Evergreen globalization: past, present, future

Dan Scott
dscott@laurentian.ca

Evergreen User Conference
May 20, 2009

http://creativecommons.org/licenses/by-sa/2.5/ca/

Agenda

- Evergreen past: 1.0 globalization
- Evergreen present: 1.4 globalization
  - Translation framework
  - Translation tools
  - Translation process
- Evergreen future: 2.0 and beyond
  - Mo' better translation and localization

My personal agenda

- I live in an officially bilingual country
- I work for an officially bilingual university
- I have friends in other countries (hello Tigran!) where English is a second or third language
- First blog post on the subject: Evergreen internationalization chat, November 17, 2006

Evergreen past: 1.0 / 1.2

- A pony with one internationalization trick: enabling the translation of static (X)HTML text
- Languages supported in 1.0: 1 - English (United States)
- Languages supported in 1.2: 2 – English (United States); French (Canada) (OPAC only)

Photo: http://www.flickr.com/photos/treehouse1977/2253328426/sizes/l/

Static (XM|XU|X?HT)ML text

- Most catalogue and staff client files are XML, XUL or XHTML composed of static text
  - Text is converted to entities in (XM|XU|X?HT)ML files
  - Entities are defined in DTD files in /openils/var/web/opac/locale/ll-LL/
  - Correct DTD is loaded via server-side include
  - XMLENT Apache extension replaces that entity inline
- Aside: never create strings by concatenating entities together!

Raw XUL file

```
<?xml version="1.0"?>
<!-- LOCALIZATION -->
<!DOCTYPE window PUBLIC "" ""[
    <!--#include virtual="/opac/locale/${locale}/lang.dtd"-->
]>

<window id="cat_marc_view_win"
    onload="try { my_init(); font_helper(); } catch(E) { alert(E); }"

xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
>

    <groupbox flex="1">
        <caption label="&staff.cat.marc_view.title;"/>
        <iframe id="marc_frame" flex="1"/>
    </groupbox>

</window>
```

It's just XML!

- Given XML input like

```
<!DOCTYPE window PUBLIC "" ""[
    <!--#include virtual="/opac/locale/${locale}/lang.dtd"-->
]>
<groupbox flex="1">
    <caption label="&staff.cat.marc_view.title;"/>
    <iframe id="marc_frame" flex="1"/>
</groupbox>
```

- ... XMLENT resolves a DTD entity defined in lang.dtd:

```
<!ENTITY staff.cat.marc_view.title “View MARC”>
```

- And generates the following output:

```
<groupbox flex="1">
    <caption label="MARC View"/>
    <iframe id="marc_frame" flex="1"/>
</groupbox>
```

Evergreen present: 1.4

- A stable of POnies, each with their own tricks

Photo: http://www.flickr.com/photos/nikonvscanon/1279007842/sizes/o/

Language support in 1.4

- Armenian
- Czech
- English (United States)
- English (Canada) – it's spelled “catalogue”, eh?
- French (Canada)
- And others are out there:
  - The Georgians have been flirting with us
  - A Simplified Chinese version of the OPAC was demonstrated back in the 1.2 era

Translation framework

All roads lead to PO

Photo: http://www.flickr.com/photos/19001426@N08/2345284953/sizes/l/

Translation process

- Philosophy: let the developers develop, and let the translators translate

1) Development occurs in the en-US locale under the /Open-ILS/ source directory
2) POT files are generated from the en-US native files and used to update PO files in the /build/ source directory
3) Locale-specific PO files are translated
4) At release packaging time, PO files generate locale-specific native files in /Open-ILS/ source directory

From entities to PO

- moz2po from Translate Toolkit takes entities defined in the DTD like this:

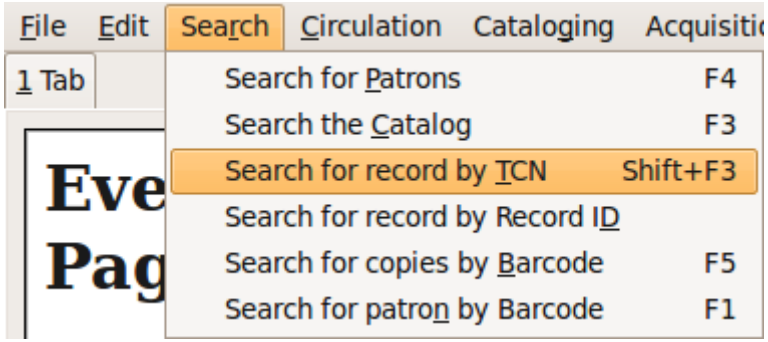
```
<!ENTITY staff.cat.marc_view.title “View MARC”>
```

- And generates round-trippable PO entries like this:

```
#: staff.cat.marc_view.title
msgid "MARC View"
msgstr ""
```

XUL accelerator keys

- XUL widgets like buttons or menu items can be associated with *accelerator keys*



- Problem – how do you keep the accelerator key associated with the label through the translation process?

XUL accelerator keys

- **Translate toolkit** combines accelerator keys with widget labels in PO entries – if you name the entities foo.accesskey and foo.label, respectively
  - The following lang.dtd entries:

```
<!ENTITY staff.main.menu.search.record.accesskey "T">
<!ENTITY staff.main.menu.search.record.label "Search for record by TCN">
```

  - Generate the following PO entry:

```
#: staff.main.menu.cat.search_tcn.label
#: staff.main.menu.cat.search_tcn.accesskey
msgid "Retrieve record by &TCN"
msgstr ""
```

XMLENT challenges

- Impossible to properly support parameterized strings
- Invasive; often breaks inline JavaScript outside of protected <![CDATA[...]]> blocks
- Custom solution that only the Evergreen project uses

- That said, it's fast and it does work...

Dynamic text: staff client

- Catalogue uses JavaScript to display contents
  - A bit nasty, and no parameterized argument support
- XUL supports JavaScript bundles of translatable strings with parameterized arguments
  - Perfect! But client-side only, for security reasons
  - And much of the Evergreen staff client is server-side
- So Mike and Jason built a custom XUL chrome widget (*messagecatalog*) that supports two functions:
  - getString(*stringID* string)
  - getFormattedString(*stringID* string, *params* array)

messagecatalog example

- Strings are defined in chrome and server .properties files; from `xul/server/locale/en-US/circ.properties`:

```
staff.circ.copy_status.sel_renew.not_circulating
  = Item with barcode %1$s is not circulating.
```

- Corresponding `circ.properties.po` entry:

```
#: staff.circ.copy_status.sel_renew.not_circulating
msgid "Item with barcode %1$s is not circulating."
msgstr ""
```

- As invoked in `xul/server/circ/copy_status.js`:

```
alert(
    $('circStrings').getFormattedString(
        'staff.circ.copy_status.sel_renew.not_circulating',
        [barcode])
);
```

messagecatalog challenges

- Linefeeds are not properly handled; you have probably seen “\n” in various alerts and dialogues
- Only available in the staff client at the moment
- Another custom solution that only the Evergreen community uses

JavaScript text: Dojo interfaces

- Dojo Toolkit JavaScript framework supports string substitution with parameterized arguments
- Load the localized strings:

```
dojo.requireLocalization("openils.reports", "reports");
var rpt_strings =
    dojo.i18n.getLocalization("openils.reports", "reports");
```

- Use a simple string:

```
label : rpt_strings.FILTERS_LABEL_GT_EQUAL,
```

- Use a parameterized string:

```
dojo.string.substitute( rpt_strings.RPT_BUILDER_CONFIRM_SAVE,
    [tmpl.name(), tmpl.description()] )
```

Dojo string bundles

- Defined as plain old JSON hashes; for example, from `Open-ILS/web/js/dojo/openils/reports/nls/reports.js`:

```
{
    "RPT_BUILDER_CONFIRM_SAVE":
        "Name : ${0}\nDescription: ${1}\nSave Template?",

    "FILTERS_LABEL_GT_EQUAL": "Greater than or equal to"
}
```

