

Fuentes de iconos, contenido HTML y estrategia de iconos en producción

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

El último mecanismo y la política que los une a todos: una hoja de estilos de fuente de iconos cargada desde Default.html, AllowHtml en los controles que deben interpretar marcado, glifos de icono dentro de celdas de DataGridView con clics de elemento según su rol, y una lista de comprobación de producción que cubre accesibilidad, licencias, caché, dependencias externas y coherencia visual. Qué significa esto para un desarrollador que conoce C# básico y ya ha trabajado algo con los controles y el diseñador de Wisej.NET — y cómo leer este módulo.



Enfoque de producción

El último mecanismo y la política que los une a todos: una hoja de estilos de fuente de iconos cargada desde Default.html, AllowHtml en los controles que deben interpretar marcado, glifos de icono dentro de celdas de DataGridView con clics de elemento según su rol, y una lista de comprobación de producción que cubre accesibilidad, licencias, caché, dependencias externas y coherencia visual.

1. El problema en producción

Después de seis módulos, IconDesk sabe dibujar un icono de cinco formas distintas, y todas funcionan. Lo que al proyecto le sigue faltando es una regla de decisión: cuando una pantalla nueva necesita un símbolo de Save, nadie debería tener que adivinar si viene del tema, de un pack oficial de Wisej-4 o de IconDesk.Icons. Este módulo añade la última opción, las fuentes de iconos renderizadas como HTML, y escribe después la política que dice a cuál recurrir y qué comprobar antes de publicar una pantalla.

2. Conceptos clave

Concepto	Por qué importa
fuentes de iconos	Una fuente web cuyos glifos son imágenes en lugar de letras, que se eligen con nombres de clase CSS y dibuja el navegador dentro del contenido HTML, no a través de ninguna propiedad de imagen.
AllowHtml	El interruptor por control que le dice a Wisej.NET que interprete el texto de un control como marcado; sin él las etiquetas aparecen como caracteres literales, porque el texto es texto plano de forma predeterminada.
cabecera de Default.html	La página anfitriona de la aplicación, donde un único enlace a una hoja de estilos pone el CSS y los archivos de fuente del pack de iconos a disposición de todas las pantallas que abra la sesión.
clic de elemento según su rol	Un clic dentro de contenido HTML puede informar de qué elemento en línea se pulsó, así que una misma celda de cuadrícula puede ofrecer un destino de Edit y otro de Delete sin dos controles separados.
CDN frente a alojamiento propio	Apuntar el enlace a una CDN pública es una decisión de producción sobre disponibilidad, privacidad, Content Security Policy y licencias; las aplicaciones internas suelen alojar ellas mismas los archivos.
política de iconos por capas	Una regla escrita que le da a cada tipo de símbolo un hogar: imágenes de tema para las semánticas, un pack principal para los comandos, el pack propio para la ilustración del producto y archivos ráster para las fotos.
pasada de QA de iconos	Un recorrido por las pantallas terminadas en un tema claro y en uno oscuro con más de una escala de pantalla, comprobando que cada icono sigue viéndose y sigue teniendo una alternativa textual.

3. Desarrollo de la lección

Cargar la hoja de estilos y activar el marcado

Una fuente de iconos no forma parte en absoluto del sistema de imágenes de Wisej.NET. No se asigna nada a `Image`, no se escribe nada en `ImageSource` y el servidor nunca produce un PNG. En su lugar, el navegador descarga una hoja de estilos y un archivo de fuente, y un nombre de clase en un elemento en línea le dice a la fuente qué glifo dibujar. Eso convierte la carga en un asunto de la página y no del control: el enlace a la hoja de estilos va en la cabecera del `Default.html` de la aplicación, donde se obtiene una vez por sesión y se aplica a todas las ventanas y diálogos que abra el usuario.

