1. Text box control:

➤ Text box controls allow entering text on a form at runtime. By default, it takes a single line of text, however, you can make it accept multiple texts and even add scroll bars to it.

❖ The Properties of the TextBox Control:

▪ The following are some of the commonly used properties of the TextBox control:

Sr.No.	Property	Description
1	**AcceptsReturn**	Gets or sets a value indicating whether pressing ENTER in a multiline TextBox control creates a new line of text in the control or activates the default button for the form.
2	**AutoCompleteCustomSource**	Gets or sets a custom System.Collections.Specialized.StringCollection to use when the AutoCompleteSourceproperty is set to CustomSource.
3	**AutoCompleteMode**	Gets or sets an option that controls how automatic completion works for the TextBox.
4	**AutoCompleteSource**	Gets or sets a value specifying the source of complete strings used for automatic completion.
5	**CharacterCasing**	Gets or sets whether the TextBox control modifies the case of characters as they are typed.
6	**Font**	Gets or sets the font of the text displayed by the control.
7	**FontHeight**	Gets or sets the height of the font of the control.
8	**ForeColor**	Gets or sets the foreground color of the control.
9	**Lines**	Gets or sets the lines of text in a text box control.
10	**Multiline**	Gets or sets a value indicating whether this is a multiline TextBox control.

11	**PasswordChar**	Gets or sets the character used to mask characters of a password in a single-line TextBox control.
12	**ReadOnly**	Gets or sets a value indicating whether text in the text box is read-only.
13	**ScrollBars**	Gets or sets which scroll bars should appear in a multiline TextBox control. This property has values: None Horizontal Vertical

		Both
14	**TabIndex**	Gets or sets the tab order of the control within its container.
15	**Text**	Gets or sets the current text in the TextBox.
16	**TextAlign**	Gets or sets how text is aligned in a TextBox control. This property has values: Left Right Center
17	**TextLength**	Gets the length of text in the control.
18	**WordWrap**	Indicates whether a multiline text box control automatically wraps words to the beginning of the next line when necessary.

❖ The Methods of the TextBox Control:

. The following are some of the commonly used methods of the TextBox control:

Sr.No.	Method Name & Description
1	**AppendText** Appends text to the current text of a text box.
2	**Clear** Clears all text from the text box control.
3	**Copy** Copies the current selection in the text box to the **Clipboard**.

4	**Cut** Moves the current selection in the text box to the **Clipboard**.
5	**Paste** Replaces the current selection in the text box with the contents of the **Clipboard**.
6	**Paste(String)** Sets the selected text to the specified text without clearing the undo buffer.
7	**ResetText** Resets the Text property to its default value.
8	**ToString** Returns a string that represents the TextBoxBase control.
9	**Undo** Undoes the last edit operation in the text box.

❖
Events of the TextBox Control:

▪
The following are some of the commonly used events of the Text control:

Sr.No.	Event	Description
1	**Click**	Occurs when the control is clicked.
2	**DoubleClick**	Occurs when the control is double-clicked.
3	**TextAlignChanged**	Occurs when the TextAlign property value changes.

2. Label control:

➢
The Label control represents a standard Windows label. It is generally used to display some informative text on the GUI which is not changed during runtime.

❖
Properties of the Label Control:

▪
The following are some of the commonly used properties of the Label control:

Sr.No.	Property	Description
1	**Autosize**	Gets or sets a value specifying if the control should be automatically resized to display all its contents.
2	**BorderStyle**	Gets or sets the border style for the control.
3	**FlatStyle**	Gets or sets the flat style appearance of the Label control
4	**Font**	Gets or sets the font of the text displayed by the control.
5	**FontHeight**	Gets or sets the height of the font of the control.
6	**ForeColor**	Gets or sets the foreground color of the control.
7	**PreferredHeight**	Gets the preferred height of the control.
8	**PreferredWidth**	Gets the preferred width of the control.
9	**TabStop**	Gets or sets a value indicating whether the user can tab to the Label. This property is not used by this class.
10	**Text**	Gets or sets the text associated with this control.
11	**TextAlign**	Gets or sets the alignment of text in the label.

❖
Methods of the Label Control:

▪
The following are some of the commonly used methods of the Label control:

Sr.No.	Method Name & Description
1	**GetPreferredSize** Retrieves the size of a rectangular area into which a control can be fitted.

| 2 | **Refresh** |
| Forces the control to invalidate its client area and immediately redraw itself and any child controls. |

| 3 | **Select** |
| Activates the control. |

| 4 | **Show** |
| Displays the control to the user. |

| 5 | **ToString** |
| Returns a String that contains the name of the control. |

❖

Events of the Label Control:

.
The following are some of the commonly used events of the Label control:

Sr.No.	Event	Description
1	**AutoSizeChanged**	Occurs when the value of the AutoSize property changes.
2	**Click**	Occurs when the control is clicked.
3	**DoubleClick**	Occurs when the control is double-clicked.
4	**GotFocus**	Occurs when the control receives focus.
5	**Leave**	Occurs when the input focus leaves the control.
6	**LostFocus**	Occurs when the control loses focus.
7	**TabIndexChanged**	Occurs when the TabIndex property value changes.
8	**TabStopChanged**	Occurs when the TabStop property changes.
9	**TextChanged**	Occurs when the Text property value changes.

