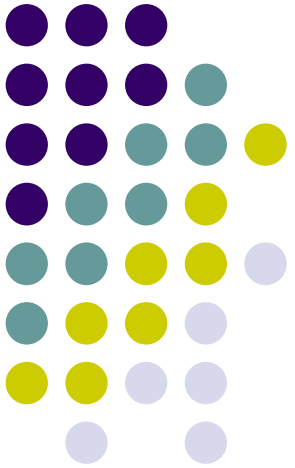
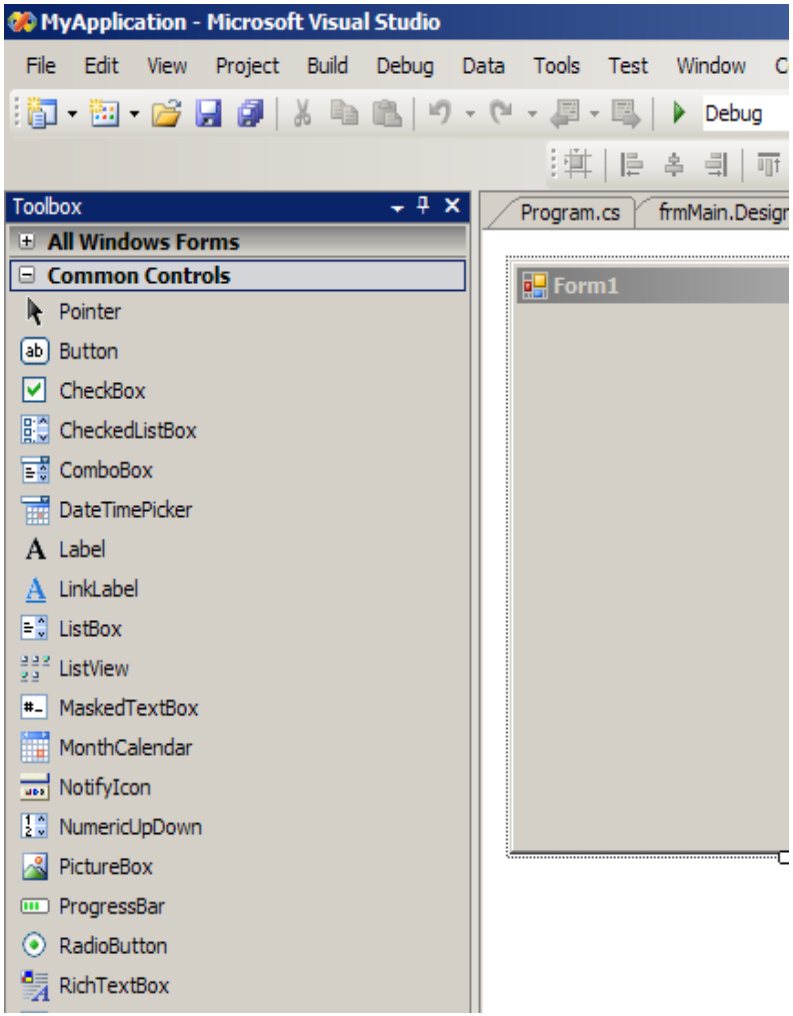
Controls



