# Information Organization and Retrieval (INFO 202)

## Assignment 2: Getting Started with XML and XML Editors

**Author(s):**
Bob Glushko
glushko@ischool.berkeley.edu

**Course:** Information Organization and Retrieval (INFO 202)
**Date:** 12 September 2008
**Title:** Assignment 2: Getting Started with XML and XML Editors
This assignment introduces you to XML using the XML Spy or the oXygen XML editors. You do not need to turn anything in, but this assignment lays the groundwork for future ones that you WILL turn in. So make sure that you get familiar with one of the two editors and understand how an XML instance, schema, and transform fit together.

**XML Spy**

XML Spy is an award-winning XML editor and development environment that has been generously provided to the ISchool for use in courses and projects, but it runs only on Windows or on Mac/Linux using a Windows emulator. XML Spy is installed on all of the computers in the computing lab on the 2nd floor of South Hall. You can use XML Spy as the primary tool for creating XML, XML schemas, XML stylesheets and transforms, and so on. You can use XML Spy in the lab or you can download it from http://www.altova.com/download.html and install it on your own computer. You can download either the "enterprise edition" or the "standard edition" - the former has way more functionality than you'll ever need, and the latter is so simple that if you get more into XML you might need more than it provides. We are getting new license keys for both products and will send them around, or give them to you at the Monday 15 September section.

**oXygen**

If you don't use Windows, an alternative is an editor called oXygen XML. It runs within a Java Virtual Machine, so it runs on any platform with a Java Runtime Environment. You can download the trial version from http://www.oXygenxml.com/ and we'll give you the full license at the section meeting.

### Assignment Instructions

1. Find XML Spy or oXygen in (some directory) and start it up.

You will be working with three files: Report.xml, Report.dtd, and Report.xsl. These three files can be downloaded from http://courses.ischool.berkeley.edu/i202/f08/assignments/a2-xml/ and must be in the same directory

2. Open an XML instance in the editor. (Report.xml)

3. Open the XML instance in a browser (IE or Mozilla). Why is it rendered this way? ("View > Source" on menu bar).

4. Back in the editor, check the XML instance for "well-formedness" - conformance to the syntax rules for XML (F7 in XML Spy; in oXygen, click the blue-checked document icon in the tool bar).

5. Delete the beginning <Name> tag. Is the instance still well-formed? Change <Para> to <para>. Is the instance well-formed? XML is enforcing more restrictive syntax rules than HTML. Or put another way, XML doesn't allow bad practices that browsers typically forgive with HTML. Undo these changes so that your instance is well-formed again.

6. Specify an XML Document Type Definition for the XML instance by inserting <!DOCTYPE Report SYSTEM "Report.dtd"> directly below the <?xml version="1.0" encoding="UTF-8"?> declaration at the top of the file.

7. Validate the XML instance. (F8 in XML Spy; in oXygen, use the red-checked document icon, near the well-formedness icon). Insert a second author element containing your name and email. Is this valid?

8. Insert a <Phone> tag, your phone number, and </Phone> after your email element. Is this valid?

9. Open the XML DTD in the editor. The syntax is a little strange but has some resemblance to the BNF you probably know from programming languages. Try to figure out how you could have answered the previous two questions by examining the DTD rather than by experimentation.

10. Specify a style transformation for the XML instance by inserting as the third line of the instance <?xml-stylesheet type="text/xsl" href="Report.xsl"?>

11. Open the XML instance in a browser again (in XML Spy, you can do this by clicking the "Browser" button at the bottom of the editor pane; in oXygen, click the red-triangle-in-a-circle icon to the right of the well-formedness icon). It should be formatted this time.

12. Delete the DTD specification. Does the style transform still work? What does this imply about XML transformation programs?

13. Open the XML transformation file (Report.xsl) in the editor. The third line of the program (where "xsl:template" occurs) matches the element named "Report" in the instance and then passes through as output everything up to the next "xsl:template" tag. Can you see how these 20 lines or so create the HTML "scaffold" for the formatted report?