sample files on clickUP, with examples on all this

Theme 6: XSL-FO

XSL-FO

XSL-FO

- **XSL-FO** is the second part of XSL (the first part = XSLT).
- The **FO** stands for **F**ormatting **O**bjects.
- XSL-FO is **a typesetting language**.
- It allows you to specify page layouts, margins...
- ...line spacing, headers/footers, table of contents, etc.

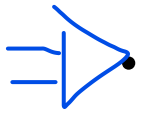
XSL-FO

- XSL-FO is also a **custom XML language**.
  - All XML grammar rules apply when coding XSL-FO.

- XSL-FO formats XML data for output to **print** (e.g. PDF).

- You need an **XSL-FO processor** to create the output.

Apache FOP



- **Apache FOP** is an XSL-FO processor that outputs PDF.
- FOP needs the **Java Runtime Environment** (JRE). not JDK
- You can download JRE and FOP from your module website.
- Follow the instructions on the website to install and use it.
- You'll need **Adobe Reader** (or equivalent) to read PDF docs.

General Structure

XSL-FO Structure

never use this actually

- An XSL-FO doc is a **text (XML) file** with a **.fo** extension.

- It starts with the **XML declaration**.

- The XSL-FO markup in the doc can be broken into **two parts**:
  - The **overall structure** (**fo:layout-master-set**) of the final output, and...
  - ...the **page content** (**fo:page-sequence**) of the final output.

layout/margins etc. is it a intro page, a index page etc

actual content like the text's decoration

XSL-FO Structure

- These two parts are enclosed in the root element, **fo:root**.

- The root must have an **xmlns:fo** attribute with the value:

```
http://www.w3.org/1999/XSL/Format
```

- In the **overall structure (fo:layout-master-set)**, we define **page masters**.
  - These are types of pages, e.g. cover pages or left-hand pages.

XSL-FO Structure

- Overall structure is represented by **fo:layout-master-set**.
  - Put **one** of these as the first child of the root.

- Inside it, we put one or more **fo:simple-page-master**'s.

margin, width, height

  - A master defines a type of page you could use in your output.
  - You must give masters unique names using **master-name**.

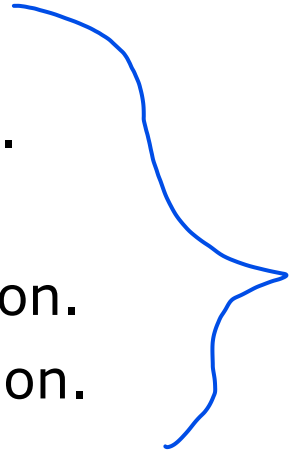
```
<fo:simple-page-master master-name="cover">
```

case matters

table of contents, cover page, bibliography page etc.

XSL-FO Structure

- Inside each master, we define **regions** for page content.
  - Add **fo:region-body** to define a body region.
  - Add **fo:region-before** to define a header region.
  - Add **fo:region-after** to define a footer region.
  - Add **fo:region-start** to define a left sidebar region.
  - Add **fo:region-end** to define a right sidebar region.



have to follow this order and everything has to be present

- Close **fo:layout-master-set** to start with **page content**.

XSL-FO Structure

you have as many page seq as you have simple page master

- Page content takes the form of **fo:page-sequence**’s.
  - Add them as the second and subsequent children of the root.

- Each sequence references a **fo:simple-page-master**.
  - You must reference an existing master in **master-reference**.

- A page sequence creates **actual output pages**…

XSL-FO Structure

content does not come inside flow...
they go inside the block

- ...based on the page layout defined by its master.

- Each sequence has a single **fo:flow.**
  - It specifies in what page region the output should go.
  - Set the region using **flow-name** (e.g. **xsl-region-body**).

- A flow will contain **fo:block**'s...

XSL-FO Structure

- ...which will contain the data to output onto the pages.
- The **fo:flow** outputs data onto an actual output page.
  - One output page = one page in the result PDF.

so if yo uhave more than one page in yourpdf you'll need more than one flow....more than one sequence

- If the data being output reaches the end of a page...
- ...the flow generates a new page and output continues.

XSL-FO Structure

- The flow will keep making new pages until it runs out of data.

- Data in a **fo:flow** is output once, but you can specify output...
- ...that must **repeat** on every page of the sequence.

- In a **fo:page-sequence**, BEFORE creating the **fo:flow**...
- ...add zero or more **fo:static-content** children.

odd and even pages headers (one has title of book, other has chapter)

XSL-FO Structure

you have to use block in order to actually show data

- Set a static-content's **flow-name** to the region...
- ...in which the repeating data must go.
  - E.g. use **xsl-region-before** for a header.
- Inside **fo:static-content**, use **fo:block**'s to output data.

Example: Header

Example: Creating a Header

- Create a master called **Content** and define a body region.

- Add **fo:region-before** to the master...
- ...to define a header region for that master.

extent vs margin (test out practically)

- Add the **extent** attribute to set the header's height.

- If you want, add other style properties (e.g. **border**).

background color etc

Example: Creating a Header

- If you use other regions together with a body region...
- ...you must set **margins** for the body...
- ...equal to the **extent** of the other regions.

- If you don’t, body content will overlap other regions.

- Add the margin attribute(s) to **fo:region-body**.