Adding Controls to Form

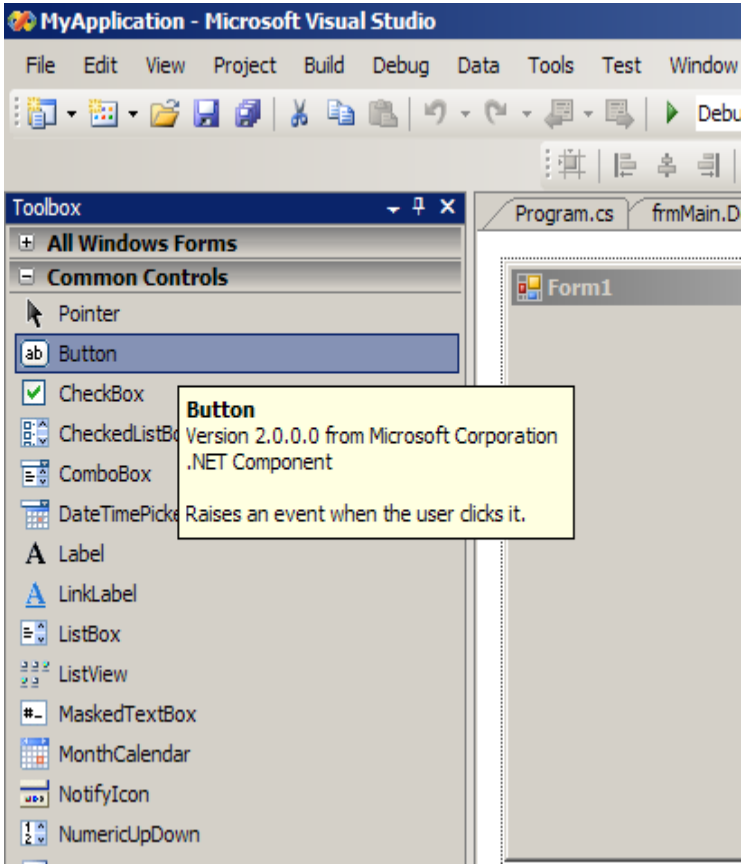


-You can pick controls from the toolbox.

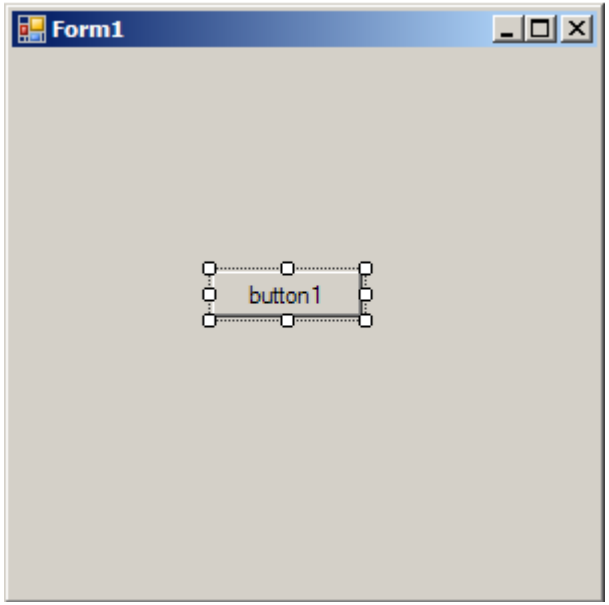
-To add the controls from Toolbox to the Form You have be in design view.

-To add the controls to the Form, simply drag and drop controls.

