7.3 Translate OpenClinica into New Languages or Maintain Existing Translations

Completing Translations

The default (English) files in your i18n folder should be considered a complete list of all the strings that should be translated. You can find existing translated files at the Transifex OpenClinica Project site. The Transifex site includes tools to allow you to manage and conduct translations. Sign up to be a translator for your target language. Once approved you can upload translations or build them online using the Transifex tools.

Visit the i18n forum to share information on what you're translating and ask others for help.

If you find your target language does not match with your OpenClinica version, run a diff to identify the changes so you do not need to start from scratch. Here are some free diff tools. Diff files for 3.1.3 are located here.

In order to complete a translation, you must translate the text to the right of the '=' sign into your language.

Default	CRF_attributes = CRF Attributes
es	CRF_attributes = Atributos del CRD

Then save the file with your language extension. For example, if you are translating the strings into Spanish, you would add _es to the names of each of the 9 default files and save them as additional files. For example, words.properties would become words_es.properties. Please make sure to keep the original default files in the i18n folder along with the translated files.

Do not change the contents of the default files. Note also there are some strings in the file which should never be translated. These are identified with an appropriate comment in each of the properties files.

Asian Languages

Asian language translations are managed through GitHub rather than Transifex. Transifex does not seem to handle Asian language well. For example, you won't be able to see human readable characters, and downloaded files will add an unnecessary back slash because it treats file encoding as Latin-1 while it should be UTF-8.

To contribute/update Asian language translations, please submit a pull request to the GitHub repository.

Character Encoding

If you are translating into a language that uses characters not found in Western Europe (specifically, outside the ISO-Latin-1 character set) you will have to save those characters as Unicode escape sequences. For example "" would be "u20AC". To convert your translated text to Unicode escape sequences (Hex value), enter them into a converter such as brahan_converter and copy the strings. Be sure to save your original unconverted text in a comment, because the escape sequences are not editable.

Contributing a Translation

If you complete or improve one of the existing translations or create a new one, please consider sharing this back with the community! It's easy, sign up the Transifex OpenClinica Project site and post your files there.

Localization of Date Formats and Calendar Widget

OpenClinica supports displaying date and time in localized formats via the date format settings in the format.properties file. By default dates are formatted in DD-MMM-YYYY in English, where 'MMM' is an alphabetical abbreviation for the month.

Other date formats may be used by changing the locale-specific date format, with the alphabetical abbreviation or using numeric formats. For instance, you may want to use DD/MM/YYYY in your format_en-gb.properties for the Great Britain locale (note: this particular format is provided as an example only and is NOT recommend due to its ambiguity with common American date formats).

OpenClinica dates are never localized in ODM XML files, web services calls, or other programmatic data formats. In these cases OpenClinica uses the universal ISO 8601 format to represent all dates, date/times, and partial dates. Regardless of the user's locale, OpenClinica also saves all dates in its database in the ISO 8601 format. See OpenClinica Date Format Specifications for more detail.

When localizing OpenClinica, you must ensure that date formats specified in the format.properties file for a given locale are also supported by the OpenClinica javascript calendar widget. This is especially important when using the 'MMM' alphabetical month name as part of the localized date format. Please refer to examples below.

By default, OpenClinica provides a calendar widget in English (en). A file named calendar-en.js is located at ${Catalina.home}webappsyour-openclinica-instanceincludesnew_callang.

OpenClinica supports localization of the calendar widget into other languages. Let's use French (fr) as example. We'll assume French properties files are installed with format_fr.properties having a date format of DD-MMM-YYYY.

In this French scenario, if French is the chosen web browser locale, but there is no calendar-fr.js available, the default English calendar widget will be used. Since server-side date validation expects a French-formatted date, the English date string provided by the calendar widget may fail server-side date format validation. The user would be required to manually change the English date to a French date. For example, the English calendar widget will populate the field with 01-May-2012 but the DD-MMM-YYYY expected for the French locale would be 01-Mai-2012. 01-May-2012 has to be changed to 01-Mai-2012 to pass the validation.

