Inside i18n

- Technical background of i18n for software and Debian packages -

Kenshi Muto
<kmuto@debian.org>

Fingerprint:
360B B879 6957 5D6F EED3 D986 40A5 BEED 72D0 3CB1

l10n/i18n talk session with Christian Perrier, debconf4 at POA.

Agenda

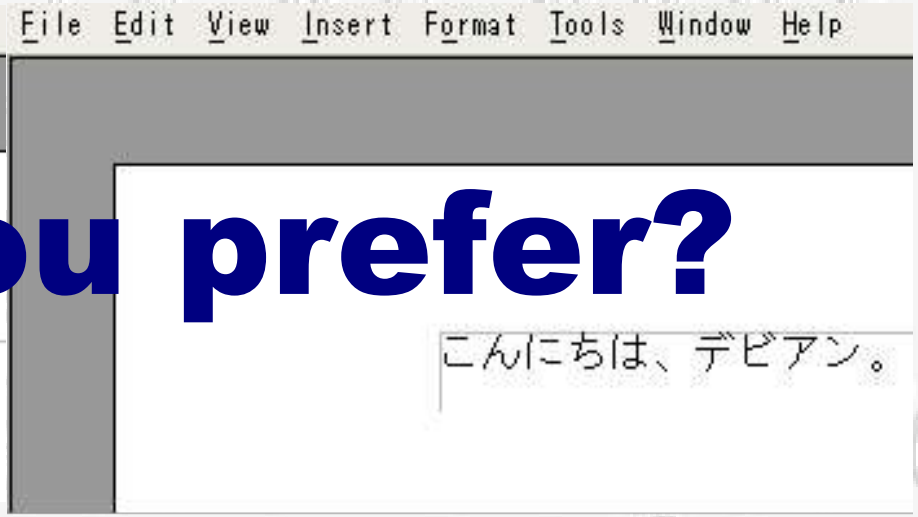
- Following Christian's talk,
- How to make your packages better about internationalization and localization
- Abbreviations
  - **i18n** : Internationalization
    18characters
  - **l10n** : Localization
    10characters

Imagine!

There are TWO Word-Processors here...

A.

B.



Which do you prefer?

Menu is localized,
Content is NOT localized

Menu is NOT localized,
Content is localized

Needs to understand somewhat...

- To understand a basic knowledge about i18n/l10n

Language

- Many many languages exist in the world
- Latin (English, French, German, ...)
- BIDI (Arabic, Hebrew)
- CJK (Chinese, Japanese, Korean)
- More (Thai, Hindi, ...)

Welcome

ما هو ديبيان

歡迎光臨

데비안이란

こんにちは、ブラジル

Character set and Encoding

```
A = 0x41 (ASCII, ISO8859-1)
€ = 0xa4 (ISO8859-15)

あ = 0xa4 0xa2 (EUC-JP)
     0x82 0xa0 (Shift_JIS)
```

- Character set
  - set of acceptable characters for each languages (ASCII, JISX0208, ...)
- Encoding (Encoded character set)
  - map character set and ID number for computer
- Many encodings...
  - ISO-8859-1 (Latin-1), ISO-8859-15 (Euro), ISO-2022-JP, EUC-JP, ShiftJIS, Big5, GB2312, KOI8-R, ...

Mojibake

- What's “mojibake”? "文字化け"
  - Broken screen (we can't read characters)
  - It's far from English developpers, but we (Japanese) meet very often
- Why?
  - Mismatch encoding or mismatch font
  - Screen problem
  - Bad toolkit or bad design

How to make your package more l10n and i18n?

gettext

msgid "Debian installer main menu" (message ID)

msgstr "Debianインストーラメインメニュー" (Japanese)

msgstr "Menu principal de l'installateur Debian" (French)

msgstr "Menu principal do instalador Debian" (Brazil Portuguese)

- Message catalog database
  - Switch messages using LANG environment variable
  - Relates msgid and localized message
  - many packages use this for l10n (such as debconf messages)
  - core architecutre of debian-installer l10n
- Many bindings
  - C, shell, Perl, Python, Ruby, Java, ...

gettext

- Misunderstood implementation...
  - Don't use non ASCII characters in msgid
  - Use `s(n)printf` and `%s` in msgid for dynamic variable
  - Does the msgid really need to make l10n?
- Gettext isn't the “Silver bullet”...
  - Remember Word Processor question

Toolkit library

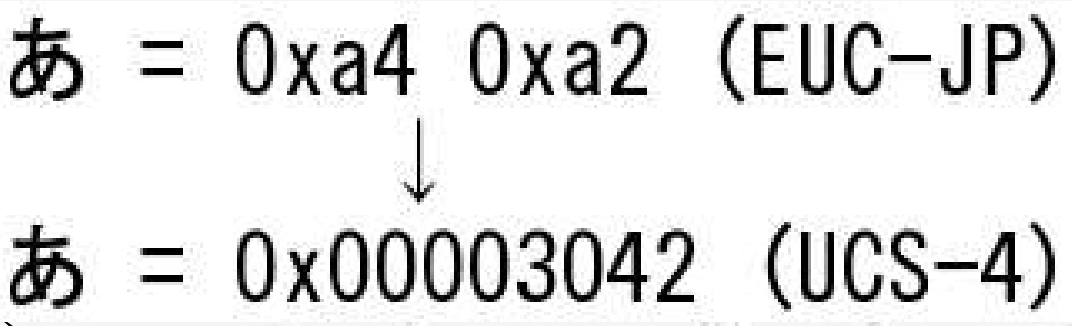
- Use i18n ready libraries
- GTK+ (2.0 is better)
- Qt
- Don't set specific font as default
  - XLFD: fixed (?)
    - FreeType: serif, sans-serif, or monospace
    - Configurable is better

Toolkit library

- Input method problem
  - How to input your local characters?
- Application on terminal
  - Depends on terminal software
- Application on X Window System
  - Only a few of modern Toolkit can handle
  - Immodule, XIM

Use internal encoding

- Before treating a string, unify its encoding
  - Use character unit instead of byte unit
- UCS-4
  - 32bit Unicode
  - GNU libc (iconv(3))
- Wide character (wchar_t)
  - C, C++
  - mbstocws(3)



Use internal encoding

- Respect user's locale setting. User's locale (encoding) can be gotten by:
  - locale charmap
    - (returns such as ANSI_X3.4-1968, EUC-JP)
    - Use 'LANG=C' or 'LC_ALL=C' if you want to ignore locale setting
- Applications should provide choice for user:
  - Input/Output file encoding

Conclusion

- There are many languages in the world! Wrong implementation causes “Mojibake”.
- Modern Toolkit for X application is recommended.
- Use internal encoding, such as UCS-4 or Wide character.
- Let's try to make your application l10n/i18n ready! More l10n/i18n makes users more happy. :-)

Happy Hacking!