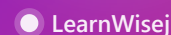


Proprietà, eventi e codice generato dal Designer

Che cosa genera il Designer, dove va il tuo codice e come collegare in modo pulito l'evento di un pulsante.



Obiettivo della lezione

Capisci come funzionano i controlli dopo averli posizionati: imposta le proprietà, collega gli eventi e tieni il tuo codice separato dal file generato dal Designer.

1. Proprietà, eventi e metodi

Una volta che un controllo è sulla pagina, tre concetti descrivono come ci lavori:

Concetto	Significato	Esempio
Proprietà	Un'impostazione o un valore di un controllo	<code>btnStart.Text = "Start"</code>
Evento	Qualcosa che succede, spesso a causa dell'utente	<code>btnStart.Click</code>
Gestore di eventi	Il metodo che viene eseguito quando l'evento si verifica	<code>btnStart_Click(...)</code>
Metodo	Un comando che chiami per far succedere qualcosa	<code>lstEventLog.Items.Add(...)</code>

2. Dove va il tuo codice

Una pagina come `DashboardWindow` è suddivisa in più file. Sapere quale file governa che cosa mantiene pulito il progetto.

File	Che cosa contiene
<code>DashboardWindow.cs</code>	I tuoi gestori di eventi, i metodi helper e la logica leggibile.
<code>DashboardWindow.Designer.cs</code>	Il codice di layout generato dal Designer. Non modificarlo a mano.
<code>Program.cs</code>	Il codice di avvio che apre la prima pagina o finestra.



IntelliSense e la documentazione delle API

Mentre scrivi, IntelliSense mostra le proprietà, i metodi e gli eventi disponibili su un controllo — `Text`, `Enabled`, `Click`, `Show`, `Hide`. Quando ti servono più dettagli, ricorri alla documentazione delle API.

3. Il ciclo di vita per chi inizia

Capire l'ordine in cui accadono le cose previene bug che confondono — come provare a usare un controllo prima che esista.

```
L'app parte da
→ viene creata la prima pagina o finestra
→ costruisce i controlli dal Designer
→ l'utente fa clic sui pulsanti o modifica gli input
→ i vengono eseguiti in risposta
```

4. Collega l'evento Click di un pulsante

Fai doppio clic su un pulsante nel Designer per crearne il gestore `Click`. Nel lab i pulsanti della dashboard non hanno bisogno di un vero database o di una vera API — l'obiettivo è un **comportamento guidato dagli eventi**: un click cambia label, indicatori di stato e messaggi di log.

```
private void btnStart_Click(object sender, EventArgs e)
{
    lblStatus.Text = "Status: Running";
    lblServerStatus.Text = "Server: Online";
    lblDatabaseStatus.Text = "Database: Online";
    lblApiStatus.Text = "API Service: Online";
    AddLog("Dashboard started.");
}

private void AddLog(string message)
{
    lstEventLog.Items.Add(DateTime.Now.ToString("hh:mm tt") + " - " + message);
}
```

Nota che il logging ripetuto è stato spostato in un piccolo helper `AddLog()`, così ogni gestore resta breve e leggibile.

5. Abitudini sicure nella gestione degli eventi

- Usa i gestori di eventi per le azioni dell'utente, come i click sui pulsanti.
- Mantieni ogni gestore facile da leggere.
- Sposta la logica ripetuta in metodi helper come `AddLog()`.
- Non modificare a mano il codice di layout generato dal Designer, a meno che tu non sappia esattamente perché.
- Non nascondere chiamate lente a database o API dentro semplici eventi dell'interfaccia, a meno che l'attività non lo richieda espressamente.



Sei pronto per il lab

Prossimo passo: guarda la videoguia sulla dashboard, supera la verifica delle conoscenze, poi costruisci ed esegui tu stesso la pagina dashboard.