To ensure the calendar widget is compatible with the localized date format, create a calendar-fr.js file which contains the translated contents from calendar-en.js file (follow the instructions in the .js file). Please refer to the examples below for more detail. Then, place calendar-fr.js file into ${Catalina.home}webappsyour-openclinica-instanceincludesnew_callang and restart Tomcat.

Based on the locale 'fr' being used, server-side validation of month names/abbreviations are automatically expected in French by the OpenClinica. The calendar widget translation MUST use the same month names/abbreviations as those expected by Java. Java 6 currently supports this list of languages for the java.text package, and the translations are provided by projects like CLDR from the Unicode Consortium. To ensure your month names/abbreviations exactly match those expected by the Java application, you can download the CLDR translations from here and find the list of months for your language in core/common/main.xml. Please note that Asian languages do not seem to follow this practice. For example, the browser default for the MMM abbreviation for Japanese and Korean are a number without leading zero when less than the month of October. This indicates both languages do not use strings for an abbreviated month.

Examples of Localized Date Formats

To ensure localized date formats work properly, you will need to modify the following items in the format.properties file located under tomcat/webapps/OpenClinica/WEB-INF/classes/org/akaza/openclinica/:

- date_format =
  - is used to build the error message
- date_format_string =
  - is used to display format by UI level
- date_time_format_string =
  - is used to format where time is applicable such as Scheduling Event
  - Note that OpenClinica allows only 24 hour format, and AM/PM markers will not be used
- date_format_calendar=
  - is used to format the date string that is sent from calendar widget when applicable

date_format, date_format_string, date_time_format_string

- Date
  - dd: 2 digits, i.e., 01, 02, 03...
  - d: digit(s) without leading zero, i.e., 1, 2, 3... 10, 11...
- Month
  - MM: month in digits. i.e., 01, 02, 03...
  - MMM: locale default abbreviated month names
    - Mar (English)
    - mars (French)
    - Mr (Deutsch)
    - 3 (Japanese, Korean)
    - ??(Chinese)
      .
      .
  - MMMMM: locale default full month names
    - December (English)
    - dcembre (French)
      .
      .
- Year
  - yyyy: Year with century
  - yy: Year without century

date_format Examples

Upper case M is used for month, while lower case is used for day and year for java class. However, upper case for all three are used to build error messages.

US English (Default): 01-Jan-2012

- date_format = DD-MMM-YYYY
- date_format_string = dd-MMM-yyyy
- date_time_format_string = dd-MMM-yyyy HH:mm:ss

Great Britain English and Spanish: 30/10/2012

- date_format = DD/MM/YYYY
- date_format_string = dd/MM/yyyy
- date_time_format_string = dd/MM/yyyy HH:mm:ss

French: 01-fvr.-2012

- date_format = DD-MMM-YYYY
- date_format_string = dd-MMM-yyyy
- date_time_format_string = dd-MMM-yyyy HH:mm:ss

Deutsch: 20.10.2012

- date_format = MM.DD.YYYY
- date_format_string = dd.MM.yyyy
- date_time_format_string = dd.MM.yyyy HH:mm:ss

Japanese: 2012/1/1

- date_format = YYYY/MMM/D
- date_format_string = yyyy/MMM/d
- date_time_format_string = yyyy/MMM/d HH:mm:ss

Japanese and Chinese: 2012?1?1? (Unicode Hex value needs to be escaped with single quote)

- date_format = YYYY'u5e74'MMM'u6708'd'u65e5'
- date_format_string = yyyy'u5e74'MMM'u6708'd'u65e5'
- date_time_format_string = yyyy'u5e74'MMM'u6708'd'u65e5' HH:mm:ss

Chinese: 2012-01-01

- date_format = YYYY-MM-DD
- date_format_string = yyyy-MM-dd
- date_time_format_string = yyyy-MM-dd HH:mm:ss