A dónde apunta ese enlace es una decisión de producción. Una CDN pública es cómoda, pero la interfaz depende entonces de que un tercero esté accesible, revela a ese servidor las peticiones de tus usuarios, tiene que pasar tu Content Security Policy y se rige por la licencia de la biblioteca de iconos. Para una aplicación interna, servir tú mismo el CSS y los archivos de fuente suele ser la mejor decisión: la disponibilidad y la privacidad dependen entonces solo de tu propio despliegue.

Con la hoja de estilos cargada, todavía no aparece nada. Los controles de Wisej.NET tratan `Text` como texto plano, así que el marcado llega escapado y el usuario ve corchetes angulares. Establecer `AllowHtml` en `true` es lo que cambia eso, y es deliberadamente por control: Label, Button, celdas de DataGridView, CheckBox, RadioButton y LinkLabel son los ejemplos con capacidad HTML. Activar el interruptor mueve además una frontera de confianza: a partir de ese momento el texto del control es marcado, así que mantenlo en manos de la aplicación y pon las palabras del usuario en un control de texto plano aparte.

Glifos en una barra de herramientas, una tarjeta y una celda de cuadrícula

En IconDesk la primera superficie con fuente de iconos es una pequeña barra de herramientas o tarjeta de resumen. Cada botón o etiqueta recibe `AllowHtml = true` y un valor de texto que pone un elemento en línea delante de una palabra, que es donde las fuentes de iconos resultan naturales: el glifo fluye con la etiqueta en lugar de ocupar un hueco de imagen aparte, y hereda el tamaño de fuente y el color de su entorno. La misma idea escala a un `DataGridView` con `AllowHtml` activado, donde una columna de acciones compacta contiene dos glifos en una celda en lugar de dos controles alojados.

Una sola celda con dos símbolos solo funciona si la aplicación puede saber cuál se pulsó. Para eso está el atributo `role` en el elemento en línea: el clic informa del elemento que se pulsó y el manejador se ramifica según él. Inspecciona los datos del evento de ratón que expone realmente tu versión del framework antes de depender de ellos, y no inventes un contrato que el evento no ofrece.

```
labelActions.AllowHtml = true;
labelActions.Text =
    "<i class='fa-solid fa-pen' role='edit'></i> " +
    "<i class='fa-solid fa-trash' role='delete'></i>";

labelActions.MouseClick += labelActions_MouseClick;
```

Lo compacto tiene un coste. Una celda llena de marcado es más difícil de alcanzar con el teclado que un botón de verdad, los lectores de pantalla se encuentran un glifo sin nombre y un icono que no se carga deja una columna vacía. Dale a cada acción un tooltip o una palabra visible, y recurre a una columna de imagen corriente o a un control alojado cuando la interacción sea más que un atajo de dos iconos.

Las fuentes de iconos junto a los SVG y los packs de iconos

Ninguno de los dos mecanismos gana de forma rotunda. El sistema propio de Wisej.NET se basa en recursos y en SVG, y ahí es donde es más fuerte: una cadena de origen puede nombrar una imagen de tema, una ruta `resource.wx` o un recurso de un pack; los SVG monocromos aceptan el sufijo `?color=` con un nombre de color del tema; el selector de imágenes del diseñador explora todos los packs instalados; la caché está pensada alrededor de esos recursos; y cualquier control con una propiedad de imagen puede usarlos directamente.

Las fuentes de iconos se ganan su sitio en otro terreno. Les resultan familiares a los desarrolladores web, encajan en un sistema de diseño HTML y CSS existente, sus catálogos son enormes y se colocan en línea con el texto de una forma que una propiedad de imagen no puede. Si el lenguaje de diseño de la empresa ya está expresado como clases de fuente, igualarlo cuesta un enlace a una hoja de estilos en lugar de redibujar todos los recursos.

Así que la respuesta honesta es que conviven, y la elección pertenece a la superficie y no al proyecto. Los comandos en controles con propiedades de imagen pertenecen a la canalización de recursos; los símbolos en línea dentro de un texto o de una celda de cuadrícula densa son un caso razonable para una fuente de iconos. Lo que no es aceptable es la deriva, cuando el mismo concepto de Delete es un SVG de un pack en una pantalla y un glifo de fuente en la siguiente.

