GUI Programming Using C#

Windows Forms Programming

- System.Drawing Namespace
  - Basic GDI+ functionality
- System.Windows.Forms Namespace
  - Creating forms applications by hand
  - Creating forms applications using Visual Studio designer
  - Higher-level controls

Form Class

- This is the top-level window class
- This class contains all other controls
- Normally, your top-level form inherits from the Form class
- Although the class has numerous methods, most of the time you interact with it via properties and delegates

Form Properties

Property	Description
Location	Point of to left corner
Size	Size of form in pixels
Text	Text displayed or caption
AutoScaleDimensions	DPI resolution of display it was built for. Will be scaled to look correct on other displays.
BackColor	Background color
ForeColor	Foreground or drawing color
ClientSize	Size of drawing area without borders or scrollbars
Controls	A collection of controls owned by the form
WindowState	Whether maximized, minimized or normal
DefaultSize	Size when initially created
MinimumSize	Minimum size window can be resized to
MaximumSize	Maximum size window can be resized to

Form Methods

Method	Description
Activate	Activates the window and gives it focus
Close	Closes the form
Show	Makes the form visible
BringToFront	Moves to top of stacking order
Hide	Makes the form invisible
Focus	Gives the form focus

Form Events

- Forms provide support for a large number of events
- You add one or more delegates to these events
- When the event happens, the delegates are invoked
- The delegates must have the signature of an event handler

```
void EventHandler(object sender,
  EventArgs e)
```

Form Events

Event	Description
Load	Just before form is loaded the first time
Closing	Just before the form is closed
Closed	When the form is actually closed
Shown	Occurs when a form is first displayed
ResizeBegin	Resize operation has begun
ResizeEnd	Resize operation has ended

Visual Studio Designer

- This is a drag and drop interface for drawing a GUI
- The code is automatically generated
- You can hook event handlers onto the events and write the code for them
- It speeds writing code
- You cannot make major modifications to the code it generates

Text Box

- This is a single line or multi-line text editor
  - `Multiline` – get/set Boolean to make multiline
  - `PasswordChar` – if this is set to a char, then the box becomes a password box

Text Box

- `ReadOnly` – if true, the control is grayed out and will not accept user input
- `TextAlign` – get/set HorizontalAlignment.Left, Center, or Right

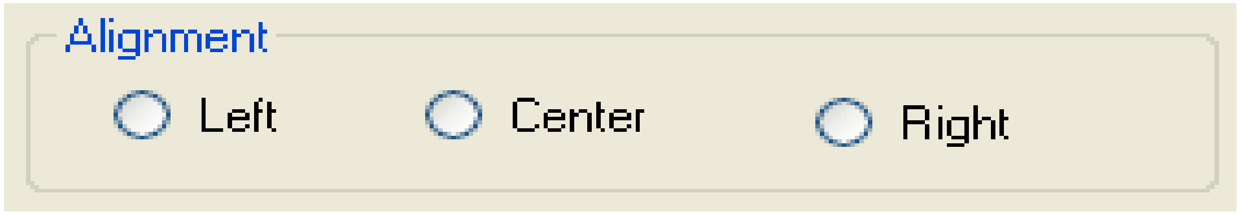
- Events
- `TextChanged` – event raised when the text is changed

Button

– A button accepts clicks.
– Text Property to set the button caption
– Name Property to set the control name

– Events
– Default and most common event is **Click event**

Group Box



- Displays a border around a group of controls
- Can have optional label controlled by Text property
- Controls can be added by
  - Placing them within the group box in the designer
  - Adding to the Controls list programmatically

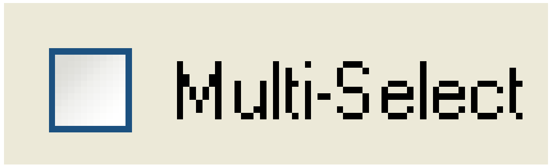
Panels

- A panel is like a group box but does not have a text label
- It contains a group of controls just like group box
  - `BorderStyle` – get/set border style as
    - `BorderStyle.Fixed3D`
    - `BorderStyle.FixedSingle`
    - `BorderStyle.None`

