Чтобы лучше организовать данные, обратим внимание на то, как мы разделяем и публикуем шаблоны. XSLT позволяет обрабатывать информацию итеративным образом. Например, можно поделить информацию на отдельные рецепты, а затем отформатировать инструкции и ингредиенты (см. листинг 9).

Листинг 9. Выделение рецептов

```
<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>

<xsl:apply-templates select="/recipes/recipe"/>

</body>

</html>

</xsl:template>

<xsl:template match="recipe">

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

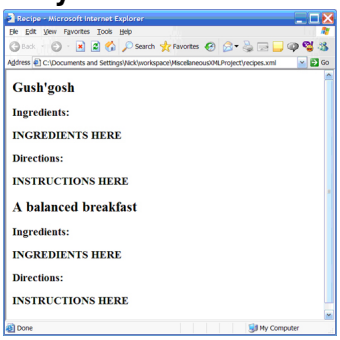
```
<h3>Ingredients:</h3>

<p><xsl:apply-templates select="./ingredients"/></p>

<h3>Directions:</h3>

<p><xsl:apply-templates

select="./instructions"/></p>

</xsl:template>

<xsl:template match="ingredients">

<h3>INGREDIENTS HERE</h3>
```

```
</xsl:template>

<xsl:template match="instructions">

<h3>INSTRUCTIONS HERE</h3>

</xsl:template>

</xsl:stylesheet>
```

В данном случае таблица стилей выводит основную страницу (HTML), а затем

просматривает каждый рецепт, выводя название, ингредиенты и инструкции для каждого рецепта. Опять-таки XPath мы будем изучать в разделе Подробнее об XPath, но в данном случае элемент recipe становится контекстным узлом, поэтому атрибуты select относятся к этому узлу так же как файлы к конкретному каталогу в файловой системе. Результат должен быть похож на рисунок 4.

Рисунок 4. Элемент recipe становится контекстным узлом



Отлично, формат приблизился к желаемому, однако нам все еще нужно отобразить фактическую информацию. Для этого надо изменить шаблоны ingredients и instructions (см. листинг 10).

Листинг 10. Работа над шаблонами ingredients и instructions

```
...

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

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

<h3>Ingredients:</h3>

<xsl:apply-templates select="./ingredients"/>

<h3>Directions:</h3>

<ol> <xsl:apply-templates

select="./instructions"/> **
**

</xsl:template>

<xsl:template match="ingredient">

<xsl:value-of select="./qty"/> <xsl:value-of

**select="./unit"/> <xsl:value-of select="./food"/>
**

</xsl:template>

<xsl:template match="instruction ">

**
<xsl:value-of select="."/>
**

</xsl:template>

</xsl:stylesheet>

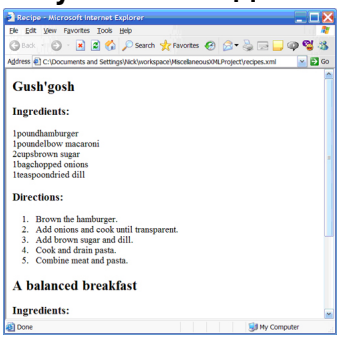
Важный момент при публикации шаблонов: вы указали процессору применять подходящие шаблоны к элементу ingredients, однако у нас нет специального шаблона для этогоэ лемента. Если вы примените шаблон к элементу, а самого шаблона не будет, то данные просто не появятся. Но это не наш случай.

Наоборот, мы воспользовались тем, что когда указали процессору применять подходящие шаблоны к элементу ingredients, он ищет не только элемент ingredients, но и дочерние элементы элемента ingredients. Таким образом, он находит шаблон для ингредиентов, в котором вы выводите количество, единицы измерения, название продукта и разрыв строки.

То же самоемы сделали для инструкций, когда отформатировали их как элементы списка. Заметьте, что мы создали фактически упорядоченный список в основном шаблоне для рецепта, а затем отправили элементы для индивидуальной обработки.

Результат должен быть похож на рисунок 5.

Рисунок 5. Создание упорядоченного списка в основном шаблоне рецепта



Определенно формат приближается к желаемому. Однако если вы посмотрите на вывод, то увидите, что проблема с пробелами все еще существует (см.листинг 11).

Листинг 11. Недоработанный вывод

```
<?xml version="1.0" encoding="UTF-8"?>

<html

xmlns="http://www.w3.org/TR/xhtml1/strict"><head><title>Recipe

</title></head><body><h2>Gush'gosh</h2><h3>

Ingredients:</h3><p>

1poundhamburger      <br/>

1poundelbow macaroni<br/>

2cupsbrown sugar      <br/>
```

1bagchopped onions

1teaspoondried dill

</p><h3>Directions:</h3>

Brown the hamburger.

Add onions and cook until transparent.

...