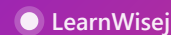


Concevoir le shell de l'application

Concevez la structure extérieure d'une vraie application — en-tête, navigation, zone de contenu et barre d'état — et faites-la se redimensionner proprement.



Objectif de la leçon

Passer d'écrans isolés à un **shell applicatif** simple — la structure extérieure commune (en-tête, navigation, zone de contenu, barre d'état) qui reste à l'écran pendant que le contenu change.

1. Ce qu'est un shell applicatif

Un **shell applicatif** est la structure extérieure commune d'une vraie application. Le shell reste en place ; seule la zone de contenu change quand l'utilisateur se déplace. Un shell simple comporte quatre parties :

Partie du shell	Rôle	Exemple pour débutant
En-tête	Nom de l'application, utilisateur courant et fil d'Ariane.	ServiceDesk Application · Signed in as: Support Agent
Navigation à gauche	Les pages que l'utilisateur peut ouvrir.	Dashboard, Tickets, Customers, Settings
Zone de contenu	Affiche le contenu de la page sélectionnée.	Des cartes de tableau de bord ou une liste de tickets
Zone d'état	Affiche la dernière action ou un retour du système.	03:26 PM - Application shell loaded.

2. Page ou Window/Form1

Wisej.NET vous permet de construire le shell comme une **Page** ou comme une **Window/Form1**. Les deux offrent des expériences utilisateur différentes.

Choix	Ce que l'on ressent	Quand l'utiliser
Page	Une page web qui remplit le cadre du navigateur.	Idéal pour une application de formation moderne ou un shell de tableau de bord.
Window/Form1	Une fenêtre de type desktop dans le navigateur, que l'on peut déplacer, réduire, agrandir et fermer.	Adapté aux outils de type desktop, aux popups, ou quand vous voulez délibérément un comportement de fenêtre.



Habitude d'interface importante

Si vous construisez le shell comme une Window/Form1, **supprimez ou désactivez le bouton de réduction**, à moins de construire aussi une barre des tâches ou un moyen clair de la restaurer. Une fenêtre réduite dans le navigateur peut donner l'impression que l'application a disparu, ce qui déroute les débutants.

3. L'ancrage et le docking gardent la mise en page utilisable

L'**ancrage** (Anchor) indique à un contrôle à quels bords de son parent il reste attaché. Le **docking** (Dock) fait occuper à un contrôle un bord ou l'espace restant. Ensemble, ils gardent le shell utilisable quand le navigateur est redimensionné.

Zone de contrôles	Mise en page recommandée	Pourquoi
Panneau d'en-tête	Dock = Top (ou Anchor Top + Left + Right)	L'en-tête doit s'étirer sur toute la largeur en haut.
Panneau de navigation à gauche	Dock = Left (ou Anchor Top + Bottom + Left)	La navigation reste à gauche et conserve sa hauteur.
Panneau de contenu	Dock = Fill (ou Anchor sur les quatre bords)	La page courante utilise l'espace restant.
Libellé/panneau d'état	Dock = Bottom (ou Anchor Bottom + Left + Right)	Les messages d'état restent visibles en bas.
Boutons de navigation	Anchor Top + Left + Right dans le panneau de navigation	Les boutons gardent une largeur nette et régulière.

4. Naviguer depuis un seul endroit

Ne dispersez pas la logique de changement de page dans chaque bouton. Chaque bouton appelle plutôt **une méthode de navigation commune**. L'application reste ainsi facile à déboguer et à faire évoluer.

```

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.";
}

```

Une seule méthode vide le panneau de contenu, charge la bonne vue, met à jour le **fil d'Ariane** ([Home](#) / [Tickets](#)) et écrit une ligne dans la zone d'**état**.

5. Les permissions comme concept de conception

Les permissions n'ont pas besoin de commencer par un système de connexion complet. Elles peuvent naître d'une simple notion de **rôle** : un Support Agent peut *consulter* les paramètres mais pas enregistrer de modifications. Plus tard, cela deviendra une véritable authentification et autorisation.

Rôle	Dashboard	Tickets	Customers	Settings
Support Agent	Consulter	Créer/fermer	Consulter	Lecture seule
Manager	Consulter	Créer/fermer	Créer/modifier	Enregistrer les paramètres



Point de contrôle

Vous savez identifier l'en-tête, la navigation, la zone de contenu et la zone d'état ; utiliser l'ancrage ou le docking plutôt que des positions fixes ; expliquer en quoi Page et Window/Form1 diffèrent ; et faire passer chaque bouton de navigation par une méthode de navigation commune.