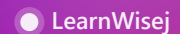


# ClientProfile y propiedades responsive

Guía de la lección · Guía del laboratorio / examen



## Guía de la lección

Cómo ClientProfiles.json se evalúa de arriba abajo para obtener Application.ActiveProfile, los valores de propiedades responsive asignados por perfil en el diseñador, ResponsiveProfileChanged para el comportamiento que las propiedades responsive no pueden expresar, y las reglas para diseñar y probar perfiles. Qué significa esto para un desarrollador de Wisej.NET y cómo leer este módulo.



### Enfoque de producción

Cómo ClientProfiles.json se evalúa de arriba abajo para obtener Application.ActiveProfile, los valores de propiedades responsive asignados por perfil en el diseñador, ResponsiveProfileChanged para el comportamiento que las propiedades responsive no pueden expresar, y las reglas para diseñar y probar perfiles.

## 1. El problema en producción

Adaptive Operations Console ya tiene un shell con tema construido con contenedores Flow, Table y Flex que resiste un redimensionado sin manejador Resize, pero muestra la misma pantalla a todo el mundo: en un teléfono de 390 píxeles, el carril de navegación, la cuadrícula de tickets y el editor de detalles compiten por el mismo ancho. Las media queries CSS no pueden resolverlo, porque los controles viven en el servidor: ocultar un panel, cambiar un botón a solo icono o reacoplar una región es un cambio de propiedad en el servidor, y un usuario de teléfono que pierde el panel de detalles necesita otra forma de editar un ticket, no una copia más pequeña de la de escritorio. Este módulo añade el mecanismo del que Wisej.NET es responsable para eso: los perfiles de cliente, las propiedades responsive y el evento `ResponsiveProfileChanged`.

## 2. Conceptos clave

| Concepto                      | Por qué importa                                                                                                                                                                                                                                                          |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ClientProfile                 | Un conjunto con nombre de reglas de coincidencia relacionadas con el dispositivo (ancho del navegador, ancho de pantalla, tipo de dispositivo, user agent, orientación horizontal) que describe cómo se usa una clase de cliente, no un nombre comercial de dispositivo. |
| ClientProfiles.json           | El archivo que lee Wisej.NET; los perfiles se evalúan de arriba abajo y gana el primero que coincide, así que el orden de las entradas forma parte de la configuración.                                                                                                  |
| Application.ActiveProfile     | El perfil seleccionado actualmente para esta sesión; su <code>Name</code> es lo que muestra la etiqueta de estado y aquello según lo cual ramifica el manejador de perfil.                                                                                               |
| propiedad responsive          | Una propiedad .NET normal marcada con <code>ResponsivePropertyAttribute</code> que guarda un valor por perfil de cliente y aplica el valor correspondiente cada vez que cambia el perfil activo.                                                                         |
| adaptación desde el diseñador | Los valores responsive se asignan solo en el diseñador de Visual Studio, lo que los mantiene serializados, visibles y deterministas; el código es para las adaptaciones que dependen de datos o del estado en tiempo de ejecución.                                       |
| ResponsiveProfileChanged      | El evento que se lanza en <code>Application</code> , en contenedores como Form, Page, Desktop y UserControl y en DataGridView y ListView cuando cambia el perfil activo; <code>ResponsiveProfileChangedEventArgs.CurrentProfile</code> contiene el nuevo perfil.         |
| matriz de QA responsive       | El plan de pruebas que enumera ancho, alto, orientación, tipo de dispositivo, táctil, barras de desplazamiento, escalado de fuente y el nombre de perfil que esperas en cada tamaño.                                                                                     |

## 3. Desarrollo de la lección

### Cómo coincide un perfil

Cada entrada de `ClientProfiles.json` tiene un `name` y cualquier combinación de `device` (Mobile, Tablet o Desktop), `landscape`, `minWidth` y `maxWidth` para el viewport del navegador, `minScreenWidth` y `maxScreenWidth` para la pantalla física, y una coincidencia `userAgent`. Wisej.NET lee el archivo, evalúa los perfiles de arriba abajo y selecciona la primera coincidencia como `Application.ActiveProfile`. Nada se combina: un cliente coincide con un perfil o con ninguno, y por eso el patrón siguiente recurre a «Default» cuando `ActiveProfile` es null.

