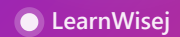


Rechts-nach-links-Sprachen und Lokalisierungssichere Layouts

Lektionsleitfaden · Lab- und Prüfungsleitfaden



Lektionsleitfaden

Unterstützung für Rechts-nach-links in Wisej.NET: RightToLeft mit Inherit, No und Yes, RightToLeftLayout, das die Kinder eines Containers spiegelt, die Anwendungseinstellung rightToLeft, Layouts, die eine Textausdehnung überstehen, der rightToLeft-Zustand des Themes und die dokumentierten Einschränkungen bei DataGridView, ListView und PropertyGrid. Was das für Entwicklerinnen und Entwickler bedeutet, die die Grundlagen von C# und die Wisej.NET-Controls kennen und die Anwendung nun in mehr als einer Sprache ausliefern müssen – und wie Sie dieses Modul lesen.



Fokus: Produktivbetrieb

Unterstützung für Rechts-nach-links in Wisej.NET: RightToLeft mit Inherit, No und Yes, RightToLeftLayout, das die Kinder eines Containers spiegelt, die Anwendungseinstellung rightToLeft, Layouts, die eine Textausdehnung überstehen, der rightToLeft-Zustand des Themes und die dokumentierten Einschränkungen bei DataGridView, ListView und PropertyGrid.

1. Das Problem im Produktivbetrieb

GlobalDesk kann die Sprache bereits im laufenden Betrieb wechseln: Die Auswahl setzt `Application.CurrentCulture`, der `Application.CultureChanged`-Handler wendet jede ressourcengestützte Beschriftung erneut an, und der Vorschaubereich formatiert seine Beispiele neu. Dann öffnet ein Tester das Dashboard auf Arabisch. Jedes Wort ist übersetzt und jedes einzelne steht am falschen Platz: Labels schmiegen sich an den linken Rand ihrer Felder, die Navigationsspalte sitzt dort, wo das Auge eines arabischen Lesers zuletzt ankommt, und die langen deutschen Beschriftungen aus dem vorigen Modul schieben schon jetzt eine Schaltfläche aus einem Panel mit fester Breite heraus.

Die Übersetzung war die leichte Hälfte. Ein lokalisierter Bildschirm hat auch eine Leserichtung und eine Breite, die er nicht kontrollieren kann, und beides sind Layout-Entscheidungen, die Sie beim Bauen des Formulars treffen, keine Reparaturen im Nachhinein.

2. Zentrale Konzepte

Konzept	Warum es wichtig ist
RightToLeft	Eine Control-Eigenschaft mit den Werten <code>Inherit</code> , <code>No</code> und <code>Yes</code> , die einem Control sagt, seinen eigenen Text und die Richtung seiner internen Elemente anzupassen.
RightToLeft.Inherit	Die normale Wahl für ein untergeordnetes Control: Es folgt seinem Elternteil, sodass der ganze Bildschirm die Richtung aus einer einzigen Entscheidung an der Spitze ändert.
RightToLeftLayout	Eine andere Eigenschaft, die bei aktivem RTL die waagerechte Position der direkten Kinder eines Containers spiegelt, sodass die visuelle Anordnung umklappt, während Ihr logisches Layout so bleibt, wie entworfen.
Die Anwendungseinstellung <code>rightToLeft</code>	Ein Eintrag in <code>Default.json</code> ; auf <code>"auto"</code> gesetzt lässt sie Wisej.NET das RTL-Verhalten aus der Kultur der Sitzung ableiten statt aus Kulturnamen-Prüfungen, die über Ihre Bildschirme verstreut sind.
<code>Application.RightToLeft</code>	Dieselbe Entscheidung im Code ausgedrückt, für Sitzungen, die bewusst umgeschaltet werden müssen statt über die Kulturerkennung.
Textausdehnung	Eine übersetzte Beschriftung hat selten die Länge der englischen, und deshalb brechen feste Pixelpositionen und von Hand ausgemessene Panelbreiten in einem lokalisierten Build als Erstes.
Dokumentierte RTL-Grenzen	Wisej.NET dokumentiert, dass <code>DataGridView</code> , <code>ListView</code> und <code>PropertyGrid</code> RTL nicht vollständig unterstützen, deshalb muss ein Testplan sie benennen, statt anzunehmen, dass sie spiegeln.

