# Internationalization in Angular

Localization translates the internationalized application for consumption in different languages and locales. This step includes generating a version of our angular application in the target language with formatting applicable to the target region. The internalization is just a general approach.

The i18n and l10n –



First, we need to internationalize our application then localized it, this is a two-step process.

We also need to consider regionality along with language difference between the countries like same language English and currency differs between countries like US, UK, Australia and Canada. For this, we need to refer the Unicode CLDR Project which provides key building blocks to support the world’s language by representing two-letter language and two-letter locale code like en-US, which is generally referred by locale itself.

Changes in angular.json file – add i18n section and replace <id> with our selected locale id and then add the build configuration for our locale

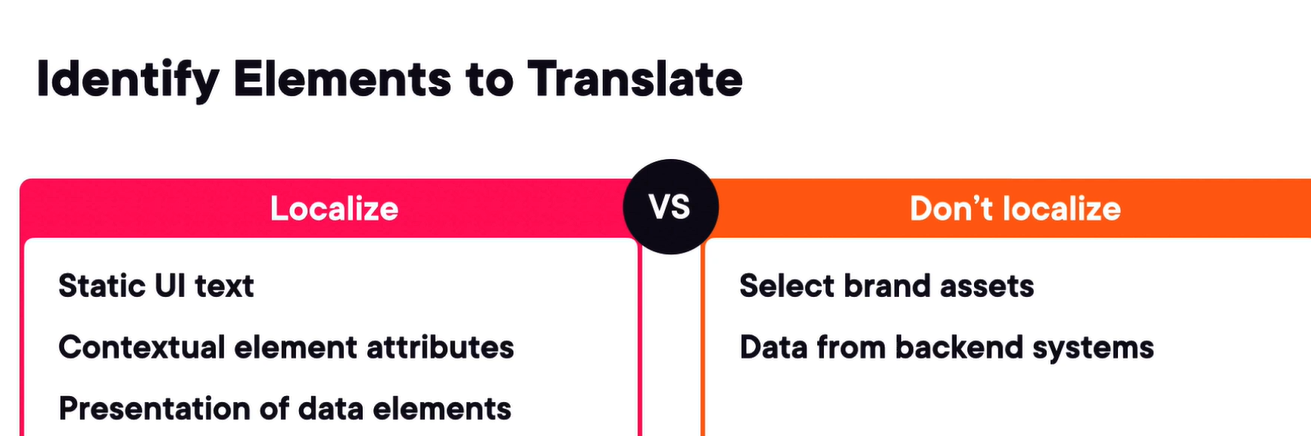


A screenshot of a computer

Description automatically generated

The localize library builds separate localized versions of the application at compile time, it does not perform localization at runtime. Angular does this for performance and maintenance reasons.

Identify elements to translate –



For each element which requires translation, we need to add i18n attribute –

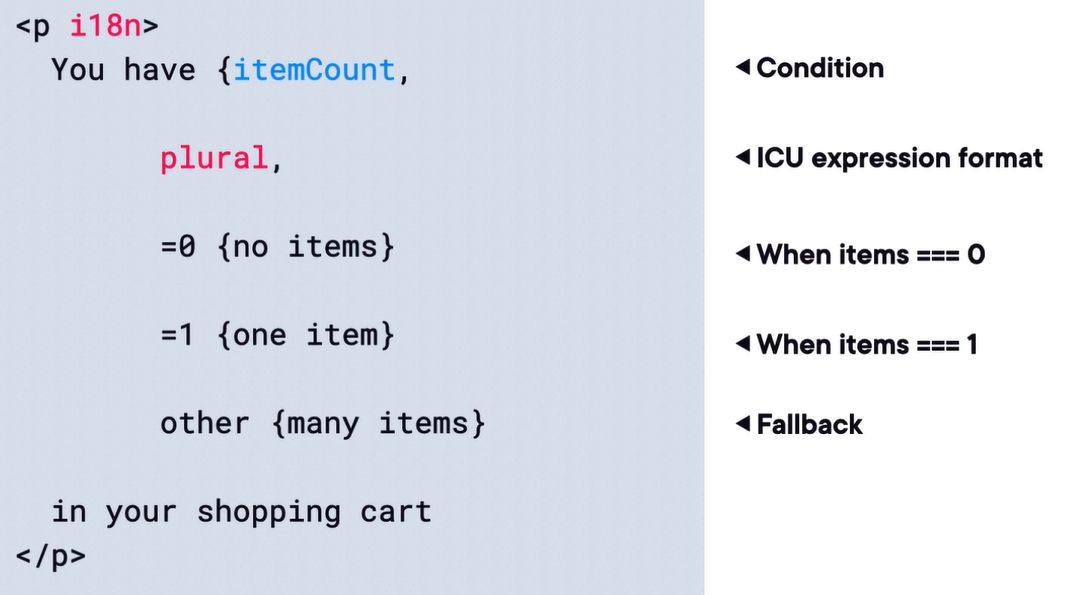
A white background with black and red text

Description automatically generated

A screenshot of a computer

Description automatically generated

Angular has a built-in support for plural and select ICU expressions which is a set of library (ICU) to support the world’s languages.



