**Localized** content. Localization in NetSuite has a big significance in the way user interface is presented, and most importantly, how your PDFs are generated. By the end of this module, you should be able to understand how printouts are localized depending on several preferences settings, learn how label values are displayed in custom forms, and dictate what contents are printed out depending on the locale.

As a NetSuite user, you can pick your preferred language, but customers and vendors have their own preference too. This overrides the users during PDF generation. If there's no preferred language in the customer or vendor side, its subsidiaries' preferences will be used.

Now the moment the user's preferred language don't match with the customer, vendor, or subsidiary's language, the Print in Customer's Locale selection appears. And in this scenario, the user clicks the Print in Customer's Locale link while the user prefers English US but the customer prefers Spanish, so the printed document will be in Spanish. If ever the customer's language field is empty, which can be since it's not required, a PDF in French would be generated instead.

In the following slides, we'll see the impact of the label field values you specify inside the custom forms we have created so far. You may remember the @label suffix that you have used in printing the labels inside of the template. This illustration shows that we can use the same expression and still come up with different label field values, depending on the language.

In this example, assuming you have English US as preferred language, this will print Employee. If language is set to Spanish, it will print [SPANISH]. Finally, if it's in German, it will print [GERMAN].

The same mechanism can be applied to custom fields. We fetch the custom entity's field label by typing in record.entity.custentity\_sdr\_sup\_phone@label. In order to set the translation values for each language, go to the translation subtab and populate each line with the translated value.

In FreeMarker, we have a special field called .locale. This field gives away the current locale the user's using. We can use a .locale field inside of an if directive. Let's look at this example top to bottom.

The first line evaluates the .locale field and see if the locale's in English US, represented by en\_US. If it is not in English US, check if it is in Spanish, represented by es\_ES. And if all else fails, finally check if the locale is in French, represented by fr\_FR.

There are few localization behaviors that you should look out for whenever you're considering translation on your PDF document. If you're planning to introduce localization, this means you need to prepare all of your field label translation. Moreover, you'll need to account some other factors.

If you're replacing string values from field labels on the fly, say, record.shippingcost@label?replacecostfee, meaning we're replacing the word cost with the word fee. Once you print the Spanish locale, the label value will not be shipping cost anymore but [SPANISH] instead, and nothing will be replaced. So keep an eye out on that.

Now, different regions may also require different tax calculations. And this is where .locale FreeMarker field is useful for. You can present different calculations depending on the recipient's locale.

Finally, labels for calculated values inside the FreeMarker doesn't have translations. Say you computed for subtotal less discount, which is equal to subtotal minus discount. You cannot print any labels for that using the @label suffix. So if you want to display different label values depending on the locale, use the .locale field inside of the if directive to display different label field values for different languages.

In the following walkthrough, we'll print an expense report in different locales and observe how preferences affect the printout language. Moreover, we'll see how the label changes according to the user's preferred language. In this walkthrough, we're going to generate two versions of the PDF, one in English, second one in German language. And as we do that, we'll also see how it affects the overall layout in the labels inside of NetSuite, as well as the labels inside of the PDF.

Let's start by going first to the expense report, Inside of this expense report, let's print it out. And let's focus our attention to the last modified date. Remember that this field doesn't have any label field property inside of the SuiteScript Records Browser. So let's see how it behaves once we change the language.

Now we go to Set Preferences. And under the Localization section, select the German language, then click Save. And the first thing you'll notice is that all of the links inside of NetSuite has changed.

Let's go back to the expense report and refresh this page. And now the label for the transaction title of the expense report has updated. And inside of the PDF, we see the same transaction title alongside with the expenses total label. And we can see the same for [GERMAN] here for employee, for purpose field, and others.

And what's interesting here is that the last modified date alongside with the supervisor email label fields are still in English. And we can fix that by going back to the SuiteDreams Expense Report template. Click this link and scroll down to the SuiteDreams Expense Report template. And let's do a quick search from last modified string.

And in here, we see the last modified date, column label, as well as the last modified date FreeMarker interpolation here. And the first thing we need to do is to determine what locale the user is using. And we can determine that by using the FreeMarker .locale special variable.

Type it in, and hit Preview, and focus our eyes on the last modified date, column header field, and we see de\_DE. And if we save this template and go back to the expense report again, we'll confirm that this locale is really being used up. And there we go. We see the same de\_DE capitalized at the last modified date column header. Now let's capitalize on that. Use that as leverage in constructing our conditionals.

So open again the SuiteDreams Expense Report template. Last modified date, search for it, and let's build our conditionals by opening up with an if directive and .locale. Remember, we don't have to type in the dollar sign and the opening curly braces because we're in FreeMarker scope.

So type in the de\_DE. If locale is equal to de\_DE, then do this. So if the locale is equal to de\_DE, print this section. And let's use a translated value for the last modified date and click Preview.

There we go. We see the German translation for the last modified date, and let's click Save to save the template. Go back to the expense report again and click Print. And there we go. We can confirm that the condition is working.

Now let's change up to English. Let's go back to English International again and see what happens if we have that conditional and we change the language. And click Save again.

If I revert back on the English International language again and click Print, now we have a problem here. The table header tag wasn't printed, and that's why the posting period column got shifted to the left. And we can fix this by going back to the template and adding a new conditional for users who use the English International language.

Let's go back to forms, the template. Open the SuiteDreams Expense Report template again, and scroll further down, and look for the German translation of last modified date. And here we go. And add a new line and initiate an elseif conditional, elseif .locale equals to "en."

And "en" is the locale value for English International. Let's leverage that. Delete the automatically added elseif, we don't need that. And copy this section over here and just change it over to last modified date again.

Just do some alignment fixes, hit Save, then go back to the expense report. Click Print. And there you have it. The last modified date is then again present inside of the PDF when using English International.

So the main takeaway for this walkthrough is if you're going to add new fields into the template and it doesn't have any label or label translation, you can take the approach of using the if directive plus the FreeMarker .locale special field. And this marks the end of the walkthrough.

Now it's your turn. It is now time for you to complete your hands on exercises for this module. Please refer to your student guide to complete the required exercises.

This concludes the Localized Content module.