- Dojo tries to match the browser's requested locale and falls back on en-US if not found
- Dojo string bundles roundtrip to PO via build/i18n/scripts/dojo_resource.py

Dojo challenges

- Default build is problematic:
  - Loading hundreds of files adds significant network overhead, even after initial load
  - Supports only a handful of locales
  - Custom builds can solve both of these problems by merging files into layers and adding required locales (like Armenian)
- Developing rapidly, deprecating rapidly
- Replacing existing solutions, such as DHTML Calendar and custom AJAX code, with Dojo is time-consuming and requires careful testing

Database strings

- Close to one thousand strings are stored in the database in a default Evergreen install
- More strings are added as you define libraries, bill types, and the like
- Two PostgreSQL functions handle storing and retrieving in-database strings:
  - oils_i18n_gettext() - marks strings for storage in the database
  - oils_i18n_xlate() - retrieves a localized version of a string from the database, if available

Storing database strings

- oils_i18n_gettext(keyval INT|TEXT, string TEXT, class_hint TEXT, property TEXT)
- At build time, the string **string** is extracted into a PO file by build/i18n/scripts/db-seed-i18n.py and associated with the property **property** of an object of type **class_hint** with identifier **keyval**

Translating database strings

- For example, from `950.data-seed-values.sql`:

```
INSERT INTO config.bib_source (id, quality, source, transcendant)
VALUES (1, 90, oils_i18n_gettext(1, 'OCLC', 'cbs', 'source'), FALSE);
```

- The following PO is generated:

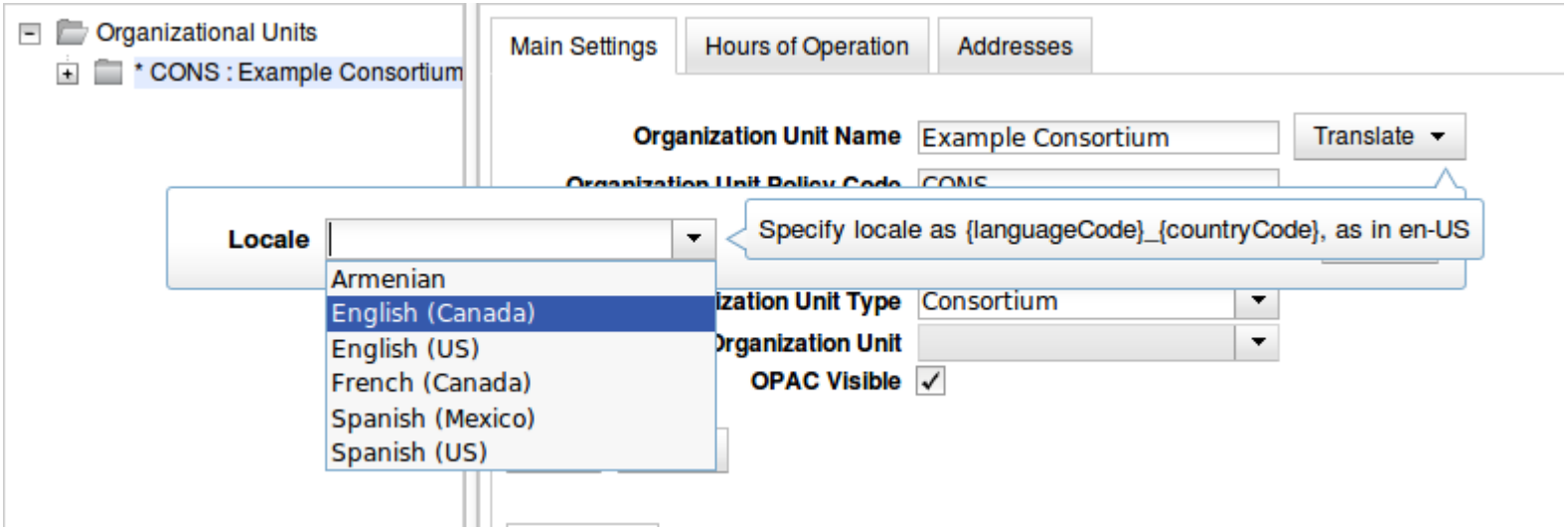
```
#: cbs.source:1
msgid "OCLC"
msgstr ""
```