Chinese: 2012-??-1

- date_format = YYYY-MMM-D
- date_format_string = yyyy-MMM-d
- date_time_format_string = yyyy-MMM-d HH:mm:ss

Korean: 2012.1.1

- date_format = YYYY.MMM.D
- date_format_string = yyyy.MMM.d
- date_time_format_string = yyyy.MMM.d HH:mm:ss

Korean: 2012? 1? 1? (Unicode Hex value needs to be escaped with single quote)

- date_format = YYYY'uB144'MMM'uC6D4'd'uC77C'
- date_format_string = yyyy'uB144'MMM'uC6D4'd'uC77C'
- date_time_format_string = yyyy'uB144'MMM'uC6D4'd'uC77C' HH:mm:ss

CJK: 2012.01.01

- date_format = YYYY.MM.DD
- date_format_string = yyyy.MM.dd
- date_time_format_string = yyyy.MM.dd HH:mm:ss

date_format_calendar

List of parameters:

- %b: Prints the abbreviated **month name**. Range: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec. Example Output: Jan
- %B: Prints the full month name. Range: January, February, March, April, May, June, July, August, September, October, November, December. Example Output: January
- %d: Prints the **day** of the month with the leading zero. Range: 01 to 31. Example Output: 01
- %e: Prints the **day** of the month without the leading zero: Range: 1 to 31. Example Output: 1
- %m: Prints the numeric **month** with the leading zero. Range: 01 to 12. Example Output: 01
- %o: Prints the numeric representation of the **month** without the leading zero. Range: 1 - 12. Example Output: 1
  (Not supported OpenClinica version 3.1.2 or earlier)
- %y: Prints the **year** without the century with the leading zero. Range: 00 - 99. Example: 01
- %Y: Prints the **year** with the century. Example: 2001

date_format_calendar Examples

US English (Default): 01-Jan-2012

- date_format_ = %d-%b-%Y

Japanese and Chinese: 2012?1?1? (Hex escape not needed)

- date_format_ = %Yu5e74%ou6708%eu65e5
  (%o not supported OpenClinica 3.1.2 or earlier)

CJK: 2012.01.01

- date_format_ = %Y.%m-%d

Calendar Widgets

- When you translate Calendar Widgets found at ${Catalina.home}webappsyour-openclinica-instanceincludesnew_callang, you do not need to convert to Unicode escape sequence (Hex value). Translated strings pass without corruption as long as your tomcat connector is configured properly as explained before.

Example in French:

```
// short month names
Calendar._SMN = new Array
("janv.",
"fvr.",
"mars",
"avr.",
"mai",
"juin",
"juil.",
"aot",
"sept.",
"oct.",
"nov.",
"dc.");
```

- Save As the calendar-en.js file with appropriate locale marker, i.e., calendar-fr.js. This file name has to be called from the format-{lang}.properties file under "jscalendar_language_file =".

Localization of Image files

As of OpenClinica 3.1, icons and other image files do not contain any text and do not have to be localized. However it is possible to create localized images in the desired language. Under the folder ${Catalina.home}webappsyour-openclinica-instanceimages, makes a new folder with the name as desired locale code, e.g. "es" for Spanish (es). In this new folder collects all translated image files.

Notes for Developers

1. When you need Locale, please call the method org.akaza.openclinica.i18n.core.LocaleResolver.getLocale (HttpServletRequest request).
    org.akaza.openclinica.i18n.core.LocaleResolver class locates in Core module. Its resolveLocale(HttpServletRequest request) method implements OpenClinica business logic of how to determine locale (please refer to Which Language to Be Displayed on User Interface (UI)).

2. If you need Date/Time format in UI, please call the methods for pattern string or SimpleDateFormat in the class org.akaza.openclinica.i18n.util.I18nFormatUtil.
     If you need Date/Time format in database, please call methods for date format pattern string in the class org.akaza.openclinica.bean.core.ApplicationConstants.