DevExpress Reports (ASP.NET, MVC, ASP.NET Core) v20.1
How to Use Azure Text Translator API to Localize Reports Video Script

Video URL: https://youtu.be/Zq4gkCl5eOA
Product URL: https://www.devexpress.com/subscriptions/reporting/
Free Trial URL: https://www.devexpress.com/Products/Try/
GitHub Sample: https://github.com/DevExpress-Examples/Reporting-Register-Azure-Cognitive-Translation-Service

DevExpress Reports – our royalty-free .NET reporting suite – ships with full support for all major .NET platforms including WinForms, WPF, ASP.NET, MVC, and ASP.NET Core. You can download your free 30-day trial by pointing your browser to devexpress.com/try.

This short video is a follow-up to our v20.1 Report Localization tutorial. A link to our first Localization-related video is listed in the description field below.

If you've watched our first Report Localization-video, you know that v20.1 – set to ship in May 2020 – allows you and your end-users to quickly translate individual reports for global distribution.

In this video, I'll describe a v20.1 feature I did not have time to document in my first video – Specifically, I'll describe how you can use Azure Text Translation and your own custom localization service within the DevExpress Web End-User Report Designer. This feature is currently available for ASP.NET, MVC, and ASP .NET Core.

A quick word about Azure Text Translation - With the Azure's Translation Service, you can easily perform real-time text translation with a simple REST API call. The Microsoft Translator API is a neural machine translation service that developers can easily integrate into their applications, websites, tools, or any solution requiring multi-language support such as website localization and more.

OK - Let's get started.

As you can see, I have a simple report displayed within the Web Report Designer. All DevExpress reports are initially set to use a Default language – both in the design view and in report preview. Our default language is English.

To begin localization, I'll open the Localization editor from the Web Report Designer's main menu. Next, I'll select the appropriate language from the Language drop down list. I'm going to translate this particular report to German. Note that all strings in my report are based on my default language - English. I can manually enter translations, or I can automatically translate text strings by clicking the Translate button.

If you do decide to translate automatically, you can preview changes and make modifications as needed.

Note that once you select a string in the Localization Editor, the corresponding DevExpress Report element is automatically selected within the Web Report Designer. This allows you to quickly modify individual report elements - From element location to element size. Changes you make to a report element will be preserved for your target language.

Translations made in this manner are automatically visible in both our Print Preview and when exporting a report to one of our many supported file formats – such as PDF.

You can implement the custom localization service described here as a frontend or backend solution and specify it as an endpoint. In the rest of this video, I'll explain how you can use Microsoft Azure Translator Text API as a custom translation service for the DevExpress End-User Report Designer.

First things first – I'll need to create a Translator Text Cognitive Service resource on the Azure portal. The link to this reource is in the description. Once created, I'll need to obtain an endpoint and a translator text subscription key for authentication purposes.

I can store the endpoint and authentication key in my ASP.NET Core application – within the application settings file called appsettings.json.

During the initialization of the Web End-User Report Designer, I'll call the `registerTranslateService` client-side method to register the service. Here I'll need to specify my translation service endpoint.

In this example, I simply used the ~Home/GetAzureServiceTranslate~ controller action as a URL. This action uses IAzureTranslationService service implementation – which was registered as a service in the dependency injection container. It gets Translator Text authentication and endpoint key.

This then calls the TranslationTextRequest method which receives the language identifier and text strings for translation. It also creates the necessary request to Azure's Translator Text service and obtains appropriate results.

By default, the DevExpress Localization Editor displays all available .NET locales in its Language dropdown. As you might imagine, Azure Text Translator does not support this entire language set. You can customize the language list by using the client-side `CustomizeLocalization` method. This is optional of course, but we highly recommend that you customize the list to avoid usability issues.

Ok – we're just about done with this tutorial. The final step is to run this and see what it looks like. I'll load up my report and activate the Web Report Designer's Localization Editor. Notice that the drop down is now limited to the set of languages I specified earlier. I'll select Spanish and initiate auto-translation. Voila – my simple report is now translated to Spanish. If a specific text string is not translated properly, you can easily modify it within the Localization Editor. Let me quickly modify Categorias de N W I N D to Categorias Northwinds and modify Category ID.

That's it – That's all there is to it. Time to see my translated report inside the print preview.

As you can see, when you combine real-time Azure Text Translation with DevExpress Reports, you don't need to manually translate text string. This will save time and avoid the need to generate separate resource files for localization.

I hope you've enjoyed this short tutorial on DevExpress Reports and its new Localization Editor with Azure Text Translation support. You can download the example used here via GitHub. Link is in the description.

As always, if you liked this video, please give us a thumbs up. If you have questions, please comment below. And please remember to subscribe to this channel for more great DevExpress training videos.