```
<body bgcolor="white">
  <h2>Grigoris Antoniou</h2>
  Affiliation: University of Bremen<br>
  Email: ga@tzi.de
  <p>
  <h2>David Billington</h2>
  Affiliation: Griffith University<br>
  Email: david@gu.edu.net
  <p>
</body>
</html>
```

The xsl:apply-templates element causes all children of the context node to be matched against the selected path expression. For example, if the current template applies to / (that is, if the current context node is the root), then the element xsl:apply-templates applies to the root element, which in this case is the authors element (remember that / is located above the root element). And if the current node is the authors element, then the element xsl:apply-templates select="author" causes the template for the author elements to be applied to all author children of the authors element.

It is good practice to define a template for each element type in the document. Even if no specific processing is applied to certain elements, like in our example, the xsl:apply-templates element should be used. That way, we work our way from the root to the leaves of the tree, and all templates are indeed applied.

Now we turn our attention to attributes. Suppose we wish to process the element

```
<person firstname="John" lastname="Woo"/>
```

with XSLT. Let us attempt the easiest task imaginable, a transformation of the element to itself. One might be tempted to write

```
<xsl:template match="person">
```