- And from that, we generate the following SQL in `950.data-seed-values-en-CA.sql`:

```
INSERT INTO config.i18n_core
    (fq_field, identity_value, translation, string) VALUES
    ('cbs.source', '1', 'en-CA', 'Elephant');
```

Translating database strings (2)

- Mike Rylander built a Dojo widget that enables users to supply translated values for supported locales
- Not available in all interfaces, but we have the technology



Retrieving database strings

- oils_i18n_xlate (keytable TEXT, keyclass TEXT, keycol TEXT, identcol TEXT, keyvalue TEXT, raw_locale TEXT)
  - Retrieves the string for table *keyclass.keycol* where *identcol* = *keyvalue* and locale = *raw_locale* (ish)
  - Falls back to matching on language if exact locale is not found
  - Falls further back to returning the original string if language is not found

MARC editor tooltips

- Extracted from freely available online sources – currently only en-US and fr-CA

```
<field tag="022" repeatable="true">
  <name>INTERNATIONAL STANDARD SERIAL NUMBER</name>
  <description>The ISSN, a unique identification number assigned to a
continuing resource.</description>
  <indicator position="1" value="#">
    <description>No level specified</description>
  </indicator>
  <indicator position="1" value="0">
    <description>Continuing resource of international
interest</description>
  </indicator>
  <subfield code="a" repeatable="false">
    <description>International Standard Serial Number</description>
  </subfield>
</field>
```

Reporter interface

- fm_IDL.xml contains reporter:label attributes that describe classes and properties for the reporter interface
  - Extract reporter:label attributes
  - Generate PO files with the label definitions
  - Generate an entity-ized fm_IDL.xml that gets placed in the reports folder
  - Generate DTD files from the PO files
  - Then XMLENT does its magic...

ILS events

- No, not the Evergreen International Conference kind of ILS event...
- These events are defined in ils_events.xml; they have numeric code and text code identifiers, and longer locale-specific descriptions:

```
<event code="1000" textcode="LOGIN_FAILED">
    <desc xml:lang="en-US">User login failed</desc>
    <desc xml:lang="en-CA">User login failed</desc>
    <desc xml:lang="fr-CA">L'ouverture de session de
l'utilisateur a échoué</desc>
</event>
```

- ils_events.py and merge_ils_events.py do the work of extracting en-US descriptions to PO

Translation process

- Build a localized release
- Make the updated PO files available to translators
- Translate with the tool of your choice
- Return updated PO files to roll into build

Building a localized release

- Install the prerequisites:

```
$ sudo aptitude install translate-toolkit python-dev
$ sudo aptitude install python-setuptools
$ sudo easy_install polib simplejson
```

