

# Fondamenti di layout: docking, ancoraggio, AutoSize, AutoScroll e composizione dello shell

Guida della lezione · Guida al lab / esame

● LearnWisej

## Guida della lezione

Il motore di layout predefinito, ordine di docking e Padding, l'ancoraggio per le regioni simili a form, AutoSize e AutoScroll usati in modo sicuro, e la composizione dello shell con contenitori annidati e UserControl invece di gestori di ridimensionamento. Che cosa significa per uno sviluppatore Wisej.NET — e come affrontare questo modulo.



### Focus di produzione

Il motore di layout predefinito, ordine di docking e Padding, l'ancoraggio per le regioni simili a form, AutoSize e AutoScroll usati in modo sicuro, e la composizione dello shell con contenitori annidati e UserControl invece di gestori di ridimensionamento.

## 1. Il problema in produzione

Adaptive Operations Console ora ha un tema e CSS circoscritto, ma il suo shell sposta ancora i controlli a mano: un gestore **Resize** ricalcola **Bounds** per la toolbar, la barra di navigazione, la griglia dei ticket e l'editor dei dettagli ogni volta che il browser cambia dimensione. Funziona alla larghezza che lo sviluppatore ha provato, si rompe a tutte le altre e non può reggere i profili tablet e telefono che introduce il Modulo 6. Il design adattivo in Wisej.NET si basa su contenitori e regole di layout, non su coordinate: regioni che occupano i bordi, un'area di lavoro che riempie ciò che resta, ed editor che scorrono invece di sovrapporsi.

## 2. Concetti chiave

| Concetto                      | Perché è importante                                                                                                                                                                                                                                                        |
|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| motore di layout predefinito  | Ogni controllo Wisej.NET ha un motore di layout; quello predefinito supporta posizione assoluta, docking e ancoraggio, così una schermata può combinare il posizionamento preciso nel designer con regioni che si ridimensionano da sole.                                  |
| priorità di docking           | I figli in dock occupano lo spazio nell'ordine in cui compaiono nella collezione dei controlli del genitore, ed è per questo che gli stessi cinque pannelli producono uno shell corretto o uno sovrapposto a seconda dell'ordine dei figli.                                |
| DisplayRectangle e Padding    | I controlli in dock vengono disposti dentro il <code>DisplayRectangle</code> del genitore, che il <code>Padding</code> riduce; il motore predefinito ignora il <code>Margin</code> per i figli in dock, quindi gli spazi vengono dal padding o da controlli distanziatori. |
| ancoraggio                    | Un ancoraggio mantiene la distanza di un controllo dai lati selezionati del genitore, il che si adatta alle regioni simili a form: label in alto a sinistra, caselle di testo left-right, pulsanti di comando in basso a destra.                                           |
| AutoSize e layout circolare   | Un contenitore che si dimensiona in base a figli in dock o ancorati a esso non ha una risposta stabile; quel ciclo è il principale avvertimento del modulo su AutoSize.                                                                                                    |
| AutoScroll con barre nascoste | <code>AutoScroll</code> permette a un contenitore di scorrere quando il contenuto non ci sta, e <code>ScrollBars.Hidden</code> mantiene lo scorrimento al tocco sui telefoni senza disegnare barre di scorrimento.                                                         |
| regione UserControl           | Una regione dello shell con il proprio layout locale e una piccola superficie pubblica, così lo shell mostra, nasconde, mette in dock o sposta l'intera regione per profilo invece di conoscerne le coordinate interne.                                                    |

