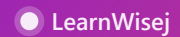


AI Coding Exercise

Build this module's VisualOperationsStudio sample with ChatGPT or Claude from a ready-made prompt that points the model at this module's specification, then review, run and extend what it gives you.



The assistant is the fast pair of hands. You are still the architect.

An assistant can produce this module's sample in minutes. What it cannot do is decide whether the result is the right design for your application — that call stays with you. You supply the judgement: brief it properly, then check what comes back against what Module 2 just taught you.

What you are asking for

Add a reusable TelemetryGauge control to VisualOperationsStudio that paints a ring, coloured threshold zones, a needle, a numeric value and a caption. Give it Value, Minimum, Maximum, WarningThreshold and CriticalThreshold properties whose setters clamp the input, return early when nothing changed and call Invalidate() once. Put the coordinate maths in a pure GaugeGeometry helper that takes a Rectangle and returns the ring bounds, sweep angles, needle points and label rectangle, then draw from those values with using-scoped pens, brushes and a GraphicsPath, leaving the supplied Graphics alone. Handle a zero-sized control, an inverted range and a value outside the range without throwing, and expose the reading through AccessibleDescription so it is not pixels-only.

The module asks for these deliverables, and the prompt below asks the assistant for exactly them:

- ✓ TelemetryGauge control with clamping Value, Minimum, Maximum and threshold properties that Invalidate() once per real change
- ✓ Pure GaugeGeometry helper returning ring bounds, sweep angles, needle points and the label rectangle
- ✓ Paint handler that disposes every pen, brush, font and path it creates and never disposes e.Graphics
- ✓ Guards for zero size, inverted range and out-of-range values, with an AccessibleDescription carrying the reading
- ✓ PaintCost.md noting the repaint size, how often it repaints and when a themed control would be cheaper

Why it is shaped this way

Module 1 put one telemetry reading on four surfaces so you could see where each pixel is produced. VisualOperationsStudio now wants that reading in six places, and copying a Paint handler into six panels is how a screen ends up with six slightly different gauges, six pens that are never disposed, and a repaint whenever an unrelated field changes. The fix is to stop writing handlers and write a control: one

TelemetryGauge whose public properties are the only thing that can change its picture, and whose cost you can state in kilobytes.

These are the terms the assistant has to get right. If what comes back uses them loosely, it has not understood the module:

Term	What it means here
Wisej.Web.PaintEventArgs	The argument of the Paint event; it hands you a System.Drawing.Graphics you did not create and a rectangle to draw inside, and your code runs on the server.
ClipRectangle	The region Wisej.NET provided for this paint operation; layout can consult the control's client size, but drawing stays bounded by the rectangle you were given.
GaugeGeometry helper	A pure function from a Rectangle plus a normalised value to ring bounds, sweep angles, needle points and a label rectangle, so the maths can be checked without a live control.
Invalidate()	The one call that asks Wisej.NET to re-render the control's surface; a property setter makes it exactly once, and only when the change can alter a pixel.
Resource ownership	The handler disposes every Pen, Brush, Font, GraphicsPath and Bitmap it creates, and never disposes the Graphics the framework supplied.
Paint cost budget	Three numbers per painted control: server CPU spent drawing, server memory for the temporary image, and the bytes the browser must fetch and decode.
AccessibleDescription	Carries the same reading as text, so a value that exists as a drawn arc is still available to a user who never sees the arc.

Build it — pick your assistant

Both buttons open a new tab with the **same ready-made prompt** already filled in. The prompt points the model at this module's specification — a single Markdown file holding the lesson, the lab brief and the deliverables — so it works from the real material instead of guessing what Module 2 is about. You can open [module-02.md](#) yourself to see exactly what it will read.



Use ChatGPT

Opens [chatgpt.com](#) with this module's prompt loaded. Works with GPT-5 or Codex; any plan that can browse the web will fetch the spec.



Use Claude

Opens [claude.ai](#) with the same prompt loaded. Claude Code users can paste it into a terminal in an empty folder instead.



Send it as-is and you get a project, not a snippet

The prompt ends by asking for the whole solution as a single `.zip` with build-and-run steps. Download it, open the solution in Visual Studio, press F5 and VisualOperationsStudio comes up on `localhost`. You need the .NET SDK and the `Wisej-4` NuGet package; everything else the prompt asks the assistant to write itself.

The prompt, in full — read it, then edit it if you want a different sample

I am working through Module 2 – "Mastering the Control Paint Event" – of the LearnWisej Drawing & Painting in Wisej.NET course, and I want to build the module sample with you.

