

Volume

3

FAMILY FEATURES EDITORIAL SYNDICATE
eSyndicate

XMLFEED Development Guide

Family Features Editorial Syndicate eSyndicate

XML Feed Development Guide

Family Features Editorial Syndicate
5825 Dearborn St.
Mission, KS 66202
Contact: Media Communications
Phone: (888) 824-3337
Email: Support@familyfeatures.com

Written by
Galen Blakeman
Modified by
Cody Inman
Blue Ocean Consulting
3320 Mesa Way, Suite B
Lawrence, KS 66044
Phone 785.842.5154
Fax 785.331.4696

Contents

CONTENTS	3
CHAPTER 1 :: OVERVIEW	4
PRODUCT	4
ATTRIBUTION.....	4
AUDIENCE.....	4
SYSTEM.....	4
CHAPTER 2 :: XML SCHEMA	6
FOOD: WEEKLY FEATURES SCHEMA	6
FOOD: WEEKLY RECIPES SCHEMA	6
FOOD: FEATURE SCHEMA	6
FOOD: RECIPE SCHEMA.....	7
NON FOOD: WEEKLY ARTICLES.....	7
NON FOOD: ARTICLE INFORMATION	7
LINKS: CATEGORIES LINKS	8
LINKS: SUB CATEGORIES LINKS.....	8
GENERAL: FORMAT SCHEMA	8
GENERAL: IMAGE SCHEMA	9
GENERAL: SECTION INFORMATION	9
GENERAL: LINK SCHEMA	9
GENERAL: SPONSOR SCHEMA	9
GENERAL: SUB-CATEGORY SCHEMA	9
GENERAL: COURSE SCHEMA	10
GENERAL: WEEK SCHEMA	10
CHAPTER 3 :: XML FEEDS	11
FOOD: WEEKLY FEATURE XML DOCUMENT	11
FOOD: FEATURE XML DOCUMENT	12
FOOD: WEEKLY RECIPE XML DOCUMENT	13
NON FOOD: WEEKLY ARTICLES XML DOCUMENT.....	14
NON FOOD: ARTICLE XML DOCUMENT	15
LINKS: CATEGORY LINKS XML DOCUMENT.....	15
CHAPTER 4 :: XSLT	17
FOOD: WEEKLY FEATURES TRANSFORMATION EXAMPLE	17
FOOD: WEEKLY RECIPES TRANSFORMATION EXAMPLE.....	19
FOOD: FEATURE TRANSFORMATION EXAMPLE	20
NON FOOD: WEEKLY ARTICLES TRANSFORMATION	20
NON FOOD: ARTICLE TRANSFORMATION	21
LINKS: CATEGORY LINKS TRANSFORMATION.....	22
APPENDIX A :: RESOURCES	23
BOOKS	23
URL'S	23

Chapter 1 :: Overview

Product

Family Features Editorial Syndicate provides weekly updated features, recipes, articles, tips and links as XML sources for incorporation into your Web site. Our eSyndicate XML feed delivers high quality, color editorial features showcasing products and services from America's favorite companies, associations and their agencies. These eye-catching themed features are used by print and online editors on the front page of their food and lifestyle sections to build readership and add value to their publications and web sites.

This page provides a summary of XML feeds, schemas, XSLT examples and resources.

Attribution

To use Family Features' eSyndicate XML Feed we require attribution as follows:

eSyndicate

A service of Family Features

Audience

This guide is written for technical developers that are integrating Family Features XML feeds into their Web sites. A general understanding of HTML, XML, and XSLT is assumed. Entire books are written on each of these subjects, so if you would like more general information please see the references section.