La coincidencia puede producirse cuando se carga la página o cuando se redimensiona el navegador, según la propiedad que use la regla: una regla de ancho del viewport se vuelve a evaluar mientras se arrastra el navegador, mientras que el tamaño de pantalla, el tipo de dispositivo y el user agent describen al propio cliente. Cada vez que cambia el perfil seleccionado, Wisej.NET lanza `ResponsiveProfileChanged`. Para el laboratorio escribes seis perfiles en orden de primera coincidencia: Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop y Desktop. Las dos entradas de teléfono comparten `device` Mobile y solo se diferencian en `landscape`; Small Desktop es un dispositivo Desktop

con `maxWidth` 1024; Desktop es la regla general al final. Si la pones arriba, oculta todos los perfiles que tiene debajo, y nada te avisa.

## Las propiedades responsive pertenecen al diseñador

Las propiedades responsive son propiedades normales marcadas con `ResponsivePropertyAttribute`: `Visible`, `Display`, `Size`, `Location`, `Dock`, `Anchor`, `Font` y valores relacionados con el diseño cuando el control los admite. En el diseñador, una propiedad así guarda un valor distinto para cada perfil de `ClientProfiles.json`, y cuando cambia el perfil activo el control aplica el valor que le corresponde. El diseñador es la única forma admitida de asignar estos valores, una regla y no una preferencia: los valores asignados ahí se serializan con el formulario, son visibles para el siguiente desarrollador y son deterministas en tiempo de ejecución, mientras que los valores asignados por código son invisibles en tiempo de diseño y se desvían de la imagen de la pantalla que tiene el diseñador.

En el laboratorio, cada diferencia determinista de la consola es trabajo del diseñador. En ambos perfiles Phone, el UserControl del carril de navegación y el panel de detalles reciben `Visible` false. Los botones de la barra de herramientas reciben un `Display` solo icono en Phone y Tablet para que la barra de comandos quepa en una fila. En los perfiles Tablet, el `Dock` de la región de detalles pasa de Right a Bottom para que quede bajo la cuadrícula de tickets en lugar de a su lado. Ninguno de esos cambios depende de datos, así que ninguno pertenece al código; el código es para las adaptaciones que dependen de datos, permisos, un cálculo en tiempo de ejecución o el propio evento de perfil.

## ResponsiveProfileChanged para el comportamiento

Algunas adaptaciones no son valores. En un teléfono, el editor de detalles no puede ser un panel lateral; tiene que abrirse como `Form` modal mediante `ShowDialog`, y la etiqueta de estado debe mostrar y registrar el nombre del perfil para que el soporte sepa qué diseño ve un usuario. Para eso sirve `ResponsiveProfileChanged`. Se lanza en `Application`, en tipos `ContainerControl` como `Form`, `Page`, `Desktop` y `UserControl`, y en algunos controles complejos como `DataGridView` y `ListView`. Gestiónalo en `Application` para el comportamiento de toda la aplicación y en un contenedor clave cuando la lógica es local a esa región; los argumentos del evento exponen `CurrentProfile`.

