store process communicate

2011-10-04

- store as XML
- communicate using HTML and CSS
- process with XSL

```
<!DOCTYPE html>
<html lang="en">
  <!--This is a comment.
      Comments are not displayed
      in the browser-->
   <head>
     <title>Recipe</title>
   </head>
   <body>
     …
     <section>
        <!--ingredients-->
        …
     </section>
     <section>
        <!--method-->
        …
     </section>
   </body>
</html>
```

putting the meaning back

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Test</title>
  </head>
  <body>
    <section id="ingredients">
      …
    </section>
    <section id="method">
      …
    </section>
  </body>
</html>
```

css selectors:

```
#ingredients{
}

#method{
}
```

each id is unique within document

putting the meaning back

```
<section id="method">
  <ol>
    <li>beat the eggs</li>
    <li>squeeze the lemons</li>
        …
  </ol>
</section>
```

css selector:

```
#method li{
}
```

putting the meaning back

```
<p class="ingredient">2 eggs</p>
<p class="ingredient">4 lemons</p>
...
```

css selector:

```
.ingredient{
}
```

```
p.ingredient{
}
```

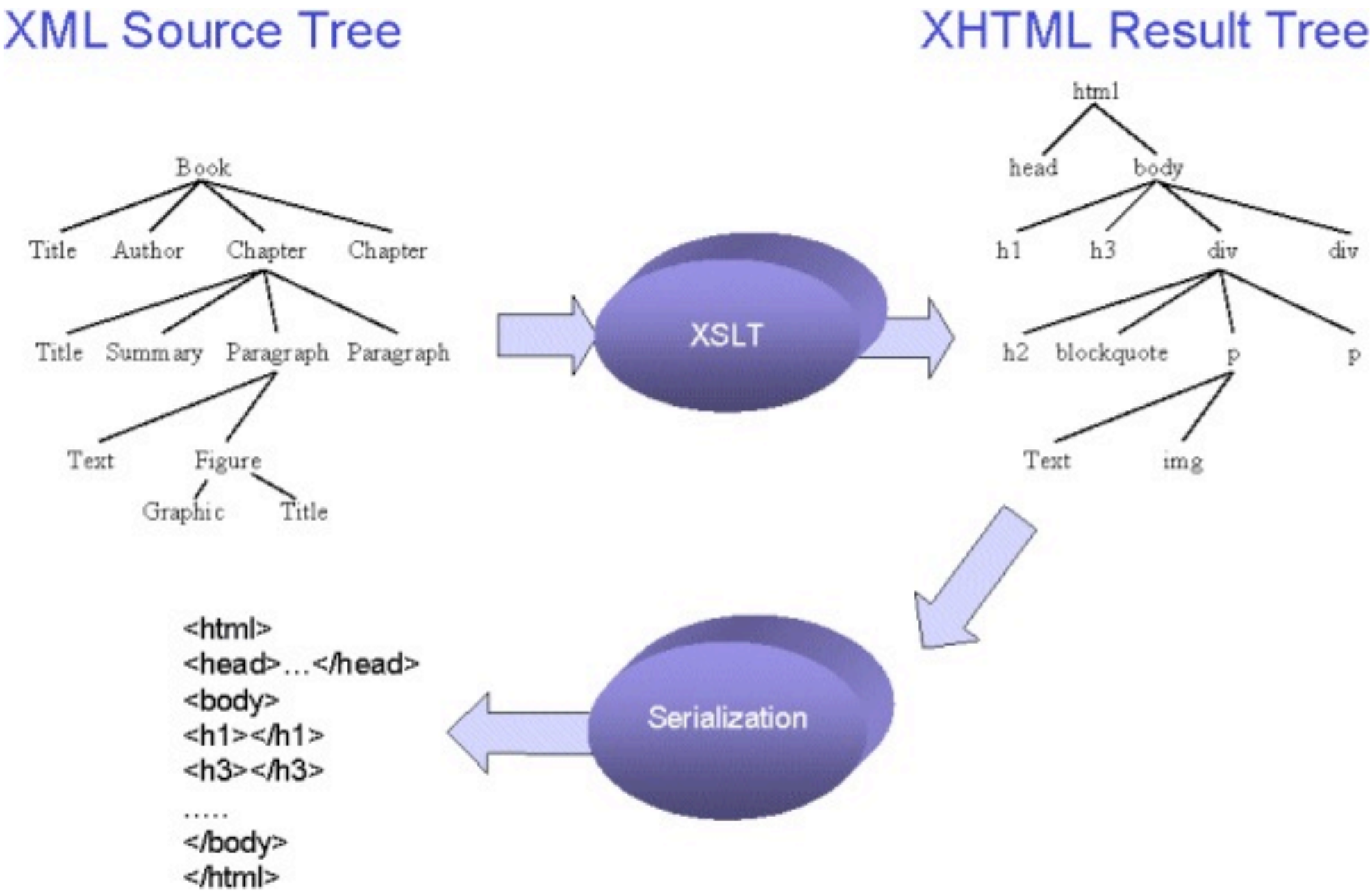
putting the meaning back

