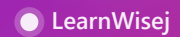


Cultura del navegador, cultura de sesión y cambio de idioma en tiempo de ejecución

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

De dónde procede la cultura de una sesión y qué tiene que ocurrir cuando cambia: el ajuste culture de Default.json, la detección automática del navegador, el parámetro de URL lang, Application.CurrentCulture como estado por sesión, el evento Application.CultureChanged y la reaplicación del texto respaldado por recursos para que ninguna pantalla a medio traducir sobreviva al cambio. 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

De dónde procede la cultura de una sesión y qué tiene que ocurrir cuando cambia: el ajuste culture de Default.json, la detección automática del navegador, el parámetro de URL lang, Application.CurrentCulture como estado por sesión, el evento Application.CultureChanged y la reaplicación del texto respaldado por recursos para que ninguna pantalla a medio traducir sobreviva al cambio.

1. El problema en producción

Tras el Módulo 3, GlobalDesk tiene un archivo de recursos neutro, uno alemán a su lado, un helper `Texts`, un editor de clientes localizado en el diseñador e incluso una etiqueta de sistema de Wisej propia. Lo que todavía no tiene es respuesta para la pregunta que un usuario real hace primero: ¿cómo consigo esto en mi idioma y qué le pasa a la pantalla cuando lo pido? Todo lo anterior se decidió mientras se construía la aplicación; este módulo trata de los dos momentos en que la decisión se toma de verdad: cuando arranca una sesión y cuando un usuario cambia de opinión a mitad de camino.

2. Conceptos clave

Concepto	Por qué importa
detección automática del navegador	Wisej.NET puede leer el idioma que informa el navegador y arrancar la sesión con él, así que cada usuario obtiene una cultura sensata sin que un único ajuste del servidor decida por todos.
<code>culture</code> en <code>Default.json</code>	El ajuste de la aplicación que decide de dónde procede la cultura de una sesión nueva; su valor predeterminado es <code>auto</code> , y un nombre fijo como <code>de-DE</code> fija en cambio todas las sesiones a una única configuración regional.
<code>rightToLeft</code> en <code>Default.json</code>	Su ajuste vecino, cuyo valor <code>auto</code> sigue a la cultura con la que acabe la sesión; el Módulo 5 se apoya en él, pero lo conoces aquí en el mismo archivo.
<code>Application.CurrentCulture</code>	Obtiene o establece la cultura de la <i>sesión actual de Wisej.NET</i> , que es justo el ámbito que necesita una aplicación web multiusuario: la elección de un usuario nunca debe repintar la pantalla de otro.
parámetro de URL <code>lang</code>	Una forma admitida de pedir un idioma al entrar, por ejemplo <code>?lang=de</code> , que hace triviales los enlaces de prueba y las URL de entrada específicas de un idioma.
<code>Application.CultureChanged</code>	Se genera cuando la cultura actual cambia respecto a aquella con la que arrancó la sesión; es el único sitio donde volver a aplicar el texto respaldado por recursos y reformatear lo que ya está en pantalla.
controles localizados en el diseñador	Sus valores localizados se aplican cuando se crean los controles, así que un cambio de cultura posterior no los reescribe: recreas el contenedor o vuelves a aplicar los valores desde el código.

3. Desarrollo de la lección

De dónde procede la cultura de una sesión

Los navegadores anuncian en cada petición los idiomas que prefiere su usuario, y Wisej.NET puede usar ese anuncio para elegir la cultura de la sesión que está a punto de iniciar. Es un valor predeterminado mucho mejor que un ajuste del servidor, porque el servidor tiene exactamente una configuración regional y sus usuarios no: dos personas que abren GlobalDesk en el mismo segundo desde navegadores configurados en inglés y en alemán obtienen cada una su sesión, cada una con su cultura, en el mismo proceso.

La detección es solo un punto de partida, no una decisión de la que no puedas salir. El idioma de un usuario puede venir igualmente de la fila de su perfil, de una preferencia de inicio de sesión o del selector que estás a punto de construir. La regla que importa es la misma gane quien gane: resuélvelo una sola vez, al principio de la sesión, y cárgalo ahí en `Application.CurrentCulture`. El código que lea recursos o dé formato a un valor más adelante nunca debería volver a preguntar de dónde salió el idioma.

Configurar el ajuste culture en Default.json

`Default.json` es donde la aplicación declara su intención. El ajuste `culture` vale `auto` de forma predeterminada, que es el comportamiento de detección del navegador que acabamos de describir, y el ajuste vecino `rightToLeft` se comporta igual: `auto` sigue a la cultura. En el laboratorio dejas `culture` en `auto` para que GlobalDesk arranque desde el navegador, y después lo demuestras cambiando el idioma preferido de tu navegador y recargando.