El patrón siguiente es la forma que espera el laboratorio: suscríbete una vez, aplica el perfil una vez al arrancar a partir de `Application.ActiveProfile` para que el diseño inicial sea correcto, pasa el evento por el mismo método `ApplyProfile`, y cancela la suscripción en `Dispose` porque `Application` sobrevive a la página. Mantén `ApplyProfile` ligero e idempotente: un teléfono que gira entre Phone y Phone Landscape lo lanza repetidamente, y un editor que ya está abierto debe reutilizarse, nunca abrirse de nuevo. Trabajo típico del evento: cargar menos columnas en la cuadrícula, cambiar el intervalo de debounce de una búsqueda, convertir un panel de detalles en un formulario modal, aplicar una plantilla de datos distinta.

```
// MainPage.cs – one method applies the profile at startup and on every change
Application.ResponsiveProfileChanged += Application_ResponsiveProfileChanged;
ApplyProfile(Application.ActiveProfile?.Name);

private void Application_ResponsiveProfileChanged(object sender,
ResponsiveProfileChangedEventArgs e)
{
    ApplyProfile(e.CurrentProfile?.Name);
}

private void ApplyProfile(string profileName)
{
    var phone = profileName != null && profileName.StartsWith("Phone");
    this.lblProfile.Text = profileName ?? "Default"; // shown and logged
    if (phone && this.ticketEditor == null) OpenDetailsAsModal(); // side panel → modal,
    once
}
```

## Variantes de diseño que cambian el comportamiento

Los perfiles deben describir comportamientos, no dispositivos. Phone, Tablet Landscape, Small Desktop, Kiosk y Touch Workstation nombran cada uno una forma de usar la consola, y cada variante debe cambiar lo que hace la consola: el teléfono recibe un editor modal y un botón de menú, la tableta apila la región de detalles bajo la cuadrícula, el escritorio pequeño mantiene el shell con una barra de herramientas solo icono, y el escritorio lo muestra todo. Usa reglas de ancho para el diseño, reglas de dispositivo para las suposiciones de entrada como los objetivos táctiles y las barras de desplazamiento ocultas, `userAgent` solo cuando sea necesario, y reglas landscape solo cuando la orientación cambie realmente la interfaz. Un perfil que no cambia nada más que una dimensión CSS no necesitaba existir.

Prueba de las dos formas. El redimensionado real del navegador pone a prueba las reglas de ancho; la emulación de dispositivos pone a prueba las reglas de dispositivo, ancho de pantalla y user agent, y después de cambiar de dispositivo en el emulador móvil de Chrome debes recargar la página para que esos perfiles se vuelvan a evaluar. La matriz de QA responsive tiene una fila por escenario, desde Desktop a 1440 por 900 hasta Phone a 390 por 844 y Phone Landscape a 844 por 390, cada una con el nombre de perfil esperado y las comprobaciones que demuestran el comportamiento: navegación visible, detalles apilados, editor modal accesible, sin desbordamiento horizontal.

- Perfiles estrechos o específicos antes que los generales; gana la primera coincidencia y nada se combina.
- Valores responsive en el diseñador; `ResponsiveProfileChanged` solo cuando un valor no basta, y entonces ligero e idempotente.
- Muestra el nombre del perfil en las compilaciones de prueba y documenta cada control oculto con su alternativa móvil.



### Errores habituales

Poner primero el perfil general Desktop, de modo que los perfiles de teléfono y tableta nunca coincidan. Asignar valores responsive por código en lugar de en el diseñador. Meter todas las adaptaciones en el manejador del evento, incluidas las que habría expresado un valor de `Visible` o `Dock`. Volver a abrir el editor modal en cada giro porque el manejador no es idempotente. Ocultar un panel en el teléfono sin ninguna alternativa móvil para lo que contenía.



### 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:

- ✓ Explicar cómo se lee y se evalúa `ClientProfiles.json` y cómo se selecciona `Application.ActiveProfile`.
- ✓ Usar propiedades responsive en el diseñador para propiedades como `Visible`, `Display`, `Size`, `Location`, `Dock`, `Anchor`, `Font` y valores relacionados con el diseño cuando estén disponibles.
- ✓ Gestionar `ResponsiveProfileChanged` en `Application` y en contenedores clave cuando se necesite lógica específica por perfil.
- ✓ Diseñar variantes de teléfono, tableta, escritorio pequeño y escritorio que cambien el comportamiento de la interfaz, no solo las dimensiones CSS.