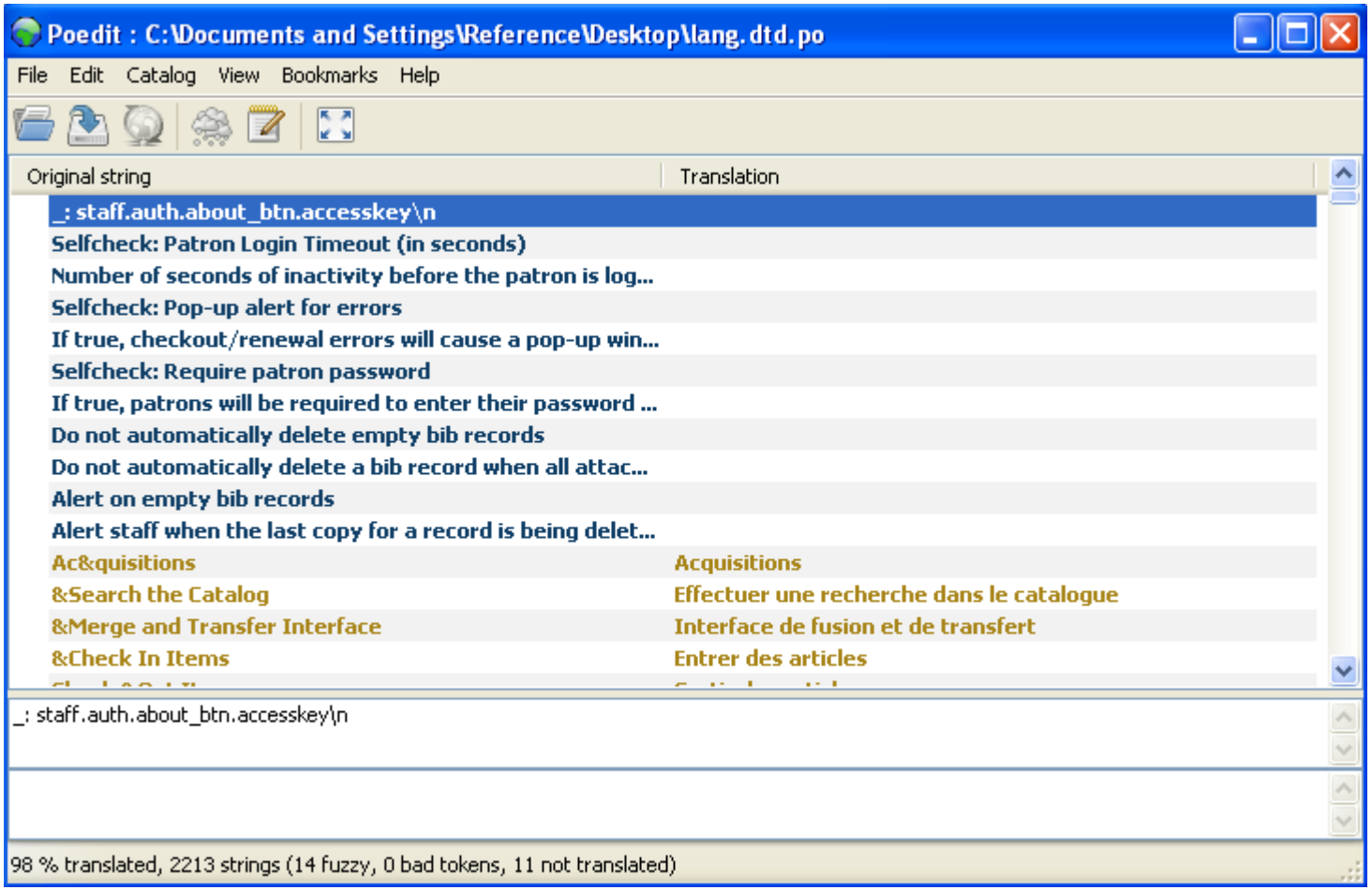
- Build the localized files:

```
$ cd ~/Evergreen-trunk/build/i18n
$ make newpot
$ make LOCALE=fr-CA install
```

