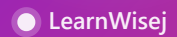


Recursos de la aplicación, ResourceManager y el texto de sistema de Wisej

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

La capa de recursos compartidos: archivos .resx neutros y específicos de cultura que llevan claves idénticas, la búsqueda de Wisej.Resources.ResourceManager por nombre base y ensamblado, marcadores de posición formateados con string.Format y la cultura activa, servicios que devuelven claves en lugar de frases en inglés, y Resources-[LANG].resx para las etiquetas de sistema de Wisej. Qué significa esto para un desarrollador que conoce los fundamentos de C# y los controles de Wisej.NET y ahora tiene que publicar la aplicación en más de un idioma, y cómo leer este módulo.



Enfoque de producción

La capa de recursos compartidos: archivos .resx neutros y específicos de cultura que llevan claves idénticas, la búsqueda de Wisej.Resources.ResourceManager por nombre base y ensamblado, marcadores de posición formateados con string.Format y la cultura activa, servicios que devuelven claves en lugar de frases en inglés, y Resources-[LANG].resx para las etiquetas de sistema de Wisej.

1. El problema en producción

El Módulo 2 dejó el editor de clientes de GlobalDesk con un aspecto convincente en alemán: todas las etiquetas que posee el diseñador cambian con la cultura y el diseño aguanta. Entonces el usuario pulsa Save y lee *Customer saved.* en inglés, porque esa frase no es una propiedad de ningún control: es un literal de cadena dentro de un manejador de clic, al lado de *This field is required.* y de la línea de estado que cuenta los contactos cargados.

La localización en el diseñador nunca puede alcanzar esas cadenas: las genera el código, las comparten varias pantallas o las produce un servicio que no tiene ni idea de que existe una interfaz de usuario. Este módulo le da a GlobalDesk una segunda capa de recursos justo para ese texto y después baja un nivel más: las etiquetas que el propio Wisej.NET dibuja en un `MessageBox` o en un calendario.

2. Conceptos clave

Concepto	Por qué importa
archivo de recursos compartido de la aplicación	<code>Resources/Strings.resx</code> contiene el texto visible para el usuario que no pertenece a ninguna pantalla diseñada concreta: confirmaciones de guardado, mensajes de validación, líneas de estado, títulos de informes y resultados de servicios.
archivo compañero por cultura	<code>Strings.de.resx</code> y <code>Strings.it.resx</code> se colocan junto al archivo neutro y llevan las mismas claves ; un traductor cambia valores, nunca nombres de claves.
clave semántica estable	Un nombre como <code>CustomerEditor.Saved</code> describe el significado del texto, así que reescribir la frase en inglés nunca obliga a cambiar código ni rompe las traducciones.
<code>Wisej.Resources.ResourceManager</code>	El objeto de búsqueda que sigue el modelo de recursos de .NET: dados un nombre base y un ensamblado, encuentra el valor para la cultura activa y se repliega hacia el archivo neutro.
marcador de clave ausente	Un helper que devuelve <code>[Key]</code> cuando una búsqueda falla convierte un defecto invisible en algo que puedes ver y buscar, en lugar de en una etiqueta en blanco.
cadena de formato con marcadores de posición	La frase entera con sus <code>{0}</code> y <code>{1}</code> vive en el recurso y <code>string.Format</code> la rellena, porque cada idioma ordena el nombre, la fecha y el verbo de forma distinta.
<code>Resources-[LANG].resx</code>	Un archivo de recursos de sistema de Wisej.NET aparte, en la raíz del proyecto o en una carpeta <code>/Resources</code> , que sobrescribe o añade etiquetas del framework como <code>\$Ok</code> , <code>\$Cancel</code> y <code>\$Offline</code> .

3. Desarrollo de la lección

Dos archivos, un único conjunto de claves

Empieza por el archivo neutro. En GlobalDesk es `Resources/Strings.resx`, y ya lleva las claves que introdujo el Módulo 1: `Navigation.Customers`, `CustomerEditor.Save`, `CustomerEditor.Cancel`, `CustomerEditor.Saved`, `Validation.Required`. El archivo neutro es aquel al que .NET se repliega cuando no existe nada mejor, así que nunca se le permite estar incompleto. Una clave que falte aquí es un defecto del desarrollador, no un hueco de traducción.