System

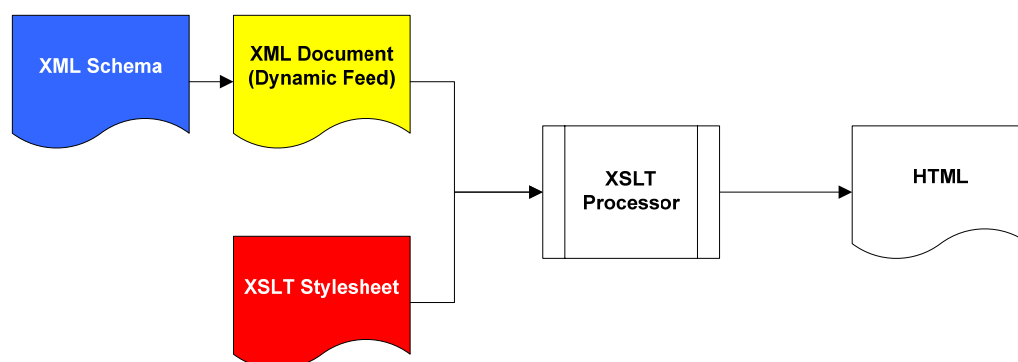


Figure 1 – XML + XSLT Example

The eSyndicate Feed is defined by the XML Schema as (in blue) as shown above. The XML Schema spells out the exact data structure of the XML feed and the relationships between data components. The XML documents (shown in yellow) are Web data sources that provide features and recipes in XML format. These XML documents validate to defined XML schemas (shown in blue). The XSLT style sheet (shown in red) defines how to transform the XML into another format. For this example we are transforming the content into HTML. The XSLT processor is software that transforms the XML document using the XSLT style sheet. The XSLT processor can be part of your dynamic web page code or built right into a browser like Microsoft Internet Explorer 6.

Chapter 2 :: XML Schema

XML Schema's are similar to document type definitions (DTD). Like DTD's Schema's define the fields and structure, but they also provide data typing and hierarchal data types. For example you can define an element to be constrained to numeric values or as a pointer to another schema. Theses XML schemas are often compared to relational database schemas. This chapter does not attempt to explain XML Schema syntax, it gives relevant schemas for XML documents provided by FFES feed. Please see references section for more information on XML schemas.

The following sections provide the schema's for all of the XML feed data sources. These schemas are hierarchal by design. For example the recipe schema builds upon the format and image schemas.

Food: Weekly Features Schema

<http://feed2.familyfeatures.com/services/xsd/weeklyfeatures.xsd>

The weekly features consist of an unbounded sequence of features. As shown in the schema the feature schema is pulled in using `xs:include` statement in second line. The root element of this document is `<weeklyfeatures>` and within is a series of `<feature>` elements, `<subcategory>` elements, `<course>` elements, and `<week>` elements.

Food: Weekly Recipes Schema

<http://feed2.familyfeatures.com/services/xsd/weeklyrecipes.xsd>

Similar to the weekly features the weekly recipes are an unbounded sequence of recipes. As shown in the schema the recipe schema is pulled in using `xs:include` statement in second line. The root element of this document is `<weeklyrecipes>` and within is a series of `<recipe>` elements and `<week>` elements.

Food: Feature Schema

<http://feed2.familyfeatures.com/services/xsd/feature.xsd>

A feature pulls together recipes, sponsors, images, and formatting into a full page color feature. Because of its composite nature this scheme is quite complex and best shown graphically, as follows:

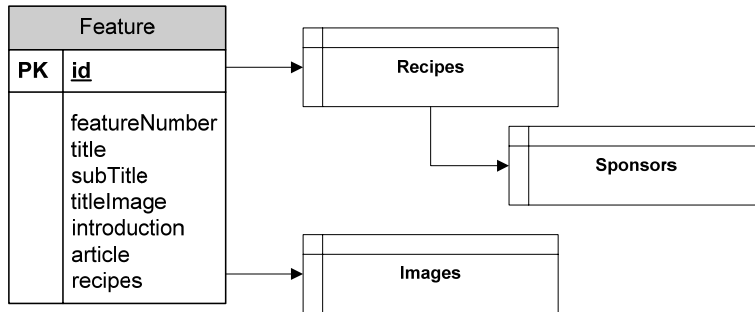


Figure 2 – Feature Schema and relationships

To keep it simple this diagram doesn't reflect that title, subtitle, introduction, and article are of type html text. See format scheme below on what html formatting is passed.