Un valor fijo no es un error, es una intención distinta. Fijar el ajuste a un nombre de cultura es la opción honesta para un producto que se publica a propósito en una única configuración regional, y es la forma más rápida de reproducir un informe sobre un idioma mientras depuras: todas las sesiones caen en esa cultura diga lo que diga el navegador del tester. Anota cuál de las dos elegiste y por qué, porque el siguiente desarrollador leerá ese ajuste como una declaración sobre el producto.

CurrentCulture es por sesión, nunca un campo estático

Cambiar el idioma en tiempo de ejecución es una sola asignación: establece

`Application.CurrentCulture` con la cultura que eligió el usuario, construida con `CultureInfo.GetCultureInfo` a partir de un nombre como `"en-US"`, `"de-DE"` o `"fr-CA"`. Como la propiedad tiene ámbito de sesión de Wisej.NET, esa asignación cambia el mundo de una pestaña del navegador y nada más. El ComboBox de idioma del laboratorio hace exactamente esto en su manejador `SelectedIndexChanged`, guardando los nombres de cultura como valores de los elementos para que el manejador nunca tenga que analizar una etiqueta visible.

El atajo tentador es un `static CultureInfo` en algún sitio cómodo, "para que todo pueda verlo". Los campos estáticos viven en el proceso del servidor, y un servidor Wisej.NET aloja todas las sesiones en ese único proceso, así que el campo contendría el valor que seleccionó el usuario más reciente. Un compañero en otro país vería cómo su pantalla cambia de idioma porque tú probaste el alemán. Si varias clases necesitan la cultura activa, léela cada vez de `Application.CurrentCulture`; eso ya es el almacenamiento por sesión que estabas a punto de reinventar.

El parámetro lang y el evento CultureChanged

Dos funciones de Wisej.NET enmarcan el cambio. A la entrada, el parámetro de URL `lang` pide un idioma antes de que se ejecute nada de tu código, así que `?lang=de` abre GlobalDesk en alemán sin tocar la configuración del navegador: ideal para un enlace de prueba, un informe de error o una URL de entrada específica de un idioma que le das a un cliente. Durante la sesión, `Application.CultureChanged` se genera cuando la cultura actual cambia respecto a la cultura inicial de la sesión, que es tu punto de enganche para todo lo que ya está en pantalla.

Suscríbete una sola vez, al construirse la página, y haz el trabajo en un único manejador para que nada dependa de qué control provocó el cambio:

```

public MainPage()
{
    InitializeComponent();
    Application.CultureChanged += Application_CultureChanged;
    ApplyTextResources();
}

private void Application_CultureChanged(object sender, EventArgs e)
{
    ApplyTextResources(); // every resource-backed caption
    UpdateCulturePreview(); // date, number and currency samples
    RecreateCustomerEditor(); // designer-localized: rebuild it
}

```

Repintar lo que el cambio dejó atrás

Un cambio de cultura no viaja hacia atrás por las etiquetas que ya asignaste. `ApplyTextResources` recorre los controles cuyo texto vino de la capa de recursos y vuelve a pedir a `Texts.Get` cada clave, para que el encabezado, el botón Customers y todos los mensajes lean sus nuevos valores. `UpdateCulturePreview` es su gemelo de formato: los datos almacenados no han cambiado en absoluto, así que toma la misma fecha, la misma cantidad y el mismo importe decimal y los muestra con la cultura ahora activa, y por eso conservas el valor y nunca la cadena formateada.

El editor de clientes localizado en el diseñador es la parte que pilla a la gente. Sus valores de propiedad localizados se aplican cuando se crean sus controles, así que conserva las etiquetas con las que nació por muchas veces que cambie la cultura después. La solución es estructural, no ingeniosa: haz Dispose del editor y añade una instancia nueva, ciérralo y vuelve a abrirlo, o navega a una página recién creada. Elijas lo que elijas, hazlo en el mismo manejador de `CultureChanged` y después mira la pantalla entera: un cambio que deja media ventana en el idioma anterior es peor que no cambiar nada, porque ahora el usuario desconfía de las dos mitades.



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:

- ✓ Explicar la detección automática del idioma del navegador en Wisej.NET.
- ✓ Configurar culture="auto" o una cultura fija en Default.json.

- ✓ Cambiar la cultura de la sesión con `Application.CurrentCulture`.
- ✓ Usar el parámetro de URL `lang` y gestionar `Application.CultureChanged`.
- ✓ Volver a aplicar la interfaz respaldada por recursos tras un cambio de idioma en tiempo de ejecución.

