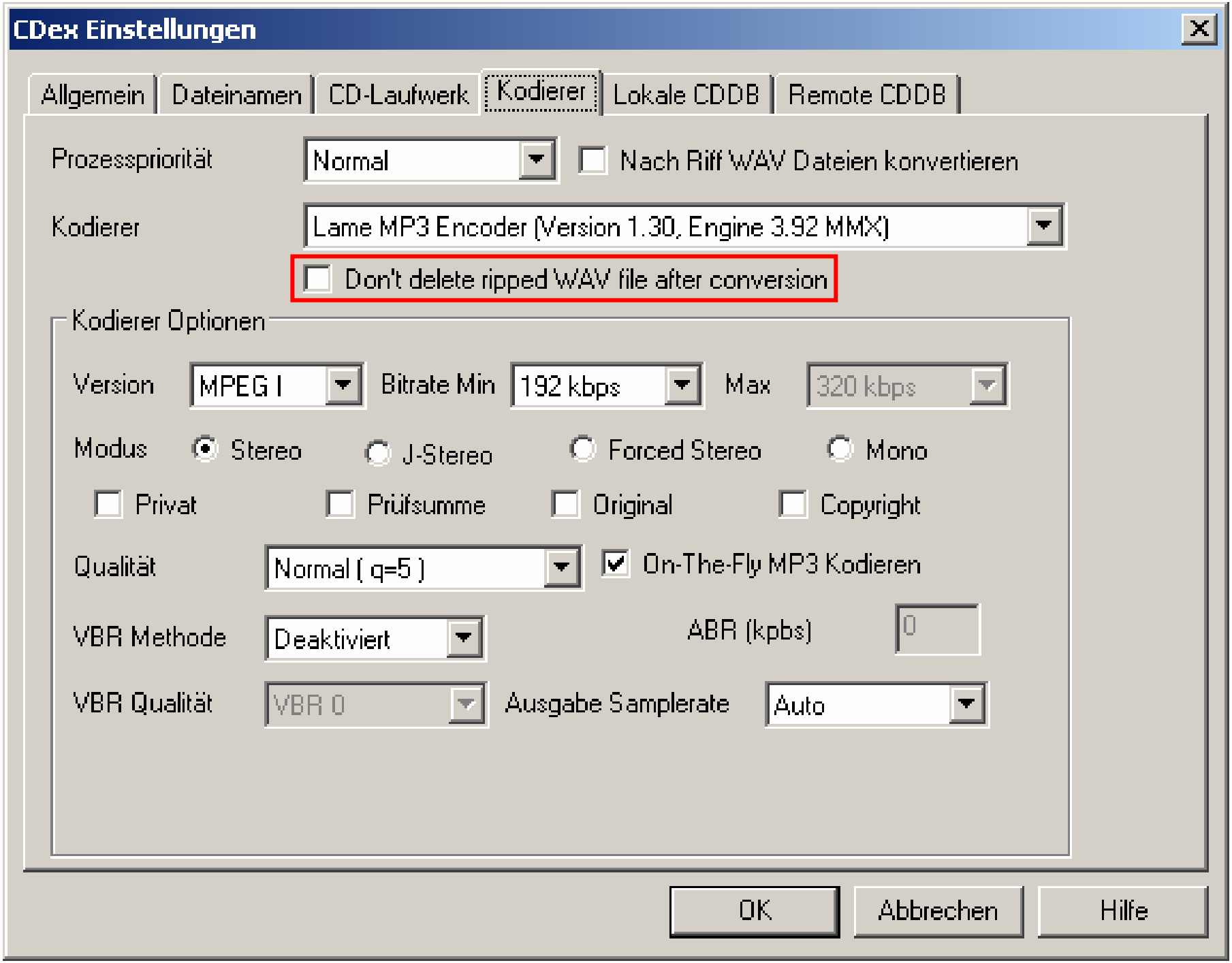
Internationalisierung in Open-Source-Projekten

Tobias Vogel


CDex Einstellungen
Allgemein
Dateinamen
CD-Laufwerk
Kodierer
Lokale CDDB
Remote CDDB
Prozesspriorität
Normal
Nach Riff WAV Dateien konvertieren
Kodierer
Lame MP3 Encoder (Version 1.30, Engine 3.92 MMX)
Don't delete ripped WAV file after conversion
Kodierer Optionen
Version
MPEG I
Bitrate Min
192 kbps
Max
320 kbps
Modus
Stereo
J-Stereo
Forced Stereo
Mono
Privat
Prüfsumme
Original
Copyright
Qualität
Normal (q=5)
On-The-Fly MP3 Kodieren
VBR Methode
Deaktiviert
ABR (kpbs)
0
VBR Qualität
VBR 0
Ausgabe Samplerate
Auto
OK
Abbrechen
Hilfe