Adding Button Control to Form

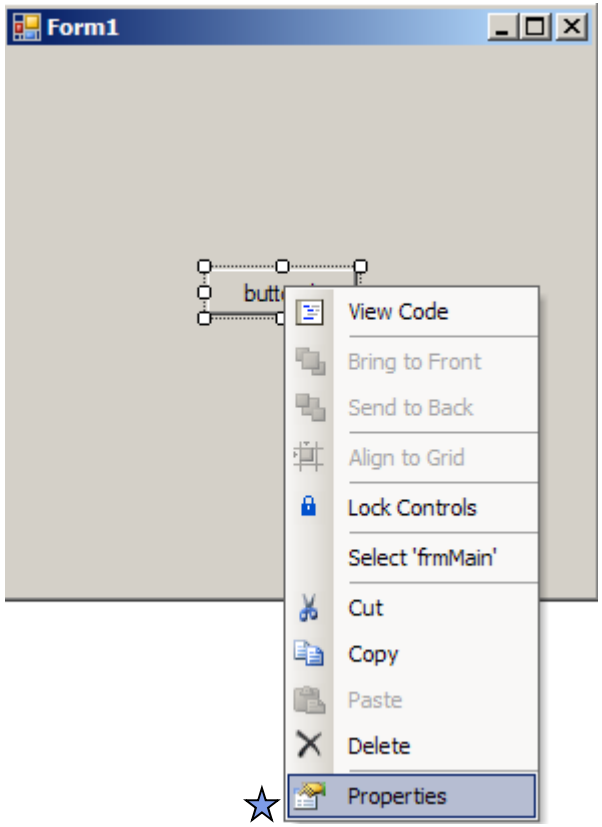


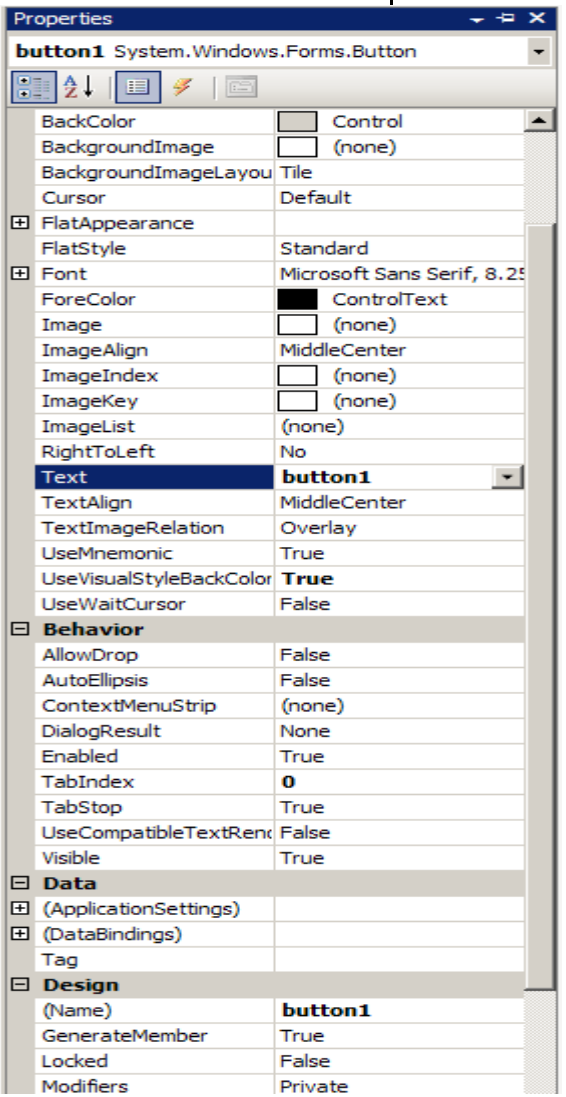
Drag and Drop



Adding Button Control to Form

To open properties of the button; right click on the button and choose properties.





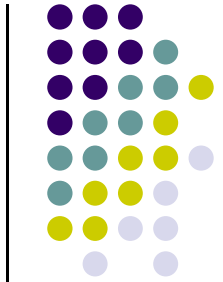
Fundamental Properties of the Button

- BackColor
- Cursor
- FlatStyle
- Font
- ForeColor
- Image
- ImageAlign
- Text
- TextAlign
- Enabled
- Visible
- Name
- Anchor
- Location
- Size

Writing Code for Button

- Every control has default event.
- Default event for Button control is Click.
- We have to write the code in to the handler funtion which will be executed when button is pressed.
- To create handler function, simply double click to the button while you are in the designer view.

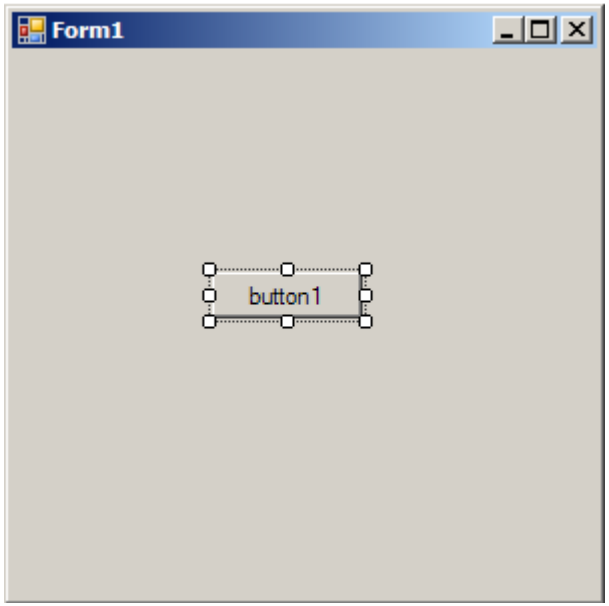
Writing Code for Button



Double Click

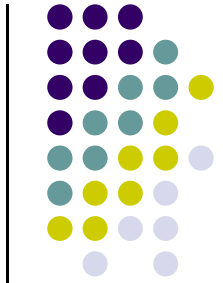
Designer View

Generated code



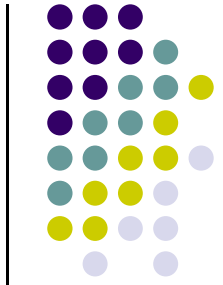

```
namespace MyApplication
{
    public partial class frmMain : Form
    {
        public frmMain()
        {
            InitializeComponent();
        }

        private void btnMerhaba_Click(object sender, EventArgs e)
        {

        }
    }
}
```