Guía del laboratorio / examen

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



Laboratorio práctico

Dale a GlobalDesk un cambio de idioma que termine el trabajo. Pon culture en auto en Default.json para que la sesión arranque con el idioma del navegador, añade un ComboBox de idioma que ofrezca inglés, alemán y una cultura de idioma y región, y establece Application.CurrentCulture con CultureInfo.GetCultInfo en su manejador SelectedIndexChanged sin guardar nunca la elección en un campo estático. Gestiona Application.CultureChanged para llamar a un método ApplyTextResources que actualice todas las etiquetas respaldadas por recursos y a un método UpdateCulturePreview que vuelva a formatear los ejemplos de fecha, número y moneda, y recrea en ese mismo manejador el editor de clientes localizado en el diseñador. Verifica la misma página con el parámetro de URL lang y después escribe una nota de laboratorio que describa qué le haría un campo de cultura estático a un segundo usuario conectado al mismo tiempo.

Punto de partida

Abre GlobalDesk tal y como lo dejaste tras el Módulo 3: el dashboard lee todas sus etiquetas de `Texts.Get`, `Resources/Strings.resx` y `Strings.de.resx` llevan las mismas claves, el panel de vista previa de la cultura da formato a una fecha, un número y un importe, y el editor de clientes tiene una variante alemana en el diseñador. Todavía no hay nada en pantalla que permita a un usuario cambiar de idioma, y `Application.CultureChanged` no tiene ningún suscriptor.

Enfoque sugerido

- ✓ Abre `Default.json`, confirma que `culture` está en `auto` y demuéstalo: cambia el idioma preferido de tu navegador a alemán, recarga GlobalDesk y anota qué etiquetas llegan traducidas. Escribe en una línea por qué fijarías el ajuste a un nombre de cultura concreto.
- ✓ Añade al dashboard un ComboBox `languageCombo` con tres elementos cuyos valores sean nombres de cultura (`en-US`, `de-DE` y una cultura de idioma y región como `fr-CA`) y cuyas etiquetas sean a su vez claves de recursos, no palabras escritas en el código.
- ✓ En `languageCombo_SelectedIndexChanged`, asigna `Application.CurrentCulture = CultureInfo.GetCultInfo(name)` y nada más. Resiste toda tentación de guardar la elección en un campo `static`; la sesión ya la almacena.
- ✓ Suscríbete a `Application.CultureChanged` en el constructor de la página y escribe los dos métodos que llama: `ApplyTextResources` vuelve a leer todas las etiquetas respaldadas por recursos a través de `Texts.Get`, y `UpdateCulturePreview` vuelve a formatear la misma fecha, cantidad e importe decimal almacenados con `Application.CurrentCulture`.
- ✓ En ese mismo manejador, haz Dispose del editor de clientes localizado en el diseñador y añade una instancia nueva a su contenedor anfitrión; después cambia de idioma con el editor abierto y comprueba todas las etiquetas de la pantalla. Termina abriendo `?lang=de` en una segunda sesión del navegador y añadiendo tu nota de laboratorio sobre el estado de cultura estático.

Entregables requeridos

- ✓ Default.json con culture en auto, más una nota sobre cuándo una cultura fija es la opción correcta
- ✓ Manejador del ComboBox de idioma que establece `Application.CurrentCulture` con `CultureInfo.GetCultreInfo`
- ✓ Manejador de `Application.CultureChanged` que llama a `ApplyTextResources` y `UpdateCulturePreview`
- ✓ Editor localizado en el diseñador recreado tras el cambio para que ninguna etiqueta se quede en el idioma anterior
- ✓ Verificación con el parámetro de URL `lang` y una nota sobre por qué la cultura no debe vivir en un campo estático

Preguntas de repaso

- ✓ Tu método `ApplyTextResources` actualiza el dashboard a la perfección, pero el editor de clientes conserva sus etiquetas en inglés tras cambiar al alemán. Explica qué tienen de distinto esas etiquetas y nombra dos estrategias que lo resuelvan.
- ✓ Un compañero propone guardar en caché el `CultureInfo` elegido en un campo estático "para no buscarlo dos veces". Describe qué vería en realidad un segundo usuario conectado desde otro país en ese mismo momento.
- ✓ La vista previa de la cultura muestra un importe monetario. Tras cambiar de `en-US` a `de-DE`, ¿qué ha cambiado en memoria y qué ha cambiado solo en pantalla, y cuál de las dos cosas escribirías alguna vez en la base de datos?



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.