La lista de comprobación de producción y la política de IconDesk

El entregable de verdad del proyecto final es una página de política. Una regla por capas funciona bien: imágenes de tema para las semánticas comunes, un pack oficial de Wisej-4 principal para los comandos amplios, el pack propio IconDesk.Icons para la ilustración propietaria, archivos ráster comprimidos para las fotografías y fuentes de iconos solo donde la integración con HTML lo merezca. Como la política nombra una primera opción para cada situación, una pantalla nueva es coherente por defecto y no por revisión.

Alrededor de eso está la lista de comprobación que el proyecto ejecuta antes de publicar una pantalla:

- Todo icono con significado tiene una palabra visible, un tooltip u otra alternativa textual, y el estado nunca se señala solo con color.
- Toda URL externa o referencia a una CDN está aprobada de forma explícita, y los términos de licencia de la biblioteca están registrados.
- El marcado de la fuente de iconos es propiedad de la aplicación o está codificado; los valores no confiables nunca llegan a un control con `AllowHtml`.
- Las URL de los recursos son estables para que la caché del navegador pueda funcionar, y ninguna cadena de consulta aleatoria la echa por tierra.
- La pantalla se ha abierto en un tema claro y en uno oscuro y con una escala de pantalla mayor, y los hallazgos están anotados.

Esa pasada de QA es donde los cinco mecanismos quedan por fin uno al lado del otro. En la página de resumen de IconDesk aparecen a la vez una imagen de tema, un icono de un pack oficial, un icono de un pack propio, un recurso incrustado al que se llega mediante `resource.wx` y un SVG recoloreado, y cambiar el tema muestra cuáles lo siguieron y cuáles se quedaron exactamente como estaban dibujados. Explicar, para cada imagen visible, dónde se almacena y qué API la resolvió es la destreza que ha ido construyendo este curso.



Nombra primero el mecanismo

Antes de asignar cualquier imagen, di qué mecanismo estás usando: un objeto `System.Drawing.Image` que el servidor renderiza como PNG, una cadena de origen de imagen que resuelve el navegador, una imagen de tema con nombre, un recurso incrustado servido por `resource.wx` o una fuente de iconos dibujada como HTML. Asigna una sola propiedad de imagen por control para que las demás no puedan restablecerla en silencio, mantén la cadena de origen con significado en lugar de una ruta mágica pegada, prefiere SVG para los iconos de interfaz para que se mantengan nítidos en cualquier tamaño y escala, y dale una alternativa textual a todo icono que transmita significado.



Objetivos de aprendizaje

Al terminar este módulo serás capaz de:

- ✓ Cargar una hoja de estilos de fuente de iconos y renderizar iconos en controles de Wisej.NET con AllowHtml.
- ✓ Usar fuentes de iconos en el contenido de Label, Button y DataGridView y entender los clics de elemento según su rol.
- ✓ Comparar las fuentes de iconos con los recursos SVG y de packs de iconos de Wisej.NET.
- ✓ Aplicar una lista de comprobación de producción que cubra accesibilidad, seguridad, caché, dependencias externas y coherencia.

Guía del laboratorio / examen

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



Laboratorio práctico

Termina el proyecto final de IconDesk. Referencia una hoja de estilos de fuente de iconos desde Default.html, activa AllowHtml en una pequeña barra de herramientas o tarjeta y renderiza sus iconos como glifos de fuente, después construye una columna de DataGridView cuyas celdas contengan acciones con fuente de iconos y lee qué elemento pulsó el usuario para que Edit y Delete hagan cosas distintas. Construye una página de resumen que demuestre a la vez todos los mecanismos vistos en el curso: una imagen de tema, un icono de un pack oficial, un icono de un pack propio, un logotipo incrustado y un SVG recoloreado. Después escribe la política de iconos de producción del proyecto y haz una pasada de QA en un tema claro y en uno oscuro.