Tab Control

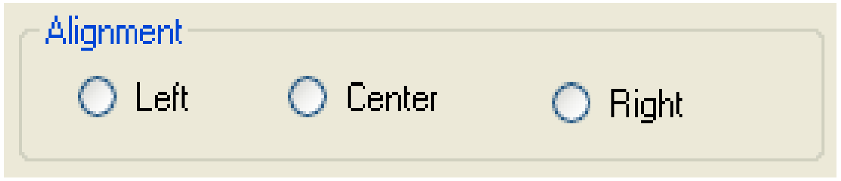
- Presents a tabbed layout in the user interface.
- TabPage Collection Editor can be used, if you need to add an additional tabs.
- Button appearance: TabControl has is the ability to change the appearance of the tabs to buttons or flat buttons.

Check Boxes



- Labeled boxes which can be checked or unchecked
  - `Checked` – get/set Boolean to determine if box is checked
  - `CheckedChanged` – delegate called when the box is checked or unchecked

Radio Buttons



- Radio buttons are similar to checkboxes, but
  - Appear slightly different
  - Allow buttons to be grouped so that only one can be checked at a time
- A group is formed when the radio buttons are in the same container – usually a group box or panel

Radio Buttons

- `Checked` – get/set Boolean indicating if the button is checked
- `CheckedChanged` – delegate invoked when the button is checked or unchecked

Combo Box

- A combo box is like a list but lets you displays a selected value.
- The list pulls down when a selection is being made.
- Options allow the selected text to be editable or to require it to be selected from the drop-down list

Combo Box

- `DropDownStyle` –
  - `Simple` – text is editable & list always visible
  - `DropDown` – default indicating text is editable & user must click to see list
  - `DropDownList` – value is not editable & user must click to see list
- `Items` – the collection of items in the list

Combo Box

- `MaxDropDownItems` – max number of items in pulldown before scrollbar used
- `SelectedIndex` – index of selection
- `SelectedItem` – selected item
- `Sorted` – whether entries are sorted
- `SelectedIndexChanged` – event raised when selection changes

List Box

- The ListBox presents a list of items which can be selected
- A scrollbar is displayed if needed
  - `MultiColumn` – displays list as multiple columns
  - `SelectedIndex` – index of selected item
  - `SelectedIndices` – collection of selected indices
  - `SelectedItem` – the selected item

List Box

- `SelectedItems` – collection of selected items
- `SelectionMode` – how items can be selected
  - `None` – no selection
  - `One` – single selection
  - `MultiSimple` – each click selects additional item
  - `MultiExtended` – uses shift and control keys
- `Sorted` – if true the items will be sorted alphabetically

List Box

- `Items` – a collection of items in the list box
- `ClearSelected` – method to clear selection
- `GetSelected` – returns true if the parameter passed is selected
- `SelectedIndexChanged` – event when selection changes

Populating a List Box

- Any object can be placed into a ListBox
- The display is generated by ToString()

```
for(int i = 0; i < 50; i++) {
    listBox1.Items.Add(
        "Item " + i.ToString());
}
```