3. Der rote Faden der Lektion

Zwei Eigenschaften, zwei Aufgaben

Der erste Fehler, den fast alle machen, ist, Rechts-nach-links als Einstellung zur Textausrichtung zu behandeln. Arabisch, Hebräisch und die übrigen Rechts-nach-links-Sprachen brauchen, dass der Text in die andere Richtung läuft *und* dass üblicherweise auch die Anordnung des Bildschirms gespiegelt wird. Wisej.NET trennt diese beiden Aufgaben in zwei Eigenschaften, und zu wissen, welche Sie brauchen, ist der größte Teil dieses Moduls.

`RightToLeft` gehört zu einem einzelnen Control und verändert, was in ihm geschieht: die Richtung seines Textes und der Elemente, die es für sich selbst zeichnet. Es nimmt `Inherit`, `No` oder `Yes`. Lassen Sie es fast überall auf `Inherit`, denn ein Kind folgt dann seinem Elternteil, und das ganze GlobalDesk-Dashboard antwortet auf eine einzige Entscheidung an der Spitze. `Yes` auf einzelnen Controls zu setzen gibt Ihnen einen Bildschirm, auf dem die Richtung an fünfzig Stellen konfiguriert ist und niemand sagen kann, welche falsch ist.

`RightToLeftLayout` ist die andere Aufgabe. Es spiegelt bei aktivem RTL die waagerechte Position der direkten Kinder eines Containers, sodass Sie den Kundeneditor weiterhin in der Reihenfolge entwerfen, die

für Sie sinnvoll ist, während Wisej.NET die visuelle Anordnung umklappt. Beachten Sie die Worte „direkte Kinder“: Auf dem Dashboard gesetzt spiegelt es die Panels, die das Dashboard hält, und jedes Panel entscheidet selbst, ob seine eigenen Controls spiegeln. Deshalb setzt das Lab es auf dem Dashboard und auf dem Kundeneditor.

Lassen Sie die Anwendungseinstellung entscheiden

`Default.json` trägt eine Einstellung `rightToLeft` neben der Einstellung `culture`, die Sie im vorigen Modul konfiguriert haben. Mit dem Wert `"auto"` leitet Wisej.NET das RTL-Verhalten aus der Kultur ab, unter der die Sitzung läuft, sodass es genügt, Arabisch in die Auswahl aufzunehmen, damit die Richtung der Sprache folgt. `Application.RightToLeft` stellt dieselbe Entscheidung dem Code zur Verfügung, wenn eine Sitzung bewusst gesteuert werden muss.

Die Alternative ist das Muster, das zu vermeiden ist: eine Kulturnamen-Prüfung in jedem Bildschirm, dann eine zweite Liste von Präfixen, wenn Hebräisch dazukommt, dann ein Bildschirm, den jemand vergessen hat. Eine einzige Anwendungseinstellung ersetzt das alles, und deshalb können Ihre Controls auf `Inherit` bleiben. Die Richtung wird zu Sitzungszustand, genau wie `Application.CurrentCulture`: Ein Tester, der zu Arabisch wechselt, darf niemals den Bildschirm eines Kollegen spiegeln.

Ein Layout entwerfen, das sich spiegeln lässt

Spiegelung sieht nur dann richtig aus, wenn das Layout von Anfang an flexibel war. Ein Control an einer von Hand ausgemessenen `Location` in einem Panel fester Größe spiegelt in eine Position, die nie entworfen wurde, und eine Beschriftung, die auf Deutsch 40 % länger ist, läuft schlicht über ihren Rand hinaus. Die Werkzeuge sind die gewöhnlichen Layout-Werkzeuge von Wisej.NET: Docking, damit eine Region einen Rand besitzt, Anchoring, damit ein Control mit seinem Container wächst, `AutoSize`, wo ein Label so lang sein darf wie seine Übersetzung, ein Container-Layout, wo eine Schaltflächenreihe neu umbrechen muss, und sinnvolle Mindestgrößen, damit nichts zusammenfällt.

