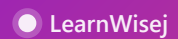


FlowLayoutPanel, TableLayoutPanel, FlexLayoutPanel y propiedades de diseño extendidas

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

Elegir contenedores flow, table o flex según el comportamiento del contenido, las propiedades extendidas de los hijos FillWeight, FlowBreak, AlignX, AlignY, Row, Column, spans y estilos, cuándo respetan los márgenes los motores de diseño, y las extensiones de marcado fluent para diseños creados por código. Qué significa esto para un desarrollador de Wisej.NET y cómo leer este módulo.



Enfoque de producción

Elegir contenedores flow, table o flex según el comportamiento del contenido, las propiedades extendidas de los hijos FillWeight, FlowBreak, AlignX, AlignY, Row, Column, spans y estilos, cuándo respetan los márgenes los motores de diseño, y las extensiones de marcado fluent para diseños creados por código.

1. El problema en producción

El Módulo 4 dio a Adaptive Operations Console un shell de regiones acopladas, pero el acoplamiento solo decide dónde se sitúa una región, no cómo se comporta el contenido que hay dentro cuando el navegador mide 1400 píxeles de ancho el lunes y 700 en una tableta el martes. Una fila de tarjetas de métricas debe pasar a la línea siguiente, un editor de tickets debe mantener alineados sus etiquetas y editores por larga que sea la traducción al alemán, y las regiones de lista y detalles deben repartirse el área de trabajo en la proporción que tú decidas. Cada uno es un comportamiento de contenido distinto, y Wisej.NET incluye un contenedor para cada uno: `FlowLayoutPanel`, `TableLayoutPanel` y `FlexLayoutPanel`. Elegir el equivocado es la forma en que los equipos acaban volviendo a escribir código de redimensionado.

2. Conceptos clave

Concepto	Por qué importa
FlowLayoutPanel	Coloca los hijos en filas o columnas automáticas a lo largo de <code>FlowDirection</code> , pasa a la línea siguiente cuando <code>WrapContents</code> es true e ignora Location, Dock y Anchor de cada hijo, así que los chips y las tarjetas se recolocan solos.
TableLayoutPanel	Coloca los hijos en celdas identificadas por Row y Column, con spans para las celdas que se extienden, de modo que etiquetas y editores permanecen alineados mientras cada hijo puede seguir acoplándose o anclándose dentro de su celda.
FlexLayoutPanel	Coloca los hijos en una sola fila o una sola columna, siempre ocupa su área cliente y reparte el espacio restante por peso, lo que encaja con las regiones principales que dividen el área de trabajo de forma proporcional.
FillWeight	Una propiedad extendida del hijo que leen los motores flow y flex: un peso mayor que cero estira el hijo en el espacio restante en proporción a los demás hijos con peso; cero mantiene su propio tamaño.
FlowBreak	Una propiedad extendida de los hijos de FlowLayoutPanel que fuerza al siguiente hijo a empezar una fila o columna nueva, de modo que una barra de filtros termina una línea de forma deliberada en lugar de depender del ancho.
Estilos de fila y de columna	<code>RowStyles</code> y <code>ColumnStyles</code> declaran si una fila o columna ajusta su tamaño automáticamente, toma un tamaño fijo en píxeles o reparte el resto por porcentaje; ahí residen las proporciones de un formulario.
Margin y Padding	Padding es el espacio interior del contenedor alrededor de todos los hijos; Margin es la distancia del hijo a sus vecinos, y solo la usan los motores que respetan los márgenes.

3. Desarrollo de la lección

Elige el contenedor según el comportamiento del contenido

La regla de decisión es lo bastante breve para recordarla y lo bastante estricta para revisar con ella. Usa `FlowLayoutPanel` para contenido que puede pasar a la línea siguiente: chips de filtro, tarjetas de métricas, zonas de desbordamiento de la barra de herramientas, grupos de comandos responsive. Usa `TableLayoutPanel` para formularios estructurados: etiquetas de campos, editores alineados, filas de resumen y cualquier cuadrícula de controles que deba resistir la localización. Usa `FlexLayoutPanel` para las regiones principales que deben ocupar el espacio disponible de forma proporcional, como un carril de navegación junto a una región de contenido y un panel lateral de detalles, o secciones móviles apiladas que reparten la altura por peso. El acoplamiento y el anclaje siguen siendo adecuados para las regiones estables del shell y los formularios sencillos; los tres contenedores son para el contenido dentro de esas regiones.