```
<div class="ingredients">
  <p>2 eggs</p>
  <p>4 lemons</p>
   …
</div>
```

css selector:

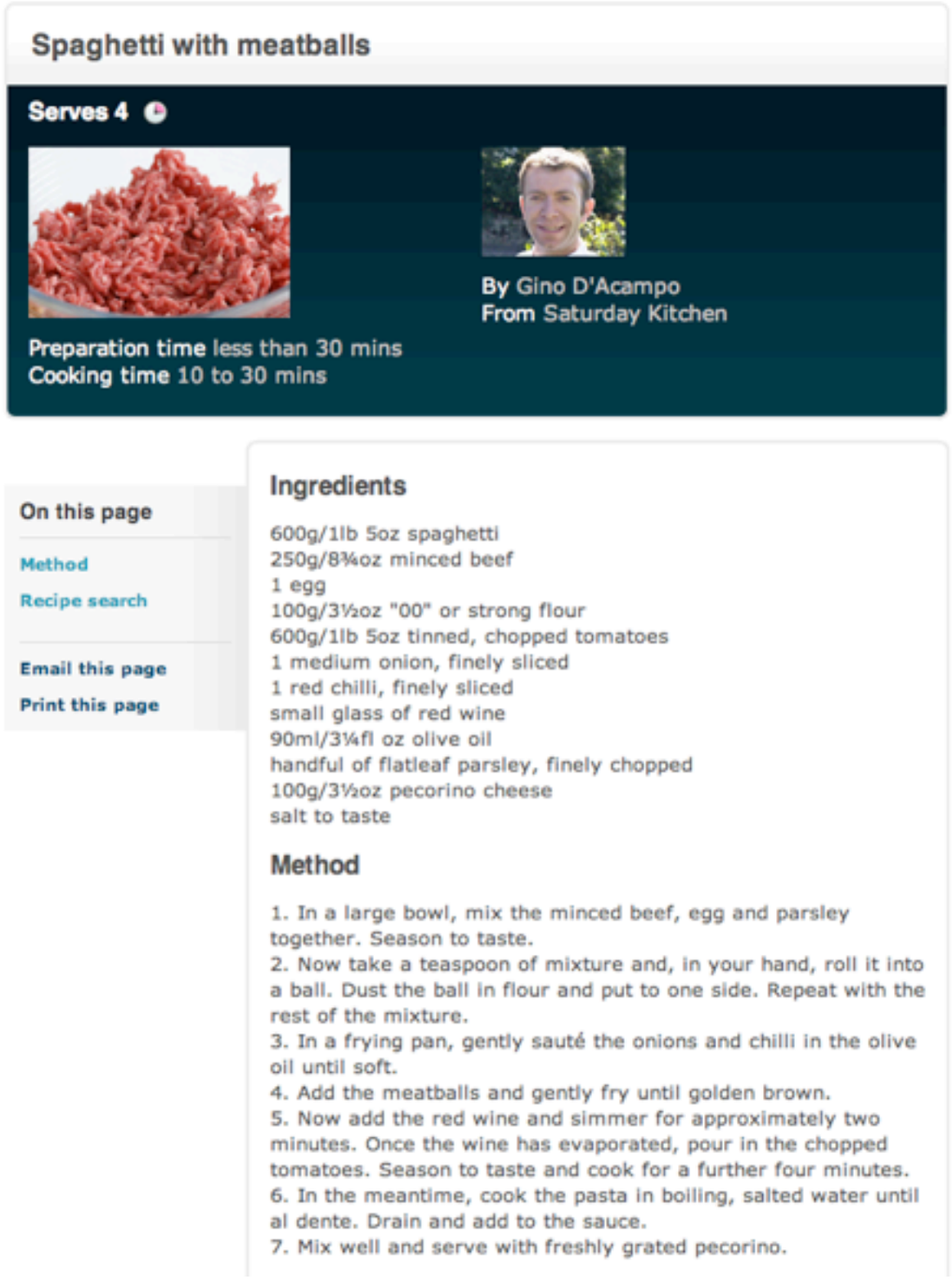
```
.ingredients p{
}
```

http://nwalsh.com/docs/tutorials/xsl/xsl/slides.html



http://www.inf.ed.ac.uk/teaching/courses/il1/practicals/2009-09-28/xslurl.xml