Food: Recipe Schema

<http://feed2.familyfeatures.com/services/xsd/recipe.xsd>

The recipe schema consists of a number of html fields and a sequence of sponsors. The schema is diagramed below:

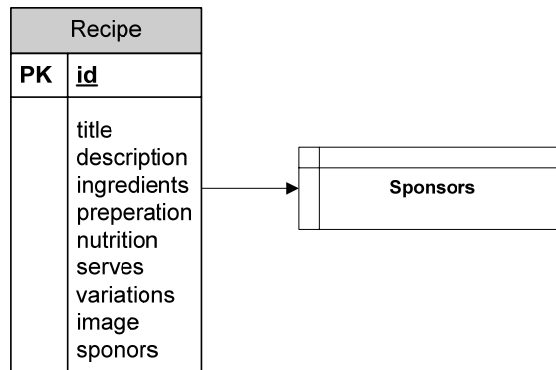


Figure 3 – Recipe Schema and relationships

Non Food: Weekly Articles

<http://feed2.familyfeatures.com/services/xsd/weeklyarticles.xsd>

Weekly articles consist of an unbounded sequence of articles. As shown in the schema the article schema is pulled in using xs:include statement in second line. The root element of this document is <weeklyarticles> and within is a series of <article> elements, <subcategory> elements, and <week> elements.

Non Food: Article Information

<http://feed2.familyfeatures.com/services/xsd/article.xsd>

An article schema pulls together sections and links into a full page color article. The sections and links are unbounded sequences, which are described below. Because of its composite nature this scheme is complex and is shown graphically, as follows:

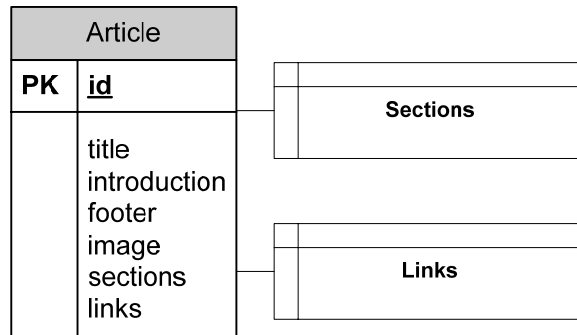


Figure 4 – Article Schema and relationships

Links: Categories Links

<http://feed2.familyfeatures.com/services/xsd/categorylinks.xsd>

Categories links consists of category name and an unbounded sequence of subcategories. The root element of this document is <categorylinks> and within is a series of <subcategories> elements, which in turn contain sub category links. The schema is diagrammed below:

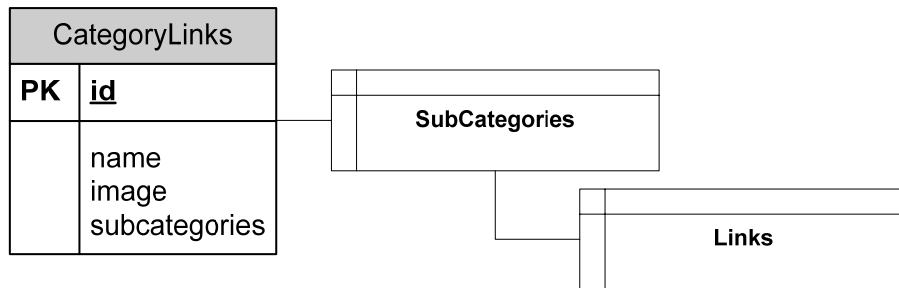


Figure 5 – Category Links Schema and relationships

Links: Sub Categories Links

<http://feed2.familyfeatures.com/services/xsd/subcategorylinks.xsd>

Subcategories links consists of sub category name and an unbounded sequence of links. The root element of this document is <subcategorylinks> and within is a series of <link> elements.

General: Format Schema

<http://feed2.familyfeatures.com/services/xsd/format.xsd>

This schema sets pattern for supporting basic HTML formatting. The following HTML tags are supported:

B	STRONG	STRIKE	I	U
P	BR	UL	OL	LI
A	FONT	DIV	SPAN	BLOCKQUOTE

Table 7 – HTML Text tags

General: Image Schema

<http://feed2.familyfeatures.com/services/xsd/image.xsd>

This schema provides supporting for all the HTML image attributes. Any standard HTML reference can detail these attributes. All you need to worry about with the FFES feeds are “src” and “alt” attributes.