3. Button control:

➢ The Button control represents a standard Windows button. It is generally used to generate a Click event by providing a handler for the Click event.

❖ Properties of the Button Control:

. The following are some of the commonly used properties of the Button control:

Sr.No.	Property	Description
1	**AutoSizeMode**	Gets or sets the mode by which the Button automatically resizes itself.
2	**BackColor**	Gets or sets the background color of the control.
3	**BackgroundImage**	Gets or sets the background image displayed in the control.
4	**DialogResult**	Gets or sets a value that is returned to the parent form when the button is clicked. This is used while creating dialog boxes.
5	**ForeColor**	Gets or sets the foreground color of the control.
6	**Image**	Gets or sets the image that is displayed on a button control.
7	**Location**	Gets or sets the coordinates of the upper-left corner of the control relative to the upper-left corner of its container.
8	**TabIndex**	Gets or sets the tab order of the control within its container.
9	**Text**	Gets or sets the text associated with this control.

❖ Methods of the Button Control:

. The following are some of the commonly used methods of the Button control:

Sr.No.	Method Name & Description
1	**GetPreferredSize** Retrieves the size of a rectangular area into which a control can be fitted.
2	**NotifyDefault** Notifies the Button whether it is the default button so that it can adjust its appearance accordingly.
3	**Select** Activates the control.
4	**ToString** Returns a String containing the name of the Component, if any. This method should not be overridden.

❖ Events of the Button Control:

. The following are some of the commonly used events of the Button control:

Sr.No.	Event	Description
1	**Click**	Occurs when the control is clicked.
2	**DoubleClick**	Occurs when the user double-clicks the Button control.
3	**GotFocus**	Occurs when the control receives focus.
4	**TabIndexChanged**	Occurs when the TabIndex property value changes.
5	**TextChanged**	Occurs when the Text property value changes.
6	**Validated**	Occurs when the control is finished validating.

4. ListBox control:

➤ The ListBox represents a Windows control to display a list of items to a user.

➤ A user can select an item from the list. It allows the programmer to add items at design time by using the properties window or at the runtime.

➤ You can populate the list box items either from the properties window or at runtime. To add items to a ListBox, select the ListBox control and get to the properties window, for the properties of this control.

➤ Click the ellipses (...) button next to the Items property. This opens the String Collection Editor dialog box, where you can enter the values one at a line.

❖ Properties of the ListBox Control:

. The following are some of the commonly used properties of the ListBox control:

Sr.No.	Property	Description
1	**AllowSelection**	Gets a value indicating whether the ListBox currently enables selection of list items.
2	**BorderStyle**	Gets or sets the type of border drawn around the list box.
3	**ColumnWidth**	Gets of sets the width of columns in a multicolumn list box.
4	**HorizontalExtent**	Gets or sets the horizontal scrolling area of a list box.
5	**HorizontalScrollBar**	Gets or sets the value indicating whether a horizontal scrollbar is displayed in the list box.
6	**ItemHeight**	Gets or sets the height of an item in the list box.
7	**Items**	Gets the items of the list box.

8	**MultiColumn**	Gets or sets a value indicating whether the list box supports multiple columns.
9	**ScrollAlwaysVisible**	Gets or sets a value indicating whether the vertical scroll bar is shown at all times.
10	**SelectedIndex**	Gets or sets the zero-based index of the currently selected item in a list box.
11	**SelectedIndices**	Gets a collection that contains the zero-based indexes of all currently selected items in the list box.

12	**SelectedItem**	Gets or sets the currently selected item in the list box.
13	**SelectedItems**	Gets a collection containing the currently selected items in the list box.
14	**SelectedValue**	Gets or sets the value of the member property specified by the ValueMember property.
15	**SelectionMode**	Gets or sets the method in which items are selected in the list box. This property has values: None One MultiSimple MultiExtended
16	**Sorted**	Gets or sets a value indicating whether the items in the list box are sorted alphabetically.
17	**Text**	Gets or searches for the text of the currently selected item in the list box.
18	**TopIndex**	Gets or sets the index of the first visible item of a list box.

❖ Methods of the ListBox Control:

. The following are some of the commonly used methods of the ListBox control:

Sr.No.	Method Name & Description
1	**BeginUpdate** Prevents the control from drawing until the EndUpdate method is called, while items are added to the ListBox one at a time.
2	**ClearSelected** Unselects all items in the ListBox.
3	**EndUpdate** Resumes drawing of a list box after it was turned off by the BeginUpdate method.
4	**FindString**

		Finds the first item in the ListBox that starts with the string specified as an argument.

5	**FindStringExact**
	Finds the first item in the ListBox that exactly matches the specified string.
6	**GetSelected**
	Returns a value indicating whether the specified item is selected.
7	**SetSelected**
	Selects or clears the selection for the specified item in a ListBox.
8	**OnSelectedIndexChanged**
	Raises the SelectedIndexChanged event.
8	**OnSelectedValueChanged**
	Raises the SelectedValueChanged event.

❖ Events of the ListBox Control:

. The following are some of the commonly used events of the ListBox control:

Sr.No.	Event	Description
1	**Click**	Occurs when a list box is selected.
2	**SelectedIndexChanged**	Occurs when the SelectedIndex property of a list box is changed.

5. ComboBox control:

➢ The ComboBox control is used to display a drop-down list of various items. It is a combination of a text box in which the user enters an item and a drop-down list from which the user selects an item.