Un archivo traducido es el mismo nombre de archivo con un nombre de cultura antes de la extensión: `Strings.de.resx` para el alemán, `Strings.it.resx` para el italiano. Solo difiere la columna de **valores**. Renombra `CustomerEditor.Saved` en el archivo alemán y dejará de responder a la búsqueda, con lo que la pantalla se repliega en silencio al inglés. La lista de claves es un contrato: añadir una clave significa añadirla en todas partes.

Un único archivo alemán sirve igual a `de` , `de-DE` y `de-AT` : .NET recorre desde la cultura más específica hacia el recurso neutro y toma la primera coincidencia. Añade `Strings.de-AT.resx` solo cuando la redacción austriaca difiera de verdad.

Buscar el texto a través de ResourceManager

Wisej.NET proporciona `Wisej.Resources.ResourceManager` , que sigue el modelo de recursos de .NET y sabe localizar los recursos específicos de cada cultura. Construyes uno a partir de un nombre base y del ensamblado que contiene los recursos; el nombre base es el namespace del proyecto más la carpeta y el nombre del archivo, que para GlobalDesk es `GlobalDesk.Resources.Strings` . Equivocarse ahí es el fallo más común de quien empieza, y se manifiesta como todas las claves devolviéndose como ausentes a la vez.

Envuelve el manager en un pequeño helper estático, para que el resto de la aplicación nunca lo toque directamente y un único método decida qué ocurre con una clave ausente:

```
using Wisej.Resources;

public static class Texts
{
    private static readonly ResourceManager RS =
        new ResourceManager("GlobalDesk.Resources.Strings", typeof(Texts).Assembly);

    public static string Get(string key)
        ⇒ RS.GetString(key) ?? $"[{key}]";
}
```

El marcador entre corchetes es un buen valor predeterminado para el desarrollo: una etiqueta vacía parece un error de diseño, mientras que `[CustomerEditor.Saved]` te dice exactamente qué clave añadir y dónde. En producción quizá prefieras registrar la clave ausente, algo que el Módulo 7 convierte en una política. Lo que nunca debes hacer es dejar que un recurso ausente se convierta en texto en blanco silencioso.

Recursos del diseñador, recursos compartidos y dónde va cada cadena

Ahora conviven dos mecanismos de recursos en el mismo proyecto, y responden a preguntas distintas. Los recursos del diseñador pertenecen a una superficie diseñada, se escriben en los archivos `.resx` propios de ese contenedor y se aplican cuando se crean sus controles. Los recursos compartidos de la aplicación pertenecen a la aplicación y se leen bajo demanda a través de `Texts.Get` , así que están tan al día como el momento en que los llamas.

La prueba práctica es la propiedad. Si una etiqueta existe porque arrastraste un `Button` al editor de clientes, es del diseñador. Si el texto aparece en un `MessageBox` , en una línea de estado, en la cabecera de una columna de cuadrícula que construyes en código o en dos pantallas distintas, pertenece a `Strings.resx` . Duplicar una frase en tres archivos del diseñador es como nacen las aplicaciones a medio traducir: la tercera copia siempre es la que nadie actualiza.

El mismo razonamiento llega hasta el dominio. Un servicio de guardado no debería devolver la frase *Customer saved.*; debería devolver un resultado estable como la clave `CustomerEditor.Saved` o un código

de resultado que la interfaz asigne a una clave. Así la lógica de negocio sigue siendo comprobable, no se escapa inglés de la capa de servicios y la pantalla sigue siendo el único sitio que sabe cómo hablarle al usuario.

Las etiquetas propias de Wisej: localización de sistema y Resources-[LANG].resx

No todas las palabras de la pantalla son tuyas. Wisej.NET localiza sus propios recursos del framework (los botones del `MessageBox`, las etiquetas del calendario y el comportamiento sensible a la cultura de los editores numéricos, de fecha y de moneda) e incluye traducciones de sistema integradas para inglés, alemán, francés, italiano, turco, español y portugués. Ejecuta GlobalDesk en alemán y los botones Yes/No cambian solos.