General: Section Information

<http://feed2.familyfeatures.com/services/xsd/section.xsd>

Sections are the building blocks of articles and features and contain HTML formatted text, and image, and title.

General: Link Schema

<http://feed2.familyfeatures.com/services/xsd/link.xsd>

This schema provides support for name and URL of hyper links.

General: Sponsor Schema

<http://feed2.familyfeatures.com/services/xsd/sponsor.xsd>

The sponsor schema represents the sponsor of a article, feature, or recipe. The schema is diagramed below:

Sponsor	
PK	<u>id</u>
	name url

Figure 6 – Sponsor Schema

General: Sub-Category Schema

<http://feed2.familyfeatures.com/services/xsd/subcategory.xsd>

This schema provides support for name of subcategories.

General: Course Schema

<http://feed2.familyfeatures.com/services/xsd/course.xsd>

This schema provides support for name of courses.

General: Week Schema

<http://feed2.familyfeatures.com/services/xsd/week.xsd>

This schema provides support for start date and end date of archived weeks.

Chapter 3 :: XML FEEDS

Food: Weekly Feature XML Document

<http://feed2.familyfeatures.com/service/weeklyfeatures.ashx?xmlOnly=true>

This weekly feature feed returns a XML document containing a sequence of features. Each feature will contain unique key id attribute for linking through to feature detail. Summary information is included in the feed for feature number, title, sub-title, image of title, introduction, and feature image. The following is a simple example of a weekly features XML document.

```
<weeklyfeatures>
  <feature id="1">
    <featureNumber>00111</featureNumber>
    <title>Title 1</title>
    <subTitle>Sub title 1</subTitle>
    <titleImage src="http://titleimageurl"/>
    <introduction>Introduction Text 1</introduction>
    <introductionImage src="http://titleimageurl"/>
  </feature>
  <feature id="2">
    <featureNumber>00222</featureNumber>
    <title>Title 2</title>
    <subTitle>Sub title 2</subTitle>
    <titleImage src="http://titleimageurl"/>
    <introduction>Introduction Text 2</introduction>
    <introductionImage src="http://introductionimageurl"/>
  </feature>
</weeklyfeatures >
<subcategory id="1">
  <subCategoryName>Sub-Category</subCategoryName>
</subcategory>
<course id="1">
  <courseName>Course</courseName>
</course>
<week id="1">
  <startDate>1/1/2008</startDate>
  <stopDate>1/5/2008</stopDate>
</week>
```

Figure 8 – Weekly Feature XML Document Example

Food: Feature XML Document

<http://feed2.familyfeatures.com/service/feature.ashx?FeatureId=1530&xmlOnly=true>

Using the unique feature id from the weekly feature XML document the detailed feature XML document can be accessed. For example if the feature id in the weekly XML document was 1 you would call the above URL replacing 784 with 1. The following is an example of a feature XML document. For simplicity this example shows a single recipe associated with the feature, typically their will be multiples. Similarly there can be multiple sponsors per recipe.

```
<feature id="1">
  <featureNumber>00111</featureNumber>
  <title>Title 1</title>
  <subTitle>Sub title 1</subTitle>
  <titleImage src="http://titleimageurl"/>
  <introduction>Introduction Text 1</introduction>
  <introductionImage src="http://introductionimageurl"/>
  <section id="1">
    <title>Title 1</title>
    <content> Text ...</content>
    <image src="http://sectionimageurl" />
  </section>
  <section id="2">
    <title>Title 2</title>
    <content> Text ...</content>
    <image src="http://sectionimageurl" />
  </section>
  <link id="1">
    <name>Link Name</name>
    <url>http://linkurl</url>
  </link>
  <recipe id="1">
    <title>Recipe Title</title>
  </recipe>
  <sponsor id="1">
    <name>Sponsor Name</name>
    <url>http://sponsorurl</url>
  </sponsor>
</feature>
```