❖ Properties of the ComboBox Control

. The following are some of the commonly used properties of the ComboBox control:

Sr.No.	Property	Description
1	**AllowSelection**	Gets a value indicating whether the list enables selection of list items.
2	**AutoCompleteCustomSource**	Gets or sets a custom System.Collections.Specialized.StringCollection to use when the AutoCompleteSourceproperty is

		set to CustomSource.
3	**AutoCompleteMode**	Gets or sets an option that controls how automatic completion works for the ComboBox.
4	**AutoCompleteSource**	Gets or sets a value specifying the source of complete strings used for automatic completion.
5	**DataBindings**	Gets the data bindings for the control.
6	**DataManager**	Gets the CurrencyManager associated with this control.
7	**DataSource**	Gets or sets the data source for this ComboBox.
8	**DropDownHeight**	Gets or sets the height in pixels of the drop-down portion of the ComboBox.
9	**DropDownStyle**	Gets or sets a value specifying the style of the combo box.
10	**DropDownWidth**	Gets or sets the width of the of the drop-down portion of a combo box.
11	**DroppedDown**	Gets or sets a value indicating whether the combo box is displaying its drop-down portion.
12	**FlatStyle**	Gets or sets the appearance of the ComboBox.

13	**ItemHeight**	Gets or sets the height of an item in the combo box.
14	**Items**	Gets an object representing the collection of the items contained in this ComboBox.
15	**MaxDropDownItems**	Gets or sets the maximum number of items to be displayed in the drop-down part of the combo box.
16	**MaxLength**	Gets or sets the maximum number of characters a user can enter in the editable area of the combo box.
17	**SelectedIndex**	Gets or sets the index specifying the currently selected item.
18	**SelectedItem**	Gets or sets currently selected item in the ComboBox.
19	**SelectedText**	Gets or sets the text that is selected in the editable portion of a ComboBox.
20	**SelectedValue**	Gets or sets the value of the member property specified by the ValueMember property.
21	**SelectionLength**	Gets or sets the number of characters selected in the editable portion of the combo box.

22	**SelectionStart**	Gets or sets the starting index of text selected in the combo box.
23	**Sorted**	Gets or sets a value indicating whether the items in the combo box are sorted.
24	**Text**	Gets or sets the text associated with this control.

❖ Methods of the ComboBox Control:

. The following are some of the commonly used methods of the ComboBox control:

Sr.No.	Method Name & Description
1	**BeginUpdate** Prevents the control from drawing until the EndUpdate method is called, while items are added to the combo box one at a time.

2   **EndUpdate**

Resumes drawing of a combo box, after it was turned off by the BeginUpdate method.

3   **FindString**

Finds the first item in the combo box that starts with the string specified as an argument.

4   **FindStringExact**

Finds the first item in the combo box that exactly matches the specified string.

5   **SelectAll**

Selects all the text in the editable area of the combo box.

❖ Events of the ComboBox Control:

. The following are some of the commonly used events of the ComboBox control:

Sr.No.	Event	Description
1	**DropDown**	Occurs when the drop-down portion of a combo box is displayed.
2	**DropDownClosed**	Occurs when the drop-down portion of a combo box is no longer visible.
3	**DropDownStyleChanged**	Occurs when the DropDownStyle property of the ComboBox has changed.
4	**SelectedIndexChanged**	Occurs when the SelectedIndex property of a ComboBox control has changed.
5	**SelectionChangeCommitted**	Occurs when the selected item has changed and the

		change appears in the combo box.

6. RadioButton control:

➢ The RadioButton control is used to provide a set of mutually exclusive options.

➢ The user can select one radio button in a group. If you need to place more than one group of radio buttons in the same form, you should place them in different container controls like a GroupBox control.

➢ The Checked property of the radio button is used to set the state of a radio button. You can display text, image or both on radio button control. You can also change the appearance of the radio button control by using the Appearance property.

❖ Properties of the RadioButton Control:

. The following are some of the commonly used properties of the RadioButton control:

Sr.No.	Property	Description
1	**Appearance**	Gets or sets a value determining the appearance of the radio button.
2	**AutoCheck**	Gets or sets a value indicating whether the Checked value and the appearance of the control automatically change when the control is clicked.
3	**CheckAlign**	Gets or sets the location of the check box portion of the radio button.
4	**Checked**	Gets or sets a value indicating whether the control is checked.
5	**Text**	Gets or sets the caption for a radio button.
6	**TabStop**	Gets or sets a value indicating whether a user can give focus to the RadioButton control using the TAB key.

❖ Methods of the RadioButton Control:

. The following are some of the commonly used methods of the RadioButton control:

Sr.No.	Method Name & Description
1	**PerformClick** Generates a Click event for the control, simulating a click by a user.

❖ Events of the RadioButton Control:

. The following are some of the commonly used events of the RadioButton control:

Sr.No.	Event	Description
1	**AppearanceChanged**	Occurs when the value of the Appearance property of the RadioButton control is changed.
2	**CheckedChanged**	Occurs when the value of the Checked property of the RadioButton control is changed.

7. <u>CheckBox control:</u>

➢
   The CheckBox control allows the user to set true/false or yes/no type options. The user can select or deselect it.

➢
   When a check box is selected it has the value True, and when it is cleared, it holds the value False.