The basic idea is that we use xslt to extract data from xml. The url identifies a page where you can apply your xsl to an arbitrary web page (this will only work if the page you choose to scrape contains well-formed xml).



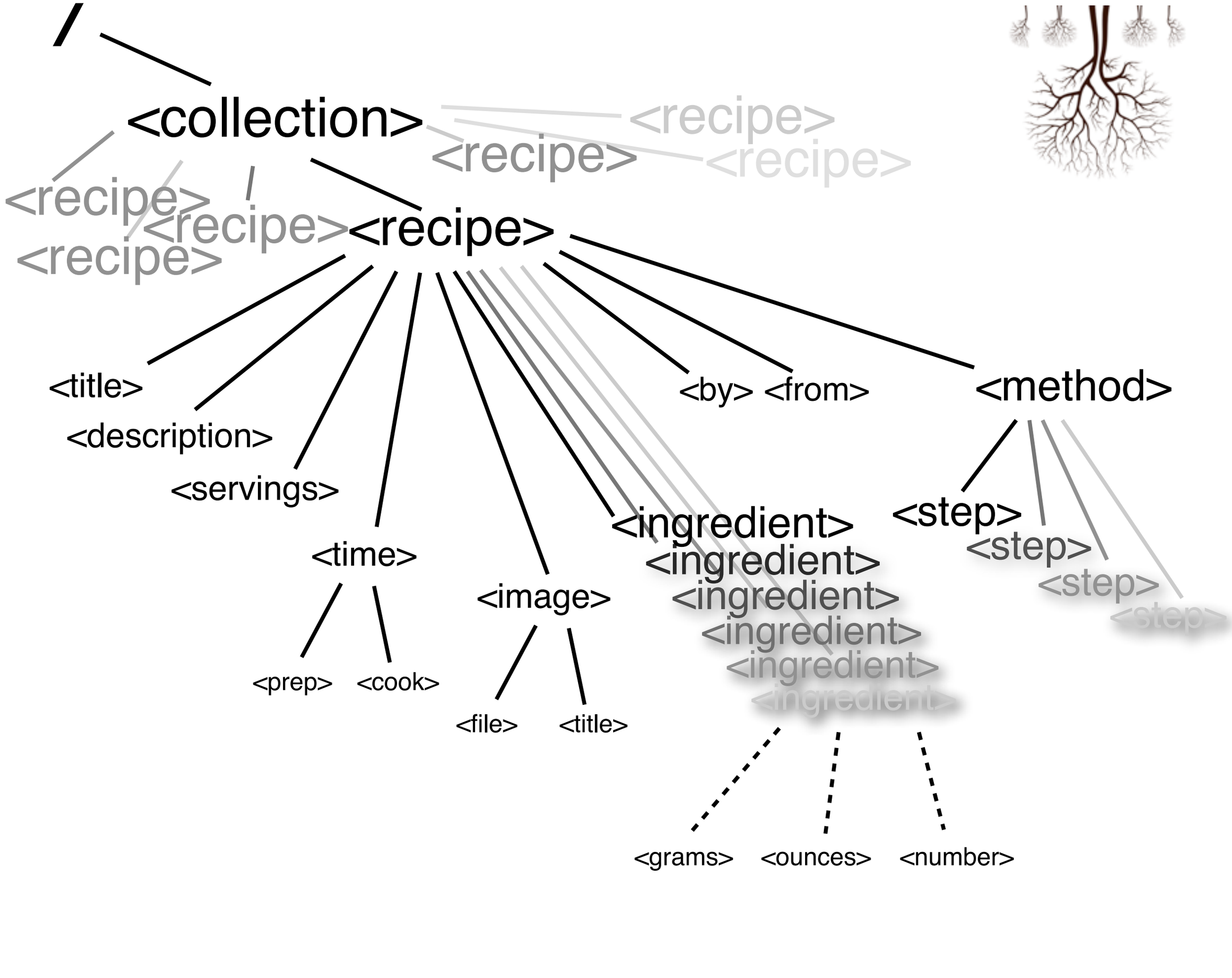
http://www.bbc.co.uk/food/recipes/database/spaghettiwithmeatbal_72227.shtml