Figure 9 –Feature XML Document Example

Food: Weekly Recipe XML Document

<http://feed2.familyfeatures.com/service/weeklyrecipes.ashx?xmlOnly=true>

The weekly recipe feed is a sequence of recipes. Unlike the weekly features feed which is separated into weekly and detail documents the recipe feed consists of single document. The following is a simple example of a weekly features XML document.

```
<weeklyrecipes>
  <recipe id="1">
    <title>Recipe Title 1</title>
    <description>Recipe Description 1</description>
    <ingredients>Text...</ingredients>
    <preparation>Text...</preparation>
    <nutrition>Text...</nutrition>
    <serves>Text...</serves>
    <variations>Text...</variations>
    <calories>Text...</ calories >
    <totalFat>Text...</totalFat>
    <cholesterol>Text...</cholesterol>
    <sodium>Text...</sodium>
    <carbs>Text...</carbs>
    <protien>Text...</protien>
    <descriptionImage src="http://descriptionimageurl"/>
    <ingredientsImage src="http:// ingredientsimageurl"/>
    <preparationImage src="http:// preparationimageurl"/>
    <nutritionImage src="http:// nutritionimageurl"/>
    <servesImage src="http:// servesimageurl"/>
    <variationsImage src="http:// variationsimageurl"/>
    <course id="1">
      <title>Course 1</name>
    </course>
    <sponsor id="1">
      <name>Sponsor 1</name>
      <URL>mailto:yyy@yyy.com</URL>
    </sponsor>
    <content id="1">
      <title>Content 1</name>
    </content>
    <related id="1">
      <title>Recipe 1</name>
    </related>
  </recipe >
  <recipe id="2">
    <title>Recipe Title 2</title>
    <description>Recipe Description 2</description>
```

```

<ingredients>Text...</ingredients>
<preparation> Text...</preparation>
<nutrition> Text...</nutrition>
<serves> Text...</serves>
<variations> Text...</variations>
<calories> Text...</ calories >
<totalFat> Text...</totalFat>
<cholesterol> Text...</cholesterol>
<sodium> Text...</sodium>
<carbs> Text...</carbs>
<protien> Text...</protien>
<descriptionImage src="http://descriptionimageurl"/>
<ingredientsImage src="http:// ingredientsimageurl"/>
<preparationImage src="http:// preparationimageurl"/>
<nutritionImage src="http:// nutritionimageurl"/>
<servesImage src="http:// servesimageurl"/>
<variationsImage src="http:// variationsimageurl"/>
<course id="2">
    <title>Course 2</name>
</course>
<sponsor id="2">
    <name>Sponsor 2</name>
    <URL>mailto:yyy@yyy.com</URL>
</sponsor>
<content id="2">
    <title>Content 2</name>
</content>
<related id="1">
    <title>Recipe 2</name>
</related>
</recipe >
</ weeklyrecipes>

```

Figure 10 – Weekly Recipe XML Document Example

Non Food: Weekly Articles XML Document

```

<weeklyarticles>
    <categoryCode>CC</categoryCode>
    <categoryName>Category 1</categoryName>
    <image src="http://categoryimageurl"/>
    <article id="1">
        <contentNumber>00001</contentNumber>
        <title>Article Title 1</title>
        <introduction>Text...</introduction>
        < introduction Image src="http://articleimageurl"/>
    </article>
    <article id="2">

```

```

        <contentNumber>00002</contentNumber>
        <title>Article Title 2</title>
        <introduction>Text...</introduction>
        < introduction Image src="http://articleimageurl"/>
    </article>
    <subcategory id="1">
        <subCategoryName>Sub-Category Name</subCategoryName>
    </subcategory>
    <week id="1">
        <startDate>1/1/2008</startDate>
        <stopDate>1/5/2008</stopDate>
    </week>
</weeklyarticles>

```

Figure 11 – Weekly Weekly Articles XML Document Example

Non Food: Article XML Document

```

<article id="1 ">
    <contentNumber>F0002</contentNumber>
    <title>Article Title 1</title>
    <introduction>Text...</introduction>
    < introduction Image src="http://articleimageurl"/>
    <section id="1">
        <title>Section Title 1</title>
        <content>Text..</content>
        <image src="http://sectionimageurl"/>
    </section>
    <section id="2">
        <title>Section Title 2</title>
        <content>Text..</content>
        <image src="http://sectionimageurl"/>
    </section>
    <link id="1">
        <name>Link Name 1</name>
        <url>http://www.1boc.com</url>
    </link>
    <sponsor id="1">
        <name>Sponsor Name 1</name>
        <url> http://www.1boc.com </url>
    </sponsor>
</article>