Para un idioma que Wisej.NET no cubra, o para una redacción que tu producto exija, Wisej.NET admite un archivo `Resources-[LANG].resx` en la raíz del proyecto o en una carpeta `/Resources`. Sus claves son las del propio framework y empiezan con un signo de dólar: `$Ok`, `$Cancel`, `$Yes`, `$No`, `$Retry`, las etiquetas de navegación del calendario y `$Offline`. Mantén las dos cosas separadas: `Strings.de.resx` es lo que dice GlobalDesk, `Resources-de.resx` es lo que dice Wisej.NET.

Mensajes, validación e imágenes bajo la cultura activa

Las frases que mencionan datos necesitan marcadores de posición, y los marcadores de posición son donde quien empieza pierde el orden de las palabras. Guarda la frase completa en el recurso — `Customer.LastOrder` contiene `{0} last ordered on {1}`. — y deja que el código aporte los valores: `string.Format(Application.CurrentCulture, Texts.Get("Customer.LastOrder"), customerName, lastOrderDate)`. Pasar la cultura también da a la fecha el formato que la sesión espera. No concatenes nunca `"Customer " + name + " saved"`: un traductor no puede mover un fragmento que el código ha pegado.

Los marcadores de posición no significan nada para alguien que solo ve la columna de valores, así que usa el campo de comentario del editor `.resx` para decir qué son `{0}` y `{1}`. Los mensajes de validación merecen el mismo cuidado: `Validation.Required` y `Validation.Email` los reutilizan varias pantallas, y por eso pertenecen al archivo compartido y no a cada formulario.

Un archivo `.resx` puede contener más que cadenas: las imágenes y otros datos son entradas legítimas, y la localización en el diseñador puede intercambiar una imagen por idioma. Hazlo a propósito, eso sí. Una imagen que solo se diferencia porque lleva palabras incrustadas es una trampa de mantenimiento; deja ese texto como texto real en la interfaz, donde `Texts.Get` lo traduce y ningún diseñador tiene que redibujar nada.

🔔 Regla de las capas de localización

Decide dónde vive cada cadena antes de teclearla. Los recursos del diseñador pertenecen a una pantalla diseñada y se aplican cuando se crean sus controles; los recursos de aplicación compartidos en `.resx` llevan el texto que genera el código o que reutilizan varias pantallas; `Resources-[LANG].resx` sobrescribe las etiquetas de sistema de Wisej.NET. Nunca guardes una fecha, un número o un importe ya formateados: conserva el valor y dale formato para la cultura activa en el momento de mostrarlo. Y recuerda que la cultura es por sesión, así que el idioma de un usuario nunca debe cambiar la pantalla de otro.



Objetivos de aprendizaje

Al terminar este módulo serás capaz de:

- ✓ Crear archivos de recursos .resx neutros y específicos de cultura.
- ✓ Recuperar valores a través de ResourceManager usando claves estables.
- ✓ Organizar las cadenas reutilizables por separado de los recursos del diseñador.
- ✓ Explicar la localización de sistema integrada de Wisej.NET y los archivos Resources-[LANG].resx personalizados.
- ✓ Usar recursos sensibles a la cultura para mensajes, validación e imágenes.

Guía del laboratorio / examen

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



Laboratorio práctico

Amplía los recursos compartidos de GlobalDesk. Añade `Strings.de.resx` junto a `Resources/Strings.resx` con exactamente las mismas claves y los valores traducidos, añade una entrada `Customer.LastOrder` que contenga dos marcadores de posición, y compón la frase con `string.Format` y `Application.CurrentCulture` en lugar de concatenar fragmentos. Haz pasar por el helper `Texts` todos los mensajes que muestra el editor de clientes, incluidas la confirmación de guardado y los dos mensajes de validación, y haz que el servicio de guardado devuelva una clave de resultado estable como `CustomerEditor.Saved` en lugar de una frase en inglés que la interfaz imprime directamente. Después añade un archivo `Resources-de.resx` que sobrescriba una etiqueta de sistema de Wisej, demuéstalo en un `MessageBox` y escribe una nota breve sobre la diferencia entre el texto de tu producto y el texto de sistema de Wisej.