➢
   The CheckBox control has three states, **checked**, **unchecked** and **indeterminate**. In the indeterminate state, the check box is grayed out. To enable the indeterminate state, the ThreeState property of the check box is set to be **True**.

❖
   Properties of the CheckBox Control:

   .
      The following are some of the commonly used properties of the CheckBox control:

Sr.No.	Property	Description
1	**Appearance**	Gets or sets a value determining the appearance of the check box.
2	**AutoCheck**	Gets or sets a value indicating whether the Checked or CheckedState value and the appearance of the control automatically change when the check box is selected.
3	**CheckAlign**	Gets or sets the horizontal and vertical alignment of the check mark on the check box.
4	**Checked**	Gets or sets a value indicating whether the check box is selected.
5	**CheckState**	Gets or sets the state of a check box.
6	**Text**	Gets or sets the caption of a check box.
7	**ThreeState**	Gets or sets a value indicating whether or not a check box should allow three check states rather than two.

❖
   Methods of the CheckBox Control:

   .
      The following are some of the commonly used methods of the CheckBox control:

Sr.No.	Method Name & Description
1	**OnCheckedChanged** Raises the CheckedChanged event.
2	**OnCheckStateChanged** Raises the CheckStateChanged event.
3	**OnClick** Raises the OnClick event.

❖

Events of the CheckBox Control:

.

The following are some of the commonly used events of the CheckBox control:

Sr.No.	Event	Description
1	**AppearanceChanged**	Occurs when the value of the Appearance property of the check box is changed.
2	**CheckedChanged**	Occurs when the value of the Checked property of the CheckBox control is changed.
3	**CheckStateChanged**	Occurs when the value of the CheckState property of the CheckBox control is changed.

8. PictureBox control:

➢
The PictureBox control is used for displaying images on the form. The Image property of the control allows you to set an image both at design time or at run time.

❖

Properties of the PictureBox Control:

.

The following are some of the commonly used properties of the PictureBox control:

Sr.No.	Property	Description
1	**AllowDrop**	Specifies whether the picture box accepts data that a user drags on it.
2	**ErrorImage**	Gets or specifies an image to be displayed when an error occurs during the image-loading process or if the image load is cancelled.
3	**Image**	Gets or sets the image that is displayed in the control.
4	**ImageLocation**	Gets or sets the path or the URL for the image displayed in the control.
5	**InitialImage**	Gets or sets the image displayed in the control when the main

		image is loaded
6	**SizeMode**	Determines the size of the image to be displayed in the control. This property takes its value from the PictureBoxSizeMode enumeration, which has values: **Normal** - the upper left corner of the image is placed at upper left part of the picture box **StrechImage** - allows stretching of the image **AutoSize** - allows resizing the picture box to the size of the image **CenterImage** - allows centering the image in the picture box **Zoom** - allows increasing or decreasing the image size to maintain the size ratio.
7	**TabIndex**	Gets or sets the tab index value.

8	**TabStop**	Specifies whether the user will be able to focus on the picture box by using the TAB key.
9	**Text**	Gets or sets the text for the picture box.
10	**WaitOnLoad**	Specifies whether or not an image is loaded synchronously.

❖

Methods of the PictureBox Control:

▪
  The following are some of the commonly used methods of the PictureBox control:

Sr.No.	Method Name & Description
1	**CancelAsync** Cancels an asynchronous image load.
2	**Load** Displays an image in the picture box
3	**LoadAsync** Loads image asynchronously.
4	**ToString** Returns the string that represents the current picture box.

❖

Events of the PictureBox Control:

▪
  The following are some of the commonly used events of the PictureBox control:

Sr.No.	Event	Description

1	**CausesValidationChanged**	Overrides the Control.CausesValidationChanged property.
2	**Click**	Occurs when the control is clicked.
3	**Enter**	Overrides the Control.Enter property.
4	**FontChanged**	Occurs when the value of the Font property changes.
5	**ForeColorChanged**	Occurs when the value of the ForeColor property changes.
6	**KeyDown**	Occurs when a key is pressed when the control has focus.

7	**KeyPress**	Occurs when a key is pressed when the control has focus.
8	**KeyUp**	Occurs when a key is released when the control has focus.
9	**Leave**	Occurs when input focus leaves the PictureBox.
10	**LoadCompleted**	Occurs when the asynchronous image-load operation is completed, been canceled, or raised an exception.
11	**LoadProgressChanged**	Occurs when the progress of an asynchronous image-loading operation has changed.
12	**Resize**	Occurs when the control is resized.
13	**RightToLeftChanged**	Occurs when the value of the RightToLeft property changes.
14	**SizeChanged**	Occurs when the Size property value changes.
15	**SizeModeChanged**	Occurs when SizeMode changes.
16	**TabIndexChanged**	Occurs when the value of the TabIndex property changes.
17	**TabStopChanged**	Occurs when the value of the TabStop property changes.
18	**TextChanged**	Occurs when the value of the Text property changes.

9. Progress Bar control:

➢ It represents a Windows progress bar control. It is used to provide visual feedback to your users about the status of some task. It shows a bar that fills in from left to right as the operation progresses.

➢ The main properties of a progress bar are Value, Maximum and Minimum. The Minimum and Maximum properties are used to set the minimum and maximum values that the progress bar can display.

➢ The Value property specifies the current position of the progress bar.