```

Figure 12 – Article XML Document Example

Links: Category Links XML Document

```

<categorylinks>
    <name>Category Name</name>
    <image src="http://categoryimageurl "/>

```

```
<subcategory id="22">
  <name>Associations</name>
  <link id="1">
    <name>Link Name 1</name>
    <url>http://www.athenix.com</url>
  </link>
  <link id="2">
    <name> Link Name 2</name>
    <url> http://www.athenix.com </url>
  </link>
</subcategory>
</categorylinks>
```

Figure 13 – Category Links XML Document Example

Chapter 4 :: XSLT

Extensible Stylesheet Language Transformations (XSLT) is a syntax that is explicitly designed to transform XML into some other format. XSLT isn't required to use the FFES XML feeds; however they are the recommended method for transforming the XML feeds. For the examples in this chapter we will be transforming into HTML. This chapter does not attempt to explain XSLT syntax, it gives simple transformations of the XML documents provided by FFES feed. Please see references section for more information on XSLT.

Food: Weekly Features Transformation Example

<http://feed2.familyfeatures.com/services/xsl/weeklyfeatures.xml>

This XSLT stylesheet takes the weekly features XML document and converts it to a HTML document that places features in a 440 width table. Each feature is displayed with feature title and introduction placed in separate rows in table. The following breaks down the key sections of transformation:

```
<xsl:template match="/">
  <html>
  <head/>
  <body>
    <table width="440" border="0" cellpadding="8" cellspacing="0"
    bgcolor="#ffffff">
```

`<xsl:template match="/">` Says match every element in XML document. The next few lines are standard HTML that starts the page and table. Next:

```
<xsl:for-each select="weeklyfeatures">
  <xsl:for-each select="feature">
    <xsl:variable name="id" select="@id"/>
    <xsl:variable name="title" select="title"/>
```

`xsl:for-each` tells the processor to process the nested tags for each of the elements found in XML document. The `<weeklyfeatures>` element is the root element, so in reality this code is only executed once. There will be multiple `<feature>` elements, so this code will be looped over for each feature. The first step we take for each feature is to create variables for `id` and `title`. This is used to feed XML data into HTML tag attributes. Next:

```
<tr><td colspan="2">
```

```
<br /></td></tr>
<tr>
```

This code adds a row to the table with line image in first row and then opens HTML table row. Next:

```
<td>
<a href="feature.jsp?featureId={ $id }">
<xsl:for-each select="titleImage">
  <xsl:for-each select="@src">
    <img vspace="5" hspace="0" border="0" alt="">
      <xsl:attribute name="src"><xsl:value-of select="." /></xsl:attribute>
      <xsl:attribute name="alt"><xsl:value-of select="../title" /></xsl:attribute>
    </img>
  </xsl:for-each>
</xsl:for-each>
</a>
</td>
```

Using standard HTML TD we add a table column. The information in the column is surrounded by a hyper link. The source to this hyper link is a reference to the feature XML data feed. Notice that we use the { \$id } variable created earlier to feed the specific feature into the feature XML data request. After building the “href” we select the titleImage, which contains an “src” attribute. Within the “for-each select” tags for the title image we create the HTML image tag. The “xsl:attribute” tags within the HTML image tag lookup to the parent image tag and populate the “src” and “alt” attributes. Next:

```
<td>
<a href="feature.jsp?featureId={ $id }">
<xsl:for-each select="introductionImage">
  <xsl:for-each select="@src">
    <img align="right" vspace="5" hspace="8" border="1" alt="">
      <xsl:attribute name="src"><xsl:value-of select="." /></xsl:attribute>
      <xsl:attribute name="alt"><xsl:value-of select="../title" /></xsl:attribute>
    </img>
  </xsl:for-each>
</xsl:for-each>
</a>
</td>
</tr>
```

This code feeds in a table column with the feature image. This column works just like the above column that feed in the feature title image. Note that we close out the table row opened at the very beginning. Next:

```
<tr>
<td colspan="2">
<xsl:value-of select="introduction/text()" disable-output-escaping="yes"/>&#160;
```

```

<a href="feature.jsp?featureId={ $id }">View</a>
</td>
</tr>

```

As you can see this code is quite a bit simpler. It creates a HTML row and it places the introduction within a column that spans the two columns above. We use "<xsl:value-of select='introduction/text()' disable-output-escaping='yes'/>" to place the introduction in the column, but also to apply basic html formatting. We also create a "View" link that goes to feature detail. Next:

```

</xsl:for-each>
</xsl:for-each>
</table>
</body>
</html>

```

The first for-each is the end of the loop on features; the second is for the weekly features. The rest of the code is HTML that closes out table, body, and HTML.

