Getting Your Language In

LibreOffice goes BCP 47
– or – Language Tags and all that it entails

Eike Rathke (erAck) – Red Hat, Inc.

Agenda

- About the Speaker
- Locales in LibreOffice
- The Obstacles
- Seeing Light
- The Solution
- Results
- More TODO
- Q&A

About the Speaker

- Eike Rathke, known on the net as erAck
- Based in Hamburg, Germany
- Worked on StarOffice from 1993 to 2000 for Star Division
- Worked on OpenOffice.org from 2000 to 2011 for Sun Microsystems and one other company
- Works on LibreOffice since 2011, employed by Red Hat, Inc.
- Areas of expertise:
  - Calc core, formula compiler and interpreter
  - number formatter/scanner
  - i18n framework, locale data
- Also mentor and knowledge spreader whenever possible

Locales in LibreOffice

A short overview.

Locales in LibreOffice

- Until LO 4.1 known locales use only ISO 639-2 or 639-3 alpha language codes and ISO 3166 alpha country codes
  - For example, *en-US* or *de-DE* or *pjt-AU*
    - (Pitjantjatjara in Australia in case you didn't know)
- UNO API uses the com::sun::star::lang::Locale struct
  - Designed after the Java Locale
  - 3 struct members / fields
    - Language
      - Restricted to ISO 639 alpha codes
    - Country
      - Restricted to ISO 3166 alpha codes
    - Variant
      - “free-form” field, platform and application specific

The Obstacles

Something is in the way to that mountain tribe.

The Obstacles

- Application modules locale-wise didn't know about anything else than language and country and were not prepared to encounter anything else
- No ISO 15924 script codes
    - For example, Serbian in Latin script or Cyrillic script is not distinguishable by only language and country codes, the LibreOffice hack was to use *sr* for Cyrillic and *sh* for Latin, with the deprecated *sh* code being abused
  - Undefined how to store a script code in a Locale struct
  - OpenDocument Format (ODF) 1.0/1.1 only defined *fo:language* and *fo:country* to be used
    - Luckily ODF 1.2 additionally defines *fo:script*
- Unix Locale Identifier, yet another free-form *@modifier*
    - For example, *be_BY@latin* or *ca_ES@valencia*

The Obstacles

- No languages for that no ISO 639 code exists
  - For example, there is Catalan Valencian, for the UI locale the hack *ca-XV* was used with *XV* being an ISO 3166 reserved for private use country code, those private use codes must not be stored in ODF's *fo:country*
- No dialects or variants, ISO 639 defines only languages
- No specifics like old and new grammar of one language
- Luckily ODF 1.2 defines **:rfc-language-tag* (AHAAA!) attributes that can be used in these cases
- The API's com::sun::star::lang::Locale can not be changed or replaced, otherwise almost all existing extensions would cease to work
- Additionally, LibreOffice must know about and in core heavily uses the MS locale identifier, or *LangID*, a 16-bit value

Seeing Light

Nice guys bringing sunshine.

Seeing Light

- BCP 47, Best Current Praxis
  - Currently RFC 5646, Tags for the Identification of Languages (previously RFC 4646, 3066, 1766)
  - For pointers see http://www.langtag.net/ and my selection at http://erack.de/bookmarks/D.html#Language_Tags
- language ["-" script] ["-" region] *("-" variant) *("-" extension) ["-" privateuse]
  - For example *ca-valencia* for Catalan Valencian
    - Where *ca* is language and *valencia* is variant
- Tags are registered with IANA (Internet Assigned Numbers Authority)
    - Reuse of codes/tags for different languages or countries will not happen, like it was the case with ISO 3166 *CS*, first Czechoslovakia then Serbia and Montenegro

Seeing Light

- New attributes will be possible, for example specify that a document uses
  - “German in the traditional spelling/orthography before 1996”
    - *de-DE-1901*
  - “British English with Oxford English Dictionary spelling”
    - *en-GB-oed*
    - important because it is mandatory for UN documents

The Solution

Throw away, replace and polish.

The Solution

- Use *liblangtag* to parse, validate and canonicalize language tags if they are not of simple cases already known to the application
  - http://tagoh.bitbucket.org/liblangtag/
  - Will be included in upcoming or is already in latest Debian and Fedora releases
  - Validation: language alpha code and subtags must be registered with IANA
  - Canonicalization: transform a valid language tag to the most concise form, for example *de-Latn-DE* becomes *de-DE* because Latin is the default script for German

The Solution

- Define a convention to transport language tags in com::sun::star::lang::Locale
  - If a locale can be expressed as ISO language and country codes only, use those in Language and Country fields
    - Maintains compatibility with currently used locales
  - Else, if a locale contains a script code or needs to be expressed as language tag, set the Language field to the *qlt* ISO 639-3 reserved for private use code and set the Variant field to the full language tag
    - The Country field may contain the corresponding ISO 3166 alpha country code, if any, or otherwise must be empty (if applicable, language tag contains the region subtag)

The Solution

- Create a central service to accept, store, convert between and obtain
  - Language tags
  - Locales
  - MS LangIDs
  - Unix Locale Identifiers
  - Single tags like language, script, country, …
  - Information how a locale can be expressed for ODF
- Get rid of all handcrafted places that extract or assemble such information, usually only handling language and country because nothing else was known at that time
  - Replace with the central service

Results for LibreOffice 4.0

In the middle of everywhere.