## 3. Il filo della lezione

### Docking, ordine dei figli e lo spazio che resta

Il docking attacca un figlio a un bordo del genitore ( `DockStyle.Top` , `Bottom` , `Left` , `Right` ) oppure gli fa riempire ciò che resta ( `DockStyle.Fill` ). Il motore distribuisce lo spazio ai figli in dock uno alla volta, seguendo il loro ordine nella collezione del genitore, e ogni figlio in dock riduce il rettangolo disponibile per il successivo. È tutto qui il segreto dello shell della console: la toolbar prende una striscia in alto, la regione di stato una in basso, la barra di navigazione una a sinistra, il pannello dei dettagli una a destra, e l'area di lavoro riempie ciò che resta. Cambia l'ordine e il pannello di stato nasconde l'ultima riga della griglia. Nel designer di Visual Studio, `BringToFront` , `SendToBack` e le operazioni sull'indice dei figli cambiano quell'ordine, quindi quando uno shell sembra sbagliato controlla l'ordine dei figli prima di toccare una dimensione. Il codice dello shell del pacchetto è volutamente breve, e il lab lo riproduce nel designer:

```
// MainPage.cs - lo shell della console, un Dock per regione
this.toolbarPanel.Dock = DockStyle.Top;    this.toolbarPanel.Height = 56;
this.statusPanel.Dock = DockStyle.Bottom; this.statusPanel.Height = 28;
this.navigationPanel.Dock = DockStyle.Left; this.navigationPanel.Width = 240;
this.detailsPanel.Dock = DockStyle.Right;  this.detailsPanel.Width = 360;

this.workspacePanel.Dock = DockStyle.Fill;
this.workspacePanel.MinimumSize = new System.Drawing.Size(320, 240);

// gli spazi tra le regioni in dock vengono dal contenitore, non da Margin
this.Padding = new Padding(8);
this.workspacePanel.Padding = new Padding(12, 8, 12, 8);
```

I figli in dock vengono collocati dentro il `DisplayRectangle` del genitore, e il `Padding` restringe quel rettangolo da ogni lato. Il motore di layout predefinito non usa il `Margin` per i controlli in dock: un margine sulla barra di navigazione non cambia nulla nello spazio accanto a essa, quindi dai un padding al contenitore oppure metti in dock un sottile pannello distanziatore. L'ancoraggio riguarda i controlli dentro una regione. Un ancoraggio conserva la distanza dai lati selezionati del genitore: label in alto a sinistra così restano ferme, caselle di testo left-right così crescono con il pannello dei dettagli, l'editor largo delle note top-left-right, e i pulsanti Save e Cancel in basso a destra. Un controllo senza ancoraggio su un asse resta libero e mantiene una posizione proporzionale, giusta per centrare una card di accesso e sbagliata per una griglia.

## Contenitori invece di gestori di ridimensionamento

Il `MainPage_Resize` della console è il segnale d'allarme che il modulo esiste per eliminare. Ogni riga che imposta `Bounds`, `Location` o `Size` in risposta a un ridimensionamento è una regola che il motore di layout avrebbe potuto applicare per te, e a differenza del motore viene eseguita dopo che il browser ha già disegnato il layout sbagliato, non sa nulla del padding e non si compone: nel momento in cui il pannello dei dettagli diventa uno `UserControl`, il gestore deve imparare anche i suoi dettagli interni. Il refactoring è meccanico: elenca le regioni, decidi il bordo che ciascuna occupa, aggiungile in quell'ordine con `Dock` ed elimina il gestore. Le coordinate assolute restano utili per un lavoro preciso nel designer dentro una finestra di dialogo a dimensione fissa; sono lo strumento sbagliato per una schermata che deve girare anche su un telefono.

Uno shell duraturo è un insieme di regioni annidate: barra dei comandi in alto, navigazione a sinistra, area di lavoro al centro, dettagli a destra, stato in basso. Ogni regione è un `Panel`, `UserControl`, `TabControl`, `SplitContainer` o `Page` con il proprio layout locale, ed è l'annidamento che rende stabile lo shell: all'area di lavoro non importa come la toolbar dispone i suoi pulsanti, e alla toolbar non importa che l'area di lavoro contenga card delle metriche sopra una griglia dei ticket. Anche ancorare molti controlli in combinazioni complicate è un segnale: una `TableLayoutPanel` o una `FlexLayoutPanel`, entrambe trattate nel Modulo 5, esprimono campi allineati o regioni proporzionali in modo più chiaro di dodici impostazioni di ancoraggio.

## AutoSize, AutoScroll e i limiti di dimensione

`AutoSize` è prezioso dove è il contenuto a decidere la dimensione: una label che deve contenere un testo tradotto, un badge che cresce con il suo numero, una card che avvolge contenuto dinamico;

`AutoSizeMode` decide se il controllo può solo crescere o anche restringersi. Il pericolo è il layout circolare. Se un contenitore si dimensiona automaticamente in base a figli in dock o ancorati a quello stesso contenitore, il genitore chiede ai figli quanto devono essere grandi, i figli chiedono al genitore quanto sono grandi, e il motore non ha una risposta stabile. L'incidente tipico è un pannello con `AutoSize` che riempie anche il suo genitore. Tieni il dimensionamento automatico sui controlli foglia e sui contenitori i cui figli hanno dimensioni fisse o guidate dal contenuto.

