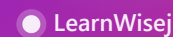


Widgets complejos enlazados a datos: patrones generales con ejemplos de Kendo y DevExtreme

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

Integra widgets de datos avanzados sin que el curso dependa de una única biblioteca de proveedor. Qué significa esto para un desarrollador de integraciones con Wisej.NET — y cómo leer este módulo.



Enfoque de producción

Integra widgets de datos avanzados sin que el curso dependa de una única biblioteca de proveedor.

1. El problema de integración

Las bibliotecas avanzadas como Kendo UI y DevExtreme ofrecen cuadrículas, tablas dinámicas, hojas de cálculo y almacenes de datos potentes, pero cada una tiene su propia API, su propio ciclo de vida y su propio modelo de operaciones de datos.

2. Conceptos clave

Concepto	Cómo usarlo
Operaciones de datos	Asigna la paginación, la ordenación, el filtrado, la agrupación y la edición a manejadores de servidor.
Origen de datos del proveedor	Trata Kendo/DevExtreme como ejemplos de un patrón de integración general.

3. Lección de producción

Kendo UI y DevExtreme son ejemplos avanzados porque contienen widgets completos con un comportamiento de datos real. No son la definición de la integración con Wisej.NET. Los mismos patrones se aplican a cualquier biblioteca seria de widgets de JavaScript: cargar los recursos en el orden correcto, crear el widget en el contenedor adecuado, pasar opciones, conectar eventos, suministrar datos, gestionar operaciones remotas y exponer una API .NET adecuada para la aplicación de destino. En las cuadrículas y tablas dinámicas, la decisión de diseño más importante es el contrato de datos. Una tabla dinámica de solo lectura puede devolver un array JSON o una respuesta de estilo OLAP. Una cuadrícula editable necesita operaciones de carga, inserción, actualización y borrado. La paginación y el filtrado remotos requieren modelos de petición. La serialización del diseño requiere un objeto de estado independiente. La exportación requiere funcionalidad del proveedor en el cliente o un pipeline en el servidor. Kendo y DevExtreme tienen API distintas, pero las responsabilidades del adaptador de Wisej.NET son las mismas.

Diseñar primero el contrato de datos

Un widget completo de cuadrícula o de tabla dinámica se define por las operaciones que realiza sobre los datos, así que el contrato se diseña antes de escribir código del proveedor. Una vista de solo lectura necesita una operación de carga que devuelva filas. Una vista paginada necesita skip y take, o un índice y un tamaño de página. Una vista ordenada o filtrada necesita una descripción de la ordenación y el filtro que aplicó el widget. Una vista agrupada devuelve agregados, y una tabla dinámica devuelve una estructura organizada por campos de fila y de columna. Una vista editable añade inserción, actualización y borrado, cada una con una clave y los valores modificados.

El laboratorio lo recoge como un `GridOperationRequest` tipado en el servidor, con skip, take, sort y filter, y un resultado con las filas y el recuento total. Cualquier biblioteca de proveedor, llame como llame a estas cosas, puede asignarse a esa petición. Diseñar la petición primero en .NET mantiene al servidor al mando de lo que puede contener un filtro y de lo grande que puede ser una página.

El estado del diseño, como el ancho de las columnas y las columnas visibles, es un contrato aparte, guardado por usuario y nunca mezclado con las operaciones de datos.

Asignar las operaciones del proveedor a endpoints

El DataSource de Kendo y el CustomStore de DevExtreme difieren en la sintaxis, pero ambos llaman a una URL o a una función para la carga, y ambos vuelven a llamar para la inserción, la actualización y el borrado cuando se habilitan las operaciones remotas. El trabajo del wrapper es traducir. En un origen basado en URL, el InitScript dirige al proveedor a la URL de postback, y el manejador `WebRequest` lee los nombres de los argumentos del proveedor, los convierte en el `GridOperationRequest` y devuelve el resultado con la forma que espera el proveedor.

En un origen basado en funciones, las funciones del InitScript llaman a un `[WebMethod]` con argumentos tipados y devuelven al proveedor el resultado esperado. En ambos casos, el lado del servidor es el mismo manejador con la misma validación, y cambiar de proveedor solo cambia la traducción del InitScript.

Habilitar las operaciones remotas es el momento en que la cuadrícula se convierte en una integración real. Hasta entonces, el proveedor ordena y pagina en memoria y al servidor solo se le pregunta una vez; a partir de ahí, cada interacción del usuario es una llamada al servidor que debe ser rápida y segura.

Opciones JSON frente a propiedades tipadas

Tanto Kendo como DevExtreme se configuran mediante grandes objetos de opciones, y la forma más rápida de poner una cuadrícula en pantalla es asignar ese objeto a Options directamente desde C#. Es lo adecuado para el prototipo de tabla dinámica de solo lectura del laboratorio, cuyo objetivo es demostrar que el widget funciona y que el contrato de datos se cumple.