Es gibt eine bewusste Ausnahme, und das Lab bittet Sie, den Kommentar zu schreiben, der sie erklärt. Manche Zeichenfolgen sind kein Fließtext: ein Kundencode, eine Seriennummer, der Inhalt eines Code-Editors. Diese lesen sich in jeder Kultur von links nach rechts, und sie umzudrehen macht aus einem korrekten Bezeichner einen verwirrenden. Dafür ist `RightToLeft.No` da — sparsam eingesetzt und immer mit einer Begründung daneben.

```
private void ConfigureDirection()
{
    // Default.json carries rightToLeft "auto", so Application.RightToLeft
    // already follows the session culture. Children stay on Inherit.
    this.RightToLeft = RightToLeft.Inherit;
    this.RightToLeftLayout = true;

    // The customer code is a technical identifier, not prose:
    // it must keep reading left to right in every culture.
    txtCustomerCode.RightToLeft = RightToLeft.No;
}
```

Ausdehnung und gespiegelte Controls testen

Keines der beiden Probleme ist im Code sichtbar, deshalb endet das Modul in einer Testliste und nicht in einer Funktion. Führen Sie jeden lokalisierten Bildschirm durch mindestens fünf Fälle:

- Kurzer Text in der neutralen Sprache, also der Zustand, in dem alles entworfen wurde.
- Eine Sprache mit spürbar längeren Übersetzungen, etwa Deutsch.
- Ein schmales Browserfenster, wo sich Anchoring und Mindestgrößen bezahlt machen.
- Eine RTL-Kultur wie Arabisch, mit Blick auf die Anordnung und nicht nur auf die Ausrichtung.
- Gemischter Inhalt, etwa ein arabisches Label neben einem lateinischen Produktcode.

Eine pseudo-lokalisierte Ressourcendatei macht die Ausdehnungsfälle ehrlich. Kopieren Sie die neutralen Schlüssel und verlängern und klammern Sie dann jeden Wert, sodass eine Beschriftung etwa zu `[[Save changes ~~~~]]` wird. Alles, was noch als schlichtes Englisch zu lesen ist, war nie ein Ressourcenschlüssel, und alles, was abgeschnitten wird, sagt Ihnen, welches Control Anchoring braucht. Auch Wisej.NET-Themes kennen einen `rightToLeft`-Zustand, sodass Symbole und einseitige Gestaltung sich ohne ein zweites Theme anpassen können; eigene Theme-Regeln und eigene Widgets sind die Teile, die unter Umständen weiterhin von Hand geschriebene Spiegelung brauchen.

Die Controls mit dokumentierten Grenzen

Und schließlich der ehrliche Teil. Wisej.NET dokumentiert, dass `DataGridView`, `ListView` und `PropertyGrid` Rechts-nach-links nicht vollständig unterstützen. GlobalDesk zeigt seine Kontakte in einer `DataGridView`, deshalb bittet das Lab Sie, festzuhalten, was dieses Grid unter Arabisch tatsächlich tut, und nicht, was Sie erwartet haben. Spaltenreihenfolge, Ausrichtung der Kopfzeile und die Seite der Bildlaufleiste sind die Dinge, auf die Sie schauen.

Das ist kein Grund, diese Controls zu meiden; es ist ein Grund, sie im Testplan zu benennen und sie in genau der Version und Konfiguration zu prüfen, die Sie ausliefern. Ein lokalisiertes Layout sollte niemals eine übersetzte Beschriftung abschneiden, ein Control aus dem sichtbaren Bereich schieben, von englischer Wortlänge abhängen, einen Richtungspfeil ohne RTL-Variante in ein Bild einbrennen oder einen Bezeichner in die falsche Leserichtung zwingen — jedes davon ist auf dem ersten Bildschirm günstiger zu beheben als auf dem hundertsten.