```
<?xml version="1.0" encoding="utf-8"?>
<collection>
  <recipe>
    <title>Spaghetti with meatballs</title>
    <description>
       A classic vegetarian dish.
    </description>

    <ingredient>spaghetti
        <grams>600</grams><ounces>21</ounces>
    </ingredient>
    <ingredient>minced beef
        <grams>250</grams><ounces>8.75</ounces>
    </ingredient>
    <ingredient>egg<number>1</number></ingredient>
    ...
    <method>
      <step>
        In a large bowl, mix the minced beef,
        egg and parsley together. Season to taste.
      </step>
      <step>
         Now take a teaspoon of mixture and, in your
         hand, roll it into a ball. Dust the ball in
         flour and put to one side. Repeat with the
         rest of the mixture.
      </step>
      ...
    </method>
    <time>
      <prep>less than 30 mins</prep>
      <cook>10 to 30 mins</cook>
    </time>
    <servings>4</servings>

    <image>
      <file>1.jpg</file><title>Beef Mince</title>
    </image>
    <by>Gino D'Acampo</by>
    <from>Saturday Kitchen</from>
  </recipe>
  ...
</collection>
```


We represent the information in the recipe using XML



XML organises information in labelled trees. We draw them upside-down.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
     <title>IL recipes</title>
      </head>
      <body>


      </body>
    </html>
</xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
          <title>IL recipes</title>
      </head>
      <body>
          <section class="foreword">
            <h1>IL recipes</h1>
            <p>An exercise in the collection, processing and
                 communication of information.</p>
          </section>
          <xsl:for-each select="collection/recipe">
            <section class="recipe">
              <h1><xsl:value-of select="title" /></h1>
              <aside><xsl:value-of select="description" /></aside>
              <h2>Ingredients</h2>
              <ul>
                <xsl:for-each select="ingredient">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ul>
              <h2>Method</h2>
              <ol>
                <xsl:for-each select="method/step">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ol>
            </section>
          </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>IL recipes</title>
```


```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
```


```
            <h1><xsl:value-of select="title" /></h1>
            <aside><xsl:value-of select="description" /></aside>
            <h2>Ingredients</h2>
            <ul>
              <xsl:for-each select="ingredient">
                  <li><xsl:value-of select="." /></li>
              </xsl:for-each>
            </ul>
            <h2>Method</h2>
            <ol>
              <xsl:for-each select="method/step">
                  <li><xsl:value-of select="." /></li>
              </xsl:for-each>
            </ol>
          </section>
        </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>IL recipes</title>
      </head>
      <body>
        <section class="foreword">
          <h1>IL recipes</h1>
          <p>An exercise in the collection, proce
             communication of information.</p>
        </section>
        <xsl:for-each select="collection/recipe">
          <section class="recipe">
            <h1><xsl:value-of select="title" /></h1
            <aside><xsl:value-of select="descript
            <h2>Ingredients</h2>
            <ul>
              <xsl:for-each select="ingredient">
                <li><xsl:value-of select="." /></li>
              </xsl:for-each>
            </ul>
            <h2>Method</h2>
            <ol>
              <xsl:for-each select="method/step">
                <li><xsl:value-of select="." /></li>
              </xsl:for-each>
            </ol>
          </section>
        </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```
<xsl:template match="/">
  <html>
    <head>
      <title>IL recipes</title>
    </head>
    <body>
```