Writing Code for Button



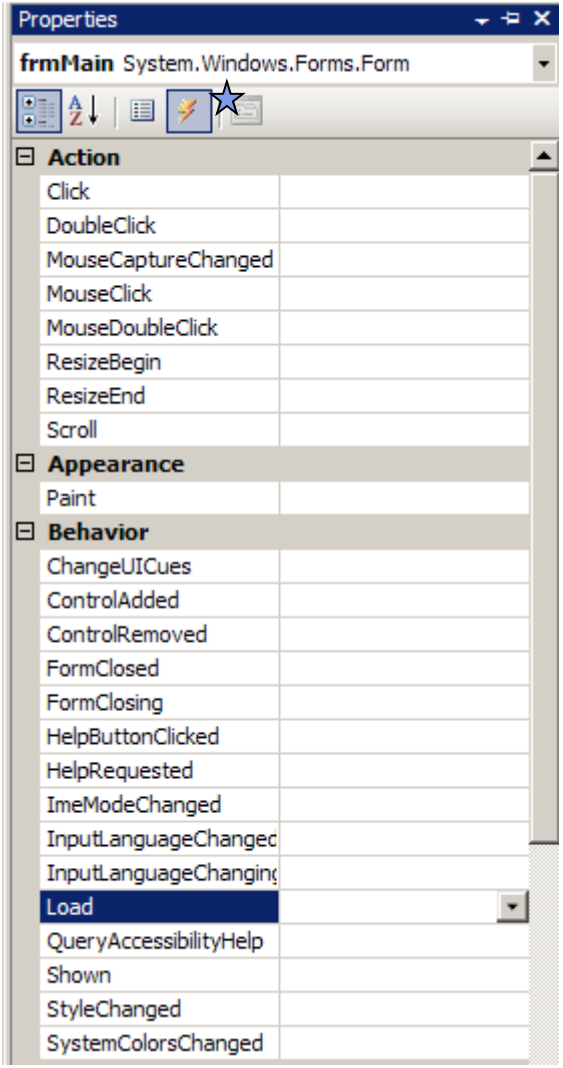
- private void btnHello_Click(object sender, System.EventArgs e) ????? !!!!
- “btnHello_Click” function is generated by using both controls name and events name.
- We should write the codes in the that function which are meant to be executed when button clicked.
- Event handler function takes parameters which includes useful information about the fired event.
- These parameters are set by the system during the running the runtime.
- Handler function is executed when user clicks the button at the runtime.

More Events



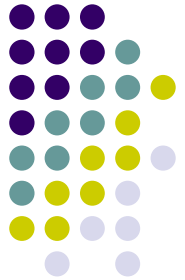
- Events are the way of notifying fro a control when something of interest happened.
- Some events are exclusive to a control, some can be found in almost every control.
- Press events button on the Properties window to access and manage events that are assigned to a control in a Form.

More Events



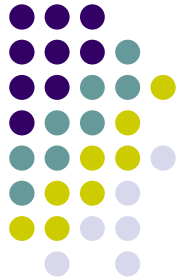
To create a handler function to an event of the control, just double click to the empty area at the right of the event.

Fundamental Events of the Form



- Click
- DoubleClick
- Closed
- Closing
- Focus Events
- Mouse Events

Fundamental Events of the Button



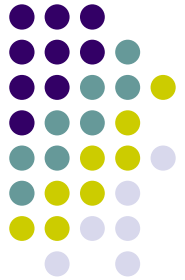
- Click
- Focus Events
- Mouse Events

Writing code for Button Control (Continues)

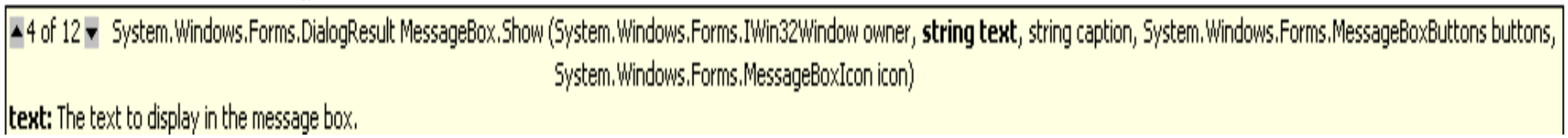
- Lets make program to give a message when Button is pressed.