A screen shot of a computer

Description automatically generated

A blue circle with a blue text

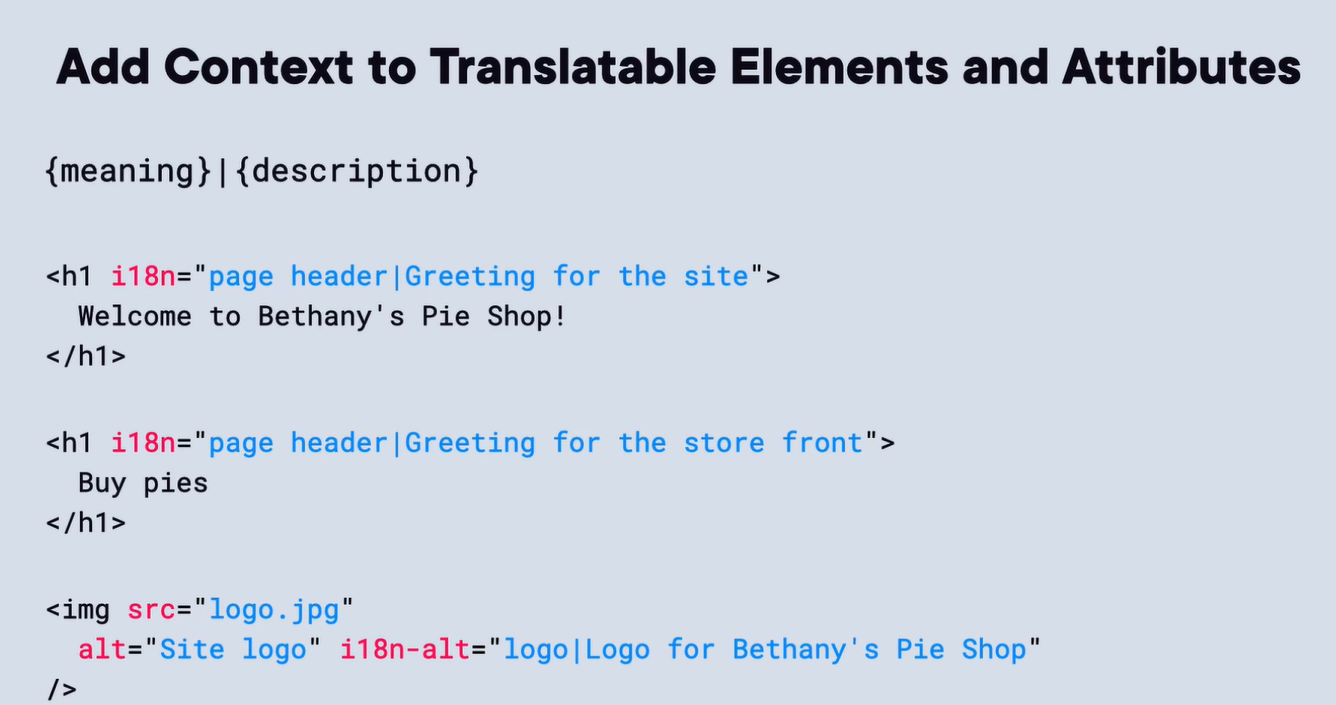
Description automatically generated

A screen shot of a computer

Description automatically generated

We can also perform the localization through code when code conditions necessitate examples like in case of dynamic menu options, table row headers, tricky cases using templates. We need to use the $localize API for this.

We can also provide additional context to the translatable elements and attributes –



Angular generates an unique id for each translatable element. And it generates a new id when the source text changes, which may affect updates. We can add custom Ids to translation elements using @@ –



Angular provide locale context under the LOCALE\_ID injection token. In one of the example, we can use it further to get the locale specific data.



Currency is different from locale, due to which we either need to provide the CURRENCY\_CODE or provide the DEFAULT\_CURRENCY\_CODE injection token.

A computer screen shot of a code

Description automatically generated