Food: Weekly Recipes Transformation Example

<http://feed2.familyfeatures.com/services/xsl/weeklyrecipes.xml>

This XSLT stylesheet takes the weekly recipes XML document and converts it to a HTML document that places features in a 440 width table. The core concept of this transformation is the same as the weekly feature transformation example. The following sections of the transformation highlight what is different:

```

<xsl:for-each select=" weeklyrecipes">
  <xsl:for-each select="recipe">
    <a>
      <xsl:attribute name="href">#<xsl:value-of select="@id"/></xsl:attribute>
      <xsl:for-each select="title"><xsl:apply-templates/></xsl:for-each>
    </a><br/>
  </xsl:for-each>
</xsl:for-each>

```

This code creates a series of in page anchors at top of the page to recipes displayed after the feature. This allows the user to jump down to the recipes. The "for-each" tags loop over the recipes and the nested tags create anchors using the recipe titles and recipe ids.

```

<xsl:for-each select="descriptionImage">
  <xsl:if test="string-length(@src)>0">
    <xsl:for-each select="@src">
      <img align="right" vspace="5" hspace="8" border="1">
        <xsl:attribute name="src">
          <xsl:value-of select="." />
        </xsl:attribute>

```

```

        <xsl:attribute name="alt">
            <xsl:value-of select="../alt" />
        </xsl:attribute>
    </img>
</xsl:for-each>
</xsl:if>
</xsl:for-each>

```

This code shows an example of including a recipe description image, but only if the XML document contains a “src” attribute that has a string length greater than zero. The tag that executes this logic is the “<xsl:if test="string-length(@src)>0">”.

Food: Feature Transformation Example

<http://feed2.familyfeatures.com/services/xsl/feature.xml>

This XSLT stylesheet takes the feature XML document and converts it to a HTML document that places features in a 340 width table. The core concept of this transformation is the same as the weekly feature transformation example. The following sections of the transformation highlight what is different:

```

<head>
<xsl:for-each select="feature">
    <xsl:for-each select="title">
        <title>
            <xsl:apply-templates select="text()" />
        </title>
    </xsl:for-each>
</xsl:for-each>
</head>

```

The above example demonstrates using the feature title to set the HTML page title. “<xsl:apply-templates select="text()" />” grabs the text for feature title and inserts it. We use this tag instead of “<xsl:apply-templates/>” because we don’t need to apply formatting.

```

<xsl:if test="subTitle/text()">
    <xsl:for-each select="subTitle">
        <b><xsl:apply-templates/></b><br/>
    </xsl:for-each>
</xsl:if>

```

The above example shows using a “<xsl:if>” to see if subtitle has text. If it does exist it is included and break is added.

Non Food: Weekly Articles Transformation

<http://feed2.familyfeatures.com/services/xsl/weeklyarticles.xml>

This XSLT stylesheet takes the weekly articles XML document and converts into a HTML document that places article introductions within 440 width table. The core concept of this transformation is the same as the weekly features and recipes transformation examples. The following sections of the transformation highlight what is different:

```

<xsl:variable name="categoryCode" select="categoryCode"/>
<tr><td colspan="2">
<xsl:for-each select="image">
  <xsl:if test="string-length(@src)>0">
    <xsl:for-each select="@src">
      <img align="right" vspace="2" hspace="2" border="1">
        <xsl:attribute name="src"><xsl:value-of select="." /></xsl:attribute>
        <xsl:attribute name="alt"><xsl:value-of select="../alt" /></xsl:attribute>
      </img>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>
<b><font size="4">
<xsl:for-each select="categoryName"><xsl:apply-templates/></xsl:for-each> Resources
</font></b><br/>
<a href="categorylinks.jsp?category={ $categoryCode }">Helpful Links</a>
</td></tr>

```

The above example creates a row in HTML table for helpful links and populates it with category image and links HREF.