`AutoScroll` è la risposta sicura quando il contenuto non ci sta: il contenitore scorre invece di allargare lo shell. L'editor dei dettagli della console ha più campi di quanti ne possa mostrare lo schermo di un telefono, quindi il lab attiva `AutoScroll` e, alle larghezze strette, imposta `ScrollBars` su `Hidden`, che mantiene lo scorrimento al tocco senza disegnare barre di scorrimento che rubano larghezza.

`AutoScrollMargin` aggiunge spazio dopo l'ultimo controllo e `AutoScrollMinSize` garantisce un'area scorrevole minima. `MinimumSize` e `MaximumSize` sono gli altri limiti di sicurezza: un `MinimumSize` sull'area di lavoro e sulla griglia fa sì che la regione su Fill non possa mai ridursi a una striscia quando il browser si restringe, e un `MaximumSize` impedisce al pannello dei dettagli di inghiottire un monitor ultra-wide.

## Regioni UserControl che lo shell può riorganizzare

L'ultimo passo del lab è quello che ripaga nel Modulo 6. Una volta che la barra di navigazione e l'editor dei dettagli sono `UserControl`, ciascuno con il proprio docking e ancoraggio interno, lo shell smette di conoscere le coordinate interne di qualsiasi editor. Conosce cinque regioni, e per profilo mostra, nasconde, mette in dock o sposta regioni intere: la barra di navigazione può diventare una stretta striscia di icone, l'editor dei dettagli può spostarsi sotto la griglia su un tablet o aprirsi in un `Form` modale su un telefono, e nessuno di questi spostamenti tocca una casella di testo. La regione espone pochi membri significativi, come una proprietà `Ticket` e un evento `Saved`, e tiene privati i suoi controlli figli così che una rinomina interna non rompa mai la pagina.

La checklist di layout del modulo è breve: preferisci i contenitori ai gestori di ridimensionamento, controlla l'ordine di docking, metti il `Padding` sulle regioni in dock, imposta `MinimumSize` sulle regioni su Fill, e lascia scorrere il contenuto che non ci sta. L'elenco dei segnali d'allarme per il refactoring del pacchetto è l'immagine speculare di quella checklist, e rivedere una schermata rispetto a esso è una buona abitudine prima di ogni test sui profili.



### Segnali d'allarme per il refactoring

Codice `Resize` manuale che imposta ripetutamente `Bounds` su molti controlli. `Dock` e `Anchor` mescolati su controlli dove si contrastano. Coordinate assolute in una schermata che deve girare sui telefoni. Griglie senza `MinimumSize`. Pannelli con `AutoSize` e docking `Fill` che creano cicli. Layout di form che nascondono controlli invece di spostare il contenuto a bassa priorità in una sezione scorrevole.



### **La regola del sistema visivo**

Usa il meccanismo Wisej.NET che possiede il problema: il tema per l'identità dei controlli standard, AppearanceKey per le varianti semantiche, CssClass per il CSS circoscritto e specifico dell'applicazione, CssStyle solo per valori dinamici una tantum, i contenitori di layout invece del codice di ridimensionamento, le proprietà responsive per i valori di profilo impostati nel designer, e ResponsiveProfileChanged per il comportamento che non è semplicemente visivo.



### **Obiettivi di apprendimento**

Alla fine di questo modulo sarai in grado di:

- ✓ Usare docking e ancoraggio in modo intenzionale e capire come l'ordine dei figli influisce sulla priorità di docking.
- ✓ Comporre le schermate con contenitori annidati invece che con gestori di ridimensionamento e aggiornamenti manuali delle coordinate.
- ✓ Usare in modo sicuro AutoSize, AutoSizeMode, AutoScroll, ScrollBars, MinimumSize, MaximumSize, Padding e Margin.
- ✓ Creare regioni UserControl riutilizzabili che possono essere riorganizzate tra i profili desktop, tablet e telefono.

## Guida al lab / esame

Che cosa costruirai nel lab pratico, l'approccio suggerito e i deliverable richiesti.