➢ The ProgressBar control is typically used when an application performs tasks such as copying files or printing documents.

➢ To a user the application might look unresponsive if there is no visual cue. In such cases, using the ProgressBar allows the programmer to provide a visual status of progress.

❖ Properties of the ProgressBar Control:

▪ The following are some of the commonly used properties of the ProgressBar control:

Sr.No.	Property	Description
1	**AllowDrop**	Overrides Control.AllowDrop.
2	**BackgroundImage**	Gets or sets the background image for the ProgressBar control.
3	**BackgroundImageLayout**	Gets or sets the layout of the background image of the progress bar.
4	**CausesValidation**	Gets or sets a value indicating whether the control, when it receives focus, causes validation to be performed on any controls that require validation.
5	**Font**	Gets or sets the font of text in the ProgressBar.
6	**ImeMode**	Gets or sets the input method editor (IME) for the ProgressBar.
7	**ImeModeBase**	Gets or sets the IME mode of a control.
8	**MarqueeAnimationSpeed**	Gets or sets the time period, in milliseconds, that it

		takes the progress block to scroll across the progress bar.
9	**Maximum**	Gets or sets the maximum value of the range of the control.
10	**Minimum**	Gets or sets the minimum value of the range of the control.
11	**Padding**	Gets or sets the space between the edges of a ProgressBar control and its contents.
12	**RightToLeftLayout**	Gets or sets a value indicating whether the ProgressBar and any text it contains is displayed from right to left.
13	**Step**	Gets or sets the amount by which a call to the PerformStep method increases the current position of the progress bar.
14	**Style**	Gets or sets the manner in which progress should be indicated on the progress bar.
15	**Value**	Gets or sets the current position of the progress bar.

❖
Methods of the ProgressBar Control:

▪
The following are some of the commonly used methods of the ProgressBar control:

Sr.No.	Method Name & Description
1	**Increment** Increments the current position of the ProgressBar control by specified amount.
2	**PerformStep** Increments the value by the specified step.
3	**ResetText** Resets the Text property to its default value.
4	**ToString** Returns a string that represents the progress bar control.

❖
Events of the ProgressBar Control:

▪
The following are some of the commonly used events of the ProgressBar control:

Sr.No.	Event	Description
1	**BackgroundImageChanged**	Occurswhenthevalueofthe BackgroundImage property changes.
2	**BackgroundImageLayoutChanged**	Occurswhenthevalueofthe BackgroundImageLayout property changes.
3	**CausesValidationChanged**	Occurswhenthevalueofthe CausesValidation property changes.
4	**Click**	Occurs when the control is clicked.
5	**DoubleClick**	Occurs when the user double-clicks the control.
6	**Enter**	Occurs when focus enters the control.
7	**FontChanged**	Occurs when the value of the Font property changes.
8	**ImeModeChanged**	Occurs when the value of the ImeMode property changes.
9	**KeyDown**	Occurs when the user presses a key while the control has focus.
10	**KeyPress**	Occurs when the user presses a key while the control has focus.
11	**KeyUp**	Occurs when the user releases a key while the control has focus.
12	**Leave**	Occurs when focus leaves the ProgressBar control.
13	**MouseClick**	Occurs when the control is clicked by the mouse.
14	**MouseDoubleClick**	Occurs when the user double-clicks the control.
15	**PaddingChanged**	Occurs when the value of the Padding property changes.
16	**Paint**	Occurs when the ProgressBar is drawn.
17	**RightToLeftLayoutChanged**	Occurs when the RightToLeftLayout property changes.
18	**TabStopChanged**	Occurs when the TabStop property changes.
18	**TextChanged**	Occurs when the Text property changes.

10. ScrollBar control:

➢ The ScrollBar controls display vertical and horizontal scroll bars on the form. This is used for navigating through large amount of information.

➢ There are two types of scroll bar controls: **HScrollBar** for horizontal scroll bars and **VScrollBar** for vertical scroll bars. These are used independently from each other.

❖ Properties of the ScrollBar Control:

▪ The following are some of the commonly used properties of the ScrollBar control:

Sr.No.	Property	Description
1	**AutoSize**	Gets or sets a value indicating whether the ScrollBar is automatically resized to fit its contents.
2	**BackColor**	Gets or sets the background color for the control.
3	**ForeColor**	Gets or sets the foreground color of the scroll bar control.
4	**ImeMode**	Gets or sets the Input Method Editor (IME) mode supported by this control.
5	**LargeChange**	Gets or sets a value to be added to or subtracted from the Value property when the scroll box is moved a large distance.
6	**Maximum**	Gets or sets the upper limit of values of the scrollable range.
7	**Minimum**	Gets or sets the lower limit of values of the scrollable range.
8	**SmallChange**	Gets or sets the value to be added to or subtracted from the Value property when the scroll box is moved a small distance.
9	**Value**	Gets or sets a numeric value that represents the current position of the scroll box on the scroll bar control.

❖ Methods of the ScrollBar Control:

▪ The following are some of the commonly used methods of the ScrollBar control:

Sr.No.	Method Name & Description

1	**OnClick**
	Generates the Click event.

2	**Select**
	Activates the control.

❖

Events of the ScrollBar Control:

.