...

```
    </body>
  </html>
</xsl:template>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:te
    <htm
      <he
        <
      </he
      <bo
        <
          <section class="foreword">
            <h1>IL recipes</h1>
            <p>An exercise in the collection, processing and
                communication of information.</p>
          </section>
          <
          <
          <xsl:for-each select="collection/recipe">
            <section class="recipe">
              <h1><xsl:value-of select="title" /></h1>
              <aside><xsl:value-of select="description" /></aside>

                                ...

              </section>
          </xsl:for-each>
          <
      </bo
    </htm
  </xsl:te...plate>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/
  <xsl:template match="/">
    <html>
      <head>
        <title>IL recipes</title
      </head>
      <body>
        <section class="forew
          <h1>IL recipes</h1>
          <p>An exercise in th
              communication c
        </section>
        <xsl:for-each select="
          <section class="reci
            <h1><xsl:value-of s
            <aside><xsl:value-
            <h2>Ingredients</h
            <ul>
              <xsl:for-each sele
                <li><xsl:value-
              </xsl:for-each>
            </ul>
            <h2>Method</h2>
            <ol>
              <xsl:for-each sele
                <li><xsl:value-
              </xsl:for-each>
            </ol>
          </section>
        </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```
<h2>Ingredients</h2>
<ul>
    <xsl:for-each select="ingredient">
        <li><xsl:value-of select="." /></li>
    </xsl:for-each>
</ul>
<h2>Method</h2>
<ol>
    <xsl:for-each select="method/step">
        <li><xsl:value-of select="." /></li>
    </xsl:for-each>
</ol>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
          <title>IL recipes</title>
      </head>
      <body>
          <section class="foreword">
            <h1>IL recipes</h1>
            <p>An exercise in the collection, processing and
                 communication of information.</p>
          </section>
          <xsl:for-each select="collection/recipe">
            <section class="recipe">
              <h1><xsl:value-of select="title" /></h1>
              <aside><xsl:value-of select="description" /></aside>
              <h2>Ingredients</h2>
              <ul>
                <xsl:for-each select="ingredient">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ul>
              <h2>Method</h2>
              <ol>
                <xsl:for-each select="method/step">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ol>
            </section>
          </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
<collection>
  <recipe>
    <title>Spaghetti with meatballs</title>
    <description>
      A classic vegetarian dish.
    </description>

    <ingredient>spaghetti
       <grams>600</grams><ounces>21</ounces>
    </ingredient>
    <ingredient>minced beef
       <grams>250</grams><ounces>8.75</ounces>
    </ingredient>
    <ingredient>egg<number>1</number></ingredient>
    <method>
      <step>
        In a large bowl, mix the minced beef,
        egg and parsley together. Season to taste.
      </step>
      <step>
         Now take a teaspoon of mixture and, in your
         hand, roll it into a ball. Dust the ball in
         flour and put to one side. Repeat with the
         rest of the mixture.
      </step>
      ...
    </method>
    <time>
      <prep>less than 30 mins</prep>
      <cook>10 to 30 mins</cook>
    </time>
    <servings>4</servings>

    <image>
      <file>1.jpg</file><title>Beef Mince</title>
    </image>
    <by>Gino D'Acampo</by>
    <from>Saturday Kitchen</from>
  </recipe>
  ...
</collection>
```


XML represents the information.
XSL transforms it to a form suited for presentation.
CSS provides the styling.

```
<?xml version="1.0" encoding="utf-8"?>
<collection>
  <recipe>
    <title>Spaghetti with meatballs</title>
    <description>
      A classic vegetarian dish.
    </description>

    <ingredient>spaghetti
       <grams>600</grams><ounces>21</ounces>
    </ingredient>
    <ingredient>minced beef
       <grams>250</grams><ounces>8.75</ounces>
    </ingredient>
    <ingredient>egg<number>1</number></ingredient>
    <method>
      <step>
        In a large bowl, mix the minced beef,
        egg and parsley together. Season to taste.
      </step>
      <step>
         Now take a teaspoon of mixture and, in your
         hand, roll it into a ball. Dust the ball in
         flour and put to one side. Repeat with the
         rest of the mixture.
      </step>
      ...
    </method>
    <time>
      <prep>less than 30 mins</prep>
      <cook>10 to 30 mins</cook>
    </time>
    <servings>4</servings>

    <image>
      <file>1.jpg</file><title>Beef Mince</title>
    </image>
    <by>Gino D'Acampo</by>
    <from>Saturday Kitchen</from>
  </recipe>
  ...
</collection>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
          <title>IL recipes</title>
      </head>
      <body>
          <section class="foreword">
            <h1>IL recipes</h1>
            <p>An exercise in the collection, processing and
                communication of information.</p>
          </section>
          <xsl:for-each select="collection/recipe">
            <section class="recipe">
              <h1><xsl:value-of select="title" /></h1>
              <aside><xsl:value-of select="description" /></aside>
              <h2>Ingredients</h2>
              <ul>
                <xsl:for-each select="ingredient">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ul>
              <h2>Method</h2>
              <ol>
                <xsl:for-each select="method/step">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ol>
            </section>
          </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```


XML represents the information.
XSL transforms it to a form suited for presentation.
CSS provides the styling.

```
<?xml version="1.0" encoding="utf-8"?>
<collection>
  <recipe>
    <title>Spaghetti with meatballs</title>
    <description>
      A classic vegetarian dish.
    </description>

    <ingredient>spaghetti
       <grams>600</grams><ounces>21</ounces>
    </ingredient>
    <ingredient>minced beef
       <grams>250</grams><ounces>8.75</ounces>
    </ingredient>
    <ingredient>egg<number>1</number></ingredient>
    <method>
      <step>
        In a large bowl, mix the minced beef,
        egg and parsley together. Season to taste.
      </step>
      <step>
         Now take a teaspoon of mixture and, in your
         hand, roll it into a ball. Dust the ball in
         flour and put to one side. Repeat with the
         rest of the mixture.
      </step>
      ...
    </method>
    <time>
      <prep>less than 30 mins</prep>
      <cook>10 to 30 mins</cook>
    </time>
    <servings>4</servings>