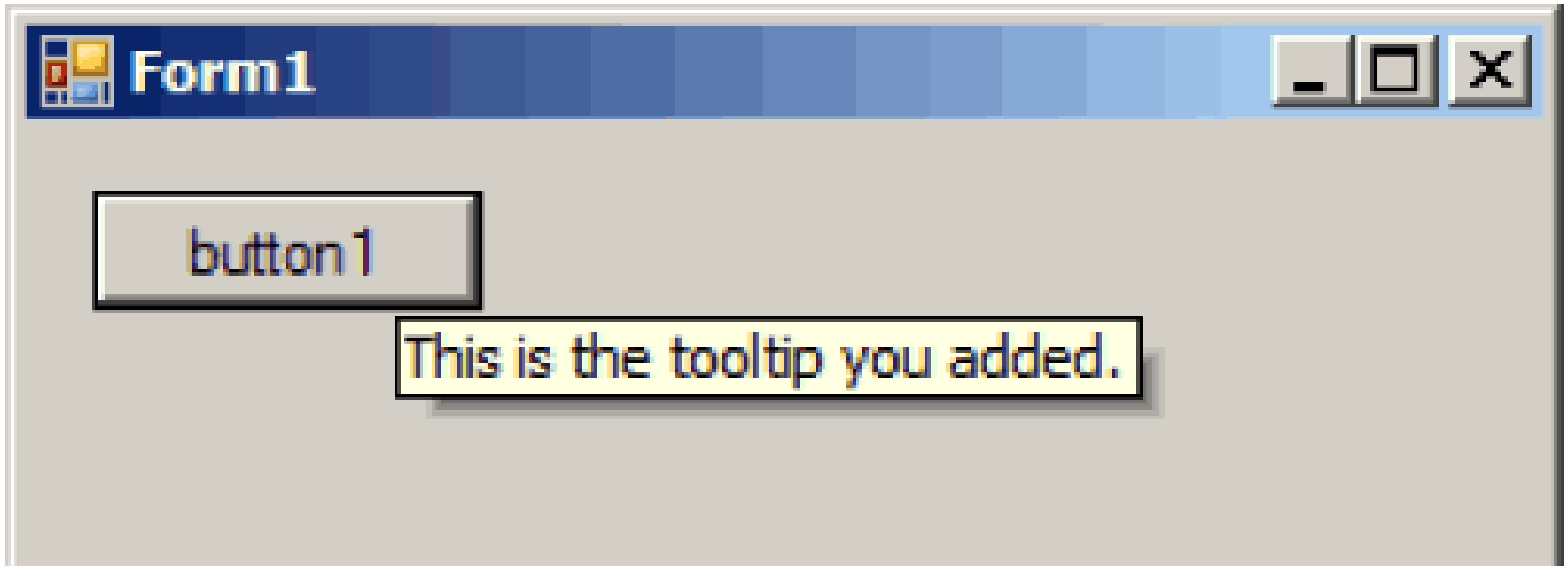
Tool Tips

- These are the small pop-up boxes which explain the purpose of a control
- To use
  - Create a new tooltip in the designer
  - Drop the tooltip onto the form
  - The tooltip will appear on a tray below the form

Tool Tips

- After the tooltip appears in the tray, a new tooltip property appears for every component
- This can be assigned different text for each component
- That text will be displayed when the mouse hovers over that component