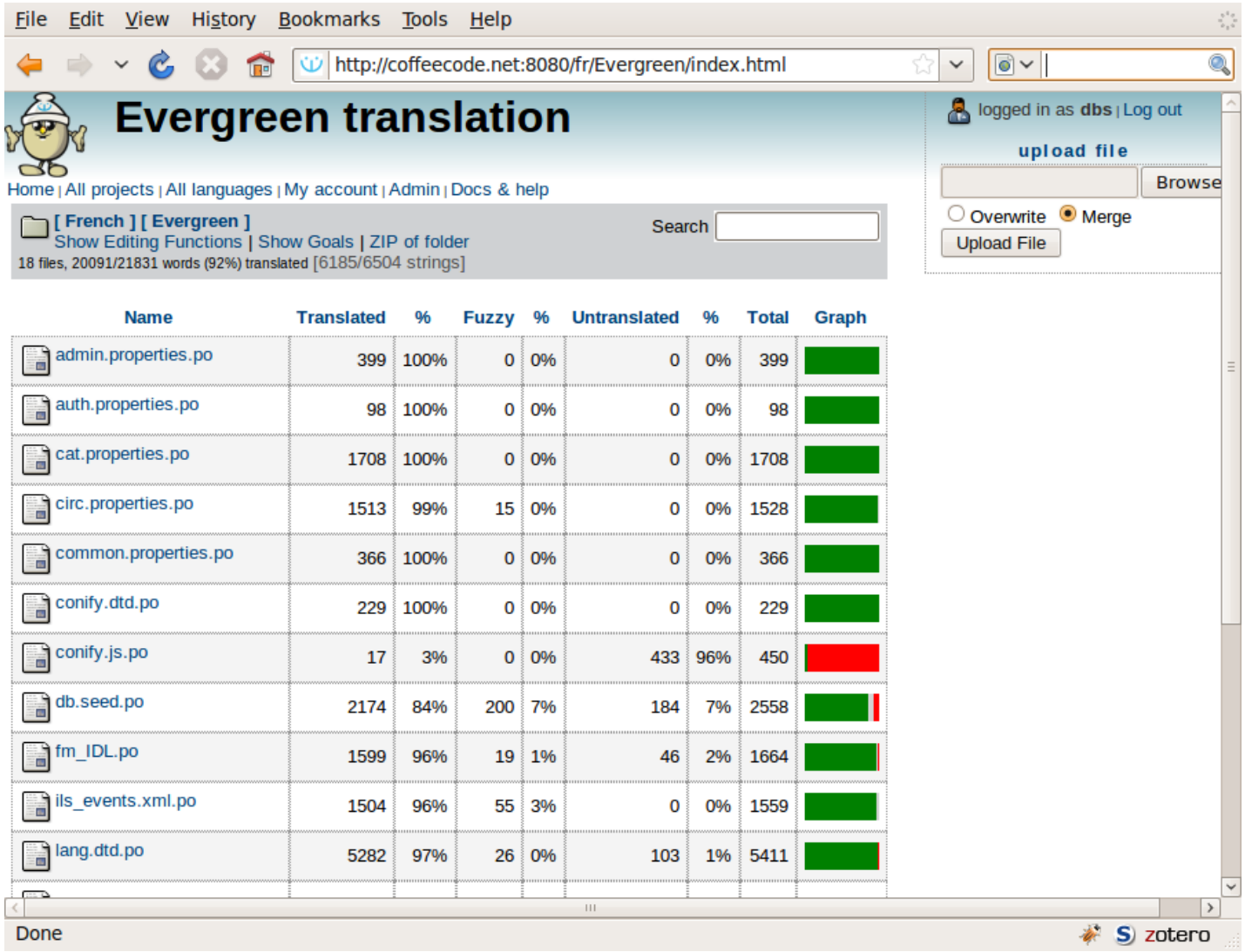
- Build and install Evergreen:

```
$ cd ~/Evergreen-trunk/
$ ./autogen.sh
$ ./configure --prefix=/openils --sysconfdir=/openils/conf
$ make
$ sudo make install
```

POEdit


Poedit : C:\Documents and Settings\Reference\Desktop\lang.dtd.po
File Edit Catalog View Bookmarks Help
Original string
Translation
_: staff.auth.about_btn.accesskey\n
Selfcheck: Patron Login Timeout (in seconds)
Number of seconds of inactivity before the patron is log...
Selfcheck: Pop-up alert for errors
If true, checkout/renewal errors will cause a pop-up win...
Selfcheck: Require patron password
If true, patrons will be required to enter their password ...
Do not automatically delete empty bib records
Do not automatically delete a bib record when all attac...
Alert on empty bib records
Alert staff when the last copy for a record is being delet...
Ac&quisitions
Acquisitions
&Search the Catalog
Effectuer une recherche dans le catalogue
&Merge and Transfer Interface
Interface de fusion et de transfert
&Check In Items
Entrer des articles
_: staff.auth.about_btn.accesskey\n
98 % translated, 2213 strings (14 fuzzy, 0 bad tokens, 11 not translated)