Non Food: Article Transformation

<http://feed2.familyfeatures.com/services/xsl/article.xml>

This XSLT stylesheet takes the article XML document and converts into a HTML document that places article within 450 width table. The core concept of this transformation is the same as the feature transformation example. The following sections of the transformation highlight what is different:

```

<xsl:for-each select="sections">
  <xsl:for-each select="section">
    <tr><td>
      <a>
        <xsl:attribute name="name"><xsl:value-of select="@id"/></xsl:attribute>
      </a>
      <xsl:for-each select="image">
        <xsl:if test="string-length(@src)>0">
          <xsl:for-each select="@src">
            <img align="right" vspace="5" hspace="8" border="1">
              <xsl:attribute name="src"><xsl:value-of select="." /></xsl:attribute>
              <xsl:attribute name="alt"><xsl:value-of select="../alt" /></xsl:attribute>
            </img>
          </xsl:for-each>
        </xsl:if>
      </xsl:for-each>
    </td>
  </xsl:for-each>
</xsl:for-each>

```

```

        </xsl:for-each>
        <xsl:if>
    </xsl:for-each>
    <xsl:if test="title/text()">
        <b><font size="4">
            <xsl:for-each select="title"><xsl:apply-templates/></xsl:for-each>
            </font></b>
        </xsl:if>
        <p><xsl:value-of select="content/text()" disable-output-escaping="yes"/></p>
    </td></tr>
</xsl:for-each>
</xsl:for-each>

```

The above example creates a row in HTML table for each section and places section image, title, and content with row.

Links: Category Links Transformation

<http://feed2.familyfeatures.com/services/xsl/categorylinks.xml>

This XSLT stylesheet takes the category links XML document and converts it to a HTML document that breaks links into subcategories and places content in a 440 width table. The core concept of this transformation is the same as the weekly feature or recipe transformation examples. The following section of the transformation highlights what is different:

```

<xsl:for-each select="subcategory">
    <tr><td>
        <a>
            <xsl:attribute name="name"><xsl:value-of select="@id"/></xsl:attribute>
            </a>
            <b><font size="4">
                <xsl:for-each select="name"><xsl:apply-templates/></xsl:for-each>
            </font></b><br/>
            <xsl:for-each select="links">
                <xsl:for-each select="link">
                    <a>
                        <xsl:if test="string-length(url)>0">
                            <xsl:attribute name="href"><xsl:value-of select="url"/></xsl:attribute>
                        </xsl:if>
                        <xsl:for-each select="name"><xsl:apply-templates/></xsl:for-each>
                    </a><br/>
                </xsl:for-each>
            </xsl:for-each>
        </td></tr>
    </xsl:for-each>

```

The above example creates a row in HTML table for each subcategory, creates in page anchor for jumps, writes out subcategory name, then each subcategory link.



Appendix A :: Resources

Here are some books and URL's that can provide further information on XML & XSLT

Books

- Tidwell, D. (2010). XSLT: Mastering XML transformations. O'Reilly. Sebastopol, CA.
- DuCharme, B. (1999). XML: The Annotated Specification. Prentice Hall PTR. Upper Saddle River, NJ.
- Burke, E. (2001). Java and XSLT: Embedded XML Processing. Into Java Applications O'Reilly. Sebastopol, CA.
- Laughlin, B. (2000). Java and XML. O'Reilly. Sebastopol, CA.
- St. Laurant, S. (2000). Building XML Applications. McGraw Hill. New York.
- St. Laurant, S. (2000). XML Elements of Style. McGraw-Hill. New York.

URL's

- <http://www.xml.com/> O'Reilly XML resource center
- <http://www.w3c.org/> World Wide Web Consortium (W3C)
- <http://www.vbxml.com/xsl/tutorials/intro/default.asp> XSLT & XPath Tutorial
- <http://www.mulberrytech.com/xsl/xsl-list/> XML & XSLT User Forum