**Android 7.0 – Easier localization and handling multilingual apps**

Android 7.0 provides enhanced support for multilingual users. The new version of Android expands greatly the number of locales supported and changes the way the system resolves resources.

This tutorial will explain how to take advantage of the expanded number of locales to support more multilingual users.

Android 7.0 (API level 24) brings more robust resource resolution, and finds better fallbacks automatically. However, to speed up resolution and improve maintainability, you should store resources in the most common parent dialect. Prior to that the process looked like this:

***(blog\_table-03.png)***

The app’s default language is en\_US (US English) and lets assume it also has localized strings in fr\_FR (French, France)

The device is set to fr\_CH (Swiss French)

When your Java code refers to strings, the system would load strings from the default (en\_US) resource file, even if the app has French resources localized under fr\_CH. This is because when the system cannot find an exact match, it continues to look for resources by stripping the country code off the locale. Finally, if no match is found, the system falls back to the default, which is en\_US.

With Android 7.0 (API level 24) the user gets French resources instead of English. This example also shows why you should store French strings in fr rather than fr\_FR for Android 7.0 or higher. Here the course of action is to match the closest parent dialect, making resolution faster and more predictable.

***(blog\_table-04.png)***

**Localization tips for easier translation**

* Don’t hard code strings or string constants; Instead use the strings.xml files.
* Don’t hard code images or layouts; use R.drawable and R.layout making organisalion of multilingul content later on much easier
* Translate the strings.xml files and localize your images.
* Import your localized resources in the appropriate directories under ‘res/’.

**How to localize your App**

Building multilingual android apps is essential to reach users in different countries and locales. Text United already supports android .xml strings, it will filter out your textual content and professional translators will translate it without touching any of your code. It also integrates and syncs with your GitHub and BitBucket repositories seamlessly.

To start building a multilingual App you need to collect the content from your app into resource files (.xml)

*(folder tree illustration)*

Once you provide the translation, the Android OS will choose the resources that match the user’s locale – as described above. If your application has several languages available – the system will select the language that the user’s device is using.

The content of your app is loaded in your project’s “res” folder. Additionally, Android can load resources from different folders based of the device configuration and locale. As described above android will choose the correct value for that string at runtime by loading the appropriate strings.xml file from a matching “res/values” directory.

Translate your resource files and create new folder for res (the original values folder already exists) so you create new folders appropriately to the languages you translated you content to. For example:

*(folder tree illustration)*

The users default language acts as a preferred option when a translation exists, it’s used instead. If the user is located in Switzerland, and the phone language is set to French, android will first look for a folder called fr-rCH, if it can’t find it, it looks for a folder called values-fr and displays the appropriate language according to the user’s device.