### Lab pratico

Riorganizza lo shell di Adaptive Operations Console in contenitori annidati: una toolbar in alto, una barra di navigazione a sinistra, un'area di lavoro su Fill, un pannello dei dettagli a destra e una regione di stato, ciascuno un Panel o uno UserControl con il proprio layout locale. Elimina ogni gestore Resize che imposta Bounds, correggi l'ordine di docking nel designer, usa Padding per gli spazi tra le regioni in dock, dall'area di lavoro e alla griglia un MinimumSize, e fai scorrere l'editor dei dettagli con AutoScroll e barre di scorrimento nascoste alle larghezze strette.

## Punto di partenza

Apri la soluzione `AdaptiveOperationsConsole` come l'hai lasciata dopo il Modulo 3: la `Page` principale porta il tema AdaptiveOps, il foglio di stile circoscritto e il cambio di tema per sessione, ma le sue cinque regioni sono ancora semplici pannelli i cui `Bounds` vengono ricalcolati in `MainPage_Resize`, e l'editor dei dettagli si sovrappone alla griglia dei ticket sotto i 1100 pixel circa. Esegui una volta alla larghezza desktop, poi restringi il browser trascinandolo e annota che cosa viene tagliato, si sovrappone o scompare prima di cambiare qualsiasi cosa.

## Approccio suggerito

- ✓ Elenca ogni metodo `Resize`, `Layout` o di caricamento che imposta `Bounds`, `Location` o `Size`, scrivi negli appunti del lab le cinque regioni e il bordo che ciascuna occupa, poi elimina quel codice prima di toccare il designer.
- ✓ Nel designer imposta `Dock` sui cinque pannelli (toolbar Top, stato Bottom, navigazione Left, dettagli Right, area di lavoro Fill) e correggi l'ordine dei figli con `BringToFront` e `SendToBack` finché i bordi non vengono occupati per primi e l'area di lavoro prende ciò che resta.
- ✓ Dai alla pagina e alle regioni un `Padding` per gli spazi tra le aree in dock, imposta `MinimumSize` sull'area di lavoro e sulla griglia dei ticket, e verifica che il motore predefinito ignori il `Margin` sui figli in dock.
- ✓ Dentro il pannello dei dettagli ancora le label in alto a sinistra, gli editor left-right, l'editor delle note top-left-right e i pulsanti di comando in basso a destra; attiva `AutoScroll`, imposta `ScrollBars` su `Hidden` per le larghezze strette e regola `AutoScrollMargin`.
- ✓ Estrai `NavigationRail` e `DetailsEditor` come `UserControl` con il proprio layout locale e una piccola superficie pubblica, reinseriscili nello shell e ridimensiona il browser dalla larghezza desktop a quella telefono per verificare che nulla si sovrapponga o venga tagliato.

## Deliverable richiesti

- ✓ Shell ricostruito con contenitori annidati con le regioni toolbar, navigazione, area di lavoro, dettagli e stato
- ✓ Rimosso tutto il codice Resize manuale che impostava Bounds

- ✓ Ordine di docking, Padding e MinimumSize impostati in modo intenzionale su ogni regione
- ✓ Editor dei dettagli che scorre tramite AutoScroll con barre di scorrimento nascoste alle larghezze strette
- ✓ Regioni di navigazione e dettagli estratte in UserControl

## Domande di revisione per lo studente

- ✓ Se nell'ordine dei figli il pannello dell'area di lavoro fosse messo in dock prima del pannello di stato, che cosa vedrebbe l'utente in fondo alla griglia, e quale comando del designer lo corregge senza cambiare una sola dimensione?
- ✓ Hai impostato un `Margin` di 12 pixel sulla barra di navigazione e lo spazio non è comparso. Perché, e quali due meccanismi producono invece uno spazio tra le regioni in dock?
- ✓ L'editor dei dettagli è uno `UserControl` con `AutoScroll`. Quali dei suoi membri deve conoscere lo shell per spostarlo sotto la griglia su un tablet, e di quali non deve mai avere bisogno?



### La regola del sistema visivo

Usa il meccanismo Wisej.NET che possiede il problema: il tema per l'identità dei controlli standard, AppearanceKey per le varianti semantiche, CssClass per il CSS circoscritto e specifico dell'applicazione, CssStyle solo per valori dinamici una tantum, i contenitori di layout invece del codice di ridimensionamento, le proprietà responsive per i valori di profilo impostati nel designer, e ResponsiveProfileChanged per il comportamento che non è semplicemente visivo.