Fíjate en el comportamiento, no en el aspecto. La zona de tarjetas de métricas de la consola parece una cuadrícula, pero su comportamiento es «tantas tarjetas por fila como quepan, y luego pasar a la siguiente», es decir, flow. El editor de tickets parece una lista de campos, pero su comportamiento es «el

borde izquierdo de todos los editores en la misma línea», es decir, table. El área de trabajo parece dos paneles, pero su comportamiento es «la lista ocupa dos tercios, los detalles un tercio y ninguno se colapsa», es decir, flex. El laboratorio construye los tres para que puedas redimensionar el navegador y ver qué contenedor resiste cada escenario; esa comparación por escrito es el entregable que más importa.

Las propiedades extendidas de los hijos

Los contenedores funcionan mediante propiedades extendidas: valores que pertenecen al hijo pero que guarda y lee el padre, y que solo aparecen en la cuadrícula de propiedades del hijo mientras está dentro de ese contenedor. `FlowLayoutPanel` lee `Margin`, `FillWeight`, `MinimumSize`, `MaximumSize` y `FlowBreak`, que se asignan por código a través del contenedor: `SetFillWeight(search, 1)` permite que el cuadro de búsqueda use el ancho que le quede a la fila, y `SetFlowBreak(apply, true)` termina la fila para que las tarjetas de debajo empiecen en una línea nueva. `FlowDirection` elige de izquierda a derecha, de derecha a izquierda, de arriba abajo o de abajo arriba; con `WrapContents` a false los hijos se recortan o se desplazan en lugar de pasar a la línea siguiente, que es lo que quiere una barra de herramientas de una sola línea con `AutoScroll`.

`TableLayoutPanel` lee `Row`, `Column`, `RowSpan` y `ColumnSpan` de cada hijo y dimensiona la cuadrícula a partir de `RowStyles` y `ColumnStyles`: tamaño automático, un valor absoluto en píxeles o un porcentaje. El editor de tickets usa una columna de etiquetas absoluta de 140 píxeles y una columna de editores del 100 %, de modo que los editores crecen con el panel mientras las etiquetas se quedan en su sitio, y el editor de Notes abarca ambas columnas en una fila que ocupa la altura restante. `GrowStyle` decide qué ocurre cuando se añade un control con las celdas llenas: añadir filas, añadir columnas o negarse a crecer. Como un hijo puede seguir acoplándose dentro de su celda, un editor acoplado con `Fill` en una columna porcentual es el formulario alineado más limpio que produce Wisej.NET.

`FlexLayoutPanel` lee `Margin`, `MinimumSize`, `MaximumSize`, `FillWeight`, `AlignX` y `AlignY`. Su `LayoutStyle` es horizontal (una fila) o vertical (una columna); el contenedor siempre ocupa su área cliente y distribuye el espacio restante entre los hijos con peso. Cuando un hijo no ocupa una dimensión, `AlignX` y `AlignY` sobrescriben la alineación del padre solo para ese hijo, así que una región de detalles con `AlignY` arriba se pega al borde superior de un área de trabajo alta en lugar de centrarse. La regla que comprueba el laboratorio: cada hijo con `FillWeight` recibe un `MinimumSize`; de lo contrario, un navegador estrecho reduce una región a cero y el usuario ve una franja vacía donde estaba el editor de detalles. `MaximumSize` limita esa misma región para que un monitor 4K no estire un formulario hasta llenarlo de espacio en blanco.

```
// DashboardWorkspace : FlexLayoutPanel – the list/details split
this.Dock = DockStyle.Fill;
this.LayoutStyle = FlexLayoutStyle.Horizontal;
this.Padding = new Padding(8);

var list = new Panel { MinimumSize = new Size(300, 200) };
var details = new Panel { MinimumSize = new Size(280, 200),
                          MaximumSize = new Size(520, 0) };

this.Controls.Add(list);
this.Controls.Add(details);

this.SetFillWeight(list, 2); // two thirds of the spare width
this.SetFillWeight(details, 1); // one third, never below 280 px
this.SetAlignY(details, VerticalAlignment.Top);
```

Cuándo respetan los márgenes los motores de diseño

Margin es la propiedad que más se asigna y menos se respeta, porque que haga algo depende del motor que coloca al hijo. Los motores flow y flex aplican los márgenes: cada hijo de un `FlowLayoutPanel` o `FlexLayoutPanel` mantiene la distancia que declara su Margin. El motor de diseño predeterminado no usa los márgenes para el acoplamiento; un Panel acoplado a Left con un Margin de 8 queda pegado al borde de todos modos, y por eso el Módulo 4 usó Padding en las regiones acopladas para los espacios. `TableLayoutPanel` tiene en cuenta los márgenes dentro de la celda, así que el Margin de un hijo se convierte en su distancia al borde de la celda, mientras que el Padding de la tabla añade el marco exterior.

