От преобразования нет никакой пользы, если оно не приводит данные именно к той форме, которая нам нужна. Чтобы это сделать, нужно уметь использовать шаблоны, чему и посвящен данный раздел.

Созданиешаблонов

Большинство таблиц стилей не имеют такой простой формы, какую вы только что видели в предыдущем разделе. Вместо этого они разбиваются на группу шаблонов, каждый из которых применяется к определенному типу данных. Давайте переведем нашу таблицу стилей в эту форму (см. листинг 7).

Листинг 7. Переделанная таблица стилей

```
<xsl:stylesheet

 version="1.0"

 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

 xmlns="http://www.w3.org/TR/xhtml1/strict">
```

<xsl:template match="/">

<html>

<head>

<title>Recipe</title>

</head>

<body>

<h2><xsl:value-of select="/recipes/recipe/name"/></h2>

<h3>Ingredients:</h3>

<xsl:value-of

select="/recipes/recipe/ingredients"/>

<h3>Directions:</h3>

<xsl:value-of

select="/recipes/recipe/instructions"/></p>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

Здесь информация не изменилась, за исключением того, что процессор смотрит на таблицу стилей и начинает с шаблона, совпадающего с корневым элементом документа, как указано в атрибуте match. Затем он выводит этот шаблон, включая все значения, так же как и раньше. Если вы выполните сейчас преобразование, то должны увидеть точно такие же результаты, как и в листинге6.

Но это нето, что мы хотим. Мы хотим иметь возможность форматировать ингредиенты и инструкции. Для этого мы можем создать отдельные шаблоны для каждого из этих элементов и включить их в таблицу стилей (см. листинг8).

Листинг 8. Создание дополнительных шаблонов

```
<xsl:stylesheet

version="1.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

xmlns="http://www.w3.org/TR/xhtml1/strict">



<xsl:template match="/">

<html>

<head>

<title>Recipe</title>

</head>
```

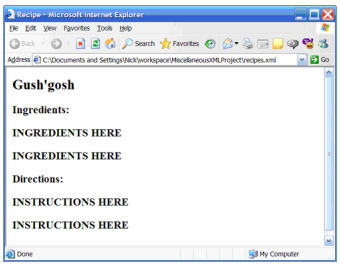
```
<body>

<h2><xsl:value-of select="/recipes/recipe/name"/></h2>

<h3>Ingredients:</h3>

<p><xsl:          apply          -templates

select="/recipes/recipe/ingredients"/></p>

<h3>Directions:</h3>

<p><xsl:          apply          -templates

select="/recipes/recipe/instructions"/></p>

</body>

</html>

</xsl:template>
```

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

 <h3>INGREDIENTS HERE</h3>

</xsl:template>

<xsl:template match="instructions">

 <h3>INSTRUCTIONS HERE</h3>

</xsl:template>
```

```
</xsl:stylesheet>
```

Обратите внимание, что вместо того, чтобы просто вывести элемент value-of, мы теперь указываем таблице стилей применять соответствующие шаблоны к элементам ingredients и instructions. Затем мы создали отдельные шаблоны для этих элементов, задав их в атрибуте match. Добравшись до элемента apply-templates, процессор выбирает все элементы ingredients в документе. Затем он ищет шаблон для ингредиентов, а, найдя его, выводит этот шаблон. То же самое он делает с элементом instructions. Результат должен быть похож на изображенный на рисунке 3.

Рисунок 3. Применение соответствующих шаблонов к элементам ingredients и instructions



Так, это уженемного ближе, хотя бы понятно, что здесь два рецепта, однако вы вряд ли захотите объединять ингредиенты и инструкции для всех рецептов. К счастью, эту проблему можно решить лучшей организацией шаблонов.