Tool Tips

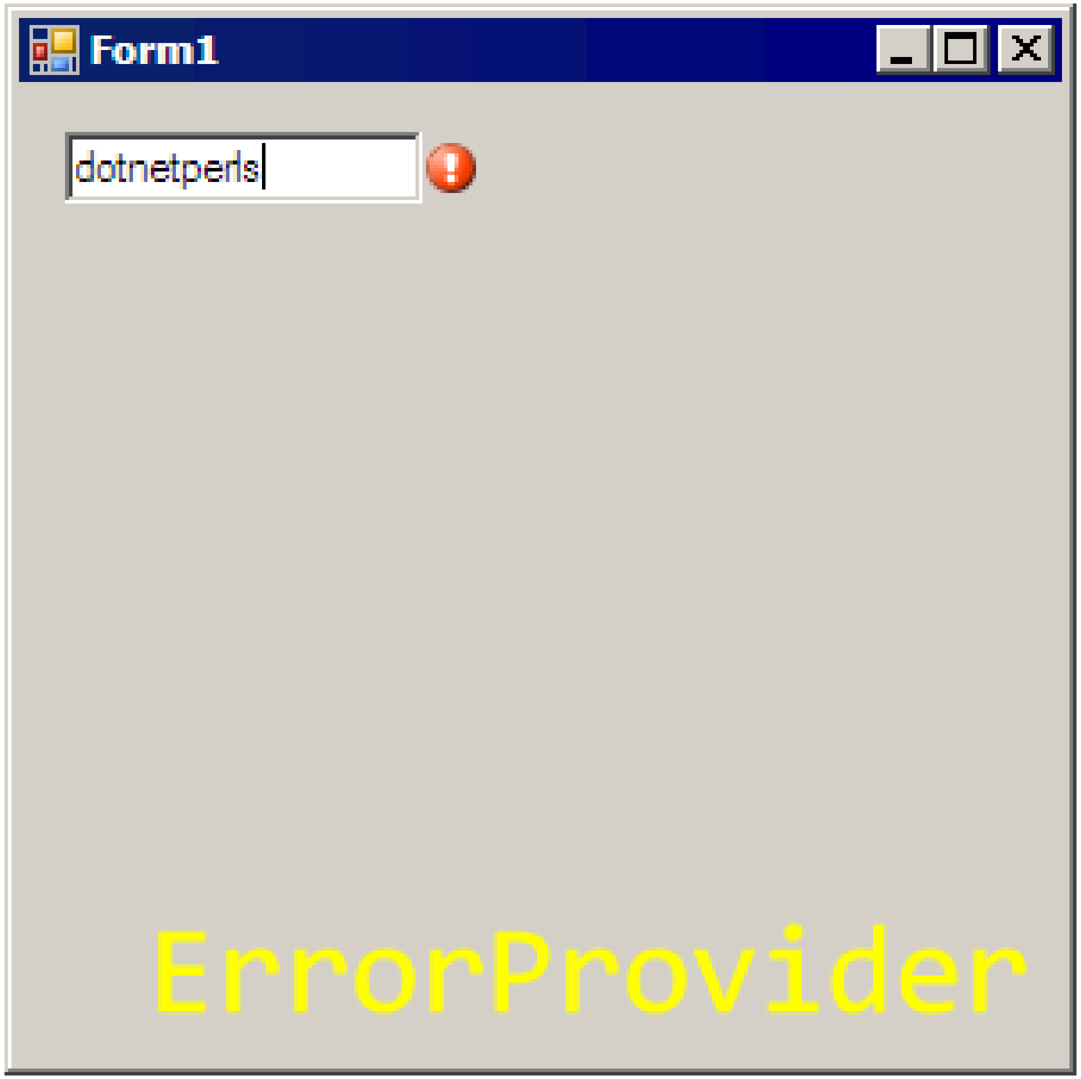


This is the tooltip you added.

Error Provider

- **ErrorProvider** simplifies and streamlines error presentation.
- It is an abstraction that shows errors on your form.

Error Provider

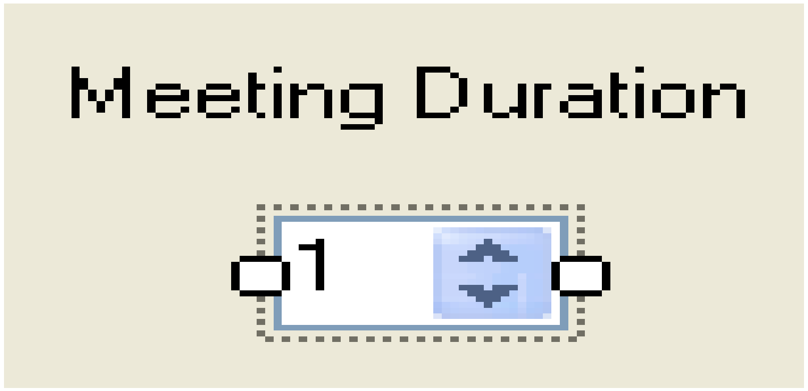


Picture Box

- This displays an image
  - `Image` – assigned an Image object to display
  - `SizeMode` – determines what to do if the image does not fit into the window
    - `Normal`
    - `StretchImage`
    - `AutoSize`
    - `CenterImage`
    - `Zoom`

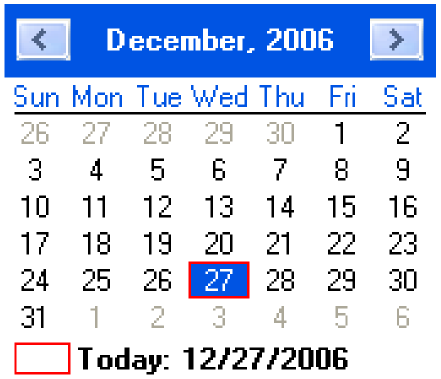
NumericUpDown

- This allows the selection of an integer from a limited range
- Also called a spinner
  - `Minimum` – smallest selectable value
  - `Maximum` – largest selectable value
  - `Increment` – size of increment per click
  - `Value` – the selected value
  - `ValueChanged` – event raised when the value changes



MonthCalendar

- A control which displays a calendar for the selection of a range of dates
  - `MinDate` – the first selectable date
  - `MaxDate` – the last selectable date
  - `SelectionStart` – DateTime of start of selection
  - `SelectionEnd` – DateTime of end of selection
  - `DateChanged` – event raised when date is changed