The following are some of the commonly used events of the ScrollBar control:

Sr.No.	Event	Description
1	**Click**	Occurs when the control is clicked.
2	**DoubleClick**	Occurs when the user double-clicks the control.
3	**Scroll**	Occurs when the control is moved.
4	**ValueChanged**	Occurs when the Value property changes, either by handling the Scroll event or programmatically.

11. DateTimePicker control:

➤
The DateTimePicker control allows selecting a date and time by editing the displayed values in the control.

➤
If you click the arrow in the DateTimePicker control, it displays a month calendar, like a combo box control.

➤
The user can make selection by clicking the required date. The new selected value appears in the text box part of the control.

➤
The **MinDate** and the **MaxDate** properties allow you to put limits on the date range.

❖

Properties of the DateTimePicker Control:

.

The following are some of the commonly used properties of the DateTimePicker control:

Sr.No.	Property	Description
1	**BackColor**	Gets or sets a value indicating the background color of the DateTimePicker control.
2	**BackgroundImage**	Gets or sets the background image for the control.
3	**BackgroundImageLayout**	Gets or sets the layout of the background image of the DateTimePicker control.
4	**CalendarFont**	Gets or sets the font style applied to the calendar.
5	**CalendarForeColor**	Gets or sets the foreground color of the calendar.

6	**CalendarMonthBackground**	Gets or sets the background color of the calendar month.
7	**CalendarTitleBackColor**	Gets or sets the background color of the calendar title.
8	**CalendarTitleForeColor**	Gets or sets the foreground color of the calendar title.
9	**CalendarTrailingForeColor**	Gets or sets the foreground color of the calendar trailing dates.
10	**Checked**	Gets or sets a value indicating whether the Value property has been set with a valid date/time value and the displayed value is able to be updated.
11	**CustomFormat**	Gets or sets the custom date/time format string.

12	**DropDownAlign**	Gets or sets the alignment of the drop-down calendar on the DateTimePicker control.
13	**ForeColor**	Gets or sets the foreground color of the DateTimePicker control.
14	**Format**	Gets or sets the format of the date and time displayed in the control.
15	**MaxDate**	Gets or sets the maximum date and time that can be selected in the control.
16	**MaximumDateTime**	Gets the maximum date value allowed for the DateTimePicker control.
17	**MinDate**	Gets or sets the minimum date and time that can be selected in the control.
18	**MinimumDateTime**	Gets the minimum date value allowed for the DateTimePicker control.
19	**PreferredHeight**	Gets the preferred height of the DateTimePicker control.
20	**RightToLeftLayout**	Gets or sets whether the contents of the DateTimePicker are laid out from right to left.
21	**ShowCheckBox**	Gets or sets a value indicating whether a check box is displayed to the left of the selected date.
22	**ShowUpDown**	Gets or sets a value indicating whether a spin button control (also known as an up-down control) is used to adjust the date/time value.
23	**Text**	Gets or sets the text associated with this control.
24	**Value**	Gets or sets the date/time value assigned to the control.

❖

Methods of the DateTimePicker Control:

▪
The following are some of the commonly used methods of the DateTimePicker control:

Sr.No.	Method Name & Description
1	**ToString** Returns the string representing the control.

❖

Events of the DateTimePicker Control:

▪
The following are some of the commonly used events of the DateTimePicker control:

Sr.No.	Event	Description
1	**BackColorChanged**	Occurs when the value of the BackColor property changes.
2	**BackgroundImageChanged**	Occurswhenthevalueofthe BackgroundImage property changes.
3	**BackgroundImageLayoutChanged**	Occurswhenthevalueofthe BackgroundImageLayout property changes.
4	**Click**	Occurs when the control is clicked.
5	**CloseUp**	Occurs when the drop-down calendar is dismissed and disappears.
6	**DoubleClick**	Occurs when the control is double-clicked.
7	**DragDrop**	Occurs when a drag-and-drop operation is completed.
8	**ForeColorChanged**	Occurs when the value of the ForeColor property changes.
9	**FormatChanged**	Occurs when the Format property value has changed.
10	**MouseClick**	Occurs when the control is clicked with the mouse.
11	**MouseDoubleClick**	Occurs when the control is double-clicked with the mouse.
12	**PaddingChanged**	Occurs when the value of the Padding property changes.
13	**Paint**	Occurs when the control is redrawn.
14	**RightToLeftLayoutChanged**	Occurs when the RightToLeftLayout property changes.

| 15 | **TextChanged** | Occurs when the value of the Text property changes. |
| 16 | **ValueChanged** | Occurs when the Value property changes. |

12. TreeView control:

➤
The TreeView control is used to display hierarchical representations of items similar to the ways the files and folders are displayed in the left pane of the Windows Explorer. Each node may contain one or more child nodes.

❖
Properties of the TreeView Control:

➤
The following are some of the commonly used properties of the TreeView control:

Sr.No.	Property	Description
1	**BackColor**	Gets or sets the background color for the control.
2	**BackgroundImage**	Gets or set the background image for the TreeView control.
3	**BackgroundImageLayout**	Gets or sets the layout of the background image for the TreeView control.
4	**BorderStyle**	Gets or sets the border style of the tree view control.
5	**CheckBoxes**	Gets or sets a value indicating whether check boxes are displayed next to the tree nodes in the tree view control.
6	**DataBindings**	Gets the data bindings for the control.
7	**Font**	Gets or sets the font of the text displayed by the control.
8	**FontHeight**	Gets or sets the height of the font of the control.
9	**ForeColor**	The current foreground color for this control, which is the color the control uses to draw its text.
10	**ItemHeight**	Gets or sets the height of each tree node in the tree view control.
11	**Nodes**	Gets the collection of tree nodes that are assigned to the tree view control.
12	**PathSeparator**	Gets or sets the delimiter string that the tree node

		path uses.
13	**RightToLeftLayout**	Gets or sets a value that indicates whether the TreeView should be laid out from right-to-left.

