EuroPython.org by Localizer

Internationalization and Localization of EuroPython.org with Localizer

J. David Ibáñez

`j-david@noos.fr`

Overview

This talk will explain how the EuroPython.org web site was internationalized and localized using the Localizer product. The discussion will be drived by examples and at the end some ideas to improve this task will be exposed.
The different sections are:

- Introduction
- The Internationalization process, exampe by example
- What was not internationalized and why
- The Localization process
- How to do better next time

Introduction, the context

The internationalization process of EuroPython.org has been determined by several constrains, these are:

- The web site was originally built as a monolingual one, without the support of multiple languages in mind.
- The internationalization started when the web site was already in production.
- Many developers did it, with different styles (some used ZPatterns, others developed a Python product, etc..).
- There was very few time to do it, the work took around a month spending few hours every week.
- Limited access to the web site, only to the management screens.

Introduction, the approach

As a consequence of these constrains, the web site internationalization is not ideal, basically:

- Different sections had different solutions because they were already different.
- None of the sections was rebuilt, instead workarounds were used when needed. Not always the most elegant solution.

This was a challenging task, a test for Localizer.

i18n, master_zpt

The overall layout and design of the web site is defined in the `master_zpt` template, which contains the menus and other texts. It was internationalized using a message catalog and few local content objects.

- A message catalog in the root of the site, `gettext`;

  ```
  <b tal:content="python:here.gettext('Register NOW with')" />
  ```

- Two local content objects, `footer` and `header`.

  ```
  <tal:block content="structure container/footer/body" />
  ```

i18n, index_html

The home page was a ZPT template, its internationalization consisted to use a local content object instead, and add the required `default_template`.

```
<html metal:use-macro="container/master_zpt/macros/page">
  <span metal:fill-slot="body">
    <tal:block content="structure here/body" />
  </span>
</html>
```

i18n, dates

To internationalize the dates a `LocalFolder` object was used, its name is `date` and contains a method for each language, for example, `long_en` looks like:

```
import DateTime

date = DateTime(int(date))
return date.strftime('%A, %d %B %Y')
```

It's used as:

```
container.date.long(date)
```

i18n, others

Other sections were internationalized too, using always the resources already described, the message catalog, local content objects and local folders.
However, the internationalization not always was as clean as the examples described above. For example:

- The talks are stored in a ZPattern based solution, to internationalize it without having to rewrite everything a non intrusive quick and dirty solution was used.

i18n, language negotiation

Nothing special was done in this area, the default features provided by Localizer were used.
An instance of the Localizer meta type was created in the root and the builtin form to change the language was used as a quick solution:

```
<tal:block content="structure here/Localizer/changeLanguageForm" />
```

Not internationalized

Some things in the web site weren’t internationalized for different reasons, they’re:

- Images with translatable text
  Technically are easy to internationalize, just using the `LocalFolder` meta type. However, the localization cost is too high, so they weren’t internationalized because they wouldn’t be localized anyway.
- Python products: ZWiki and Formulator
  These two products are monolingual and generate html directly. To internationalize them was out of the scope of this effort, and would have required much more resources than availiable.
  However, Localizer also provides facilities to make Python products as these multilingual. It was just of the scope.

Localization, who

The localization was done through the web using the interfaces provided by the different meta types. People that did it includes:

- Godefroid Chapelle
- Thomas Reulbach
- Nicolas Chauvat

Coordination was done through a wiki web.

Things to improve

Many things can be done to improve this task next time, but mainly two look as the most important ones:

- Everybody involved in the development of the web site should be aware of the internationalization requirements to avoid bad practices, such as using images with translatable text.
- A system should be developed to notify translators when an object has changed so they can update its translations.

Conclusions

I think Localizer has passed the test, EuroPython.org is now a multilingual web site. But there's still a lot of work to do to improve the internationalization and localization process. Localizer related links:

- Localizer, http://www.nuxeo.org/localizer
- CMFLocalizer, http://zope.org/Members/fafhrd/CMFLocalizer
  Extends Localizer with ZPT and CMF features.
- Base18, http://www.nexedi.org/software/Base18.stx
  Extends Localizer improving the multilingual content management.

Thank you for staying here until the end!!