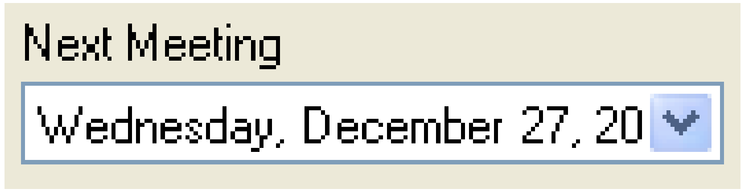
December, 2006

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Today: 12/27/2006

DateTimePicker

- Similar to a month calendar but
  - Calendar pulls down and selection displayed
  - More configurable
  - Selects a single value, not a range
- Properties/methods
  - `Format` – Long, Short, Time, Custom
  - `Value` – DateTime value selected
  - `ValueChanged` – event which fires when date or time changes



System.DateTime Structure

- A structure representing a date and time
- Constructors
  - `DateTime(int d, int m, int y)`
  - `DateTime(int d, int m, int y, int h, int m, int s)`
- Properties
  - `Now` – returns a DateTime object set to the current local time

DateTime

- `Day` – day from 1-31
- `Month` – month from 1-12
- `Year` – tear from 1-9999
- `Hour` – from 0-23
- `Minute` – minute from 0 -59
- `Second` – second from 0 -59
- `Millisecond` – millisecond from 0-999

DateTime

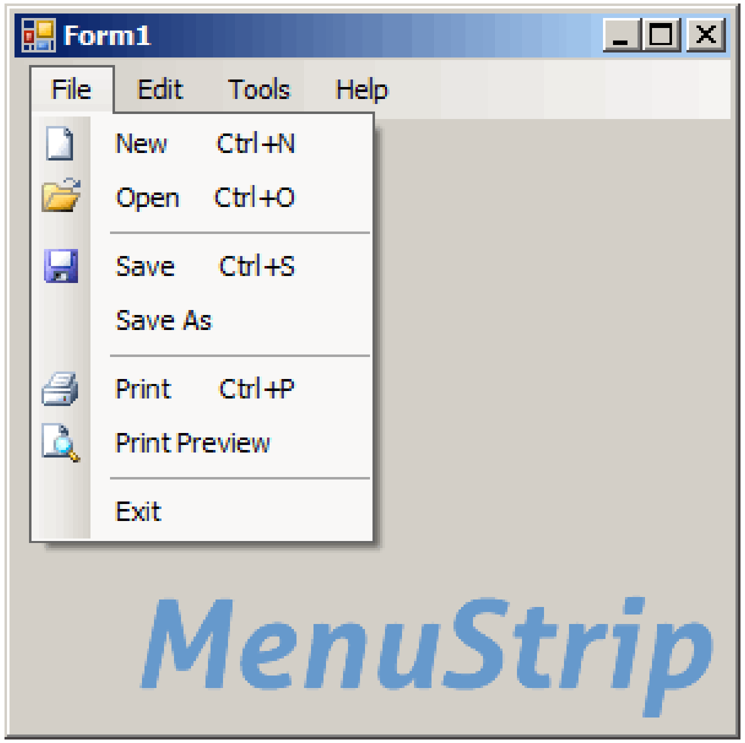
  - `DayOfWeek` – get enumeration of Sunday, Monday,…
  - `DayOfYear` – day of year from 1 – 366
- Methods
  - `DateTime AddYears(double value)`
  - `DateTime AddMonths(double value)`
  - `DateTime AddDays(double value)`
  - `DateTime AddHours(double value)`
  - `DateTime AddSeconds(double value)`
  - `DateTime AddMilliseconds(double value)`

DateTime

- `TimeSpan Subtract(DateTime)`
- `int CompareTo(DateTime)`
- `static DateTime Parse(string)`
- `ToLongDateString()`
- `ToShortDateString()`
- `ToLongTimeString()`
- `ToShortTimeString()`