Results for LibreOffice 4.0

- Introduced class LanguageTag
  - Now (master and 4.2) include/i18nlangtag/languagetag.hxx, i18nlangtag/source/languagetag/*
  - Single central place that uses *liblangtag*, encapsulated
  - Internally uses some of the old but now adapted MsLangId::convert...() methods for known locales for quick and easy implementation
- Replaced all uses of MsLangId::convert...() methods throughout the entire code base with instantiation of and calls to LanguageTag
- Removed most MsLangId::convert...() methods and made remaining methods used by LanguageTag private to prevent further access by application modules

Results for LibreOffice 4.0, continued

- Consolidated various different uses of an empty locale meaning, depending on context
  - System locale
  - Absence of language
  - Undetermined language or all languages
- Empty locale or language tag now means system locale, except in a linguistic service to obtain all available writing aids, for API stability
- Absence of language information consistently expressed as *zxx* ISO 639 special code
- Undetermined language consistently expressed as *und* ISO 639 special code

More Results, LibreOffice 4.2

Completing a milestone.

More Results, LibreOffice 4.2

- Introduced the LanguageTag handling to low level code interfacing with *rtl_Locale*
- Prepared the i18n framework for language tags
  - Special care needed for places where it interfaces with ICU
    - Conversion from language tag to ICU locale when necessary
  - Introduced *qlt* to locale data's <LC_INFO><Language> element
  - Made the locale data's <LC_INFO><Variant> element hold the BCP 47 language tag string

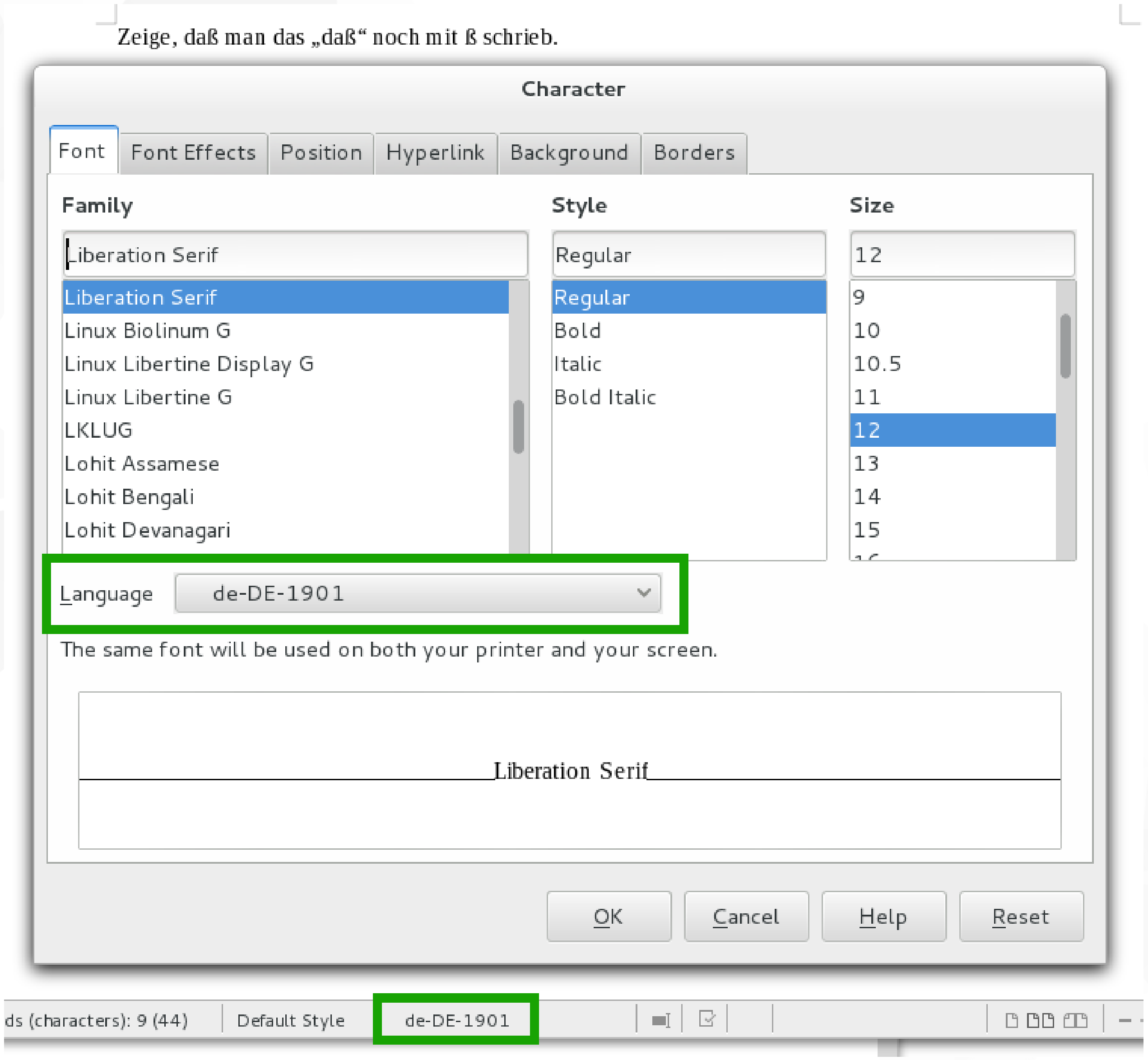
More Results, LibreOffice 4.2, continued

