Object-oriented programming - lab in .NET environment

Lecture 02

Container controls

- **Container controls** contain other controls
  - A container control can contain other container controls thus forming **control hierarchy**
- Goals:
  - Provide the user with a meaningful user interface
  - Easier management of the controls contained in them
- Examples of container controls:
  - Panel, GroupBox, FlowLayoutPanel, TableLayoutPanel, TabControl, SplitContainer

Adding controls to a container

- When we add a control to another control, we call it a child control, and the second control (parent) is called parent control
- Two ways to add:
  - At design time
  - At runtime
- Each container control contains a property **Controls**
  - Property **Controls** is a collection of objects type of **Control**
- Class **Form** (inherited from Control) also contains a property **Controls** and we consider it container control

Visual Studio designer

- It is possible to mark multiple controls and thus change their common properties:
  - Hold down the Control key (Ctrl) when marking
  - Mark with a lasso
- In the menu **Format** are options for arranging the controls:
  - Align, Make Same Size, Horizontal Spacing, Vertical Spacing, Center in Form, Order (Bring to Front / Send to Back)
- Alignment lines (**snaplines**) enable quick and easy correct stacking of controls
- **Smart tags** enable the most common operations to be performed with control
  - An arrow with additional options appears next to the upper right edge of the control

Properties Anchor and Dock

- The Anchor and Dock properties define how children will behave inside the parent
- **Anchor**
  - Defines the distance between one/more edges of the child and parent when resizing the parent
  - The default value is **Top, Left**
- **Dock**
  - Enables attachment of the child to the edge of the parent
  - The default value is **None**
  - For example, we bind both buttons to the right edge of the form

Panel and GroupBox controls

- **Panel**
  - Basic container control
  - It is most often used
  - Default is no border, but can be set (BorderStyle)

- **GroupBox**
  - By default, it contains a box and text that describes the group
  - It is primarily used for grouping **RadioButton** controls
  - Within one **GroupBox** only one **RadioButton** control can be selected
  - It can be used to group any controls

FlowLayoutPanel control

- Inherited from **Panel** controls
- Added dynamic redistribution of children in case of resizing
  - A principle similar to distribution **HTML** elements on the website
- The predefined flow direction of children is from left to right (property **FlowDirection**)
- Whether children will move to a new row/column is determined by the property **WrapContents** (by default **true**)
- Manual transition to a new row/column can be done by calling the method **SetFlowBreak()**
  - Reading with **GetFlowBreak()**

TableLayoutPanel control

- It represents a table where each cell serves as a control container
- Essential properties:
  - **CellBorderStyle** determines the display of cell frames
  - **RowStyles** and **ColumnStyles** represent collections of rows or columns and we can adjust the width and height through them
  - **Controls** allows adding controls:
    - In the first free cell according to the property **GrowStyle**
    - Right into a specific cell

TabControl control

- Enables grouping of controls using tabs
- **TabControl** contains the following essential properties:
  - **TabPages** is a collection of controls type of **TabPage**
  - **SelectedIndex** determines the displayed **TabPage**
- Each card is one **TabPage** control
  - **TabPage** contains property **Controls**
- Important **TabControl** events:
  - **SelectedIndexChanged**
  - **Selecting**

SplitContainer control

- It consists of a splitter that separates the two **SplitterPanel** controls
  - **SplitterPanel** control is very similar to the Panel control
    - Available through properties **Panel1** and **Panel2**
- Essential properties:
  - **Orientation** determines the orientation of the splitter
  - **FixedPanel** defines which panel will remain fixed if the entire control is resized
  - **IsSplitterFixed** makes it impossible to move the splitter
  - **SplitterDistance** sets the distance of the splitter from the left edge of the control
  - **SplitterWidth** determines the width/height of the splitter

Important inherited properties

- All built-in controls inherit from the class **Control**, for example:
  - FlowLayoutPanel inherits from the Panel class
  - The panel inherits the class ScrollableControl
  - ScrollableControl inherits from the Control class

Dialog forms

- Two ways of displaying forms:
  - **Modal** – the user must close the form in order to be able to work with other forms of the same application
  - **Modeless** – the user can switch between forms as desired
- Modeless forms are displayed by method **Show()**
- Modal forms are displayed by method **ShowDialog()**
  - The method "blocks" work with other forms until the user closes the modal form
- For each button we can define a property **DialogResult** in order to control the behavior of the modal form