- To achieve this first examine MessageBox class.

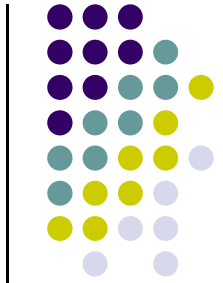
MessageBox



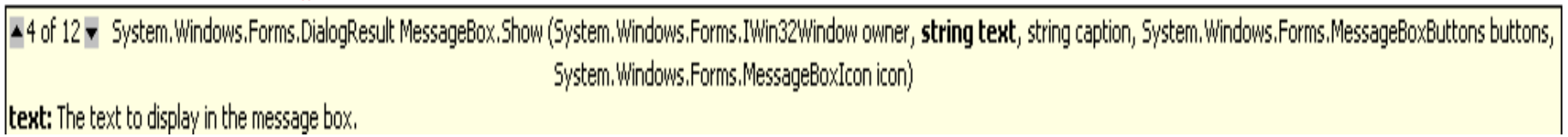
- MessageBox class is used for giving information to the user during execution of the program.
- A message could be information, error, warning or question.
- To display MessageBox, we should call Show method.



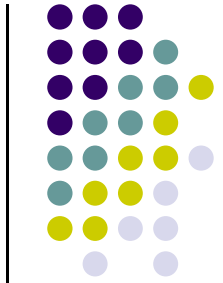
MessageBox



- Parameters
  1. The control which MessageBox belongs to
  2. Message body
  3. Header
  4. Button options
  5. Icon options



MessageBox



```
public partial class frmMain : Form
{
    public frmMain()
    {
        InitializeComponent();
    }

    private void btnMerhaba_Click(object sender, EventArgs e)
    {
        MessageBox.Show(this, "Merhaba dünya!", "Selam", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
```

Exercise

- Try different combinations of MessageBoxButtons and MessageBoxIcon enumerations.

Exercise

- When user tries to exit the program, usually programs ask to the user that whether he is sure.

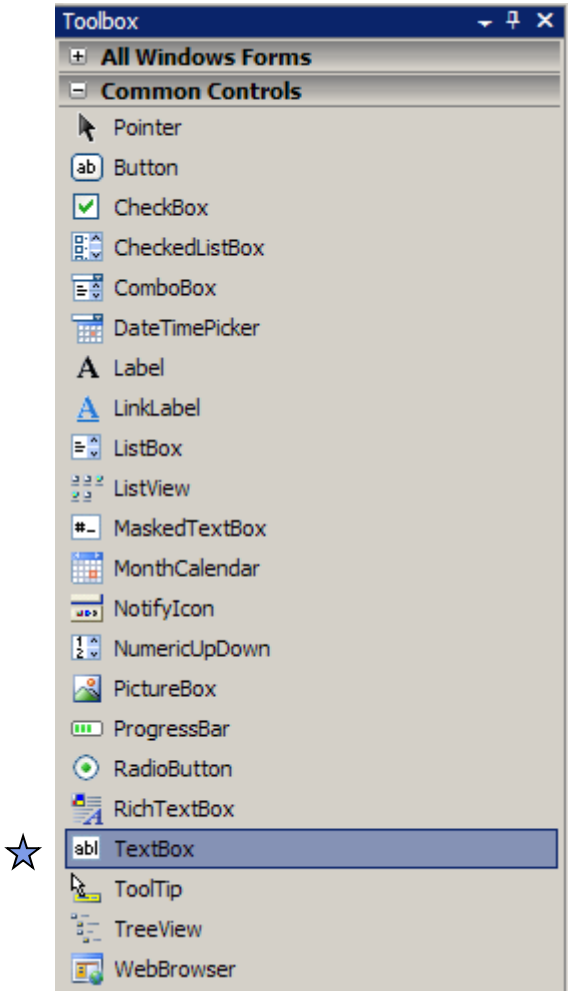
- Write a program that asks to the user “Are you sure?” when form is closing.