Trata ambas propiedades como pertenecientes a responsables distintos: Padding es la decisión del contenedor sobre el espacio interior alrededor de todos sus hijos, y Margin la decisión del hijo sobre su distancia a los controles adyacentes. Un sistema de diseño coherente para la consola define valores estándar una sola vez, por ejemplo modos denso, normal y amplio de 4, 8 y 16 píxeles, y los aplica en todas partes mediante esas dos propiedades. Cuando no aparezca un espacio, pregúntate qué motor es responsable del hijo antes de cambiar ningún número.

Extensiones de marcado fluent

El espacio de nombres `Wisej.Web.Markup` añade métodos de extensión fluent que se leen como una descripción del diseño. `FlowLayoutPanelExtensions` puede asignar FillWeight y FlowBreak, `FlexLayoutPanelExtensions` puede asignar FillWeight, AlignX y AlignY, y `TableLayoutPanelExtensions` puede asignar el número de columnas, el estilo de crecimiento y los ajustes de diseño. Cada método devuelve el contenedor, así que un shell creado por código se convierte en una sola cadena: `panel.FillWeight(nav, 0).FillWeight(content, 1).AlignY(nav, VerticalAlignment.Top)`. El laboratorio te pide construir una región de esta forma; la candidata natural es una región que ya creas por código, como un panel de control cuyas tarjetas dependen del rol del usuario conectado.

Usa el estilo fluent donde haga legible un diseño creado por código y deja las pantallas creadas en el diseñador al diseñador. Mezclar en una misma clase la serialización del diseñador, llamadas imperativas a `SetFillWeight` y cadenas fluent obliga al siguiente desarrollador a leer tres notaciones para un solo panel, así que mantén una notación por contenedor. En cualquier caso se aplica la lista de verificación: contenedores en lugar de manejadores de redimensionado, orden de acoplamiento comprobado, Padding

en las regiones acopladas, flow para el ajuste de línea, table para formularios, flex para regiones proporcionales, y un `MinimumSize` en cada control con `FillWeight`.



Errores habituales

Asignar `Location`, `Dock` o `Anchor` a un hijo de `FlowLayoutPanel` y preguntarse por qué se ignoran. Regiones con `FillWeight` sin `MinimumSize` que se colapsan en un teléfono. Esperar que `Margin` cree espacios entre `Panels` acoplados con el motor de diseño predeterminado. Recurrir a `FlexLayoutPanel` para un formulario con etiquetas alineadas, o a `TableLayoutPanel` para una fila de tarjetas que debería pasar a la línea siguiente.



Regla del sistema visual

Usa el mecanismo de `Wisej.NET` responsable del problema: el tema para la identidad de los controles estándar, `AppearanceKey` para las variantes semánticas, `CssClass` para CSS específico de la aplicación con ámbito, `CssStyle` solo para valores dinámicos puntuales, contenedores de diseño en lugar de código de redimensionado, propiedades `responsive` para los valores por perfil definidos en el diseñador, y `ResponsiveProfileChanged` para el comportamiento que no es simplemente visual.



Objetivos de aprendizaje

Al terminar este módulo serás capaz de:

- ✓ Elegir `FlowLayoutPanel`, `TableLayoutPanel` o `FlexLayoutPanel` según el comportamiento del contenido y la finalidad de la pantalla.
- ✓ Usar `FillWeight`, `FlowBreak`, `AlignX`, `AlignY`, `Row`, `Column`, `RowSpan`, `ColumnSpan`, `RowStyles`, `ColumnStyles`, `GrowStyle`, `MinimumSize` y `MaximumSize`.
- ✓ Entender cuándo usan los márgenes los motores de diseño y cuándo no.
- ✓ Usar métodos de extensión de marcado para definiciones de diseño fluent cuando corresponda.

Guía del laboratorio / examen

Lo que construirás en el laboratorio práctico, el enfoque sugerido y los entregables requeridos.