Start by reading the module specification. It is a single Markdown file holding the full lesson, the lab brief and the deliverables:

<https://learnwisej.com/assets/courses/drawing-and-painting/specs/module-02.md>

Then build a complete Wisej.NET 4 project I can open in Visual Studio:

1. Project "VisualOperationsStudio", C#, net8.0 or later, NuGet package Wisej-4 4.1.x, Kestrel host calling app.UseWisej().
2. Build what the specification's lab asks for: Add a reusable TelemetryGauge control to VisualOperationsStudio that paints a ring, coloured threshold zones, a needle, a numeric value and a caption. Give it Value, Minimum, Maximum, WarningThreshold and CriticalThreshold properties whose setters clamp the input, return early when nothing changed and call Invalidate() once. Put the coordinate maths in a pure GaugeGeometry helper that takes a Rectangle and returns the ring bounds, sweep angles, needle points and label rectangle, then draw from those values with using-scoped pens, brushes and a GraphicsPath, leaving the supplied Graphics alone. Handle a zero-sized control, an inverted range and a value outside the range without throwing, and expose the reading through AccessibleDescription so it is not pixels-only.
3. Produce these deliverables, and tell me where each one lives in the solution:
 - TelemetryGauge control with clamping Value, Minimum, Maximum and threshold properties that Invalidate() once per real change
 - Pure GaugeGeometry helper returning ring bounds, sweep angles, needle points and the label rectangle
 - Paint handler that disposes every pen, brush, font and path it creates and never disposes e.Graphics
 - Guards for zero size, inverted range and out-of-range values, with an AccessibleDescription carrying the reading
 - PaintCost.md noting the repaint size, how often it repaints and when a themed control would be cheaper
4. Follow the specification wherever it names a Wisej.NET class, property or event – use exactly that one. Where it leaves a choice open, tell me which API you picked and why.
5. Keep the server authoritative: state and validation live in .NET, and the browser renders and reports. Never move a decision into JavaScript because it is easier there.
6. Make the result demonstrable: I should be able to run it and see the behaviour this module is about, including what happens when something goes wrong.

Before you write any code, restate in your own words what you are going to build and where each piece will live, so I can correct the design while it is still cheap.

When the code is done, package the whole solution as a single .zip I can download, and give me the exact steps to build and run it.

Finally, answer these questions from the specification, using your own implementation as the evidence:

- A colleague moves the ring and needle maths directly into the Paint handler because "it is only a few lines". Which two things become impossible to do cheaply afterwards, and which one of them does Module 6 depend on?
- Your gauge is bound to a machine record that also carries an operator name and a maintenance date. If those fields change, should the control invalidate? Explain your answer

in terms of visual dependencies rather than in terms of the model.

– The same gauge is going onto a dashboard that shows sixty of them and refreshes every two seconds. What would you measure first, and what would you change if the measurements were bad?

Ask me about anything that is ambiguous before you start.

Then review what came back

Code that compiles is not yet code that is right. The interesting questions are design questions: is this the Wisej.NET control the job actually calls for, does it follow the lifecycle the framework gives you, is the authoritative state on the server rather than parked in the browser because that was convenient? Read the result as a reviewer, not as a customer.

This module gives you the checklist. Answer each one against the code you just received:

- ✓ A colleague moves the ring and needle maths directly into the Paint handler because "it is only a few lines". Which two things become impossible to do cheaply afterwards, and which one of them does Module 6 depend on?
- ✓ Your gauge is bound to a machine record that also carries an operator name and a maintenance date. If those fields change, should the control invalidate? Explain your answer in terms of visual dependencies rather than in terms of the model.
- ✓ The same gauge is going onto a dashboard that shows sixty of them and refreshes every two seconds. What would you measure first, and what would you change if the measurements were bad?

Anything you had to correct is worth writing down. A short note of the conventions and decisions you settled on, pasted into the first message of your next session, is worth more than any single clever prompt.



No reference sample for this course yet — yours is the first

The [LearnWisej samples repository](#) carries worked builds for most courses, but not yet for this one. That means there is nothing to compare against and no shortcut: the specification and your own review are the whole safety net. Keep what you build — and if it turns out well, we would like to see it.

Three prompts to take it further

Point your assistant at the project you just built and try these. Each one is a review-and-extend exercise, not a rewrite.

- ✓ **Prove it actually works.** *"Go through this project deliverable by deliverable against the module specification at <https://learnwisej.com/assets/courses/drawing-and-painting/specs/module-02.md> and show me the evidence that each one is genuinely done — the code path, what to click, and what I should see. Where a deliverable is only partly met, say so plainly instead of arguing it through. Then add whatever is missing."*
- ✓ **Review it like a pull request.** *"Walk me through this VisualOperationsStudio code as if you were reviewing a colleague's pull request and did not write it. Where does state live, what happens end to end on one user action, what would you send back for changes, and which parts would you not*

be comfortable putting in front of a customer? Be specific and quote the lines you mean — do not summarise the design back to me."

- ✓ **Swap a piece without moving the seams.** *"Pick the part of this VisualOperationsStudio project that is most obviously a stand-in for something real — a fake data source, an in-memory store, a simplified service — and replace it with a production-shaped implementation. The contract either side of it must not change: same types, same calls, same behaviour under failure. Tell me what the real implementation forced you to handle that the stand-in did not."*



Carry it into the lab

The hands-on lab that closes this module asks for the same deliverables you just had the assistant produce. Use what you have here as the shape of your answer and the evidence behind it — not as the answer itself. The lab is graded on your reasoning, and that part has to be yours.