Handout WPF

Was ist WPF?

- UI-Framework von Microsoft für Windows-Desktop-Anwendungen.
- Verwendet XAML zur Gestaltung der Benutzeroberfläche.
- Hardwarebeschleunigt mit DirectX für flüssige Grafiken.
- Unterstützt Datenbindung & MVVM für saubere Trennung von UI und Logik.
- Flexibles Layout & moderne UI-Elemente mit Animationen und Stilen.

Vorteile von WPF!

- Trennung von UI und Logik durch XAML und Code-Behind (bessere Wartbarkeit).
- Skalierbare & flexible Benutzeroberfläche dank Vektorgrafiken und dynamischem Layout.
- Leistungsstarke Datenbindung & MVVM für strukturierte und effiziente Entwicklung.
- Hardwarebeschleunigung mit DirectX für bessere Performance und flüssige Animationen.
- Erweiterbarkeit & Anpassbarkeit durch Styles, Templates und Custom Controls.

Erstellen eines WPF-Projekts in Visual Studio

- WPF-Anwendung C# in Visual Studio auswählen

Grundlegende Komponenten von WPF

XAML (Extensible Application Markup Language)

- Markup-Sprache für WPF zur Definition der Benutzeroberfläche.
- Trennung von UI und Logik, ähnlich wie HTML/CSS für Webanwendungen.
- Hierarchische Struktur für einfache Verschachtelung von UI-Elementen.
- Unterstützt Datenbindung für dynamische Inhalte ohne viel Code.
- Ermöglicht Wiederverwendbarkeit durch Styles, Templates und Ressourcen.

Steuerelemente (Controls)

- Grundlegende Controls: Button, TextBox, Label, CheckBox, ComboBox.
- Container-Elemente: Grid, StackPanel, DockPanel, WrapPanel zur Anordnung von UI-Elementen.
- Erweiterte Controls: ListView, TreeView, DataGrid für komplexe Datenanzeige.
- Interaktive Controls: Slider, ProgressBar, DatePicker, RichTextBox.
- Anpassbar durch Styles & Templates für individuelles Design.

Layouts

- Grid: Flexibles Rasterlayout mit Zeilen und Spalten.
- StackPanel: Anordnung von Elementen vertikal oder horizontal.
- WrapPanel: Elemente werden automatisch umgebrochen, wenn der Platz nicht ausreicht.
- DockPanel: Elemente können an den Rändern (oben, unten, links, rechts) angedockt werden.
- Canvas: Freie Platzierung von Elementen mit festen Koordinaten (X, Y).

Wichtige Ereignisse in WPF

Maus- und Tastaturereignisse

- Mausereignisse: MouseEnter, MouseLeave, MouseDown, MouseUp, MouseMove, Click
- Tastaturereignisse: KeyDown, KeyUp, PreviewKeyDown für Tastatureingaben
- Fokussteuerung: GotFocus, LostFocus für UI-Elemente, die Eingaben erhalten
- Modifier Keys: Prüfen von Ctrl, Shift, Alt mit Keyboard.IsKeyDown()

Routing Events

- Ereignisweiterleitung in der WPF-Elementhierarchie (von Kind- zu Elternelementen oder umgekehrt)
- Bubbling-Ereignisse: Wandern vom auslösenden Kind-Element zum Eltern-Element (Button.Click)
- Tunneling-Ereignisse: Wandern vom Root-Element zum Kind-Element (PreviewMouseDown)
- Direkte Ereignisse: Werden nur vom auslösenden Element verarbeitet (TextChanged)
- Ermöglicht zentrale Ereignisbehandlung ohne direkten Event-Handler auf jedem Element