Quality Control

DTD design tips

- Choose names that make sense when designing your xml file so that your DTD makes sense.
- Hierarchy is important, a blog has posts that contain paragraphs and headings. Containers add depth but don’t add too much depth. A big box of millions of little boxes is much harder to work with than a big box with a few medium boxes and smaller boxes inside those, and so on.
- Know when to use elements over attributes. An element holds content that is part of your document. An attribute modifies the behaviour of an element. The trick is to find a balance between using general elements with attributes to specify purpose and using an element for every single contingency.

Modularisation

- A DTD does not have to be stored in a single file. It often makes sense to store it in multiple files. You may wish to borrow from someone else, importing their DTD into your own as a subset. Or you may want to make the DTD a little neater by separating pieces into different files.
- To import a whole DTD or parts of DTD's, we use an external parameter entity.

Importing DTD

<!ELEMENT catalog (title, metadata, front, entries+)>

<!ENTITY % basic.stuff SYSTEM “basics.mod”>

%basic.stuff;

<!ENTITY % front.matter SYSTEM “front.mod”>

%front.matter;

<!ENTITY % metadata PUBLIC “//standards/DTDmeta/en”
http://www.standards-stuff.org/dtds/metadata.dtd>

%metadata;

Importing DTD

- This DTD had two local components, which are specified by SYSTEM identifiers. Each has a .mod extension, which is traditional to show that a file contains declarations but should not be used as a DTD on its own. The last component is a DTD that can stand on its own and is a PUBLIC resource.