Site Tree

Initial version

by Christian Egli, Andreas Hartmann

NOTICE:

Proposal for a a general framework for management of navigation for a publication.

Table of contents

1. Motivation

Currently there is no standard way to manage the navigation in Lenya. Previous attempts have shown their own merits and drawbacks. A general and standard approach is needed.

2. Rationale

The basic problems we're trying to solve are as follows:

1. Maintain the navigation in an easy format independent of the actual storage of the documents.
2. Enable moving of documents.
3. Enable querying of doctype independent of request URI.

2.1. Navigation

There are a couple of known implementations to choose from:

- The existing Lenya tree.xml
- The Forrest site.xml.
- The hashmap way: A sitetree.xml and purlspace.xml combination where the purlspace.xml contains all existing documents and sitetree.xml contains the navigation hierarchy.

2.2. Moving Of Documents

The Forrest site.xml solves this quite elegantly: Due to the use of XPath and some ambiguity the documents can be found almost independent of their actual location

With the existing Lenya or the hashmap solution changing of document location requires editing of the tree.xml or the purlspace.xml respectively.

2.3. Doctype Querying

For question 3 there are a couple of known approaches:

2.3.1. Hashmap

This method uses an action which contains for each request URI a mapping to attributes such as doctype, srcfile, xslt, etc.

2.3.2. Forrest SourceTypeAction

Forrest contains an action which for a given request opens the source file and determines its doctype based on the dtd declaration.

2.3.3. URIParametrizer

The URIParametrizer is a black box (probably an action) which internally issues cocoon requests to different sitemaps which return different attributes (doctype, etc.) for a given request URI.

This amounts to a combination of the hashmap solution with the basic cocoon concepts: The request URIs aren't matched against a hashmap but are handled by a sitemap instead. The sitemap contains normal pipelines and matchers which match on the request. As a catch-all there is still the possibility of using a hashmap for URIs that didn't match any pipelines.

Finally a new (internal) request is generated using the parameter values. This request is handled by a normal cocoon sitemap as if the original request had contained these parameters.

Diagram of URIParametrizer

3. The Sitetree Schema

The proposed sitetree schema looks as follows:

```
<?xml version="1.0"?>

<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://apache.org/cocoon/lenya/sitetree/1.0"
    xmlns="http://apache.org/cocoon/lenya/sitetree/1.0"
    elementFormDefault="qualified">

  <xs:element name="site">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="node"
            minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="node">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="node"
            minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="href" type="xs:string"/>
      <xs:attribute name="label" type="xs:string"/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

An example `sitetree.xml` could look like follows:

```
<site
    xmlns="http://apache.org/cocoon/lenya/sitetree/1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
        "http://apache.org/cocoon/lenya/sitetree/1.0
        ../../../../resources/entities/sitetree.xsd">

  <node href="index.html" label="Home"/>

  <node href="demo/" label="Demo">
    <node href="unipublic.html" label="Unipublic"/>
    <node href="news.html" label="Wyona News and Comments"/>
    <node href="oscom.html" label="OSCOM"/>
  </node>

  <node href="download/" label="Download">
    <node href="source.html" label="Source Version"/>
    <node href="binary.html" label="Binary Version"/>
  </node>

  <node href="documentation/" label="Documentation"/>

</site>
```

4. Backwards Compatibility

There has not been a previous implementation, so this is a non-issue.

5. Roadmap

- Decide which solution to choose

6. Reference Implementation

None