Pootle translation server



Evergreen future: 2.0 and beyond

- No worries, there's lots of work for everyone
  - More translations:
    - Tigran is working on a full Russian translation
    - Michigan may translate the catalogue into Spanish
- Template::Toolkit, the basis of acquisitions and many new administration interfaces, needs to learn to speak more than English
  - Locale::Maketext::Lexicon, in conjunction with Locale::Maketext::Extract::Plugin::TT2, looks like a good fit for our PO-centric universe

Simplify and enable translation

- XMLENT must die; replace with Template::Toolkit
- Continue to search and destroy hard-coded strings:
  - Staff client
  - SlimPAC
  - OpenSearch
  - SuperCat
- Move to one PO file for the whole project

Localization – qu'est-ce que c'est?

Localization refers to the **adaptation** of a product, application or document content to meet the language, cultural and other requirements of a specific target market (a "locale").

W3C Internationalization FAQ

Common localization requirements

- Numeric, date and time formats
- Use of currency
- Keyboard usage
- Collation and sorting
- Symbols, icons and colors
- Text and graphics containing references to objects, actions or ideas which, in a given culture, may be subject to misinterpretation or viewed as insensitive.
- Varying legal requirements

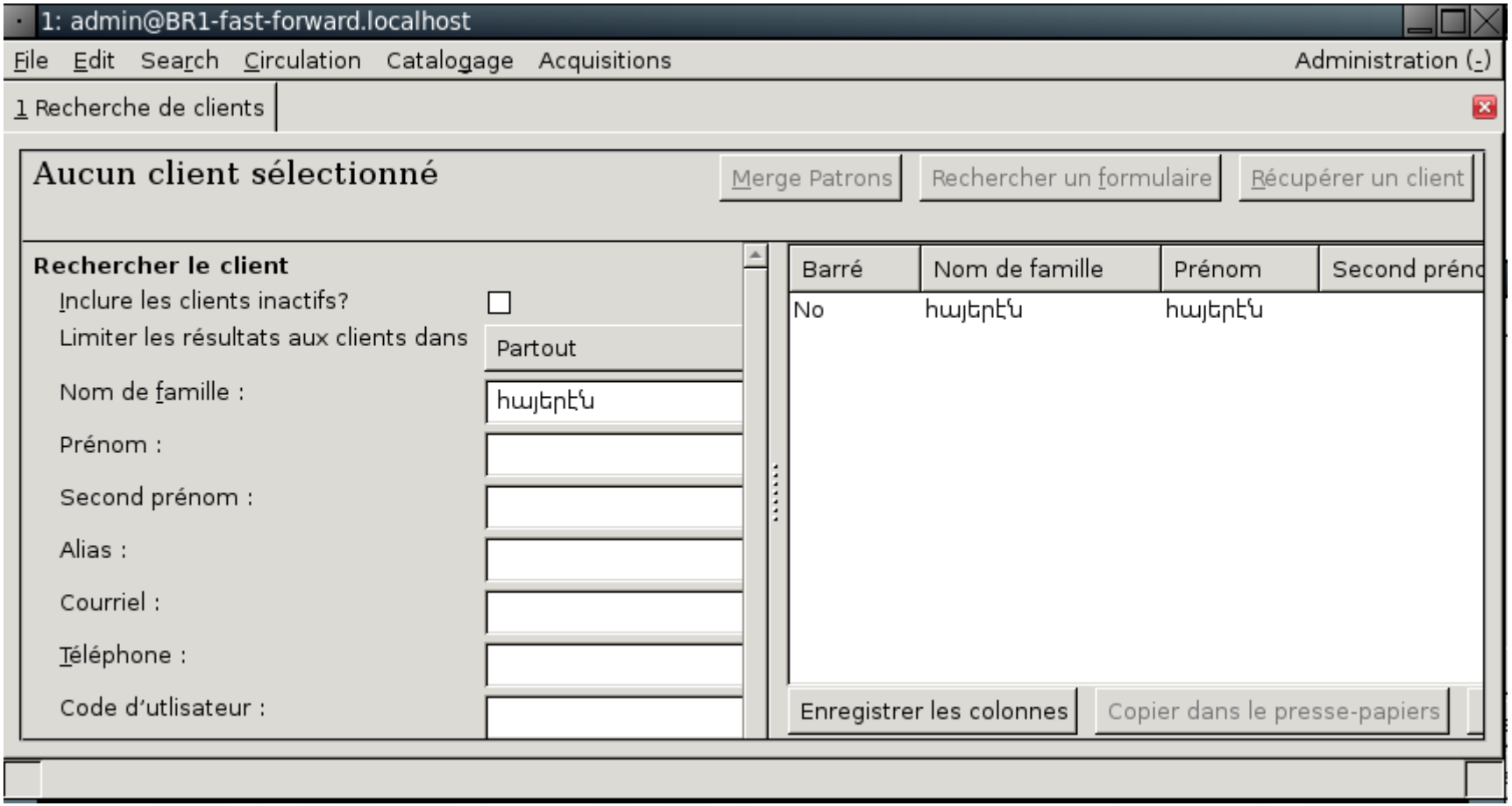
W3C Internationalization FAQ

Localization tasks

- Deploy Dojo date, time, number, and currency formatting widgets
- Move to Unicode-safe regexes in currently ASCII-only interfaces (patron registration and search)
- Integrate locale awareness into spell checker
- Support multiple collating sequences for sorting search results (depends on PostgreSQL 8.4)
- Teach user interfaces to better handle expansion (longer translated text)
- Support right-to-left and bidirectional locales

Unicode-safe regexes rock!

- Today, patron search throws out non-ASCII characters



A formal translation process?

- Translation sort of just happens, currently