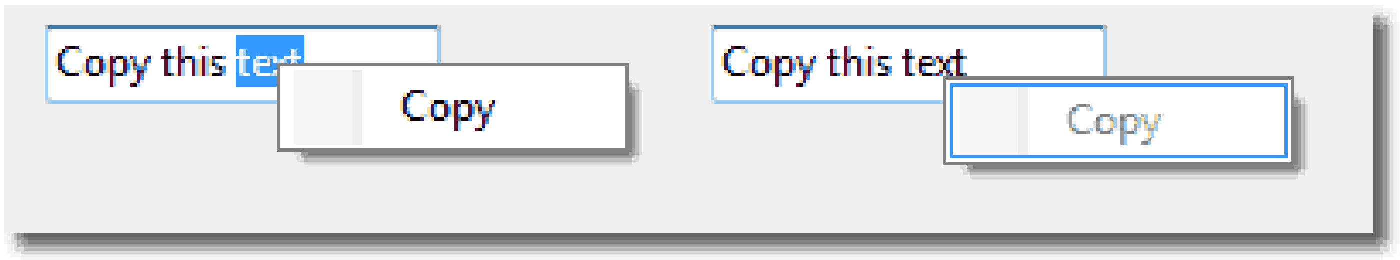
Menu Strip

- **MenuStrip** adds a menu bar to your Windows Forms program.
- ToolStripMenuItem_Click event



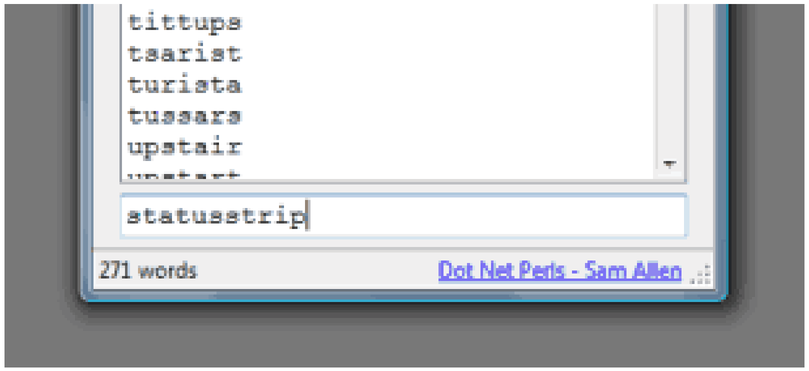
Context Menu Strip

- **ContextMenuStrip** enhances usability in programs. Context menus should appear when a user right-clicks, reacting to the surroundings.



Status Strip

- A **StatusStrip** displays window status.
- **Status Items collection.** Select the status strip control on your form, and in the Properties pane look through the entries there and select Items.


tittups
tsarist
turista
tussars
upstair

statusstrip

Tool Strip Container

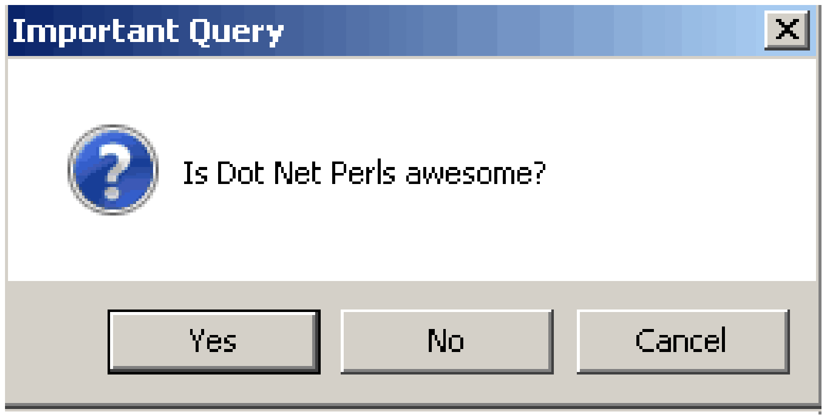
- **ToolStripContainer** adds user interface functionality. It serves as a way to allow ToolStrips (which contain buttons or other controls) to be dragged around the edges of a Form.

Multiple-Document Interface (MDI) Applications

- Multiple-document interface (MDI) applications enable you to display multiple documents at the same time, with each document displayed in its own window.
- In the Properties window, set the **IsMDIContainer** property to true.
- Then Create MDI Child Forms

Message Box

- Dialog boxes interrupt users. They force users to respond before further action is taken.
- **MessageBox.Show()**


Important Query

Is Dot Net Perls awesome?