Agenda

Introduction

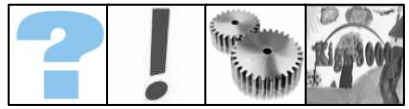
Concepts

Possible implementations

- native approach
- Java resource bundles
- gettext

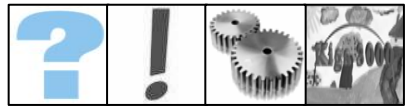
KiGa 3000

Introduction



- **What is internationalisation?**
  - Internationalisation is a technique of constructing a software product in such a way, that its presentation language can be customized afterwards easily.
  - Localisation is the process of actually adjusting an internationalised product for a given language.
  - A local is a collection of cultural dependent rules.

Introduction

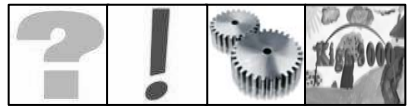


- **You notice that you need internationalisation when you use culturally dependent data, such as:**
  - messages/help pages
  - GUI component labels
  - numbers/currencies/time stamps
  - sounds
  - phone numbers
  - addresses
  - ...
- **and also if you want to**
  - compare strings
  - refer to the ASCII representation of a character

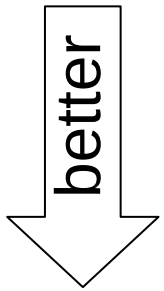
Concepts

- **global aim: separation of strings and code**
- **Mostly strings (or locales) are stored in external (text) files.**
  e.g. myProject_de_AT_VIENNA.lang
- **3 levels of identification**
  1. Language (ISO 3166)
  2. Country (ISO 639)
  3. Variant (user defined)
- **Locales are resolved hierarchically.**
  (localV, localC, localL, defaultV, defaultC, defaultL, default)

Possible implementations

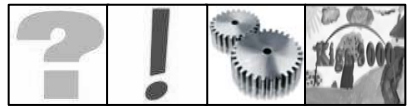


- **Challenges**
  - changeability outside the code
  - working also for languages without capability to include code at run time
  - ease of use when internationalizing later
  - compound messages
  - plural forms
- **3 different approaches**
  - native approach
  - Java resource bundles
  - gettext



- **Running example**
  - Christmas Notifier

Possible implementations ▸ native approach

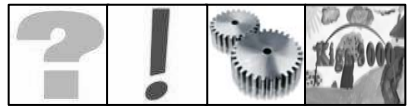


- **example code from *Christmas Notifier.php***

```
<?php
echo 'Merry Christmas';
echo 'and a happy new year!';
?>
```

- **challenges**
  - internationalisation of this page
  - available languages (localisations): English and German

Possible implementations ▸ native approach



- **first: externalize strings**

```
<?php echo 'Merry Christmas';
      echo 'and a happy new year!'; ?>
```

- **into file *CNLocales_en.php***

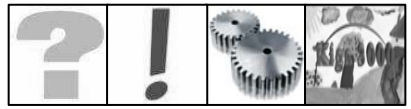
```
<?php $xmas = 'Merry Christmas';
      $neYe = 'and a happy new year!'; ?>
```

- **second: alter *Christmas Notifier.php***

```
<?php
include('CNLocales_' . $language . '.php);
echo $xmas;
echo $neYe;
?>
```

- **problem: no include-statement ⇒ no including**
- **problem: no compound messages (e.g. string + value + string)**

Possible implementations ▸ Java



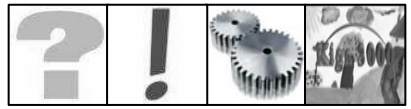
- **code *ChristmasNotifier.java***

```
public class ChristmasNotifier {
 public static void main(String[] args) {
  System.out.println("Merry Christmas");
  System.out.println("Today, I have visited" + args[1]
  + "people yet.");
}}
```

- **challenges**
  - “Merry Christmas”
  - compound message; in German other word-order possible “100.000 Menschen habe ich heute schon besucht.”
  - formatted numbers: 100,000 vs. 100.000

- **new: localised properties files**
  - *CNLocales_en.properties*

Possible implementations ▸ Java (contd.)



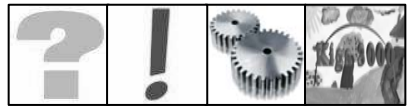
- ***CNLocales_en.properties***

```
announcement = Merry Christmas
```

- ***ChristmasNotifier.java***

```
import java.util.*;

// create locale
myLocale = new Locale(args[2], "");

// create resource bundle
ResourceBundle myBundle =
   ResourceBundle.getBundle("CNLocales", myLocale);

// display contents
System.out.println(
   myBundle.getString("announcement“)
   );
```

Possible implementations ▸ Java (contd.)