14	**Scrollable**	Gets or sets a value indicating whether the tree view control displays scroll bars when they are needed.
15	**SelectedImageIndex**	Gets or sets the image list index value of the image that is displayed when a tree node is selected.
16	**SelectedImageKey**	Gets or sets the key of the default image shown when a TreeNode is in a selected state.
17	**SelectedNode**	Gets or sets the tree node that is currently selected in the tree view control.
18	**ShowLines**	Gets or sets a value indicating whether lines are drawn between tree nodes in the tree view control.
19	**ShowNodeToolTips**	Gets or sets a value indicating ToolTips are shown when the mouse pointer hovers over a TreeNode.
20	**ShowPlusMinus**	Gets or sets a value indicating whether plus-sign (+) and minus-sign (-) buttons are displayed next to tree nodes that contain child tree nodes.
21	**ShowRootLines**	Gets or sets a value indicating whether lines are drawn between the tree nodes that are at the root of the tree view.
22	**Sorted**	Gets or sets a value indicating whether the tree nodes in the tree view are sorted.
23	**StateImageList**	Gets or sets the image list that is used to indicate the state of the TreeView and its nodes.
24	**Text**	Gets or sets the text of the TreeView.
25	**TopNode**	Gets or sets the first fully-visible tree node in the tree view control.
26	**TreeViewNodeSorter**	Gets or sets the implementation of IComparer to perform a custom sort of the TreeView nodes.
27	**VisibleCount**	Gets the number of tree nodes that can be fully visible in the tree view control.

❖
Methods of the TreeView Control:

➢
The following are some of the commonly used methods of the TreeView control:

Sr.No.	Method Name & Description
1	**CollapseAll** Collapses all the nodes including all child nodes in the tree view control.
2	**ExpandAll** Expands all the nodes.
3	**GetNodeAt** Gets the node at the specified location.
4	**GetNodeCount** Gets the number of tree nodes.
5	**Sort** Sorts all the items in the tree view control.
6	**ToString** Returns a string containing the name of the control.

❖
Events of the TreeView Control:

➢
The following are some of the commonly used events of the TreeView control:

Sr.No.	Event	Description
1	**AfterCheck**	Occurs after the tree node check box is checked.
2	**AfterCollapse**	Occurs after the tree node is collapsed.
3	**AfterExpand**	Occurs after the tree node is expanded.
4	**AfterSelect**	Occurs after the tree node is selected.
5	**BeforeCheck**	Occurs before the tree node check box is checked.
6	**BeforeCollapse**	Occurs before the tree node is collapsed.
7	**BeforeExpand**	Occurs before the tree node is expanded.
8	**BeforeLabelEdit**	Occurs before the tree node label text is edited.
9	**BeforeSelect**	Occurs before the tree node is selected.
10	**ItemDrag**	Occurs when the user begins dragging a node.
11	**NodeMouseClick**	Occurs when the user clicks a TreeNode with the mouse.

12	**NodeMouseDoubleClick**	Occurs when the user double-clicks a TreeNode with the mouse.
13	**NodeMouseHover**	Occurs when the mouse hovers over a TreeNode.
14	**PaddingChanged**	Occurs when the value of the Padding property changes.

15	**Paint**	Occurs when the TreeView is drawn.
16	**RightToLeftLayoutChanged**	Occurs when the value of the RightToLeftLayout property changes.
17	**TextChanged**	Occurs when the Text property changes.

13. <u>The TreeNode Class:</u>

➢ The TreeNode class represents a **node** of a TreeView. Each node in a TreeView control is an object of the TreeNode class. To be able to use a TreeView control we need to have a look at some commonly used properties and methods of the TreeNode class.

❖ Properties of the TreeNode Class:

➢ The following are some of the commonly used properties of the TreeNode class:

Sr.No.	Property	Description
1	**BackColor**	Gets or sets the background color of the tree node.
2	**Checked**	Gets or sets a value indicating whether the tree node is in a checked state.
3	**ContextMenu**	Gets the shortcut menu that is associated with this tree node.
4	**ContextMenuStrip**	Gets or sets the shortcut menu associated with this tree node.
5	**FirstNode**	Gets the first child tree node in the tree node collection.
6	**FullPath**	Gets the path from the root tree node to the current tree node.
7	**Index**	Gets the position of the tree node in the tree node collection.
8	**IsEditing**	Gets a value indicating whether the tree node is in an editable state.
9	**IsExpanded**	Gets a value indicating whether the tree node is in the expanded state.
10	**IsSelected**	Gets a value indicating whether the tree node is in the selected state.

11	**IsVisible**	Gets a value indicating whether the tree node is visible or partially visible.
12	**LastNode**	Gets the last child tree node.
13	**Level**	Gets the zero-based depth of the tree node in the TreeView control.