Punto de partida

Abres GlobalDesk tal y como lo dejó el Módulo 2: el dashboard lee sus etiquetas de

`Resources/Strings.resx` a través del helper `Texts`, y el control de usuario `CustomerEditor` está localizado en el diseñador con una variante alemana. La confirmación de guardado, los dos mensajes de validación y la línea del último pedido siguen siendo literales en inglés dentro del código, y todavía no existe ningún archivo de recursos compartido traducido.

Enfoque sugerido

- ✓ Añade `Resources/Strings.de.resx` junto al archivo neutro y copia la lista de claves sin cambiarla, traduciendo solo los valores; ejecuta la aplicación con `de-DE` y confirma que las etiquetas del dashboard cambian antes de tocar nada de código.
- ✓ Añade una entrada `Customer.LastOrder` cuyo valor sea una frase completa con `{0}` y `{1}`, rellena el campo de comentario para que un traductor sepa que el primero es el nombre del cliente y el segundo la fecha del pedido, y compón la línea con `string.Format(Application.CurrentCulture, Texts.Get("Customer.LastOrder"), customerName, lastOrderDate)`.
- ✓ Sustituye los literales de `CustomerEditor` por llamadas a `Texts.Get` para `CustomerEditor.Saved`, `Validation.Required` y `Validation.Email`, y elimina toda concatenación de cadenas que encuentres por el camino.
- ✓ Cambia el servicio de guardado para que devuelva un resultado estable (la clave `CustomerEditor.Saved` o un código de resultado) y deja que el editor asigne ese resultado a texto mediante `Texts.Get`, de modo que no quede ninguna frase en inglés en la capa de servicios.
- ✓ Añade `Resources-de.resx` en la raíz del proyecto o en la carpeta `/Resources`, sobrescribe una etiqueta de sistema de Wisej.NET como `$Ok` o `$Cancel`, y después muestra un `MessageBox` en alemán y compara sus botones con la ejecución en inglés.

Entregables requeridos

- ✓ Strings.de.resx con las mismas claves que el archivo neutro y solo los valores traducidos
- ✓ Recurso Customer.LastOrder compuesto con string.Format y la cultura activa, nunca concatenado
- ✓ Servicio de guardado que devuelve una clave de recurso o un código de resultado en lugar de una frase en inglés
- ✓ Resources-de.resx que sobrescribe al menos una etiqueta de sistema de Wisej, mostrada en un MessageBox
- ✓ Nota de laboratorio que separa el texto del producto, el texto de sistema de Wisej y el texto de diagnóstico

Preguntas de repaso

- ✓ Borra una clave de `Strings.de.resx` y ejecuta en alemán. Describe qué ve el usuario, por qué no se rompe nada y cómo detectarías esto en una aplicación real con doscientas claves.
- ✓ Tu servicio de guardado devuelve ahora `CustomerEditor.Saved`. ¿Qué se rompería en las pruebas unitarias, y en la pantalla en alemán, si devolviera la frase en sí?
- ✓ Un compañero añade la palabra "Cancel" a `Resources-de.resx` bajo una clave inventada por él. Explica por qué el botón del `MessageBox` no cambia y dónde debería haber ido realmente esa cadena.



Regla de las capas de localización

Decide dónde vive cada cadena antes de teclearla. Los recursos del diseñador pertenecen a una pantalla diseñada y se aplican cuando se crean sus controles; los recursos de aplicación compartidos en .resx llevan el texto que genera el código o que reutilizan varias pantallas; Resources-[LANG].resx sobrescribe las etiquetas de sistema de Wisej.NET. Nunca guardes una fecha, un número o un importe ya formateados: conserva el valor y dale formato para la cultura activa en el momento de mostrarlo. Y recuerda que la cultura es por sesión, así que el idioma de un usuario nunca debe cambiar la pantalla de otro.