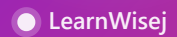


Flujo de trabajo de traducción: ResX Resource Manager y otras herramientas

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

Tratar las traducciones como trabajo de equipo: la matriz de ResX Resource Manager sobre todos los .resx de la solución, añadir una columna de idioma y filtrar las entradas sin traducir, comentarios de recurso que dan contexto a los traductores, cadenas invariantes, la traducción automática solo como borrador, la exportación e importación con una plataforma compatible con RESX, y revisar los diffs de recursos como si fueran código. 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

Tratar las traducciones como trabajo de equipo: la matriz de ResX Resource Manager sobre todos los .resx de la solución, añadir una columna de idioma y filtrar las entradas sin traducir, comentarios de recurso que dan contexto a los traductores, cadenas invariantes, la traducción automática solo como borrador, la exportación e importación con una plataforma compatible con RESX, y revisar los diffs de recursos como si fueran código.

1. El problema en producción

GlobalDesk lleva ya un `Resources/Strings.resx` neutro, un `Strings.de.resx` alemán, recursos de diseñador en el editor de clientes y un `Resources-de.resx` para las etiquetas de sistema de Wisej. Añadir el italiano con el editor de Visual Studio significa abrir esos archivos de uno en uno y comparar listas de claves a ojo, y la persona que habla italiano no suele ser la que puede abrir la solución. Este módulo convierte la traducción en un flujo de trabajo con una lista de huecos visible, una entrega y un paso de revisión.

2. Conceptos clave

Concepto	Por qué importa
Vista de matriz	Las claves en las filas y los idiomas en las columnas, así que un valor italiano que falta y un término alemán que se ha desviado se ven a la vez.
ResX Resource Manager	Una herramienta de localización .NET de código abierto, disponible como extensión de Visual Studio y como aplicación independiente, que muestra los recursos RESX de una solución en una única cuadrícula central.
Columna de idioma	Añadir un idioma de destino crea el archivo específico de esa cultura junto al neutro y te da una columna vacía que rellenar; las claves no cambian nunca.
Filtro de entradas sin traducir	Lista solo las entradas sin valor en el idioma seleccionado, de modo que "¿hemos terminado?" pasa a ser un número en lugar de una opinión.
Comentario de recurso	Una nota guardada junto al valor que le dice al traductor si Open es un verbo, un adjetivo, el estado de un ticket o una orden.
Cadena invariante	Una marca en las entradas que nunca deben traducirse, como el nombre del producto y los identificadores técnicos.
Ida y vuelta de exportación e importación	Columnas enviadas a un traductor o a una plataforma compatible con RESX y devueltas como archivos .resx específicos de cultura que la compilación trata como cualquier otro recurso.

3. Desarrollo de la lección

Una cuadrícula en lugar de seis archivos

El editor de recursos de Visual Studio está muy bien para un archivo. Deja de estarlo en cuanto la pregunta es comparativa: ¿tiene cada clave del archivo neutro un valor en alemán? Una herramienta de traducción responde a eso presentando los recursos como una matriz, las claves en las filas y cada idioma en su propia columna. ResX Resource Manager es la elección habitual: una herramienta .NET de código abierto que se distribuye como extensión de Visual Studio, que analiza la solución activa y reúne sus recursos RESX en una única cuadrícula central, y también como aplicación independiente.

Encaja con Wisej.NET porque no hay nada que adaptar. Wisej.NET guarda el texto localizado en archivos RESX estándar de .NET, así que una herramienta RESX genérica lee GlobalDesk igual que lee cualquier otro proyecto .NET; el ejemplo StockViewer de Wisej menciona la localización hecha con el diseñador junto con ResX Manager. El conjunto de funciones documentado es el propio flujo de trabajo: columnas de idioma una al lado de otra, entradas sin traducir resaltadas, traducción dentro de la cuadrícula, filtrado y ordenación, traducción automática, importación y exportación, detección de referencias en el código, cadenas invariantes, instantáneas, scripting y corrección ortográfica. La detección de referencias en el código demuestra pronto su valor: saber dónde se usa una clave suele ser la única forma de decidir qué significa una etiqueta de una sola palabra.

Lo que la herramienta no es es parte de tu aplicación. Edita archivos; en tiempo de ejecución GlobalDesk sigue haciendo la búsqueda de recursos normal de .NET a través de `Wisej.Resources.ResourceManager`, exactamente como en el Módulo 3. Nunca dejes que una herramienta de localización se convierta en una dependencia de ejecución.