Gewohnheit für die Lokalisierungsschichten

Entscheiden Sie, wo jede Zeichenfolge lebt, bevor Sie sie tippen. Designer-Ressourcen gehören zu einem entworfenen Bildschirm und werden beim Erstellen seiner Controls angewendet; gemeinsame Anwendungsressourcen in `.resx` tragen den Text, den Code erzeugt oder den mehrere Bildschirme wiederverwenden; `Resources-[LANG].resx` überschreibt die Systembeschriftungen von Wisej.NET. Speichern Sie niemals ein formatiertes Datum, eine formatierte Zahl oder einen formatierten Währungsbetrag: Behalten Sie den Wert und formatieren Sie ihn zum Anzeigzeitpunkt für die aktive Kultur. Und denken Sie daran, dass die Kultur pro Sitzung gilt, sodass die Sprache eines Benutzers niemals den Bildschirm eines anderen verändern darf.



Lernziele

Am Ende dieses Moduls können Sie:

- ✓ RightToLeft und RightToLeftLayout in Wisej.NET korrekt einsetzen.
- ✓ die Anwendungseinstellung rightToLeft="auto" erklären.
- ✓ ein Layout bauen, das für RTL-Kulturen angemessen spiegelt.
- ✓ Textausdehnung und gespiegelte Controls testen.
- ✓ Wisej-Controls benennen, deren RTL-Unterstützung dokumentierte Einschränkungen hat.

Lab- und Prüfungsleitfaden

Was Sie im praktischen Lab bauen, das vorgeschlagene Vorgehen und die geforderten Ergebnisse.



Praktisches Lab

Machen Sie GlobalDesk fit für eine Kultur, die von rechts nach links geschrieben wird. Nehmen Sie Arabisch in die Sprachauswahl auf und setzen Sie `rightToLeft` in `Default.json` auf `auto`, lassen Sie die untergeordneten Controls auf `RightToLeft.Inherit` und setzen Sie `RightToLeftLayout` auf dem Dashboard und dem Kundeneditor, damit deren Kinder gespiegelt werden. Lassen Sie genau ein Control von links nach rechts – das Feld für den Kundencode – und begründen Sie in einem Kommentar, warum dieser Bezeichner nicht umgedreht werden darf. Bauen Sie den fest positionierten Teil des Layouts mit Docking, Anchoring und einem Container-Layout neu auf, damit eine lange Übersetzung nicht abgeschnitten werden kann, fügen Sie eine pseudo-lokalisierte Ressourcendatei hinzu, deren Werte verlängert und in Klammern gesetzt sind, und halten Sie fest, was die Kontakt-DataGridView unter RTL tatsächlich tut, statt anzunehmen, dass sie spiegelt. Schließen Sie mit einer Layout-Testliste aus fünf Fällen ab, die Sie auf jedem Bildschirm erneut durchgehen können.

Ausgangspunkt

GlobalDesk nach Modul 4: Das Dashboard, der Kundeneditor und das Kontakt-Grid lesen alle ihre Beschriftungen aus Ressourcen, die Sprachauswahl bietet Englisch, Deutsch und eine Sprach-Regions-Kultur an, und ein `Application.CultureChanged`-Handler ruft bereits `ApplyTextResources` und `UpdateCulturePreview` auf. Jeder Bildschirm ist weiterhin von links nach rechts aufgebaut, und der Navigationsbereich des Dashboards verwendet noch die festen `Location`- und `Size`-Werte aus Modul 1.

