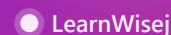


Getting Started on Wisej.NET

From an empty Visual Studio to a running app — and the habits that keep code clean.



Lesson goal

Create a first Wisej.NET project, open it in the Designer, run it locally, inspect the solution structure, and learn where the startup and configuration files live.

1. Start with the project template

In Visual Studio, create a new project from the installed Wisej templates. The exact list varies by version, but the workflow is the same: **create a new project** → **search for “Wisej”** → **pick a Wisej web application template** → **name it** → **create**.

Suggested practice project name

WisejTrainingApp



Before you start

Make sure the Wisej.NET Visual Studio extension and templates are installed. If the templates don't appear, re-check the installation or restart Visual Studio.

2. Open the Wisej Designer

The Designer is where you visually place controls. Open the main page or window and switch to design view, then drag controls from the Toolbox, set properties, and double-click a control to create an event handler.

Designer task	What you do
Add a control	Drag a <code>Button</code> , <code>Label</code> , <code>TextBox</code> , <code>Panel</code> or <code>DataGridView</code> onto the surface.
Set text	Select the control, open Properties, change the <code>Text</code> property.
Rename a control	Change <code>Name</code> to something clear like <code>btnRun</code> , <code>lblStatus</code> , <code>txtCustomerName</code> .
Create click code	Double-click a button (or use the Events panel) to create a <code>Click</code> handler.
Adjust layout	Use <code>Dock</code> , <code>Anchor</code> , <code>Size</code> , <code>Location</code> , <code>Margin</code> , <code>Padding</code> .

3. Run it locally

Running locally means testing on your own computer before publishing. Use the normal Start / Debug button — the project builds, a local server starts, and the app opens in the browser.

1. Click **Start / Debug** in Visual Studio.
2. Wait for the project to build.
3. Confirm the browser opens the Wisej.NET application.
4. Test a button click to confirm your code runs.
5. Use breakpoints to inspect event handlers and variables.



Troubleshooting habit

If the app doesn't open, read the Visual Studio **Error List** and **Output** window first. Most beginner problems are missing packages, build errors, the wrong startup project, or a renamed file whose references weren't updated.

4. Inspect the solution structure

The Solution Explorer is not random files — it shows how the app is organized.

File or folder	What it's for
<code>Program.cs</code>	Startup code that begins the application and often shows the first Page or Window.
<code>Page1.cs</code> / <code>Window1.cs</code>	Code-behind: where you write event handlers and screen behavior.
<code>*.Designer.cs</code>	Designer-generated layout and control initialization. Don't edit manually.
<code>*.resx</code>	Resource files for the designer and localized data.
Config files	<code>Web.config</code> plus Wisej-specific JSON configuration.
<code>Models/</code>	Simple data classes such as <code>Ticket.cs</code> or <code>Customer.cs</code> .
<code>Services/</code>	Business logic such as <code>TicketService.cs</code> .

5. Good developer habits

Habit	Avoid	Prefer
Name controls clearly	<code>button1</code> , <code>label2</code>	<code>btnSaveTicket</code> , <code>lblStatus</code>
Keep events readable	200 lines in <code>btnSave_Click</code>	Call <code>ValidateInput()</code> and <code>ticketService.Save(ticket)</code>
Separate business logic	All save logic in the page	A <code>TicketService</code> owns the rules
Organize files	Everything in the project root	Models, Services, Pages, Controls folders

6. Example: a readable event handler

An event handler should read like a short story — get the input, call logic, update the UI:

```
private void btnSayHello_Click(object sender, EventArgs e)
{
    string name = txtName.Text.Trim();

    if (string.IsNullOrEmpty(name))
    {
        lblStatus.Text = "Please enter a name.";
        return;
    }

    lblStatus.Text = $"Hello, {name}!";
}
```

As the app grows, move rules and data operations into service classes:

```
private readonly TicketService ticketService = new TicketService();

private void btnSaveTicket_Click(object sender, EventArgs e)
{
    Ticket ticket = ReadTicketFromScreen();
    ticketService.Save(ticket);
    lblStatus.Text = "Ticket saved.";
}
```



You're ready for the lab

Next: watch the build walkthrough, pass the knowledge check, then create and run your own first Wisej.NET screen in the hands-on lab.