Añadir el italiano y convertir "sin traducir" en un número

Añadir un idioma crea el archivo específico de esa cultura junto al neutro, `Strings.it.resx` al lado de `Strings.resx`, y te da una columna nueva y vacía. No se vuelve a generar ninguna clave; el recurso neutro sigue siendo la única fuente de verdad sobre qué claves existen. El filtro de entradas sin traducir convierte entonces una pregunta vaga en una lista que puedes trabajar y entregar.

Vale la pena escribir en una ficha el orden de trabajo seguro. Mantén `Strings.resx` completo y revisado en el control de versiones, abre ResX Resource Manager, añade el idioma de destino, traduce o importa valores conservando las claves, filtra las entradas vacías y despégalas, compila y ejecuta con la cultura de destino, revisa el diff antes de confirmar (porque, según su configuración, la herramienta puede guardar tus ediciones al instante) y, por último, prueba la interfaz de usuario y no solo la cuadrícula.

Una cuadrícula completa con un botón recortado sigue siendo un error, y las etiquetas en italiano suelen ser más largas que las inglesas. La trampa contraria es más silenciosa: una clave sin valor en italiano no revienta, se repliega al valor neutro, así que la pantalla parece terminada mientras el hueco vive solo en la cuadrícula.

La traducción automática es un borrador, y el diff es la revisión

La traducción automática es realmente útil para una primera cobertura. Es un borrador. Revisa la terminología, los marcadores de posición, la puntuación, las teclas de acceso, los nombres de producto y el contexto antes de publicar nada de eso, porque una traducción puede ser técnicamente correcta y aun así no ser las palabras adecuadas en una interfaz. La entrada `Customer.LastOrder` del Módulo 3 hace concreto el riesgo: un borrador que pierde o reordena sus dos marcadores de posición convierte una llamada de formato en un fallo.

La solución para la ambigüedad es el contexto, y el contexto vive en el comentario del recurso, que viaja con la entrada en cada ida y vuelta:

```
<data name="CustomerEditor.Open" xml:space="preserve">
  <value>Open</value>
  <comment>Button caption, verb: opens the selected customer.</comment>
</data>
<data name="App.ProductName" xml:space="preserve">
  <value>GlobalDesk</value>
  <comment>Product name, invariant: never translate.</comment>
</data>
```

Trata esos archivos como código: revisa los diffs, mantén las claves estables, evita los renombrados masivos mientras hay una ronda en marcha y no dejes nunca que un borrador automático sobrescriba una traducción humana aprobada sin revisión. Como la herramienta puede escribir mientras tecleas, el diff del control de versiones es la red de seguridad.

Entregar las cadenas a alguien que nunca abre Visual Studio

La exportación y la importación existen justo para esto. Exportas la columna neutra junto con la columna de destino, el traductor trabaja en ese archivo, y la reimportación empareja las filas devueltas con sus claves, de modo que las claves y los marcadores de posición sobreviven a una persona que nunca ha visto un `.resx`. Lo que vuelve tiene que ser el mismo conjunto de claves.

Cuando los idiomas o los revisores se multiplican, la misma forma se traslada a un sistema de gestión de traducciones en la nube. Entre las plataformas compatibles con `.resx` están Crowdin, Lokalise, Phrase y POEditor, y la ida y vuelta no cambia: subes o sincronizas el RESX neutro, traduces y revisas en la plataforma, exportas los archivos `.resx` específicos de cada cultura, los confirmas y después compilas, ejecutas y verificas en la interfaz real. Lo que compras es gobernanza: memoria de traducción, roles de revisor y terminología centralizada.

Elegir entre el editor del diseñador, ResXManager y una plataforma en la nube

La guía práctica es corta, y va del equipo más que de la herramienta:

- Un desarrollador, dos idiomas: el editor RESX de Visual Studio puede bastar ya.
- Un equipo de desarrolladores, varios idiomas: ResX Resource Manager, un flujo eficiente dentro del IDE.
- Traductores externos: exportación e importación, o una plataforma compatible con RESX.
- Muchos productos, idiomas y revisores: un TMS en la nube es más fácil de gobernar.

Elijas lo que elijas, el contrato del proyecto no se mueve: archivos `.resx` específicos de cada cultura junto al archivo neutro con claves idénticas, leídos mediante la búsqueda de recursos estándar. Escoge la opción más ligera que haga visibles las entradas sin traducir y revisable el diff, y da el salto solo cuando el cuello de botella sea la ida y vuelta.



Errores habituales

Publicar un borrador automático porque la cuadrícula se puso verde. Perder un marcador de posición en un valor traducido. Renombrar claves a mitad de una ronda. Dejar que la herramienta guarde ediciones que nunca has leído en un diff. Traducir el nombre del producto porque nadie lo marcó como invariante. Dar el trabajo por terminado sin ejecutar la aplicación con la cultura de destino.