- Proposal:
  - Development team cuts a release candidate and declares a *string freeze* for 2-3 weeks
  - Translation lead updates Pootle with the frozen strings
  - Translators translate new and changed strings
  - Development team cuts the final release, rolling in updated translations
- New translations of the current stable release can start at any time and sync up at the next string freeze

References

- Evergreen translation chat. Retrieved May 18, 2009, from http://coffeecode.net/archives/105-Evergreen-internationalization-chat.html
- Keyboard shortcuts – MDC. Retrieved May 18, 2009, from https://developer.mozilla.org/en/XUL_Tutorial/Keyboard_Shortcuts
- Translate Toolkit. Retrieved May 18, 2009, from http://translate.sourceforge.net/wiki/toolkit/index
- polib. Retrieved May 18, 2009, from http://code.google.com/p/polib/
- messagecatalog implementation. Retrieved May 18, 2009, from http://svn.open-ils.org/trac/ILS/browser/trunk/Open-ILS/xul/staff_client/chrome/content/main/bindings.xml
- W3C I18n FAQ: Localization vs. Internationalization. (n.d.). Retrieved May 18, 2009, from http://www.w3.org/International/questions/qa-i18n.
- Evergreen Internationalization (I18N), Localization (L10N), and Globalization (G11N) wiki page. Retrieved May 19, 2009 from http://open-ils.org/dokuwiki/doku.php?id=evergreen-admin:customizations:i18n

Photo credits

- Flickr Photo Download: Pony 84. (n.d.). . Retrieved May 18, 2009, from http://www.flickr.com/photos/treehouse1977/2253328426/sizes/l/. Used under the terms of the Creative Commons Attribution Share-Alike licence.
- Flickr Photo Download: Iceland Ponies 2. (n.d.). . Retrieved May 21, 2009, from http://www.flickr.com/photos/nikonvscanon/1279007842/sizes/o/. Used under the terms of the Creative Commons Attribution licence.
- Flickr Photo Download: Taylor Creek Salmon Run. (n.d.). . Retrieved May 18, 2009, from http://www.flickr.com/photos/19001426@N08/2345284953/sizes/l/. Used under the terms of the Creative Commons Attribution Share-Alike licence.