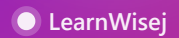


Erste Schritte mit Wisej.NET

Von einem leeren Visual Studio zu einer laufenden App – und die Gewohnheiten, die Ihren Code sauber halten.



Lernziel

Legen Sie ein erstes Wisej.NET-Projekt an, öffnen Sie es im Designer, starten Sie es lokal, untersuchen Sie die Projektmappenstruktur und finden Sie heraus, wo die Start- und Konfigurationsdateien liegen.

1. Mit der Projektvorlage beginnen

Legen Sie in Visual Studio ein neues Projekt aus den installierten Wisej.NET-Vorlagen an. Die genaue Liste hängt von der Version ab, der Ablauf ist aber immer derselbe: **neues Projekt erstellen** → **nach „Wisej.NET“ suchen** → **eine Wisej.NET-Webanwendungsvorlage auswählen** → **benennen** → **erstellen**.

Vorgeschlagener Name für das Übungsprojekt

WisejTrainingApp



Bevor Sie beginnen

Stellen Sie sicher, dass die Wisej.NET-Erweiterung für Visual Studio und die Vorlagen installiert sind. Erscheinen die Vorlagen nicht, prüfen Sie die Installation erneut oder starten Sie Visual Studio neu.

2. Den Wisej.NET-Designer öffnen

Im Designer platzieren Sie Controls visuell. Öffnen Sie die Hauptseite oder das Hauptfenster und wechseln Sie in die Entwurfsansicht. Ziehen Sie dann Controls aus der Toolbox, legen Sie Eigenschaften fest und doppelklicken Sie auf ein Control, um einen Event-Handler zu erzeugen.

Aufgabe im Designer	Was Sie tun
Ein Control hinzufügen	Ziehen Sie einen <code>Button</code> , ein <code>Label</code> , eine <code>TextBox</code> , ein <code>Panel</code> oder eine <code>DataGridView</code> auf die Oberfläche.
Text festlegen	Control auswählen, das Eigenschaftfenster öffnen und die Eigenschaft <code>Text</code> ändern.
Ein Control umbenennen	Ändern Sie <code>Name</code> in etwas Eindeutiges wie <code>btnRun</code> , <code>lblStatus</code> , <code>txtCustomerName</code> .
Klick-Code erzeugen	Doppelklicken Sie auf einen Button (oder nutzen Sie den Events-Bereich), um einen <code>Click</code> -Handler anzulegen.
Layout anpassen	Nutzen Sie <code>Dock</code> , <code>Anchor</code> , <code>Size</code> , <code>Location</code> , <code>Margin</code> , <code>Padding</code> .

3. Lokal starten

Lokal starten heißt: auf dem eigenen Rechner testen, bevor Sie veröffentlichen. Verwenden Sie die normale Schaltfläche Start / Debug – das Projekt wird gebaut, ein lokaler Server startet und die App öffnet sich im Browser.

1. Klicken Sie in Visual Studio auf **Start / Debug**.
2. Warten Sie, bis das Projekt gebaut ist.
3. Prüfen Sie, dass der Browser die Wisej.NET-Anwendung öffnet.
4. Testen Sie einen Button-Klick, um sicherzugehen, dass Ihr Code läuft.
5. Untersuchen Sie Event-Handler und Variablen mit Breakpoints.



Gewohnheit bei der Fehlersuche

Öffnet sich die App nicht, lesen Sie zuerst die **Fehlerliste** und das **Ausgabe**-Fenster von Visual Studio. Die meisten Einsteigerprobleme sind fehlende Pakete, Build-Fehler, das falsche Startprojekt oder eine umbenannte Datei, deren Verweise nicht angepasst wurden.

4. Die Projektmappenstruktur untersuchen

Der Projektmappen-Explorer ist keine zufällige Dateisammlung – er zeigt, wie die App organisiert ist.

Datei oder Ordner	Wofür sie gedacht ist
<code>Program.cs</code>	Startcode, der die Anwendung startet und oft die erste Page oder das erste Window anzeigt.
<code>Page1.cs</code> / <code>Window1.cs</code>	Code-Behind: Hier schreiben Sie Event-Handler und das Verhalten der Maske.
<code>*.Designer.cs</code>	Vom Designer erzeugtes Layout und die Initialisierung der Controls. Nicht von Hand bearbeiten.
<code>*.resx</code>	Ressourcendateien für den Designer und lokalisierte Daten.
Konfigurationsdateien	<code>Web.config</code> sowie die Wisej.NET-spezifische JSON-Konfiguration.
<code>Models/</code>	Einfache Datenklassen wie <code>Ticket.cs</code> oder <code>Customer.cs</code> .
<code>Services/</code>	Geschäftslogik wie <code>TicketService.cs</code> .

5. Gute Entwicklergewohnheiten

Gewohnheit	Vermeiden	Besser
Controls eindeutig benennen	<code>button1</code> , <code>label2</code>	<code>btnSaveTicket</code> , <code>lblStatus</code>
Events lesbar halten	200 Zeilen in <code>btnSave_Click</code>	<code>ValidateInput()</code> und <code>ticketService.Save(ticket)</code> aufrufen
Geschäftslogik trennen	Die gesamte Speicherlogik in der Page	Ein <code>TicketService</code> verantwortet die Regeln
Dateien organisieren	Alles im Projektstamm	Ordner Models, Services, Pages, Controls

6. Beispiel: ein lesbarer Event-Handler

Ein Event-Handler sollte sich wie eine kurze Geschichte lesen – Eingabe holen, Logik aufrufen, Oberfläche aktualisieren:

```
private void btnSayHello_Click(object sender, EventArgs e)
{
    string name = txtName.Text.Trim();

    if (string.IsNullOrEmpty(name))
    {
        lblStatus.Text = "Please enter a name.";
        return;
    }

    lblStatus.Text = $"Hello, {name}!";
}
```

Wächst die App, verlagern Sie Regeln und Datenoperationen in Service-Klassen:

```
private readonly TicketService ticketService = new TicketService();

private void btnSaveTicket_Click(object sender, EventArgs e)
{
    Ticket ticket = ReadTicketFromScreen();
    ticketService.Save(ticket);
    lblStatus.Text = "Ticket saved.";
}
```



Sie sind bereit für das Lab

Als Nächstes: Sehen Sie sich den Video-Walkthrough zum Aufbau an, bestehen Sie den Wissenstest und erstellen und starten Sie dann im praktischen Lab Ihre eigene erste Wisej.NET-Maske.