- ***CNLocales_en.properties***

```
report = Today, I have visited {1, number, integer} people yet.
```

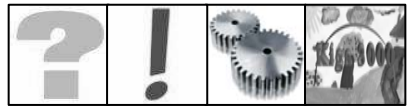
- ***CNLocales_de.properties***

```
report = {1, number, integer} Menschen habe ich heute schon besucht.
```

- ***ChristmasNotifier.java***

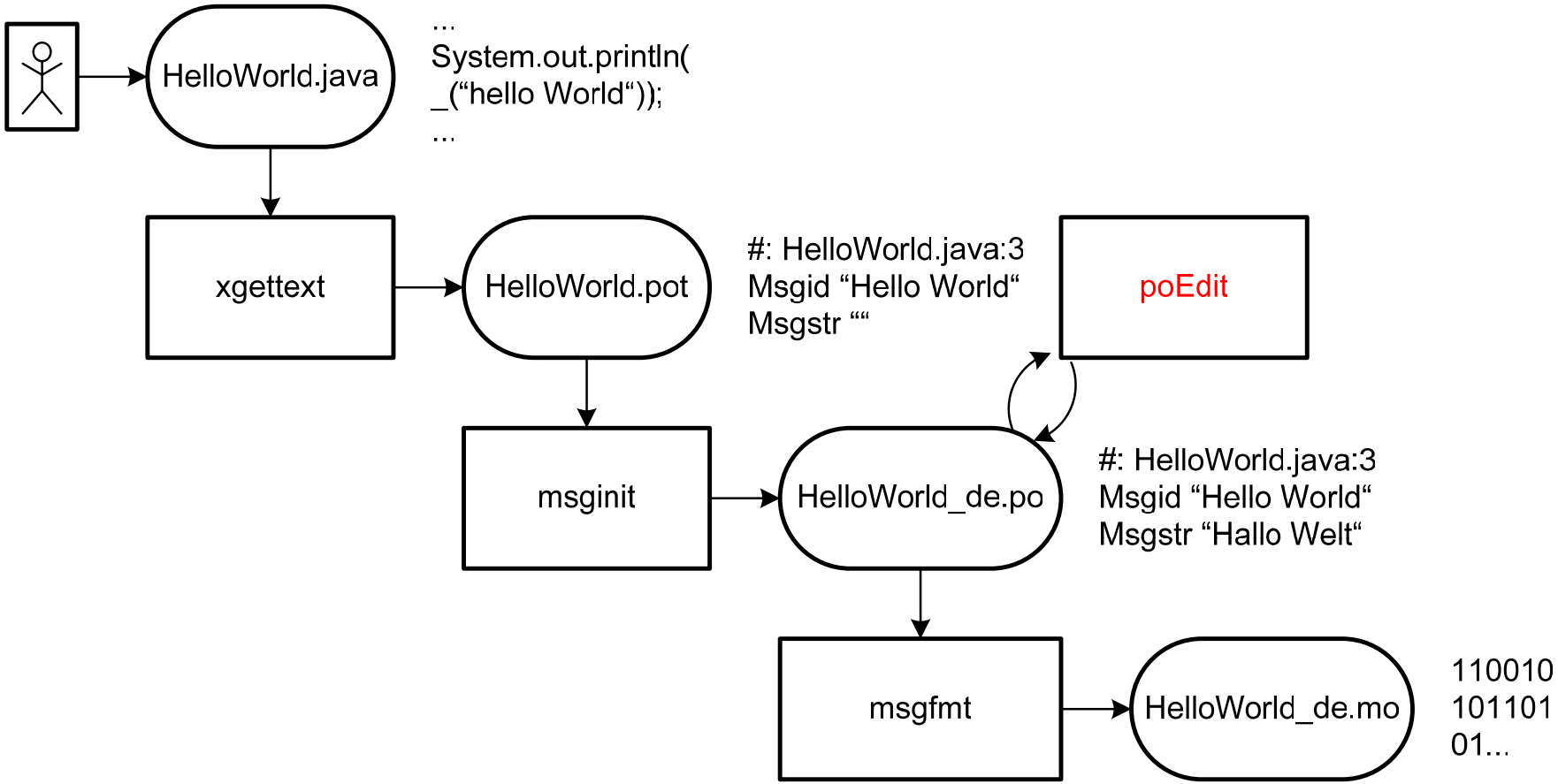
```
//prepare formatter
MessageFormat myFormatter = new MessageFormat("");
myFormatter.setLocale(myLocale);

//prepare message arguments
Object[] messageArguments = {new Integer(args[1])};

//apply pattern and display
myFormatter.applyPattern(myBundle.getString("report"));
System.out.println(
   myFormatter.format(messageArguments));
```