    <image>
      <file>1.jpg</file><title>Beef Mince</title>
    </image>
    <by>Gino D'Acampo</by>
    <from>Saturday Kitchen</from>
  </recipe>
  ...
</collection>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
          <title>IL recipes</title>
      </head>
      <body>
          <section class="foreword">
            <h1>IL recipes</h1>
            <p>An exercise in the collection, processing and
                communication of information.</p>
          </section>
          <xsl:for-each select="collection/recipe">
            <section class="recipe">
              <h1><xsl:value-of select="title" /></h1>
              <aside><xsl:value-of select="description" /></aside>
              <h2>Ingredients</h2>
              <ul>
                <xsl:for-each select="ingredient">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ul>
              <h2>Method</h2>
              <ol>
                <xsl:for-each select="method/step">
                    <li><xsl:value-of select="." /></li>
                </xsl:for-each>
              </ol>
            </section>
          </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```
body{ font-family:fantasy; }

h1 { font-family:sans-serif; }

section.foreword {font-color:#888888;}

aside { width:30%;
        float:right;
        background-color:cyan;
}
```


XML represents the information.
XSL transforms it to a form suited for presentation.
CSS provides the styling.

Wine calories table:

Wine	Wine Calories 115ml	Large Glass
Alcohol-free Wine	37 calories	74 Cals
Champagne	96 calories	192 Cals
Dry Red Wine	83 calories	166 Cals
Dry White Wine	77 calories	154 Cals
Rose	82 calories	164 Cals
Sparkling	92 calories	184 Cals
Sweet Red Wine	100 calories	200 Cals
Sweet White Wine	103 calories	206 Cals
Fortified Wines	**Wine Calories**	**Large Glass**
Bianco Vermouth	167 calories	334 Cals
Ginger Wine	190 calories	380 Cals
Martini Bianco	150 calories	300 Cals
Martini Extra Dry	150 calories	300 Cals
Martini Rose	180 calories	360 Cals
Martini Rosso	192 calories	384 Cals
Port	170 calories	340 Cals