## Guía del laboratorio / examen

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



### Laboratorio práctico

Añade a Adaptive Operations Console un ClientProfiles.json con los perfiles Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop y Desktop ordenados del más estrecho al más ancho. En el diseñador, usa propiedades responsive para ocultar el carril de navegación y el panel de detalles en el teléfono, cambiar los botones de la barra de herramientas a Display solo icono y volver a acoplar la región de detalles bajo la cuadrícula en la tableta. Gestiona ResponsiveProfileChanged para mostrar el nombre del perfil activo en la etiqueta de estado, registrar cada cambio y abrir el editor de detalles como formulario modal en el teléfono en lugar de como panel lateral.

### Punto de partida

Abre Adaptive Operations Console tal como la dejó el Módulo 5: el shell está formado por contenedores anidados con una zona de tarjetas de métricas Flow, un editor de tickets TableLayoutPanel dentro del UserControl de detalles y un área de trabajo Flex; el tema AdaptiveOps y la hoja de estilos con ámbito están cargados, y la etiqueta de estado del Módulo 1 sigue mostrando el ancho del navegador. Todavía no hay ClientProfiles.json, así que Application.ActiveProfile es null para todos los clientes y todos los anchos muestran el shell de escritorio.

### Enfoque sugerido

- ✓ Escribe ClientProfiles.json en la raíz de la aplicación con seis entradas en orden de primera coincidencia (Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop, Desktop): las dos entradas de teléfono usan device Mobile y se diferencian por landscape, la pareja de tableta hace lo mismo con Tablet, Small Desktop es un dispositivo Desktop con maxWidth 1024 y Desktop es la regla general al final. Ejecuta una vez y confirma que los nombres de los perfiles aparecen en el diseñador.
- ✓ En el diseñador, asigna al UserControl del carril de navegación y al panel de detalles un valor Visible false para ambos perfiles Phone, y comprueba que la cuadrícula del área de trabajo conserva su MinimumSize para que el diseño de teléfono no se reduzca a cero.
- ✓ Todavía en el diseñador, establece el Display de los botones de la barra de herramientas en solo icono para los perfiles Phone y Tablet, y cambia el Dock de la región de detalles de Right a Bottom en ambos perfiles Tablet para que quede bajo la cuadrícula de tickets; deja intactos los valores de escritorio.
- ✓ En MainPage, suscríbete una sola vez a Application.ResponsiveProfileChanged, escribe un método ApplyProfile(string profileName) que asigne lblProfile.Text y escriba una línea de registro por cada cambio, llámalo en el constructor con Application.ActiveProfile?.Name y desde el manejador con e.CurrentProfile?.Name, y cancela la suscripción en Dispose.
- ✓ En los perfiles Phone, haz que ApplyProfile abra el editor de tickets en un Form modal mediante ShowDialog, reutilizando el formulario si ya está abierto; después prueba arrastrando

el navegador y con la emulación de dispositivos, recargando tras cada cambio de dispositivo, y captura la etiqueta del perfil en cada tamaño.

## Entregables requeridos

- ✓ ClientProfiles.json con los perfiles Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop y Desktop en orden de primera coincidencia
- ✓ Valores de propiedades responsive asignados en el diseñador para ocultar el carril de navegación y el panel de detalles en el teléfono
- ✓ Display solo icono en la barra de herramientas y región de detalles reacoplada para tableta definidos mediante propiedades responsive
- ✓ Manejador ResponsiveProfileChanged que muestra y registra el nombre del perfil activo
- ✓ Editor de detalles abierto como formulario modal en el teléfono a través del manejador de perfil

## Preguntas de repaso

- ✓ Si Small Desktop se moviera al principio de ClientProfiles.json, ¿cuáles de los otros cinco perfiles podrían seguir coincidiendo, y qué vería una tableta en horizontal?
- ✓ El reacoplamiento de la tableta está en el diseñador y el editor modal del teléfono está en el código. ¿Qué decide qué mecanismo es responsable de cada cambio, y qué fallaría si los intercambiaras?
- ✓ Un tester giró un teléfono cuatro veces y tu manejador se ejecutó cuatro veces. ¿Qué estado debe ser idéntico tras la cuarta ejecución y tras la primera, y cómo lo garantiza tu código?



### 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.