RemoveNamespaces

Update:

Current Cocoon (2.2 trunk) contains a SAX-based Transformer `org.apache.cocoon.transformation.StripNameSpacesTransformer` that strips all namespaces. It's like the NamespaceStripperTransfomer mentioned below, but faster because it does not use the DOM.

Configuration:

```
<transformer name="strip-namespaces" src="org.apache.cocoon.transformation.StripNameSpacesTransformer"/>
```

Usage:

```
<transform type="strip-namespaces" />
```

Note: you can't configure it, it will always strip all namespace declarations found in the SAX stream.

Problem:

... I use namespaces and I want to remove the declarations and prefixes in my result xml ...

(original mail:
http://marc.theaimsgroup.com/?l=xml-cocoon-users&m=102551958020786&w=2)

Solutions, which do NOT work:

Use `exclude-result-prefixes` or `exclude-prefixes` in a stylesheet

http://marc.theaimsgroup.com/?l=xml-cocoon-users&m=100101165325379&w=2

http://marc.theaimsgroup.com/?l=xml-cocoon-users&m=101912408904652&w=2

Use <`omit-xml-declaration`> in the sitemap

http://marc.theaimsgroup.com/?l=xml-cocoon-users&m=100084611731795&w=2

Rich discussion of test cases for `exclude-result-prefixes`

http://www.dpawson.co.uk/xsl/sect2/N5536.html#d113e654

Solutions, which WORK:

1.) Serializer (by Harry Lai)

Kay's XSLT Programmer's Reference:
"The xsl:exclude-result-prefixes and exclude-result-prefixes attributes apply only to namespace nodes copied from the stylesheet using literal result elements. They do not affect namespace nodes copied from the source document using <xsl:copy> or <xsl:copy-of>: there is no way of suppressing these."

Unfortunately, since xslt's will often have a catch-all template matcher to copy elements it doesn't transform, this comes up quite a bit.

So... what I ended up doing was extending the HTMLSerializer (or whatever

serializer you're using for your pipelines), and overriding the startPrefixMapping, endPrefixMapping, methods to do nothing, effectively removing all namespaces from my HTML. Also startElement and endElement must be overridden, in order to remove namespaces from attributes. This also had the added benefit of having no performance penalties (and theoretically, a ever-so-slight speedup since we no longer process namespaces in our serializer).

You could make this more general, and use the serializer's configuration to declare which namespaces you want to exclude, but excluding all worked well for us, especially since we were outputting HTML.

(complete mail: http://marc.theaimsgroup.com/?l=xml-cocoon-users&m=102553525402606&w=2)

2.) Transformation step (by Manos Batsis)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
    <xsl:template match="*">
      <!-- remove element prefix (if any) -->
      <xsl:element name="{local-name()}">
        <!-- process attributes -->
        <xsl:for-each select="@*">
          <!-- remove attribute prefix (if any) -->
          <xsl:attribute name="{local-name()}">
            <xsl:value-of select="."/>
          </xsl:attribute>
        </xsl:for-each>
        <xsl:apply-templates/>
      </xsl:element>
  </xsl:template>
</xsl:stylesheet>
```

[ChristophHermann] If you have some IE-Fixes like <!--[if IE]> in your HTML Code, you need to add an extra Template before the above:

```
<xsl:template match="text()">
     <xsl:value-of select="normalize-space(.)" disable-output-escaping="yes"/>
</xsl:template>
```

3.) Extra Identity Transform (from XSLT FAQ above and thread above too)

```
    <xsl:template match="*" priority="-1" mode="copy">
        <xsl:element name="{name()}">
            <xsl:copy-of select="@*"/>
            <xsl:apply-templates mode="copy"/>
        </xsl:element>
    </xsl:template>
    <xsl:template match="text()" mode="copy">
        <xsl:value-of select="normalize-space(.)"/>
    </xsl:template>
```

to use, instead of `<xsl:copy-of>`, use

```
<xsl:apply-templates mode="copy"/>
```

(complete mail: http://marc.theaimsgroup.com/?l=xml-cocoon-users&m=102552029221254&w=2)

4.) NamespaceStripperTransformer

Use the attached NamespaceStripperTransformer to kill all unsused namespaces. It works for me, but isn't really fast. Read the comment of the class in the source for details.

5.) OmitNsTransformer

Updated! New version - should properly omit `xmlns:xyz=http://...` attributes too. Also clears uri's of appropriate elements. There is new parameter o`mit-plain-xmlns`, set it to `true` to omit main `xmlns=http://...` declaration. It is useful especialy for HTMLSerializer to produce correct HTML without any namespace declaration.

You may also use attached OmitNsTransformer.java - simple SAX event based transformer. This transformer omit just the `xmlns:xyz` declarations. It can omit all or few of them using special parameter `omit-namespaces`. Example included in javadoc comments. Brief excerpt from sitemap:

Declare `omitns` as `map:transformer` component:

```
<map:components>
    ...
  <map:transformes>
    ...
  <map:transformer
      logger="sitemap.transformer.omitns"
      name="omitns"
            pool-grow="2" pool-max="16" pool-min="2"
            src="com.gitus.cocoon.transform.OmitNsTransformer"/>
   </map:transformer>
   ...
```

Add transformer into pipeline:

```
<map:match pattern="*_xsp">
  <map:generate type="serverpages" src="xsp/{1}.xsp"/>
  <map:transform type="omitns" label="xml">
   <map:parameter name="omit-namespaces" value="xsp jpath xspdoc esql xsp-request"/>
  </map:transform>
  <map:serialize type="xml"/>
 </map:match>
```

Attachment: OmitNsTransformer.java

Attachment: NamespaceStripperTransformer.java