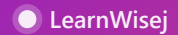


Die Anwendungsshell entwerfen

Entwerfen Sie die äußere Struktur einer echten App – Kopfzeile, Navigation, Inhaltsbereich und Statusleiste – und sorgen Sie dafür, dass sie sich sauber an jede Größe anpasst.



Lernziel

Gehen Sie von einzelnen Masken zu einer einfachen **Anwendungsshell** über – dem gemeinsamen äußeren Layout (Kopfzeile, Navigation, Inhaltsbereich, Statusleiste), das sichtbar bleibt, während sich der Inhalt ändert.

1. Was eine Anwendungsshell ist

Eine **Anwendungsshell** ist die gemeinsame äußere Struktur einer echten App. Die Shell bleibt an ihrem Platz; nur der Inhaltsbereich wechselt, während sich der Benutzer durch die App bewegt. Eine einfache Shell besteht aus vier Teilen:

Teil der Shell	Zweck	Beispiel für Einsteiger
Kopfzeile	App-Name, aktueller Benutzer und Breadcrumb.	ServiceDesk Application · Signed in as: Support Agent
Linke Navigation	Die Seiten, die der Benutzer öffnen kann.	Dashboard, Tickets, Customers, Settings
Inhaltsbereich	Zeigt den Inhalt der ausgewählten Seite.	Dashboard-Karten oder eine Ticketliste
Statusbereich	Zeigt die letzte Aktion oder Rückmeldungen des Systems.	03:26 PM - Application shell loaded.

2. Page oder Window/Form1

In Wisej.NET können Sie die Shell als **Page** oder als **Window/Form1** bauen. Beide erzeugen ein unterschiedliches Benutzererlebnis.

Variante	Wie sie sich anfühlt	Wann Sie sie verwenden
Page	Eine Webseite, die den Browserrahmen ausfüllt.	Ideal für eine moderne Trainings-App oder eine Dashboard-Shell.
Window/Form1	Ein desktopähnliches Fenster im Browser, das sich verschieben, minimieren, maximieren und schließen lässt.	Gut für Tools im Desktop-Stil, Popups oder wenn Sie bewusst Fensterverhalten wollen.



Wichtige UI-Grundregel

Wenn Sie die Shell als Window/Form1 bauen, **entfernen oder deaktivieren Sie die Minimieren-Schaltfläche** – es sei denn, Sie bauen auch eine Taskleiste oder eine klare Möglichkeit zum Wiederherstellen. Ein minimiertes Browserfenster wirkt, als wäre die App verschwunden, und das verwirrt Einsteiger.

3. Anchoring und Docking halten das Layout benutzbar

Anchoring legt fest, an welchen Seiten seines übergeordneten Elements ein Control verankert bleibt.

Docking lässt ein Control eine Seite oder den verbleibenden Platz ausfüllen. Zusammen sorgen sie dafür, dass die Shell auch nach einer Größenänderung des Browsers benutzbar bleibt.

Bereich	Empfohlenes Layout	Warum
Kopfzeilen-Panel	Dock = Top (oder Anchor Top + Left + Right)	Die Kopfzeile soll sich über die gesamte Breite erstrecken.
Linkes Navigations-Panel	Dock = Left (oder Anchor Top + Bottom + Left)	Die Navigation bleibt links und behält ihre Höhe.
Inhalts-Panel	Dock = Fill (oder Anchor an allen vier Seiten)	Die aktuelle Seite nutzt den verbleibenden Platz.
Status-Label/-Panel	Dock = Bottom (oder Anchor Bottom + Left + Right)	Statusmeldungen bleiben unten sichtbar.
Navigations-Buttons	Anchor Top + Left + Right innerhalb des Navigations-Panels	Die Buttons behalten eine saubere, einheitliche Breite.

4. Von einer zentralen Stelle aus navigieren

Verteilen Sie die Logik zum Seitenwechsel nicht auf jeden einzelnen Button. Jeder Button ruft stattdessen **eine gemeinsame Navigationsmethode** auf. So bleibt die App leicht zu debuggen und später leicht zu ändern.

```

private void btnDashboard_Click(object sender, EventArgs e)
{
    NavigateTo("Dashboard");
}

private void btnTickets_Click(object sender, EventArgs e)
{
    NavigateTo("Tickets");
}

private void NavigateTo(string pageName)
{
    pnlContent.Controls.Clear();
    if (pageName == "Dashboard")
        pnlContent.Controls.Add(new DashboardView() { Dock = DockStyle.Fill });
    else if (pageName == "Tickets")
        pnlContent.Controls.Add(new TicketsView() { Dock = DockStyle.Fill });

    lblBreadcrumb.Text = "Home / " + pageName;
    lblStatus.Text = DateTime.Now.ToString("hh:mm:ss tt") + " - Opened " + pageName + "
page.";
}

```

Eine einzige Methode leert das Inhalts-Panel, lädt die passende Ansicht, aktualisiert den **Breadcrumb** ([Home](#) / [Tickets](#)) und schreibt eine Zeile in den **Statusbereich**.

5. Berechtigungen als Designkonzept

Berechtigungen müssen nicht mit einem vollständigen Anmeldesystem beginnen. Sie können als einfache Idee einer **Rolle** starten: Ein Support Agent darf die Einstellungen *ansehen*, aber keine Änderungen speichern. Später wird daraus echte Authentifizierung und Autorisierung.

Rolle	Dashboard	Tickets	Customers	Settings
Support Agent	Ansehen	Anlegen/schließen	Ansehen	Nur ansehen
Manager	Ansehen	Anlegen/schließen	Anlegen/bearbeiten	Einstellungen speichern



Abschlusskontrolle

Sie können Kopfzeile, Navigation, Inhalts- und Statusbereich benennen, Anchoring oder Docking statt fester Positionen verwenden, den Unterschied zwischen Page und Window/Form1 erklären und jeden Navigations-Button über eine gemeinsame Navigationsmethode leiten.