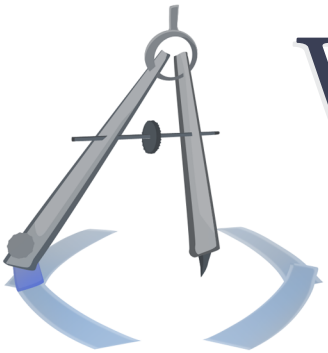

Website Architecture

Lezione 2

XML

A quick pit-stop through XML and XHTML.

Michael Serritella

Summer 2010

Intro to XML

Since HTML became so popular, someone decided to hijack its syntax for a general-purpose markup language, which can describe any kind of structured data[1]. This is e**X**tensible **M**arkup **L**anguage. For example:

```xml
<gradstudent name="Mike">
  <folio brand="Wenger">
    <pens>
      <fountainpens>
        <pen>
          <manufacturer>Pilot</manufacturer>
          <brand>Varisty</brand>
          <color>Black</color>
        </pen>
        <pen>
          <manufacturer>Pilot</manufacturer>
          <brand>Varisty</brand>
          <color>Red</color>
        </pen>
      </fountainpens>
    </pens>
  </folio>
</gradstudent>
```

[1]This is sort of a fairy tale; a grammar for general-purpose markup had existed for a long time (SGML), but HTML's popularity itself was hijacked.

The structure - the set of tags and and allowable attributes - is totally customizable. XML is useful to describe structured data in a universally-readable form. An XML document can be small, and it is often used for bite-sized pieces of data that are thrown around the Web. XML documents can also be massive, of course, but that's less common in Web.

Differences in grammar

Assuming you have common sense as a computer scientist and don't indulge in some of the more filthy habits of HTML authors, the grammars of HTML and XML are very similar. There are a few notable differences within the common features of XML.

Case-sensitivity of tags

Tags are case-sensitive.

Whitespace

In HTML, whitespace was consumed, such that any amount of whitespace reduces to one space. In XML, whitespace is preserved naturally inside of elements which do not contain others (leaves in the tree).

All tags must come to an end

In HTML, some elements had only one tag, like `<img>`. The creators of XML realized that this was a lazily-designed gram-

mar, too annoying to parse easily. So now, all tags must have an explicit closing. For *empty tags*, as they're called, this necessitates a change in grammar:

```xml
<container>
  <item /> <!-- look! -->
</container>
```

All tags must have a beginning

Sort of. All tags must be enclosed within a parent tag, until a single root tag is found.

Escape sequences

Like HTML, XML has escape sequences for special characters, but it also has sequences for arbitrarily large blocks of data.

Special characters

The following are escape sequences in XML: **<**, **>**, **&**, **'** (apostrophe), **"**.

Blobs of data

XML may contain large blobs of arbitrary data, and it may be infeasible to escape all of the special characters within it. So, blobs of literal data are enclosed like so:

```xml
<MyRawData> <!-- cute -->
  <![CDATA[
  Attack of the bloB!!! <<asfdasfa><1!&'
  ]]>
</MyRawData>
```

The CDATA stands for 'character data'.

