

Haiku locale kit overview

Adrien Destugues

BeGeistert 021

17 octobre 2009



Overview

- 1 Concepts and ideas
- 2 Inner workings
- 3 How to use it in your app
- 4 Work still needed



Makes the system more friendly

- Texts in your natural language
- Date, time, number and currency formatting and parsing
- Conventions : timezones, and other things (colors, sounds, ...)



- ICU as a backend for all formatting and parsing work, with an API wrapper
- Custom code for handling text catalogs

Example



Notes

- Define a context for each part of your app
- Enclose every string you want to use in a call to the TR() macro
- Don't forget to initialize the catalog at the start of your application and destroy it at the end.



Building

- Build your application as usual, linking to liblocale.so.
- Extract the string to translate from the file using the "collectcatkeys" tools (on a preprocessed C++ sourcefile)
- Translate the catkeys file to any language you need
- Link your catalogs into a binary format (flatenned BMessage) using linkcatkeys



Special cases

- Localizing an add-on needs to load a catalog by hand. You can't use the app one !
- Localizing a script can be done in two ways, either call to the (to-be-written) localize application or built-in support in your script language interpreter.
- Static strings can't be translated in place at compile time, so they need special handling (using TR_MARK)

In any of these cases, you can't use the TR macros directly anymore.



- More testing, most notably on the date, time, number formatting part
- Font overlay for japanese characters (and probably others)
- Converting applications to use the layout kit and localizing them
- Load gettext and zeta catalogs files and convert them
- Providing and improving catalogs for your own language