Un control de cuadrícula reutilizable no puede quedarse así. Una bolsa de opciones sin tipos no tiene validación, ni valores predeterminados, ni texto de ayuda, ni forma de saber cuáles de sus cien claves usa realmente la aplicación. Las propiedades tipadas para los ajustes de los que depende la aplicación, como el tamaño de página, la edición y las definiciones de columnas, dan al Designer algo que mostrar y a los revisores algo que comprobar. El resto de las opciones del proveedor pueden quedar tras un punto de extensión protegido para el caso poco frecuente que las necesite.

El equilibrio es velocidad ahora frente a mantenibilidad después. Haz el prototipo con JSON y convierte en propiedad tipada cada opción que sobreviva antes de entregar el control a otros desarrolladores.

Dos proveedores, un patrón

Kendo y DevExtreme aparecen en este curso porque se usan mucho y porque sus diferencias hacen visible el patrón común. Uno usa un DataSource con ajustes de transporte; el otro, un CustomStore con funciones de carga y modificación. Uno llama a la paginación skip y take; el otro la nombra de forma parecida, pero da otra forma a la expresión de filtro. Una vez asignados ambos al mismo `GridOperationRequest`, las diferencias se reducen a unas pocas líneas de traducción en cada InitScript.

Esa es la lección que hay que llevar a cualquier otra biblioteca. Las responsabilidades del adaptador no cambian nunca: cargar los recursos en orden, crear el widget en el contenedor, pasar las opciones, conectar los pocos eventos que importan, suministrar los datos mediante un contrato que controla el servidor, gestionar las operaciones remotas y exponer una API .NET dimensionada para la aplicación. Un equipo capaz de hacerlo con dos bibliotecas puede hacerlo con una tercera sin necesidad de otro curso.

Mantén el conocimiento específico del proveedor en el InitScript y en la traducción de eventos, y mantén el contrato del servidor libre de proveedores, para que un cambio de licencia o de hoja de ruta nunca obligue a reescribir el código de la aplicación.



Regla de producción

Mantén claras la propiedad del estado en Wisej.NET y el comportamiento del widget de cliente. Documenta el contrato cliente/servidor antes de confiar en la integración en producción.



Error habitual que debes evitar

Copiar JavaScript de demostración en las pantallas sin documentar el ciclo de vida, los recursos, los contratos de eventos, la limpieza ni los límites de seguridad.



Objetivos de aprendizaje

Al terminar este módulo serás capaz de:

- ✓ Diseñar contratos de datos para datos de solo lectura, editables, paginados, filtrados, ordenados, agrupados y dinámicos (pivot).
- ✓ Asignar las operaciones del origen de datos del proveedor a manejadores de postback o WebMethods de Wisej.NET.
- ✓ Entender por qué las opciones JSON son rápidas, pero las propiedades .NET tipadas son más seguras para los controles reutilizables.
- ✓ Usar Kendo y DevExtreme como implementaciones de referencia de los mismos patrones generales de integración.

Guía del laboratorio / examen

Lo que construirás en el laboratorio práctico, el flujo de trabajo recomendado, los entregables requeridos y la autoevaluación antes de entregar.



Laboratorio práctico

Crea dos ejemplos avanzados: un widget de solo lectura de tipo tabla dinámica (pivot) y un widget editable de tipo cuadrícula. Uno puede usar la semántica de DataSource al estilo Kendo y el otro la semántica de CustomStore al estilo DevExtreme.

Flujo de trabajo recomendado

- ✓ Abre el proyecto existente del curso de integración de Wisej.NET y ejecútalo en local.
- ✓ Identifica el componente de servidor, el widget de cliente, el archivo de recursos o el endpoint que intervienen en este módulo.
- ✓ Haz un solo cambio pequeño cada vez y pruébalo en el navegador antes de continuar.
- ✓ Mantén el JavaScript específico del proveedor dentro del adaptador o wrapper en lugar de repartirlo por las páginas.
- ✓ Actualiza la interfaz para que el estudiante vea con claridad el éxito, los errores, el progreso o los cambios de comunicación.
- ✓ Toma capturas de pantalla o notas que demuestren que la integración funciona y expliquen la decisión de diseño.

Entregables requeridos

- ✓ Plan de integración de la tabla dinámica de solo lectura
- ✓ Contrato de operaciones de la cuadrícula editable
- ✓ Manejadores de servidor o stubs para load/update/insert/delete
- ✓ Contratos de carga útil de eventos para el clic en celda y la actualización de fila

Autoevaluación antes de entregar

- ✓ ¿Por qué las cuadrículas son más difíciles que los indicadores?
- ✓ ¿Qué cambia cuando se habilitan las operaciones remotas?
- ✓ ¿Por qué Kendo y DevExtreme deberían enseñarse como ejemplos y no como todo el curso?