Vorgeschlagenes Vorgehen

- ✓ Fügen Sie der Sprachauswahl einen Eintrag für Arabisch hinzu, setzen Sie `rightToLeft` neben der bestehenden Einstellung `culture` in `Default.json` auf `"auto"` und führen Sie die Anwendung dann aus und wechseln Sie zu Arabisch, bevor Sie irgendetwas anderes ändern, damit Sie genau sehen, was die automatische Erkennung von sich aus tut.
- ✓ Setzen Sie `RightToLeftLayout` auf dem Dashboard und auf dem Kundeneditor auf `true` und prüfen Sie, dass jedes untergeordnete Control weiterhin auf `RightToLeft.Inherit` steht — widerstehen Sie dem Drang, `Yes` auf einzelnen Controls zu setzen, um zu „reparieren“, was der Container entscheiden sollte.
- ✓ Setzen Sie `RightToLeft.No` ausschließlich auf dem Feld für den Kundencode und schreiben Sie den Kommentar, der sagt, warum ein technischer Bezeichner seine Leserichtung behalten muss, während der Fließtext ringsum umklappt.
- ✓ Ersetzen Sie die absoluten Positionen des Navigationsbereichs durch Docking und Anchoring, legen Sie die Schaltflächenreihe des Editors in ein Container-Layout, damit sie neu umbrechen kann, und geben Sie den Beschriftungen `AutoSize` und den Panels sinnvolle Mindestgrößen.
- ✓ Fügen Sie eine pseudo-lokalisierte Ressourcendatei hinzu, deren Werte verlängert und in Klammern gesetzt sind, gehen Sie die fünf Testfälle damit durch, notieren Sie, was die Kontakt-

`DataGridView` unter Arabisch wirklich tut, und machen Sie aus dem gesamten Durchgang eine wiederverwendbare Testliste.

Geforderte Ergebnisse

- ✓ Eintrag für Arabisch in der Sprachauswahl mit `rightToLeft` auf `auto` in `Default.json`
- ✓ Dashboard und Kundeneditor mit `RightToLeftLayout`, deren Kinder auf `Inherit` belassen sind
- ✓ Ein bewusst von links nach rechts laufendes Control mit einem Kommentar, der die Ausnahme erklärt
- ✓ Mit Docking, Anchoring und einem Container-Layout neu aufgebautes Layout, damit lange Übersetzungen nicht abgeschnitten werden können
- ✓ RTL-Testliste für kurzen Text, langen Text, ein schmales Fenster, eine RTL-Kultur und die Einschränkung der `DataGridView`

Wiederholungsfragen

- ✓ Ein Kollege setzt `RightToLeft.Yes` auf zwanzig Controls, und der arabische Bildschirm sieht richtig aus, der deutsche ist jedoch unverändert. Was hat er tatsächlich konfiguriert, und welche einzelne Eigenschaft hätte die Arbeit für den ganzen Container erledigt?
- ✓ Ihr pseudo-lokalisierter Durchgang zeigt drei Beschriftungen weiterhin auf schlichtem Englisch. Nichts wurde abgeschnitten und nichts hat eine Ausnahme ausgelöst. Was sagt Ihnen das über diese drei Beschriftungen, und in welchem früheren Modul klafft eine Lücke?
- ✓ Das Kontakt-Grid spiegelt nicht so, wie der Rest des Dashboards es tut. Welche Ihrer Möglichkeiten — es hinnehmen und dokumentieren, das Control ersetzen oder die Spiegelung von Hand schreiben — können Sie gegenüber einem Reviewer vertreten, und was müssten Sie prüfen, bevor Sie eine davon zusagen?



Gewohnheit für die Lokalisierungsschichten

Entscheiden Sie, wo jede Zeichenfolge lebt, bevor Sie sie tippen. Designer-Ressourcen gehören zu einem entworfenen Bildschirm und werden beim Erstellen seiner Controls angewendet; gemeinsame Anwendungsressourcen in `.resx` tragen den Text, den Code erzeugt oder den mehrere Bildschirme wiederverwenden; `Resources-[LANG].resx` überschreibt die Systembeschriftungen von Wisej.NET. Speichern Sie niemals ein formatiertes Datum, eine formatierte Zahl oder einen formatierten Währungsbetrag: Behalten Sie den Wert und formatieren Sie ihn zum Anzeigzeitpunkt für die aktive Kultur. Und denken Sie daran, dass die Kultur pro Sitzung gilt, sodass die Sprache eines Benutzers niemals den Bildschirm eines anderen verändern darf.