Laboratorio práctico

Construye tres versiones del área de contenido de Adaptive Operations Console: una zona de filtros y tarjetas de métricas con `FlowLayoutPanel` que se ajusta a la línea y usa `FillWeight` y `FlowBreak`, un editor de tickets con `TableLayoutPanel` con etiquetas y editores alineados mediante `RowStyles` y `ColumnStyles`, y un shell de panel de control con `FlexLayoutPanel` donde las regiones de lista y detalles reparten el espacio por `FillWeight` con `AlignX`, `AlignY`, `MinimumSize` y `MaximumSize` definidos. Construye una de las tres por código con las extensiones fluent de `Wisej.Web.Markup` y compara cómo resiste cada una el redimensionado del navegador.

Punto de partida

Abre la solución `AdaptiveOperationsConsole` tal como la dejaste tras el Módulo 4: la Page principal es un shell de contenedores anidados con una barra de herramientas, un carril de navegación, un área de trabajo de relleno, un panel de detalles y una región de estado; las regiones de navegación y detalles son `UserControls`, y el área de trabajo sigue siendo un `Panel` simple cuyas tarjetas de métricas, editor de tickets y lista están posicionados a mano. Guarda las tres regiones nuevas en una carpeta `Layout` para que el proyecto final pueda elegir la ganadora de cada comparación.

Enfoque sugerido

- ✓ Crea un `FilterBar` que derive de `FlowLayoutPanel` con `FlowDirection` de izquierda a derecha y `WrapContents` activado; añade el cuadro de búsqueda, el combo de estado y el botón Apply, asigna al cuadro de búsqueda `FillWeight` 1 y un `MinimumSize`, establece `FlowBreak` en Apply, y después coloca debajo las tarjetas de métricas y observa cómo pasan a la línea siguiente.
- ✓ Construye un `UserControl` `TicketEditor` alrededor de un `TableLayoutPanel` de dos columnas: una columna de etiquetas absoluta de 140 píxeles, una columna de editores del 100 %, una fila de tamaño automático por campo, una fila Notes que abarca ambas columnas con la altura restante, y cada editor acoplado con Fill en su celda.
- ✓ Sustituye el Panel del área de trabajo por un `DashboardWorkspace` que derive de `FlexLayoutPanel` con diseño horizontal; añade las regiones de lista y detalles con `FillWeight` 2 y 1, un `MinimumSize` en ambas, un `MaximumSize` en detalles y `AlignY` arriba en detalles.
- ✓ Construye una de las tres regiones íntegramente por código con `using Wisej.Web.Markup;` y los métodos de extensión fluent `FillWeight`, `FlowBreak`, `AlignX` y `AlignY` encadenados sobre el contenedor, y mantén esa clase libre de serialización del diseñador.
- ✓ Ejecuta la consola, redimensiona el navegador de 1400 a 1024 y a 700 píxeles para cada región, haz una captura en cada ancho y anota, por región, qué contenedor resistió y qué propiedad extendida hizo el trabajo.

Entregables requeridos

- ✓ Zona de filtros y tarjetas de métricas con FlowLayoutPanel usando WrapContents, FillWeight y FlowBreak
- ✓ Editor de tickets con TableLayoutPanel con RowStyles, ColumnStyles y spans para campos alineados
- ✓ Shell de panel de control con FlexLayoutPanel con FillWeight, AlignX, AlignY, MinimumSize y MaximumSize
- ✓ Una región construida por código con las extensiones fluent de Wisej.Web.Markup
- ✓ Capturas comparativas del redimensionado y una nota sobre qué contenedor se adapta a cada región

Preguntas de repaso

- ✓ Diste a la región de detalles FillWeight 1 pero ningún MinimumSize, y a 700 píxeles desapareció. Explica qué hizo el motor flex con el espacio y por qué la solución es una restricción de tamaño y no un peso distinto.
- ✓ El botón Apply pasa a la segunda fila a 1024 píxeles aunque hay sitio para él. ¿Qué propiedad extendida lo provocó, y cómo se comportaría FilterBar de otra forma si la quitaras y pusieras WrapContents a false?
- ✓ Un Margin de 8 en las tarjetas del FlowLayoutPanel produjo espacios, pero el mismo Margin en el carril de navegación acoplado no hizo nada. ¿Qué motor respeta los márgenes en cada caso, y qué propiedad crea el espacio para el carril?



Regla del sistema visual

Usa el mecanismo de Wisej.NET responsable del problema: el tema para la identidad de los controles estándar, AppearanceKey para las variantes semánticas, CssClass para CSS específico de la aplicación con ámbito, CssStyle solo para valores dinámicos puntuales, contenedores de diseño en lugar de código de redimensionado, propiedades responsive para los valores por perfil definidos en el diseñador, y ResponsiveProfileChanged para el comportamiento que no es simplemente visual.