Sherry average	140 calories	280 Cals

All values correct at time of testing, values for wine calories may vary between different sized glasses!

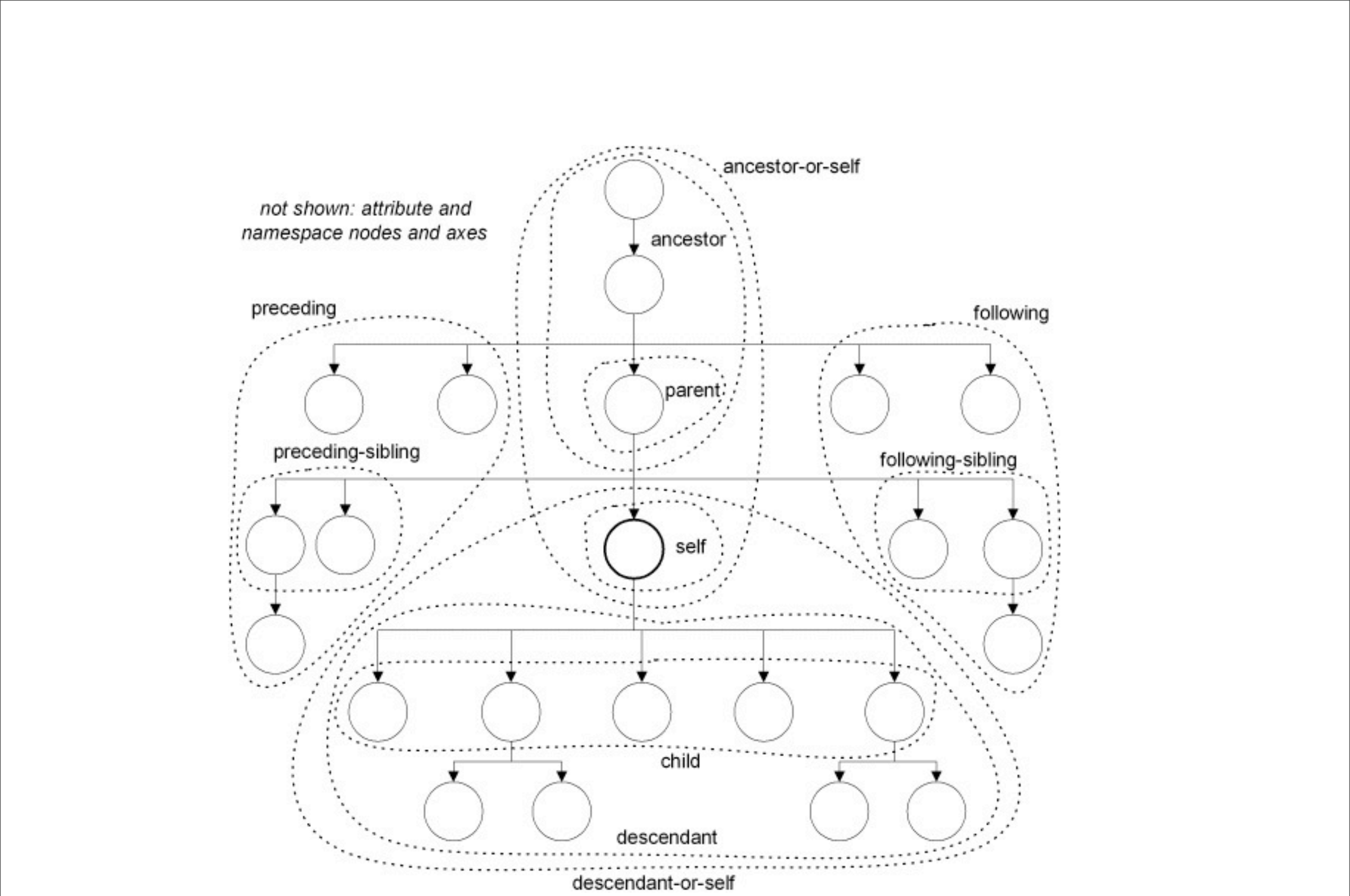
http://www.weightlossforall.com/

Here is an example

BREADS & CEREALS	Portion size *	per 100 grams (3.5 oz)	energy content
Bagel (1 average)	140 cals (45g)	310 cals	Medium
Biscuit digestives	86 cals (per biscuit)	480 cals	High
Jaffa cake	48 cals (per biscuit)	370 cals	Med-High
Bread white (thick slice)	96 cals (1 slice 40g)	240 cals	Medium
Bread wholemeal (thick)	88 cals (1 slice 40g)	220 cals	Low-med
Chapatis	250 cals	300 cals	Medium
Cornflakes	130 cals (35g)	370 cals	Med-High
Crackerbread	17 cals per slice	325 cals	Low Calories
Cream crackers	35 cals (per cracker)	440 cals	Low / portion
Crumpets	93 cals (per crumpet)	198 cals	Low-Med
Flapjacks basic fruit mix	320 cals	500 cals	High
Macaroni (boiled)	238 cals (250g)	95 cals	Low calorie
Muesli	195 cals (50g)	390 cals	Med-high
Naan bread (normal)	300 cals (small plate size)	320 cals	Medium
Noodles (boiled)	175 cals (250g)	70 cals	Low calorie
Pasta (normal boiled)	330 cals (300g)	110 cals	Low calorie
Pasta (wholemeal boiled)	315 cals (300g)	105 cals	Low calorie
Porridge oats (with water)	193 cals (350g)	55 cals	Low calorie
Potatoes** (boiled)	210 cals (300g)	70 cals	Low calorie
Potatoes** (roast)	420 cals (300g)	140 cals	Medium
Rice (white boiled)	420 cals (300g)	140 cals	Low calorie
Rice (egg-fried)	500 cals	200 cals	High in portion
Rice (Brown)	405 cals (300g)	135 cals	Low calorie
Rice cakes	28 Cals = 1 slice	373 Cals	Medium
Ryvita Multi grain	37 Cals per slice	331 Cals	Medium
Ryvita + seed & Oats	180 Cals 4 slices	362 Cals	Medium
Spaghetti (boiled)	303 cals (300g)	101 cals	Low calorie

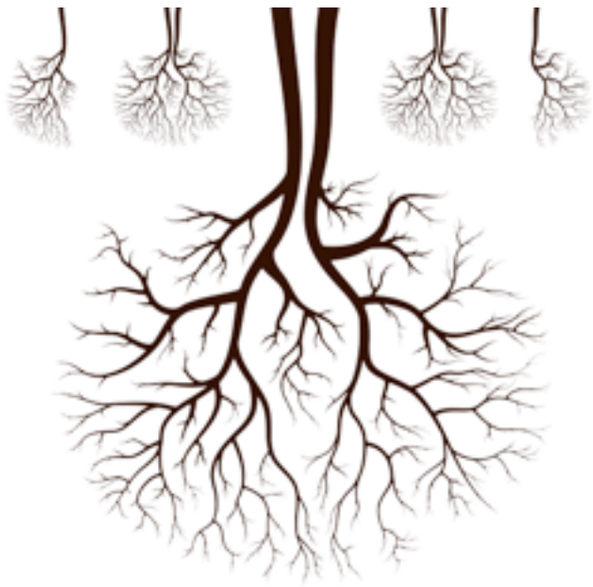
Tuesday, 6 December 11

and another



In the example, we used xsl to describe a simple path from the root (/) to the table rows we are interested in. XSL provides patterns for accessing various parts of a tree starting from the current node (self).

building
trees



