Named templates

Martin Holmes

Named templates

- So far, you've looked at templates that are invoked by means of a **match** attribute:

```
<xsl:template match="item">
  <li><xsl:apply-templates/></li>
</xsl:template>
```

- However, there is another way to invoke a template: we can give it a **name**, and call it using that name.

Naming a template, and calling it by name

Here's a named template:

```
<xsl:template name="dateToday">
  <p>Today's date is: <xsl:value-of select="current-date()"/></p>
</xsl:template>
```

and here's how you call it:

```
<xsl:call-template name="dateToday"/>
```

This is obviously useful; you can write a particular piece of logic once, and call it from multiple locations.

Passing parameters to a template

- Named templates are a little like functions in conventional programming languages.
- As with programming functions, you can also pass parameters to named XSLT templates.

Here's a named template with a parameter:

```
<xsl:template name="greetSomeone">
  <xsl:param name="whoToGreet"/>
  <p>Hello <xsl:value-of select="$whoToGreet"/>!</p>
</xsl:template>
```

and here's how you call it:

```
<xsl:call-template name="greetSomeone">
  <xsl:with-param name="whoToGreet">Fred</xsl:with-param>
</xsl:call-template>
```

Recursion using named templates

- There are many reasons to use named templates, but the most common one is **recursion**.
- In **recursion**, a template calls itself.
- This enables XSLT to perform operations that other programming languages achieve with **looping** or **iteration**.
- XSLT can't do conventional looping, because XSLT variables can't actually vary.

Recursion: how it works (1)

- Here's a simple template for outputting a number:

```
<xsl:template name="showNumber">
  <xsl:param name="theNumber" select="0"/>
  <xsl:value-of select="$theNumber"/>
</xsl:template>
```

  What exactly does this template do?
- Let's call this template like this:

```
<xsl:call-template name="showNumber">
  <xsl:with-param name="theNumber" select="3"/>
</xsl:call-template>
```

- The result should be just:

```
3
```

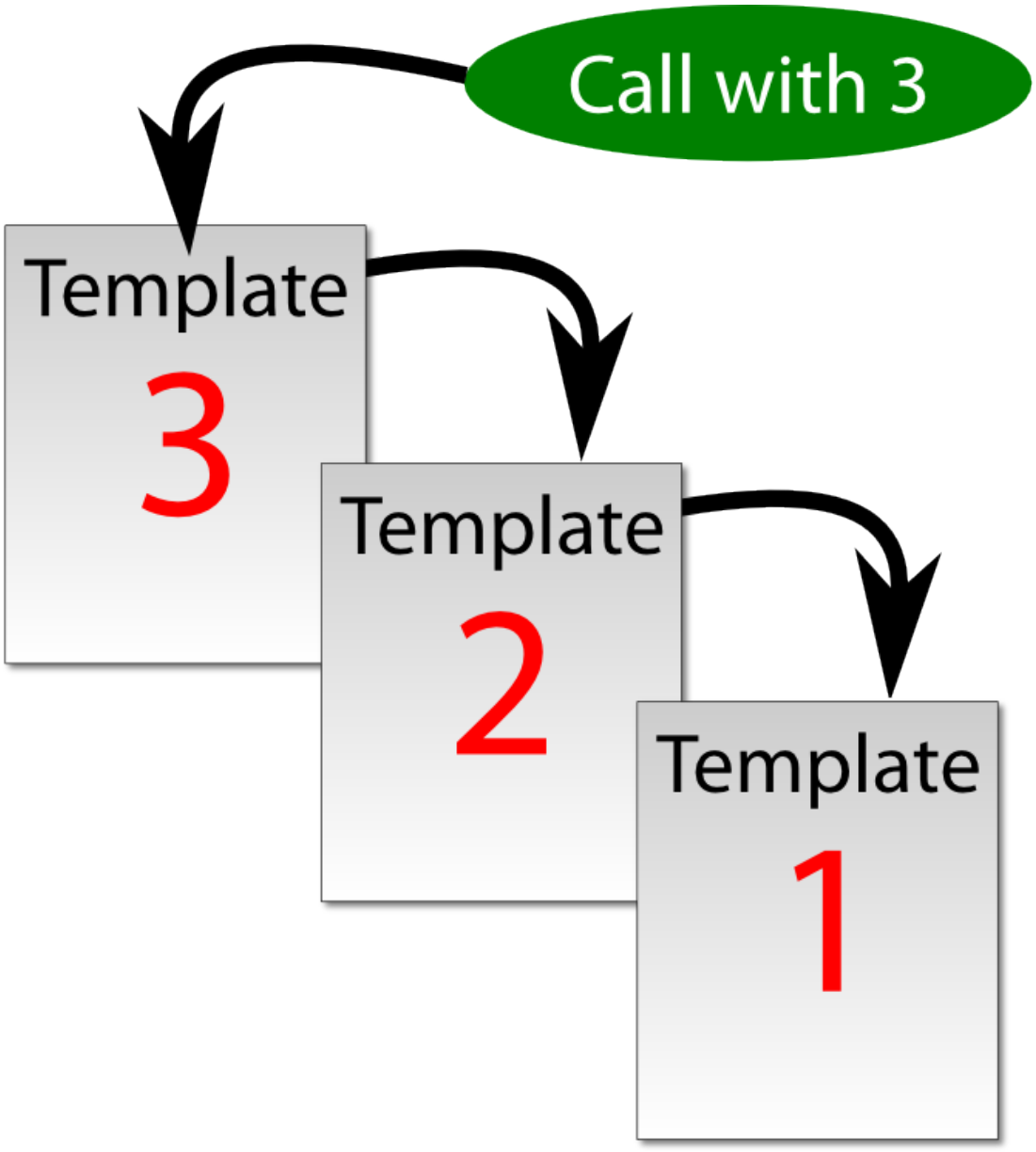
Recursion: how it works (2)

- Now let's add a new part to our named template:

```
<xsl:template name="showNumber">
  <xsl:param name="theNumber" select="0"/>
  <xsl:value-of select="$theNumber"/>
  <xsl:if test="$theNumber gt 1">
    <xsl:call-template name="showNumber">
      <xsl:with-param name="theNumber" select="$theNumber - 1"/>
    </xsl:call-template>
  </xsl:if>
</xsl:template>
```

Now what should our template do?

Recursion: how it works (3)



Named templates: Task

Write a named template which:

- is called "makeLink".
- has a parameter called "href" with a default value of "http://www.tei-c.org".
- has a parameter called "linkText" with a default value of "Click here".
- creates an XHTML anchor element using the href parameter and the link text.

Add this template to one of your XSLT files, and try calling it from elsewhere in the file.

Named templates: Task answer

```
<xsl:template name="makeLink">
  <xsl:param name="href">http://www.tei-c.org</xsl:param>
  <xsl:param name="linkText">Click here</xsl:param>
  <a href="{$href}"><xsl:value-of select="$linkText"/></a>
</xsl:template>
```