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:

- ✓ Usar ResX Resource Manager para revisar varios idiomas RESX uno al lado de otro.
- ✓ Añadir un idioma e identificar las entradas sin traducir.
- ✓ Explicar el uso seguro de la traducción automática y del control de versiones.
- ✓ Exportar e importar contenido de traducción para traductores externos o un TMS.
- ✓ Comparar los flujos de localización solo con Visual Studio, con ResXManager y en la nube.

Guía del laboratorio / examen

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



Laboratorio práctico

Ejecuta una ronda de traducción honesta sobre GlobalDesk. Abre la solución en ResX Resource Manager, añade el italiano como nueva columna de idioma y usa el filtro de entradas sin traducir para listar todas las claves que aún no tienen valor. Añade comentarios a las tres claves cuyo significado es ambiguo fuera de contexto, empezando por la etiqueta de una sola palabra Open; después traduce o genera un borrador automático de los valores en italiano y marca como invariantes el nombre del producto y los dos identificadores técnicos para que nadie los traduzca. Exporta las columnas neutra e italiana, vuelve a importarlas y revisa el diff resultante del .resx antes de confirmarlo como un único cambio revisable. Termina con una nota de una página sobre el flujo de trabajo que compare el editor de Visual Studio, ResX Resource Manager y una plataforma en la nube para un equipo cuyos traductores no son desarrolladores.

Punto de partida

GlobalDesk después del Módulo 5: `Resources/Strings.resx` y `Strings.de.resx` llevan las mismas claves, el editor de clientes tiene recursos de diseñador en alemán, `Resources-de.resx` sobrescribe una etiqueta de sistema de Wisej, el selector de idioma ya ofrece inglés, alemán y árabe, y el diseño sobrevive a las cadenas largas. No hay italiano por ninguna parte, no hay ni un comentario en una sola entrada, y no hay forma de responder a "¿cuánto queda?" salvo desplazándose por dos archivos uno al lado del otro.

Enfoque sugerido

- ✓ Antes de añadir nada, abre `Strings.resx` y `Strings.de.resx` en el editor de Visual Studio e intenta encontrar a ojo las claves que le faltan al archivo alemán. Cronométrate; ese número es el argumento del resto del laboratorio.
- ✓ Abre la solución en ResX Resource Manager y añade el italiano. Confirma que aparece `Strings.it.resx` junto al archivo neutro, que la lista de claves está intacta y que la columna nueva está vacía. Activa el filtro de entradas sin traducir y anota el número.
- ✓ Comenta antes de traducir. Dale a `CustomerEditor.Open` un comentario que diga si es un verbo o el estado de un ticket, y haz lo mismo con las otras dos claves cuyo significado no resulta obvio a partir del valor; después marca como invariantes el nombre del producto y los dos identificadores técnicos.
- ✓ Rellena la columna italiana, usando la traducción automática para una primera cobertura si quieres, y después revisa cada valor: terminología, puntuación y, sobre todo, los marcadores de posición de `Customer.LastOrder`. Vuelve a ejecutar el filtro de entradas sin traducir hasta que no devuelva nada.
- ✓ Exporta las columnas neutra e italiana, vuelve a importar el mismo archivo y lee línea por línea el `git diff` de `Strings.it.resx` antes de confirmar la ronda como un único cambio. Después ejecuta GlobalDesk con `it-IT` y comprueba las pantallas, no la cuadrícula.

Entregables requeridos

- ✓ Strings.it.resx producido con ResX Resource Manager sin ninguna clave sin traducir
- ✓ Comentarios de recurso en al menos tres claves ambiguas para que el traductor tenga el contexto
- ✓ Marcas de invariante en el nombre del producto y en los identificadores que no deben traducirse
- ✓ Exportación, reimportación y un diff del .resx revisado y confirmado como un único cambio
- ✓ Nota de flujo de trabajo que compara el editor de Visual Studio, ResX Resource Manager y un TMS en la nube

Preguntas de repaso

- ✓ Tu columna italiana está completa y la aplicación se ve bien en `it-IT`, pero un compañero afirma que dos claves nunca se tradujeron. ¿Cómo pueden ser ciertas las dos cosas a la vez y qué función de la herramienta zanja la discusión?
- ✓ Un borrador automático en italiano devolvió `Ultimo ordine: {1} il {0}` para una clave cuyo valor neutro es `Last order: {0} on {1}`. ¿Qué se rompe, cuándo se rompe y qué lo habría detectado antes de la confirmación?
- ✓ Tu equipo tiene cinco idiomas, dos de ellos traducidos por una agencia que nunca ha abierto Visual Studio. ¿Cuál de los tres flujos de trabajo adoptarías y qué tendría que cambiar antes de pasar al siguiente nivel?



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.