Customization of structure

XML documents have user-defined structure - but how? Each XML document must be preceded with a header that specifies the grammar. This header also gives the character encoding of the document. For example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE students SYSTEM "students.dtd"> <!-- Custom grammar file -->
<students>
  <gradstudent name="Mike">
    <!-- etc -->
  </gradstudent>
</students>
```

There are actually a few different document types which can provide the grammar. One type of grammar specification is called a DTD (seen here), and another is called an XSD. The DTD is a **Document Type Definition**, and we will be using them more than XSDs.

As you may expect, the XML grammar document specifies which elements may exist in the document, which elements may exist

within other elements, which are optional, etc.

DTD and XSD documents have their own grammar. DTD grammar is a custom type, and XSD grammar is based on XML. XSDs are generally considered better, but we will see why we still learn about DTDs.

There is plenty more about XML, but its rules are straightforward enough and not very architecturally relevant, so we will not study them here.

XHTML and beyond

Around the same time that someone decided to hijack HTML, someone decided to improve HTML using those same features. Perhaps it didn't happen like this.

XHTML is a reasonable subset of HTML defined in terms of XML's more stringent rules. XHTML was released about the same time as a new version of HTML - version 4.01 - which deprecated a lot of the same features as XHTML did. Both XHTML and HTML continue to exist and continue to devolve in parallel. Their browser support is basically equal if declared carefully[2]. From a practical perspective, assuming you don't write HTML like a slob, its grammar is almost identical (XHTML 1.x vs. HTML 4.01).

[2]Support is becoming more and more equal, with Internet Explorer lagging behind. And careful declaration is very subtle; we will see more in future lessons.

Grammatical differences

Basic differences are:

- All tags and attributes must be lowercase.
- Empty tags must be closed as in XML (e.g. `<br />`).
- All attribute values must be quoted; all non-empty tags must have a closing tag; the document must contain an `<html>` tag, along with `<head>`, `<title>`, and `<body>` (anti-slob rules).

DTD and header

XHTML documents must have a DTD, just like XML documents. In the DTD, they effectively inform the browser of the HTML grammar used. The document could conceivably link to a local .dtd file, but Web pages should link to a publicly available DTD on the Internet, which stored in a central location - on a server in an ivory tower. There are three choices of DTDs: **Strict**, **Transitional**, and **Frameset**. Naturally, if you have a frameset, use the frameset DTD. We will later see the difference between the **Strict** and **Transitional** ones. Example:

HTML

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xml:lang="en"
      lang="en">
      <!-- 'xmlns' Qualifies the namespace of this 'html' element,
             guarding against the possibility of this being used
             with another XML document who defines a different
             type of 'html'. The URL there is like a huge namepace
             identifier. -->
      <!-- xml:lang is the XHTML attribute for specifying language -->
  <head>
  <!-- etc -->
```

There are more subtle and more advanced differences, which we will see as we encounter other technologies, such as HTML forms and JavaScript.

Finally, new versions of HTML also use DTDs, and have the same flavors.

Why bother?

XHTML does not seem very revolutionary. Why bother?

Simplificaiton of the rendering process

The HTML 4.01 standard and XHTML 1.0 standard both aim to simplify the way in which pages are rendered by browsers.

HTML is a large and sloppily-defined language, especially when considering all past versions. And most browsers try to be backwards-compatible, so they have to be prepared for all kinds of awful. A vast majority of sites nowadays use a reasonable subset of HTML. So, wouldn't it be nice to let the browser know that you're only using that subset? Boom. Use a **Strict** doctype. Now, the browser can make tighter assumptions about how to render your page, which might be of significant benefit to browsers on embedded systems.

Use of a **Strict** DTD is considered more influential for the rendering process than the choice of HTML vs. XHTML.

Integration of XML data

Use of XHTML allows the use of other XML within the document. The other XML could be a math description language (MathML), marked-up text describing vector art to be ren-

dered on the page (as an SVG image), or your own data in XML form. An additional language, **XSLT**, allows you to describe the translation - or transformation - of one XML format into another. With this, you can describe how a custom XML format should be transformed into XHTML and, thus, rendered on the page.

This is a somewhat an exotic feature, which may be most useful in sites that serve up content from a catalogue whose native format is XML.

Future-proofing

New versions of HTML and XHTML are both being drafted, and they are expected to diverge, with XHTML becoming even less backwards-compatible with both HTML and current versions of XHTML. Big wigs in the Web community recommend that authors stick to the HTML track for acceptance by the widest audience, mainly due to the expected shortcomings of Web browsers.

Premature conclusions

Since the choice of HTML vs XHML and the choice of DTD can be made on a per-page basis (whether or not most people would consider it excellent style), the choice doesn't very much affect site architecture. If you need HTML on a certain page because of a certain JavaScript hack that you've deemed worthwhile, then you can use HTML. If you need XHTML on a certain page because of a certain XML hack that you've deemed worthwhile, then you can use HTML. Just be aware of it when you write each page.

We will see software tools which conveniently handle the problem of rendering proper HTML/XHTML, even on a per-page basis, even though heterogeneity is atypical. Simply be aware of any benefits they offer, so that you can make executive decisions.