14	**Name**	Gets or sets the name of the tree node.
15	**NextNode**	Gets the next sibling tree node.
16	**Nodes**	Gets the collection of TreeNode objects assigned to the current tree node.
17	**Parent**	Gets the parent tree node of the current tree node.
18	**PrevNode**	Gets the previous sibling tree node.
19	**PrevVisibleNode**	Gets the previous visible tree node.
20	**Tag**	Gets or sets the object that contains data about the tree node.
21	**Text**	Gets or sets the text displayed in the label of the tree node.
22	**ToolTipText**	Gets or sets the text that appears when the mouse pointer hovers over a TreeNode.
23	**TreeView**	Gets the parent tree view that the tree node is assigned to.

❖
Methods of the TreeNode Class:
➢
The following are some of the commonly used methods of the TreeNode class:

Sr.No.	Method Name & Description
1	**Collapse** Collapses the tree node.
2	**Expand** Expands the tree node.
3	**ExpandAll** Expands all the child tree nodes.
4	**GetNodeCount** Returns the number of child tree nodes.
5	**Remove** Removes the current tree node from the tree view control.
6	**Toggle** Toggles the tree node to either the expanded or collapsed state.
7	**ToString** Returns a string that represents the current object.

14. ListView control:

➢ The ListView control is used to display a list of items. Along with the TreeView control, it allows you to create a Windows Explorer like interface.

➢ The ListView control displays a list of items along with icons. The Item property of the ListView control allows you to add and remove items from it.

➢ The SelectedItem property contains a collection of the selected items.

➢ The MultiSelect property allows you to set select more than one item in the list view. TheCheckBoxes property allows you to set check boxes next to the items.

❖ **Properties of the ListView Control:**

➢ The following are some of the commonly used properties of the ListView control:

Sr.No.	Property	Description
1	**Alignment**	Gets or sets the alignment of items in the control.
2	**AutoArrange**	Gets or sets whether icons are automatically kept arranged.
3	**BackColor**	Gets or sets the background color.
4	**CheckBoxes**	Gets or sets a value indicating whether a check box appears next to each item in the control.
5	**CheckedIndices**	Gets the indexes of the currently checked items in the control.
6	**CheckedItems**	Gets the currently checked items in the control.
7	**Columns**	Gets the collection of all column headers that appear in the control.
8	**GridLines**	Gets or sets a value indicating whether grid lines appear between the rows and columns containing the items and subitems in the control.
9	**HeaderStyle**	Gets or sets the column header style.
10	**HideSelection**	Gets or sets a value indicating whether the selected item in the control remains highlighted when the control loses focus.
11	**HotTracking**	Gets or sets a value indicating whether the text of an item or subitem has the appearance of a hyperlink when the

12	**HoverSelection**	Gets or sets a value indicating whether an item is automatically selected when the mouse pointer remains over the item for a few seconds.
13	**InsertionMark**	Gets an object used to indicate the expected drop location when an item is dragged within a ListView control.
14	**Items**	Gets a collection containing all items in the control.
15	**LabelWrap**	Gets or sets a value indicating whether item labels wrap when items are displayed in the control as icons.
16	**LargeImageList**	Gets or sets the ImageList to use when displaying items as large icons in the control.
17	**MultiSelect**	Gets or sets a value indicating whether multiple items can be selected.
18	**RightToLeftLayout**	Gets or sets a value indicating whether the control is laid out from right to left.
19	**Scrollable**	Gets or sets a value indicating whether a scroll bar is added to the control when there is not enough room to display all items.
20	**SelectedIndices**	Gets the indexes of the selected items in the control.
21	**SelectedItems**	Gets the items that are selected in the control.
22	**ShowGroups**	Gets or sets a value indicating whether items are displayed in groups.
23	**ShowItemToolTips**	Gets or sets a value indicating whether ToolTips are shown for the ListViewItem objects contained in theListView.
24	**SmallImageList**	Gets or sets the ImageList to use when displaying items as small icons in the control.
25	**Sorting**	Gets or sets the sort order for items in the control.
26	**StateImageList**	Gets or sets the ImageList associated with application-defined states in the control.

27	**TopItem**	Gets or sets the first visible item in the control.
28	**View**	Gets or sets how items are displayed in the control. This property has the following values: LargeIcon - displays large items with a large 32 x 32 pixels icon. SmallIcon - displays items with a small 16 x 16 pixels icon

		List - displays small icons always in one column
		Details - displays items in multiple columns with column headers and fields
		Tile - displays items as full-size icons with item labels and sub-item information.
29	**VirtualListSize**	Gets or sets the number of ListViewItem objects contained in the list when in virtual mode.
30	**VirtualMode**	Gets or sets a value indicating whether you have provided your own data-management operations for the ListView control.

❖
Methods of the ListView Control:

➢
The following are some of the commonly used methods of the ListView control:

Sr.No.	Method Name & Description
1	**Clear** Removes all items from the ListView control.
1	**ToString** Returns a string containing the string representation of the control.

❖
Events of the ListView Control:

➢
The following are some of the commonly used events of the ListView control:

Sr.No.	Event	Description
1	**ColumnClick**	Occurs when a column header is clicked.
2	**ItemCheck**	Occurs when an item in the control is checked or unchecked.
3	**SelectedIndexChanged**	Occurs when the selected index is changed.
4	**TextChanged**	Occurs when the Text property is changed.