WISEJ.NET 2.0 CONTROLS AND A NORTHWIND APPLICATION · MODULE 6

Bind a DataGridView and Validate Shared Filters

Lesson Guide · Lab / Exam Guide

LearnWisej

Lesson Guide

Prepare for the final application by binding a small order model to DataGridView and combining tags, dates and the switch through one tested filter method.

> **Control learning focus**
>
> Prepare for the final application by binding a small order model to DataGridView and combining tags, dates and the switch through one tested filter method.

1. What you will learn

The earlier modules established how the controls work. This module supplies the data layer they will share. Use the included six-row teaching dataset to learn binding and independently verify the filter rules before combining everything in the Northwind-style application.

2. Key concepts

Concept	Why it matters
OrderRow	A small typed business record independent of the controls.
DataPropertyName	Maps a displayed grid column to a model property.
DataSource	Supplies the records displayed by the grid.
DataBoundItem	Returns the model object associated with a bound row.
Date boundaries	The course includes orders on both the start and end dates.
Pure filter method	Inputs determine outputs without modifying the original records.

3. Lesson narrative

Read the data model before binding

Open OrderData.cs and identify OrderRow.OrderId, Customer, Country, Ordered, Shipped and NetAmount. OrderId identifies the record; Customer is display data. NetAmount uses decimal because it represents money. The included records are original synthetic teaching data, not a full business database.

OrderData.Create returns a new list with new records. Keep the list in a Page instance field so each application session gets its own example state. Avoid a mutable static list shared between learners. The

filtering method may be static because it receives its input and does not retain mutable session state.

Write down the independent baseline: six orders, net amount 700.00, dates from 1 through 6 May 1998. There are two orders each for Germany, UK and USA. Three orders are unshipped. These small values let you calculate expectations by hand before trusting the screen.

Configure and bind the grid

Create a DataGridView and set AutoGenerateColumns to false. Add DataGridViewTextBoxColumn objects for OrderId, Customer, Country and NetAmount. Set each DataPropertyName to the exact corresponding model property. HeaderText is the user-facing label and can be changed independently.

Set ReadOnly to true for this exercise and use FullRowSelect. Assign the filtered List<OrderRow> to DataSource. If a column is empty, inspect the property spelling and the actual object type before changing the query. Binding names are a contract between the grid and the model.

Handle SelectionChanged and read grid.CurrentRow.DataBoundItem as OrderRow. Check for a missing CurrentRow and a failed cast before displaying details. A row's visual index is not its business identifier: sorting or filtering can change the displayed position while OrderId remains stable.

Apply one filter pipeline

Use the TagTextBox text, the CheckBox.Checked value and two DateTimePicker.Value values as inputs to OrderData.Filter. The method normalizes countries, validates the date range, applies the date and shipment predicates, and returns matching records ordered by OrderId.

The example compares dates with Date so that both boundary days are included. A start date after the end date is rejected with an ArgumentException and a clear message. Do not silently swap user input or show an unexplained empty list. Define the date convention before implementing the UI.

Use the returned collection for the grid, count and sum. Germany should produce two orders totaling 270.00. Germany with Open orders only should produce one totaling 150.00. A date range of 1 through 3 May should produce three orders totaling 410.00. These checks verify combinations, not only isolated controls.

Handle empty results and preserve clear feedback

An unknown country produces an empty collection, not an exception. Show zero matching orders and a message explaining that the learner can clear a filter. Clear the detail view when there is no current record. Do not leave the previously selected customer's information visible as though it still matches.

When the date range is invalid, show the validation message and keep the last valid data explicitly unchanged. The sample status text explains the problem. A more elaborate application may disable data actions until the range is corrected; the important point is to avoid silently presenting an invalid query as successful.

Run the included data checks and compare their results with the browser. Unit checks prove the pure filter's behavior; they do not prove column sizing, keyboard selection or detail-panel placement. Record both kinds of evidence before beginning the complete application.

Learn the control, then verify the implementation

Explain the control properties, data binding and event path yourself. Give AI the supplied example and one bounded task. Review the patch, compile against the published Wisej 2.0.54 packages, test independent expectations and observe browser behavior.

Learning objectives

By the end of this module you'll be able to:

- ✓ Bind OrderRow objects through explicit DataGridView columns.
- ✓ Explain DataPropertyName and retrieve a selected DataBoundItem.
- ✓ Apply country date and open-order filters through one pure method.
- ✓ Verify totals inclusive date boundaries invalid ranges and empty results.

Included course examples

`code/NorthwindCourse/OrderData.cs` · `code/NorthwindCourse/NorthwindPage.cs`

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

Lab / Exam Guide

Build the module slice, verify its behavior and submit the required evidence.

Hands-on lab

Bind the supplied orders to explicit DataGridView columns and prove that one filter method supplies consistent rows counts totals and selected-record details.

Starting point

Use a separate practice Page so you can understand this control in isolation. Use the published Wisej 2.0.54 packages and the supplied original examples. A licensed browser run is required for UI evidence.

Suggested approach

- ✓ Inspect the six original OrderRow records and calculate their baseline count and total.
- ✓ Create explicit read-only grid columns with correct DataPropertyName values.
- ✓ Connect tags switch and date inputs to OrderData.Filter through one refresh method.
- ✓ Read the selected OrderRow from DataBoundItem with null handling.
- ✓ Test combined filters inclusive boundaries an invalid date range and an empty result.

Required deliverables

- ✓ A model-to-column mapping table
- ✓ A grid bound to the six-row baseline
- ✓ Expected and observed results for three combined-filter cases
- ✓ Selected-record details identified by OrderId
- ✓ Data-test output and browser evidence for invalid and empty cases

Student review questions

- ✓ Why is DataPropertyName different from HeaderText?
- ✓ Why should selection use the bound record instead of a row position?
- ✓ Which checks require a browser as well as a pure data test?

Acceptance and scoring

Submit five evidence items, each worth 20 points. A working claim with repeatable evidence earns 20, a partial implementation with a documented gap earns 10, and missing or contradictory evidence earns 0. Pass at 80/100. The in-player code checks are pattern checks, not a compiler or proof of runtime correctness; the instructor or reviewer must inspect the evidence.

Control configuration example

This original one-shot fixture demonstrates control configuration. Run it once in a scratch Page. In the final application, configure existing designer controls and refresh their data instead of creating duplicate

controls on each click. Use System, System.Linq, Wisej.Web, Wisej.Web.Ext.NavigationBar and the supplied NorthwindCourse namespace as needed. Reuse existing controls in the final application.

```
private void btnRun_Click(object sender, EventArgs e)
{
    try {
        var grid = new DataGridView { Dock = DockStyle.Fill, AutoGenerateColumns = false };
        grid.Columns.Add(new DataGridViewTextBoxColumn {
            HeaderText = "Order", DataPropertyName = "OrderId"
        });
        grid.DataSource = OrderData.Filter(OrderData.Create(), "Germany", true,
            new DateTime(1998, 5, 1), new DateTime(1998, 5, 6));
        grid.SelectionChanged += (s, args) ⇒ {
            var order = grid.CurrentRow == null ? null : grid.CurrentRow.DataBoundItem as
OrderRow;
            Text = order == null ? "No selection" : "Order " + order.OrderId;
        };
        Controls.Add(grid);
    } catch (ArgumentException ex) { MessageBox.Show(ex.Message); }
}
```

Learn the control, then verify the implementation

Explain the control properties, data binding and event path yourself. Give AI the supplied example and one bounded task. Review the patch, compile against the published Wisej 2.0.54 packages, test independent expectations and observe browser behavior.