```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html"/>

    <xsl:template match="/">
      <xsl:apply-templates select="html/body/table[3]//table/tr"/>
      <hr />
    </xsl:template>

</xsl:stylesheet>
```


XSL lets us construct new trees out of old trees.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:apply-templates select="html/body/table[3]//table/tr"/>
    <hr />
  </xsl:template>

    <xsl:template match="tr">
        <br />
        <xsl:value-of select="td[1]"/> :: <xsl:value-of select="td[3]"/>
    </xsl:template>

</xsl:stylesheet>
```


Once we have identified the rows we are interested in, we add a template that specifies how we will process each row.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:apply-templates select="html/body/table[3]//table/tr"/>
     <hr />
</xsl:template>

  <xsl:template match="tr">
    <xsl:variable name="food" select="string(td[1])" />
    <xsl:variable name="calories" select="string(td[3])" />
    <br />
    <xsl:value-of select="$food"/>:: <xsl:value-of select="$calories"/>
  </xsl:template>

</xsl:stylesheet>
```


Using variables, we can give the table data items we have selected informative names - this will make it easier to understand our code and modify it later.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:apply-templates select="html/body/table[3]//table/tr"/>
     <hr />
</xsl:template>

  <xsl:template match="tr">
     <xsl:variable name="tag" select= "'b'"/>
     <xsl:variable name="food" select="string(td[1])" />
     <xsl:variable name="calories" select="string(td[3])" />

     <xsl:if test="contains($food, $tag)">
       <br /><xsl:value-of select="$food"/> :: <xsl:value-of select="$calories"/>
     </xsl:if >
  </xsl:template>

</xsl:stylesheet>
```


To select only some rows for processing, use xsl:if

Here we will only process rows where the food name contains the tag 'b'. To provide this

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:apply-templates select="html/body/table[3]//table/tr" />
    <hr />
  </xsl:template>

  <xsl:template match="tr">

    <xsl:param name="tag" select= "'b'"/>
    <xsl:variable name="food" select="string(td[1])" />
    <xsl:variable name="calories" select="string(td[3])" />

    <xsl:if test="contains($food, $tag)">
      <br /><xsl:value-of select="$food"/> :: <xsl:value-of select="$calories"/>
    </xsl:if >

  </xsl:template>

</xsl:stylesheet>
```


Here, the template for processing a table row (tr) declares tag as a parameter, which is used to select only some of the rows for processing. We can use this template just as we did before, but now 'b' is just a default value for the parameter.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:variable name="myTag" select="'a'" />
    <xsl:if test="$myTag != ''">
      <h2>Foods containing "<xsl:value-of select="$myTag" />"</h2>
    </xsl:if>
    <xsl:apply-templates select="html/body/table[3]//table/tr">
      <xsl:with-param name="tag" select="$myTag" />
    </xsl:apply-templates>
    <hr />
  </xsl:template>

  <xsl:template match="tr">

    <xsl:param name="tag" select="'b'"/>
    <xsl:variable name="food" select="string(td[1])" />
    <xsl:variable name="calories" select="string(td[3])" />

    <xsl:if test="contains($food, $tag)">
      <br /><xsl:value-of select="$food"/> :: <xsl:value-of select="$calories"/>
    </xsl:if >

  </xsl:template>

</xsl:stylesheet>
```


To supply a different value for the parameter we use xsl:with-param when the template is called (e.g. by apply-templates).

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:apply-templates select="html/body/table[3]//table/tr"/>
     <hr />
</xsl:template>

  <xsl:template match="tr">
     <xsl:variable name="tag" />
     <xsl:variable name="food" select="string(td[1])" />
     <xsl:variable name="calories" select="string(td[3])" />

     <xsl:if test="contains($food, $tag)">
       <br /><xsl:value-of select="$food"/> :: <xsl:value-of select="$calories"/>
     </xsl:if >
  </xsl:template>

</xsl:stylesheet>
```


If no default is specified, and no value is specified by xsl:apply-templates the default default is the empty string.