Example: Creating a Header

- Create a page-sequence that references **Content**.

- You'll probably want header data to repeat...
- ...on all output pages of the sequence.

- Add a static-content to the sequence...
- ...that references the **before** region (**xsl-region-before**).

Example: Creating a Header

- Add one or more blocks to the static-content...
- ...that contains and styles the header data.

- Create a flow AFTER the static-content for the body.

- Use blocks in the flow to contain and style the body data.
  - For this example, add enough data so that it generates more than one page.

Adding Content

Creating and Styling Blocks

- Content such as paragraphs are contained in **blocks.**

- The processor retrieves each block’s content...
- ...applies the block’s style properties...
- ...then stacks each block one after the other...
- ...to generate the final output.

Creating and Styling Blocks

- **fo:block** is a child of **fo:flow** or **fo:static-content**.

- Add attributes to the block to style its content.
  - Also known as **style properties**.
  - Use this **reference** to see what properties you can use.
  - XSL-FO and CSS share many of the same style properties

- You can have **fo:block**'s inside other **fo:block**'s.

Adding Images

- A **fo:block** can also contain images.
- As a child of **fo:block**, add:

adding style:

```
<fo:external-graphic src="url('graphic.uri')"/>
```

- Replace **graphic.uri** with the URL of the image.
- You can style it with properties. See the **reference**.

Inserting Page Breaks

- You can force a page break in XSL-FO output.
  - Forcing **fo:flow** to generate a new page and continue there.

- To a **fo:block**, add **break-before="page"**.
  - Now there will be a page break before that block's content.
  - Replace **-before** with **-after** to create it after the block.

Adding Page Numbers

- You can add page numbers to XSL-FO output pages.
- They would usually go in a **footer**.
- Define a **fo:region-after** for a master.
- In the corresponding sequence, add a **fo:static-content**.
  - Set its **flow-name** to **xsl-region-after**.

Adding Page Numbers

- Inside a **fo:block** within that static-content, add:

```
<fo:page-number/>
```

- The processor generates each page's current number.

cheatsheet about all this

Outputting Content in Columns

- You can output page content in columns.
- Define a master with a **fo:region-body**.
- Add **column-count=“n”** to **fo:region-body**.
  - Replace **n** with the number of columns you want.
- Create a sequence referencing that master.

Outputting Content in Columns

- Blocks of content in the body of that sequence can now be:
  - Output without using the columnal layout, or
  - Output in columns.

- Add **span="all"** to a block that should ignore columns.

- Add **break-before="column"** or **break-after="column"** to a block that should output in columns.

Practical Application

Using XSLT to Create XSL-FO

- In practice, you'd often transform XML to FO using XSLT.

- **To create a style sheet to transform XML to FO:**
  - Create the basic style sheet structure as normal.
  - In addition to **xmlns:xsl**, add **xmlns:fo** to **xsl:stylesheet**.
    - In this context, it goes in **xsl:stylesheet** instead of **fo:root**.

this allows for dynamic coding

Using XSLT to Create XSL-FO

- Instead of using **<xsl:output method="html"/>**, use:

```
<xsl:output method="xml" indent="yes"/>
```

- Put FO code in the root template instead of HTML code.
  - Start and end with **fo:root**.

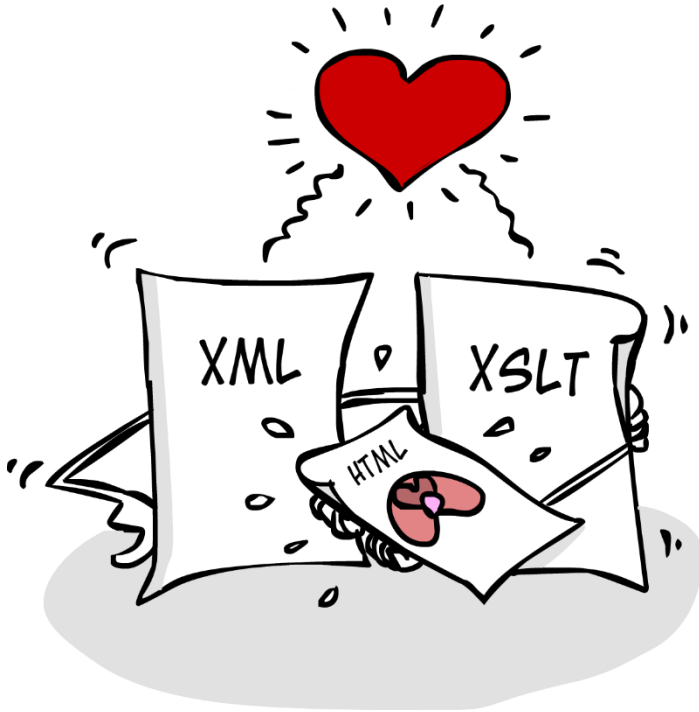
- Use XSLT instructions to generate the FO output from the XML.

do not need to link XSL-FO, cause it is a format

Using XSLT to Create XSL-FO

- Use a standalone XSLT processor to generate a FO file.
  - If browser-based, it cannot create separate output files.
- MSXSL is an example of such a processor.
  - It is included with Apache FOP on your website.
- Refer to the Apache FOP instructions on the website.

Theme 6: XSL FO



END THEME 6