WISEJ.NET 2.0 CONTROLS AND A NORTHWIND APPLICATION · MODULE 2 · A SIDE BUTTON THAT OPENS AND CLOSES A DETAIL PANEL

AI Coding Exercise

Use the course-grounded prompt, review the generated patch and record verification evidence.

LearnWisej

AI Coding Exercise

Implement module 2 as a bounded patch, then verify the result against the lab acceptance criteria.

Before prompting

Provide your current practice page and the included example files named below. Keep a copy of the prompt, response, reviewed diff and verification evidence. This exercise works with a coding assistant of your choice.

Ready-to-use prompt

```
Help me learn Wisej.NET 2.0 controls in C#. Explain the properties and event path before
implementing a change.
Module 2: A Side Button That Opens and Closes a Detail Panel
Use the original files included in this course:
code/NorthwindCourse/CollapsibleSideButton.cs. Ask for their contents if unavailable.
Compatibility: .NET Framework 4.8, Wisej-2 2.0.54 and Wisej-2-NavigationBar 2.0.54. The web
application host comes from the installed Wisej 2 template. CollapsibleSideButton is
supplied course code; TagTextBox is the tagged editor; Switch is a CheckBox appearance.
Task: Add the supplied CollapsibleSideButton to a practice Page and use it to control one
right-docked detail panel.
Acceptance criteria:
1. Add the supplied CollapsibleSideButton.cs file and read its state setter.
2. Create one button and one right-docked Panel with a visible label.
3. Initialize panel visibility from Collapsed and subscribe once to CollapsedChanged.
4. Test clicking, programmatic reset and assigning the same state twice.
5. Record state, caption, visibility and event counts at wide and narrow widths.
Initialize controls once, keep mutable data per Page, use decimal amounts and apply the
documented inclusive date rules. Explain the exact files to change. Give one small patch and
independent tests with expected results. Do not claim a command or browser check ran unless
it did.
Evidence: An explanation of the original component's property and event; A working button-
to-panel connection; A transition table for open close and reset; An event count proving
repeated assignment is ignored; Keyboard and narrow-width observations.
Review questions: Why does the host use the inverse of Collapsed? Which value is
authoritative when the page resets? What prevents the same assignment from causing repeated
work?
```

Review and verify

- ✓ Add the supplied CollapsibleSideButton.cs file and read its state setter.
- ✓ Create one button and one right-docked Panel with a visible label.
- ✓ Initialize panel visibility from Collapsed and subscribe once to CollapsedChanged.
- ✓ Test clicking, programmatic reset and assigning the same state twice.
- ✓ Record state, caption, visibility and event counts at wide and narrow widths.

Challenge the generated result

- ✓ Why does the host use the inverse of Collapsed?
- ✓ Which value is authoritative when the page resets?
- ✓ What prevents the same assignment from causing repeated work?

Record your decision

For every proposed file, record accept, revise or reject with a reason. Record exact build output and observed browser results. A plausible answer is not evidence that the application ran.

Included course examples

`code/NorthwindCourse/CollapsibleSideButton.cs`

These original files are included in the code download. Build the exercise before comparing with the completed example.