**5 Things Every Developer Should Know About Localization**

To support multiple languages, you will need to make a few adjustments to your product.

This is exactly what’s called internationalization (i18n). More specifically, it will involve planning and implementing products and services so that they can easily be adapted to specific local languages and cultures.

**Keep translatable content in external resource files**

Keeping all translatable content in external resources is usually the best plan for localization. These resource files are usually localizable xml, json, properties etc. After they are translated, place them into corresponding folders and you will be able to see how the translated version will look like.

However, the strings which are saved as .xml, .json or any other file format are usually lacking context, which is important for the translators to understand where the strings fit in your app. This can be done by providing screenshots images. The project manager in Text United can upload u each segment individually a reference screenshot and additionally, reference files which are available on a project level.

**Plan the UI ahead**

During the research stage, remember to plan the UI for localization. The general idea is that your software has to be polished in a localized version as in the ‘original’ one.

Remember that translated content can expand up to 30% in length for language combinations like English to French or English to German. In certain languages like Arabic and Hebrew, the text is read from right-to-left (RTL) requiring your entire design to be adjusted to the opposite side.

A modular design approach will come in handy while accommodating RTL languages. Did you consider these possible changes? Planning the UI ahead will save you a lot of time and money. Your UI simply has to be flexible and devoid of hard-coding, unless you want to double the workload (and probably double the money) after the product has already been translated.

**Watch out for line breaks and word wrapping**

It’s not easy when you localize your software for a country where they use the Latin alphabet, but it gets even a bit more complicated when you localize your software into East Asian languages. Why? Latin languages use spaces to separate words but languages such as Chinese, Korean and Japanese don’t. These are character-based languages that don’t use any spaces at all, and your application cannot rely on the usual line break and word wrapping algorithms for displaying text.

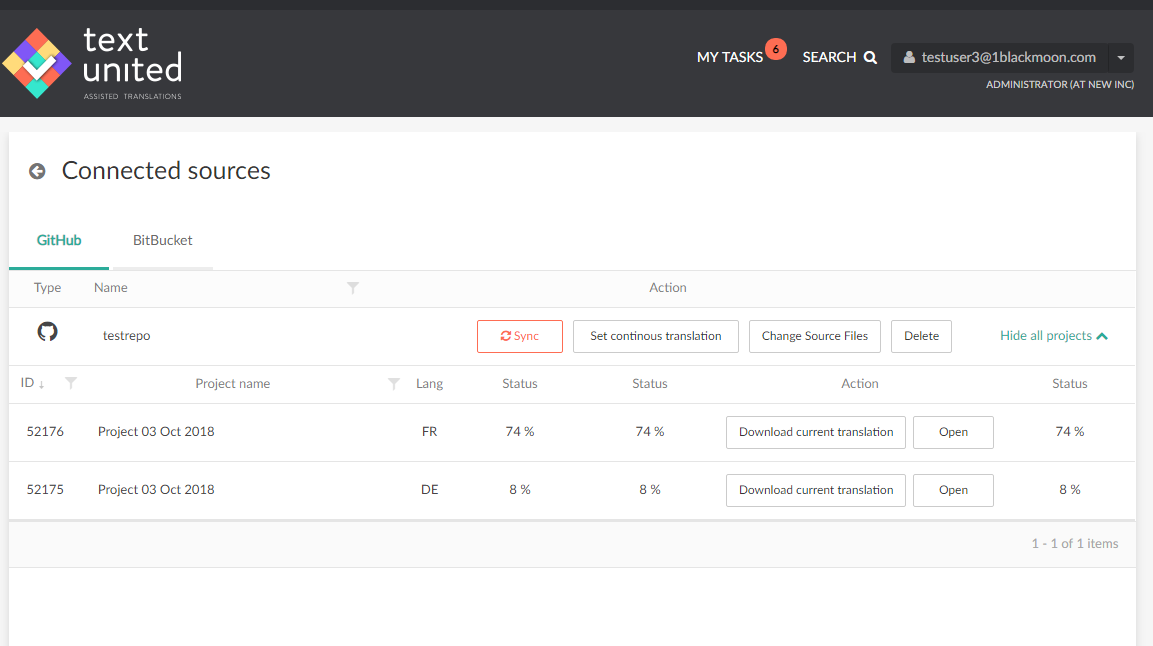
What does this mean? Well, a bit more work. Your UI will have to be adjusted specifically for these languages with the assistance of a linguistic expert. For example, text parsing for Japanese will require a specific Japanese word segmentation algorithm, which has to be highly accurate, as Japanese or Chinese words cannot simply be broken down where it seems convenient. That’s where your User Interface designer will need the help of a linguist. Which leads us to another point.

**Currency, date and time formats and measurings**

There are specific rules for text layout, formatting, measuring units etc. for each language. Date format is not the same in the UK, US and Germany for example. It’s highly recommended to use localization libraries that can help handling these aspects that include:

* Support for right to left scripts (Such as Arabic, Hebrew, Persian)
* Numeric and currency strings, date and time format based on the language
* Plurals (where the usage of plurals is predefined for each language)

**Use a TMS**



Don’t go with spreadsheets and emails for planning your localization project unless you want this project to be unproductive and to take forever. All of this manual work can nowadays be replaced by a Translation Management System (TMS). A proper TMS will have features that will allow you to manage, translate and monitor your translation project at any time.

How do I pick the right TMS? Again, research, research, research. But don’t worry, we will make it a bit easier for you and list some key elements that every TMS should have for software localization projects:

* API integration, for fast and efficient workflow
* Integrations with GitHub and BitBucket which allow for direct synchronization, versioning, and continuous translation.
* Translation Memory and Terminology Management
* Integrated machine translation
* Website translation
* Collaborative features (managing and communicating with your team through the platform)
* Context for files and strings