COM506 Professional Web Services Development

Lab Feedback

Practical	A1	**Date**	25 Sept 18

Section	**Issue**
General	Take care when copying and pasting code from the PDF workbooks. In most cases, it will work OK but when the text being copied includes certain characters (especially inverted comma characters), they are sometimes replaced by a version that is invalid in XML. You can see this visually when the character appears as a "slanted" or "rounded" character rather than the "straight" inverted comma obtained when you type SHIFT+2.
A1.2 (page 5)	The HTML 'table' example is not XML code, but an illustration of how web browsers are unforgiving of bad code. You should save it as an HTML file and load into the browser.
A1.3 (page 9)	There is no single correct answer to the XML "letter definition" problem. You need to consider how the data is likely to be used and design accordingly. For example, if the recipient address will only be accessed as a single element, then a single **`<address>`** element is appropriate. However, if you want to retrieve (e.g.) all letters sent to a specified town, then each element would be better specified as a separate field.
A1.5 (page 16)	The latest version of Notepad++ has removed the **Plugin Manager**, so we need to install the **XML Tools** plugin manually. See the separate note on this on Blackboard (Practical A1 section)
General	Possibly the biggest lesson from today's material is that XML is <u>very unforgiving</u> to work with as a developer – especially when it is complicated by namespaces and DTD specifications. Fortunately, as developers we almost never need to write XML by hand. It is usually generated as an output of some system that then allows the data to be injected into another application. We will see how this works in later classes, but for now we need to know enough about XML structure in order to use it effectively later.