- Hint: Try to use MessageBox control in the event which is fired during Form is closing.

TextBox

-Used for to show information, and take input from user.

- Data is stored in the Text property of the control.



Fundamental Properties of the TextBox

- BackColor
- Font
- ForeColor
- Text
- TextAlign
- MaxLength
- MultiLine
- PasswordChar
- ReadOnly
- Visible
- Name
- Anchor
- Dock
- Location
- Size

Fundamental Events of the TextBox

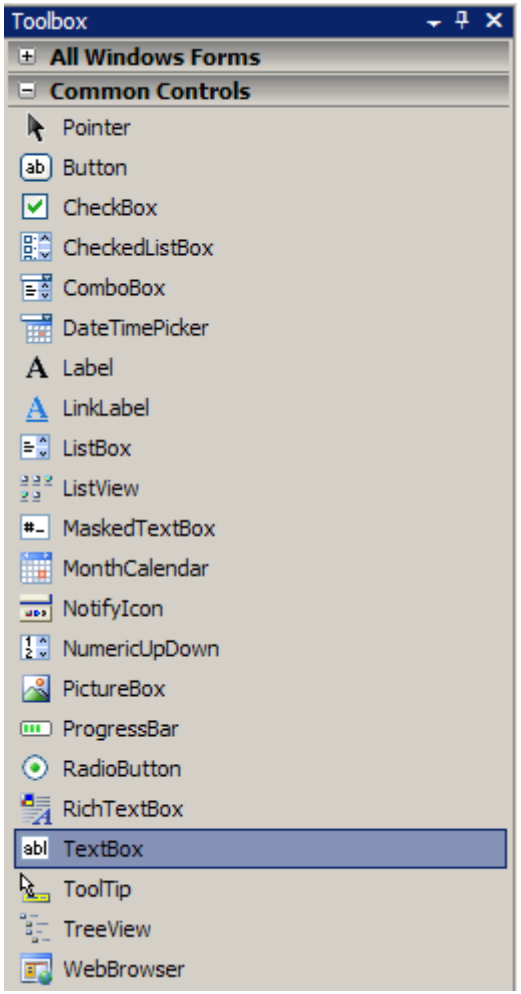
- DoubleClick
- Key Events
- Mouse Events

Adding TextBox Control to the Form

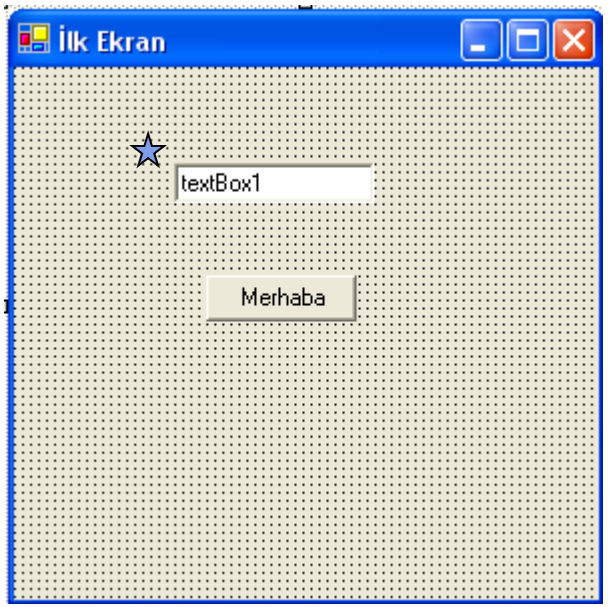
- As in the Button control, simply drag&drop TextBox control from toolbox to the Form

- Note: You may change location and size of the controls you have added in the designer view.

