Aptoide 2017 Summer Internship

Weekly report 5: Aug 07 - Aug 11

Project: Translations on demand
Intern: Válter Santos
Coordinator: Ana Lara Simões
Co-Coordinator: Diogo Loureiro

Completed tasks:

- Fixed preferences and odd case when accessing the account login;

- Attempted to create custom menu implementation but the android framework isn't very friendly towards that, so I stopped investigating that for the moment;

- Found and implemented a solution by using a factory in LayoutInflater, which allows views to stop needing a custom view class. What this means is now the views will be automatically translated without requiring to write the custom class in the xml.

Therefore, the overhead of the solution went initially from requiring to programmatically modify the view's text to no longer have an overhead on the java code, but have an overhead in the xml code (presentation) and now for layouts (which is more than 90% of the case) there's no overhead as well.

However, this only works for LayoutInflaters. So, menus (MenuInflater) will still need to be manually modified (programmatically) and preferences will need the custom class implementations. There's only one preferences xml file and a very small number of menus, so the overhead is minimal.

- The previous solution was found during the attempt to create another solution, the perfect solution, which requires access to a hidden, final, and not public class from Android. However, I found out Friday that I can use reflection to access this class and therefore modify the origin point of every view.

This class is called XmlBlock and it's what contains the binary xml code for layouts, menus and preferences. This class receives as an argument a byte array which is the binary xml. What my solution will do is intercept the byte array and modify the string references to string translations and then send the modified byte array to my XmlBlock obtained through reflection, then sending it to whatever requests this. This method will be on the custom resources, which means the total overhead of this solution would be zero except for overring the method getResources() from v8engine application and the uppermost activity class of the activity hierarchy.

This solution will almost surely work as views are created through LayoutInflater.inflate(), which then calls resources.getLayout; MenuInflater.inflate() which also calls resources.getLayout() and Preferences calls resources.getXML(), which does the same thing as getLayout(). There's also the possible bonus that strings in styles will also be translated, which would be hard to do otherwise (I haven't tested this).

The big advantages of this solution are that any new views, menus, preferences or anything that requires to access a xml view file would be automatically translated, requiring no attention to hook the new views/menus/preferences to the translations.

At the moment, I've managed to successfully alter the string references in the binary data to other strings present in the same xml at the moment of compile. What's left is to add the new translated strings to the binary xml and this will hopefully work.

Next week:

- Finish the reflection solution
- Check if it's possible to add new translations in runtime.