

Progettare lo shell applicativo

Progetta la struttura esterna di un'app reale — intestazione, navigazione, area contenuti e barra di stato — e fai in modo che si ridimensioni in modo pulito.



Obiettivo della lezione

Passa dalle singole schermate a un semplice **shell applicativo** — il layout esterno condiviso (intestazione, navigazione, area contenuti, barra di stato) che resta sullo schermo mentre il contenuto cambia.

1. Che cos'è uno shell applicativo

Uno **shell applicativo** è la struttura esterna condivisa di un'app reale. Lo shell resta al suo posto; cambia solo l'area contenuti mentre l'utente si sposta. Uno shell semplice ha quattro parti:

Parte dello shell	Scopo	Esempio per chi inizia
Intestazione	Nome dell'app, utente corrente e breadcrumb.	ServiceDesk Application · Signed in as: Support Agent
Navigazione a sinistra	Le pagine che l'utente può aprire.	Dashboard, Tickets, Customers, Settings
Area contenuti	Mostra il contenuto della pagina selezionata.	Le schede della dashboard o un elenco di ticket
Area di stato	Mostra l'azione recente o il feedback del sistema.	03:26 PM - Application shell loaded.

2. Page e Window/Form1 a confronto

Wisej.NET ti permette di costruire lo shell come **Page** o come **Window/Form1**. Le due scelte creano esperienze utente diverse.

Scelta	Come si presenta	Quando usarla
Page	Una pagina web che riempie la cornice del browser.	La scelta migliore per un'app di formazione moderna o per lo shell di una dashboard.
Window/Form1	Una finestra in stile desktop dentro il browser, che può essere spostata, ridotta a icona, ingrandita e chiusa.	Adatta a strumenti in stile desktop, popup, o quando vuoi intenzionalmente il comportamento di una finestra.



Una regola importante per l'interfaccia

Se costruisci lo shell come Window/Form1, **rimuovi o disabilita il pulsante di riduzione a icona**, a meno che tu non costruisci anche una barra delle applicazioni o un modo chiaro per ripristinarla. Una finestra ridotta a icona nel browser può sembrare un'app scomparsa, e questo confonde chi inizia.

3. Ancoraggio e docking mantengono il layout utilizzabile

L'**ancoraggio** dice a un controllo a quali lati del contenitore padre restare agganciato. Il **docking** fa sì che un controllo riempi un lato o lo spazio rimanente. Insieme mantengono lo shell utilizzabile quando il browser viene ridimensionato.

Area	Layout consigliato	Perché
Pannello dell'intestazione	Dock = Top (oppure Anchor Top + Left + Right)	L'intestazione deve estendersi lungo tutto il lato superiore.
Pannello di navigazione a sinistra	Dock = Left (oppure Anchor Top + Bottom + Left)	La navigazione resta a sinistra e mantiene la sua altezza.
Pannello dei contenuti	Dock = Fill (oppure Anchor su tutti e quattro i lati)	La pagina corrente usa lo spazio rimanente.
Label/pannello di stato	Dock = Bottom (oppure Anchor Bottom + Left + Right)	I messaggi di stato restano visibili in basso.
Pulsanti di navigazione	Anchor Top + Left + Right dentro il pannello di navigazione	I pulsanti mantengono una larghezza pulita e uniforme.

4. Naviga da un unico punto

Non spargere la logica che cambia pagina in ogni pulsante. Fai invece in modo che ogni pulsante chiami **un unico metodo di navigazione condiviso**. Così l'app resta facile da debuggare e da modificare in seguito.

```
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.";
}
```

Un solo metodo svuota il pannello dei contenuti, carica la vista giusta, aggiorna il **breadcrumb** ([Home](#) / [Tickets](#)) e scrive una riga nell'area di **stato**.

5. I permessi come concetto di design

I permessi non devono per forza partire da un sistema di login completo. Possono iniziare come una semplice idea di **ruolo**: un Support Agent può *vedere* Settings ma non salvare le modifiche. In seguito questo diventa vera autenticazione e autorizzazione.

Ruolo	Dashboard	Tickets	Customers	Settings
Support Agent	Visualizza	Crea/chiude	Visualizza	Sola lettura
Manager	Visualizza	Crea/chiude	Crea/modifica	Salva le impostazioni



Verifica di completamento

Sai individuare l'intestazione, la navigazione, l'area contenuti e l'area di stato; usare l'ancoraggio o il docking invece di posizioni fisse; spiegare in che cosa differiscono Page e Window/Form1; e far passare ogni pulsante di navigazione da un unico metodo di navigazione condiviso.