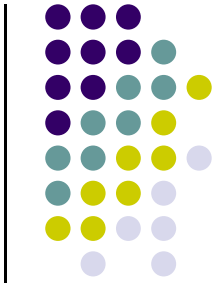
Adding TextBox Control to the Form



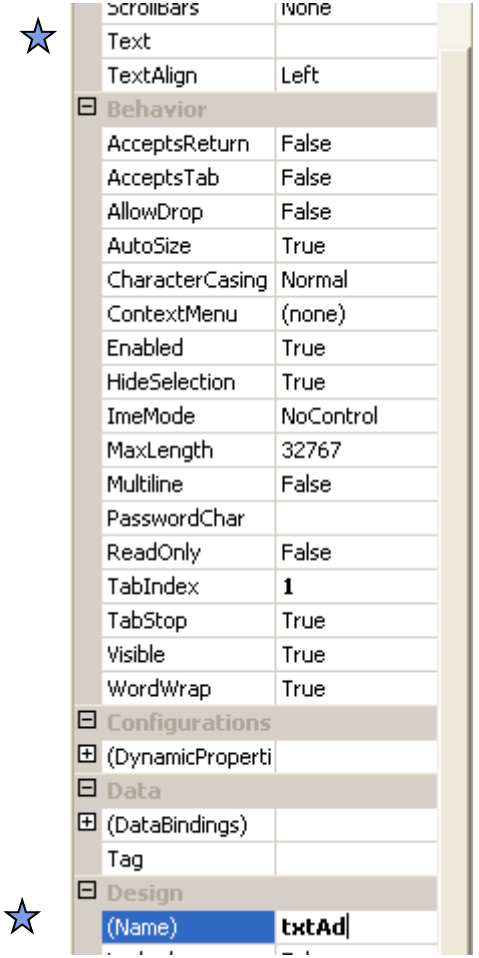
Drag&Drop



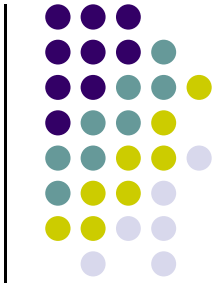
Adding TextBox Control to the Form



- Set the name property of the text control to the “txtAd”.



Adding TextBox Control to the Form



- Change the click event handler function of the Button as below

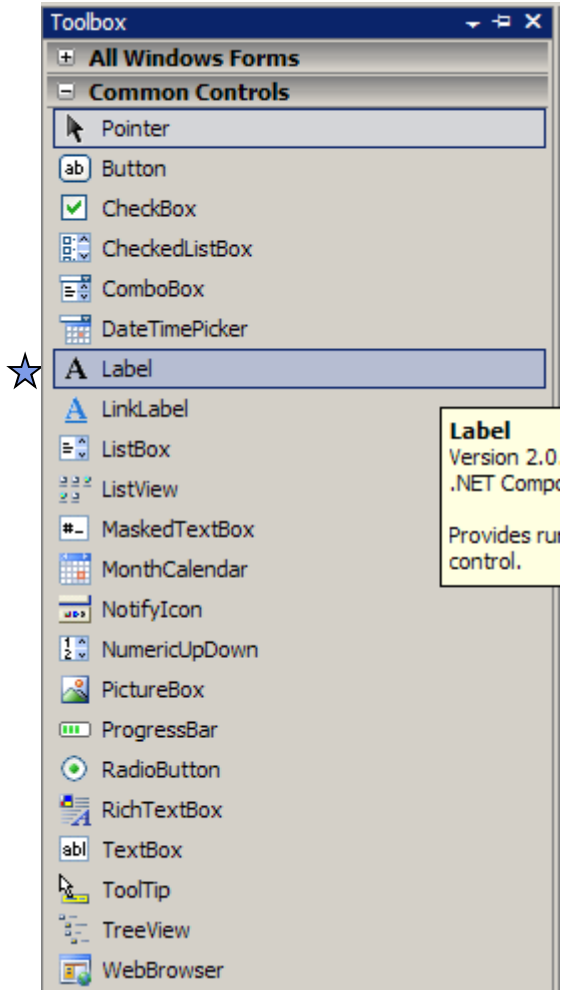
```
public partial class frmMain : Form
{
    public frmMain()
    {
        InitializeComponent();
    }

    private void btnMerhaba_Click(object sender, EventArgs e)
    {
        MessageBox.Show(this, "Merhaba, benim adım " + txtAd.Text , "Selam", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
```

- Run the program and enter your name the TextBox.

- And than click hello button.

Label Control

- Looks like TextBox control
- Used for making explanations about other controls and giving information to the user.
- User can not enter input to Label control. It is read-only for the user.



Fundamental Properties of the Label

- BackColor
- Font
- ForeColor
- Text
- TextAlign
- Location
- Size

Exercise

- Write a program that asks user for his/her name and surname. Than program will write these values to the message box when user clicks the button. If name or surname entry is missing, program should warn the user.