Possible implementations ▸ gettext

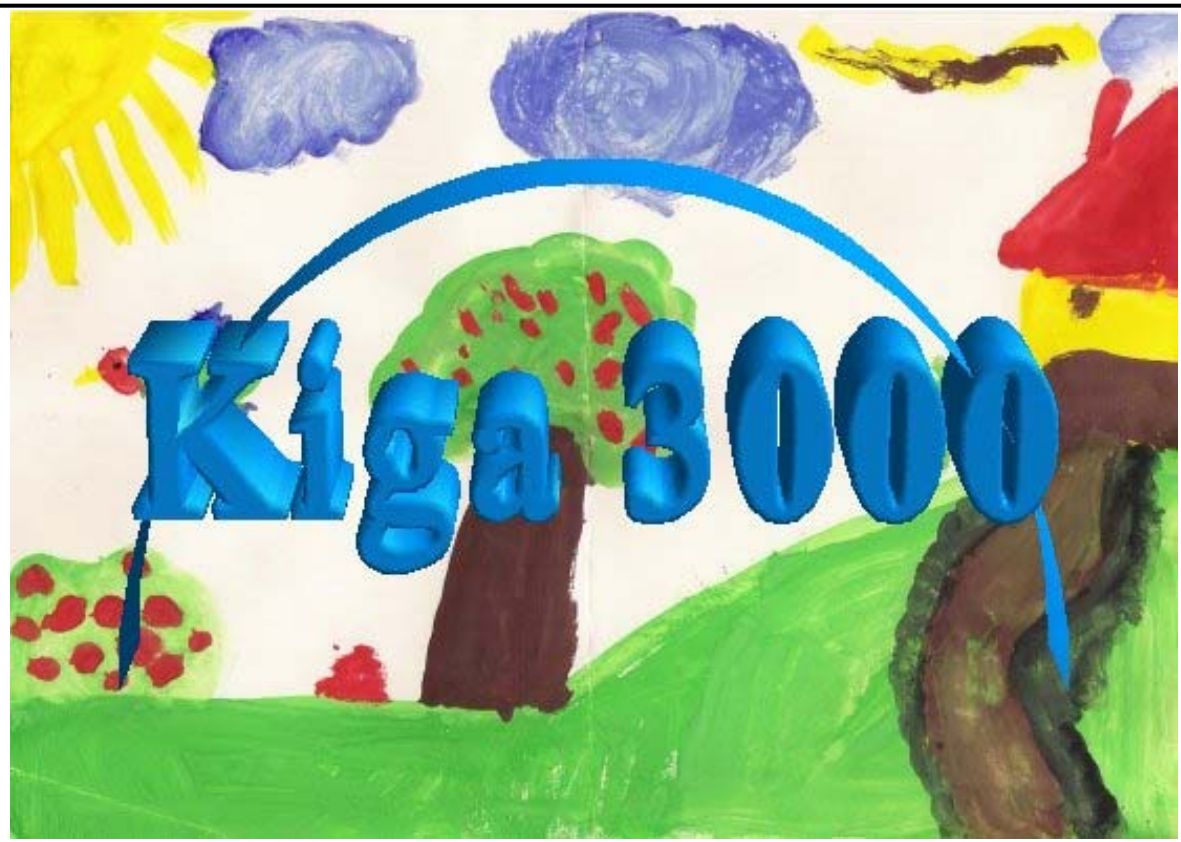


- **GNU gettext**
  - I18n library for many programming languages
  - state-of-the-art in open source projects
  - strict segregation of strings and code
  - environment variables
  - encapsulation of strings with the special function **gettext**(“foo“) or _(“foo“)
  - comfortable user interface: e.g. PoEdit
  - well-described work flow

Possible implementations ▸ gettext (contd.)



KiGa 3000



- software for managing kindergarden groups
- extensive use of database technology
- feature request: Introduce an internationalisation infrastructure in KiGa3000
- http://www.sourceforge.net/projects/kiga3000

References

- **General**
  - http://www.saphor.de/download/i18n.html
  - **http://java.sun.com/docs/books/tutorial/i18n**
  - http://www.mathema.de/event/publikation/campus_2003_11/Swing.pdf
- **Generic approach**
  - http://www.php-center.de/beitraege/detail.php?a_id=104
- **Resource Bundles**
  - http://www.galileocomputing.de/openbook/javainsel3/javainsel_100004.htm
- **Gettext**
  - http://www.gnu.org/software/gettext/gettext.html
  - http://www.phpcenter.de/de-html-manual/function.gettext.html
  - http://de.wikipedia.org/wiki/Gettext