Punto de partida

Abre la solución IconDesk tal como la dejó el Módulo 6: el panel de comandos, las páginas ImageLab e IconGallery, dos packs de iconos oficiales de Wisej-4, la carpeta Resources incrustada y servida mediante `resource.wx`, y la biblioteca de clases IconDesk.Icons con su catálogo `AppIcons`. Todavía no hay nada en el proyecto que use una fuente de iconos, y el `Default.html` del proyecto sigue teniendo la cabecera vacía salvo por el título.

Enfoque sugerido

- ✓ Decide primero si la fuente de iconos se referencia desde una CDN o se aloja junto a la aplicación, y anota las razones de disponibilidad, privacidad, Content Security Policy y licencia de esa decisión antes de añadir el enlace a `Default.html`.
- ✓ Añade una pequeña barra de herramientas o tarjeta, establece `AllowHtml` en cada Label o Button y comprueba que un control sin el interruptor muestra el marcado como texto literal, para que hayas visto los dos comportamientos.
- ✓ Activa `AllowHtml` en la cuadrícula, construye la columna de acciones con dos elementos en línea que lleven roles distintos, inspecciona después los datos del evento de ratón que expone tu versión de Wisej.NET y ramifica Edit y Delete según el elemento que se pulsó.
- ✓ Construye la página de resumen de modo que cada uno de los cinco mecanismos aparezca una vez, etiquetado con dónde se almacena la imagen y qué API la resuelve, y mantén un control por propiedad de imagen.
- ✓ Haz la pasada de QA: cambia entre un tema claro y uno oscuro, prueba una escala de pantalla mayor, revisa los tooltips y el acceso por teclado a las acciones que solo tienen icono, y anota todo lo que cambió o desapareció.

Entregables requeridos

- ✓ Hoja de estilos de fuente de iconos referenciada desde Default.html y una barra de herramientas o tarjeta con AllowHtml que renderiza glifos de fuente

- ✓ Columna de DataGridView cuyas celdas renderizan acciones con fuente de iconos y responden al elemento pulsado
- ✓ Página de resumen de IconDesk que demuestra una imagen de tema, un icono de un pack oficial, un icono de un pack propio, un recurso incrustado y un SVG recoloreado
- ✓ Política de iconos de producción escrita, que cubra accesibilidad, licencias, caché y dependencias externas
- ✓ Pasada de QA realizada en un tema claro y en uno oscuro con los hallazgos registrados

Preguntas de repaso

- ✓ El enlace a la hoja de estilos está en `Default.html` y los nombres de clase son correctos, pero una Label muestra las etiquetas en bruto mientras que el Button de al lado muestra el glifo. ¿Qué diferencia hay entre los dos controles y por qué la configuración es por control y no por aplicación?
- ✓ Tu columna de acciones funciona y un compañero pide meter el texto de la nota del cliente en esa misma celda para poder darle estilo. ¿Qué le dices y qué construirías en su lugar?
- ✓ En la página de resumen, ¿cuáles de las cinco imágenes cambian cuando la aplicación cambia de tema y cuáles se quedan exactamente como estaban dibujadas? ¿Qué te dice eso sobre dónde se almacena cada una?



Nombra primero el mecanismo

Antes de asignar cualquier imagen, di qué mecanismo estás usando: un objeto `System.Drawing.Image` que el servidor renderiza como PNG, una cadena de origen de imagen que resuelve el navegador, una imagen de tema con nombre, un recurso incrustado servido por `resource.wx` o una fuente de iconos dibujada como HTML. Asigna una sola propiedad de imagen por control para que las demás no puedan restablecerla en silencio, mantén la cadena de origen con significado en lugar de una ruta mágica pegada, prefiere SVG para los iconos de interfaz para que se mantengan nítidos en cualquier tamaño y escala, y dale una alternativa textual a todo icono que transmita significado.