I18NTransformer

Internationalization (i18n) in Cocoon

Introduction

Developing and maintaining multi-language sites is a common problem for web developers. The usage of XML and XSLT makes this task easier, especially with Cocoon's content, logic and presentation separation concept.

Internationalization (i18n)

Process of developing a product in such a way that it works with data in different languages and can be adapted to various target markets without engineering changes.

Localization (l10n)

Subsequent process of translating and adapting a product to a given market's cultural conventions.

For more info on this see search here] or [http://www.google.com/search?hl=en&ie=ISO-8859-1&oe=ISO-8859-1&q=XML+Internationalization&btnG=Google+Search

I18n transformer

This approach for internationalization (further - i18n) of applications within Cocoon is based on a transformer, which uses XML message catalogue for all the dictionary data and special markup for internationalized content.

The namespace of i18n markup in is defined as follows:

`xmlns:i18n="http://apache.org/cocoon/i18n/2.0"` - for Cocoon 2.0.x versions

`xmlns:i18n="http://apache.org/cocoon/i18n/2.1"` - Cocoon 2.1+ versions

Brief description

The following features are supported by the i18n transformer:

- Plain text translation
- Attribute translation
- Parameter substitution
- Substitution parameter translation
- Date, time, number and currency formatting
- Locale-based content filtering (2.1 version)
- Markup in translations (2.1 version)
- Named substitution parameters (2.1 version)
- Multiple catalogues

Detailed description

This description supposes that you first defined a **catalogue** file like the following :
message.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalogue xml:lang="fr">
  <message key="aKey">Une clé</message>
  ...
  <message key="zipcode">Code postal</message>
</catalogue>
```

You also must configure the main cocoon **sitemap** like this :

```
    <map:transformers>
      <map:transformer     name="i18n"
        src="org.apache.cocoon.transformation.I18nTransformer">
        <catalogue-name>messages</catalogue-name>
        <catalogue-location>../../translations</catalogue-location>
        <cache-at-startup>true</cache-at-startup>
      </map:transformer>
    </map:transformers>
```

Here is the way to call it in a **pipeline** :

```
<map:pipelines>
  <map:pipeline>
    <map:match pattern="mypage.html">
      <map:generate src="mypage.html"/>
      <map:transform type="i18n"/>
      <map:serialize type="html"/>
    </map:match>
  </map:pipeline>
</map:pipelines>
```

Here is now how you can internationalize items :

Plain text translation

Write `<i18n:text>aKey</i18n:text>` in your HTML source code. This will be replaced by the text corresponding to this key in the catalogue.

Attribute translation

Parameter substitution

Substitution parameter translation

Date, time, number and currency formatting

Multiple catalogues

If you want to use multiple catalogues, you can use this:

```
    <map:transformers>
      <map:transformer name="i18n" src="org.apache.cocoon.transformation.I18nTransformer">
        <catalogues default="other">
          <catalogue id="other" name="OtherMessages" location="../../translations"/>
          <catalogue id="woody" name="WoodyMessages" location="../../translations"/>
        </catalogues>
        <cache-at-startup>true</cache-at-startup>
      </map:transformer>
    </map:transformers>
```

Refer to the correct catalogue in your pipeline using the "default-catalogue-id" parameter:

```
<map:pipelines>
  <map:pipeline>
    <map:match pattern="mypage.html">
      <map:generate src="mypage.html"/>
      <map:transform type="i18n">
         <map:parameter name="default-catalogue-id" value="woody"/>
      </map:transform>
      <map:serialize type="html"/>
    </map:match>
  </map:pipeline>
</map:pipelines>
```

To be continued ...

Problem with Cocoon 2.1

if you get

```
java.lang.NullPointerException
        at org.apache.cocoon.components.ExtendedComponentSelector.release(ExtendedComponentSelector.java:317)
        at org.apache.cocoon.components.pipeline.AbstractProcessingPipeline.recycle(AbstractProcessingPipeline.
java:639)
        at org.apache.cocoon.components.pipeline.impl.AbstractCachingProcessingPipeline.recycle
(AbstractCachingProcessingPipeline.java:970)
        at org.apache.avalon.excalibur.pool.ResourceLimitingPool.put(ResourceLimitingPool.java:438)

...
java.lang.IllegalStateException: You cannot lookup components on a disposed ComponentLocator
        at org.apache.avalon.excalibur.component.ExcaliburComponentManager.lookup(ExcaliburComponentManager.
java:199)
        at org.apache.cocoon.components.CocoonComponentManager.lookup(CocoonComponentManager.java:315)

...
```

for Cocoon 2.1, then make sure your namespace reads

```
xmlns:i18n="http://apache.org/cocoon/i18n/2.1"
```

instead of "2.0"...

---Tom Eicher