- Implemented reading and writing ODF with fo:script and *:rfc-language-tag when necessary
- Prepared writing aids (spell checker, thesaurus) to use language tags
- Reworked the current known locales to be able to introduce script and/or full language tags
- Implemented proper handling of existing workarounds
  - For example, *ca-XV* is now *ca-ES-valencia* (full locale, or *ca-valencia* for language only) and old *sh-RS* is *sr-Latn-RS*
  - With fallbacks for old *sh-** tags to be able to read existing documents

More Results, LibreOffice 4.2, continued

- Implemented an internal language tag registration to be able to generate on-the-fly MS-LangIDs in chunks of the reserved for user-defined IDs areas
  - Reading a document that contains a valid but unmapped language tag generates such LangID on the fly and for selected text the language tag is displayed in the status bar and language listbox
    - Such unmapped tags are preserved and written to the ODF file again when saving the document
    - See also http://erack.org/blog/archives/30-LibreOffice-goes-BCP-47.html

Temporary Language Tag Read From File



Zeige, daß man das „daß“ noch mit ß schrieb.

Character

Font | Font Effects | Position | Hyperlink | Background | Borders

Family: Liberation Serif

Style: Regular

Size: 12

Liberation Serif
Linux Biolinum G
Linux Libertine Display G
Linux Libertine G
LKLUG
Lohit Assamese
Lohit Bengali
Lohit Devanagari

Regular
Bold
Italic
Bold Italic

9
10
10.5
11
12
13
14
15

Language: de-DE-1901

The same font will be used on both your printer and your screen.

Liberation Serif

OK | Cancel | Help | Reset

ds (characters): 9 (44) | Default Style | de-DE-1901

File Changes to Permanently Add New Languages

- include/i18nlangtag/lang.h
  - MS-LangID, either defined by Microsoft or new ID in the user-defined ranges
- i18nlangtag/source/isolang/isolang.cxx
  - Mapping between MS-LangID and language tag
- svtools/source/misc/langtab.src
  - Entry for the language listbox
- i18nlangtag/source/isolang/mslangid.cxx
  - MsLangId::isRightToLeft() if RTL or MsLangId::getScriptType() if CTL language
- solenv/inc/langlist.mk
  - Build entry once translation is ready to be integrated

File Changes to Permanently Add New Languages

- Full description at https://wiki.documentfoundation.org/LibreOffice_Localization_Guide/Adding_a_New_Language_or_Locale
  - or in the wiki search enter “adding language”

Still more TODO

Maybe even for you?

Still more TODO

- Introduce the LanguageTag handling to low level code interfacing with calls to *setlocale()*
  - Implement support for constructing Unix locales
    - *liblangtag* knows to parse glibc locales
      - Needs some enhancement to not just obtain from LC_CTYPE but also from LC_MESSAGES or others
    - Best also have it construct those locale strings then
    - Above needs changes to *liblangtag*
- Display descriptions for on-the-fly tags read from documents
  - Possibly using the IANA registration's *Description* field
  - Would need translation of descriptions

Still more TODO, continued

- Make adding new languages for spell-checking purposes not yet handled by LibreOffice easier
  - Could be done with configuration entries
  - Installing spell checker / dictionary extensions providing support for a new language tag could configure that new tag

Still more TODO, for you?

- There's room for improvement in various applications that don't use language tags yet but could, such as
  - Other ODF 1.2 generators and consumers, implement the *fo:script* and **:rfc-language-tag* attributes
  - The locale data generator at it46.se (let's talk)
  - LibreOffice extension developers when evaluating a Locale's content should prepare for the *qlt* code in the Language field and if present handle language tags in the Variant field
  - *fontconfig*, use language tag instead of language-territory
  - … your application?

Questions?

I hope to be able to answer.

Thank you …

- … for using LibreOffice!
- … for supporting LibreOffice!
- … for hacking LibreOffice!