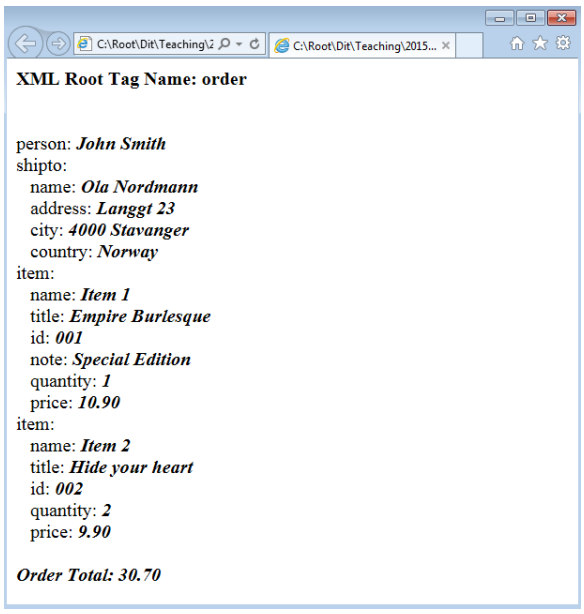
# Enterprise Systems & Architecture

## Lab 2 (Week 3): XML Exercises – XML Parsing

**The purpose of this labsheet is to provide some further exercises to review the hierarchical nature of XML data from a structural and programming perspective. To keep it simple we will use Internet Explorer (IE) and Javascript to load and parse an XML document. Note – *please only use IE (not Microsoft Edge) for these exercises, if you do not have IE on your laptop, please use a lab machine for this labsheet.***

### Exercise 1

- Below is a screenshot of an XML file displayed in Internet Explorer (IE) which uses javascript to parse the xml document and display the contents in the format shown.
- Write an XML file called *order.xml* that you think would reflect the xml document data being displayed (note, the order total is a calculated output and not part of the XML). Ensure your XML is well-formed.



## Exercise 2

- At the end of the lab sheet there is an html page containing the skeleton javascript code used to load and display the XML as above. The part that is incomplete is the code needed to parse the xml document structure and extract/display the element names and data values as well as calculate the total.
- Create the html file in the same folder as your xml file and add to the code so that when the page is opened using IE, it displays the XML as shown in the above screenshot (leave the order total calculation out of it for the moment).

***Notes***

1. Recall – XML is a tree structure
2. To get the child nodes of a parent element you can use *parentElement.childNodes* – this returns an array of child nodes.
3. A child node could be the text value node of the element or it could be a child *element* which itself could contain a text value node etc… (see example structure below)
4. To get an element's text value you call *element.firstChild.nodeValue* – when doing this you should first check that the element you are getting the value for has a child element (the *element.hasChildNodes()* method can be used for that purpose).

## Exercise 3

If you can, add some code that would calculate the total order value as shown in the screenshot.

```
<html>
<script type='text/javascript'>

/// Load XML file...
var doc = new ActiveXObject( "Microsoft.XMLDOM" )
doc.async = false;
doc.load("order.xml");

var error = doc.parseError;
if (error != "")
{
alert(error.reason);
}

/// Write HTML output...
document.write("<b>XML Root Tag Name: " + doc.documentElement.tagName + "<br/><br/></b>");

var rootChildren = doc.documentElement.childNodes;
var orderTotal = 0.00;

/// Loop all elements...
for (i=0; i<rootChildren.length; i++) {

  /// Add your code here...

}

///document.write("<br/><br/><i><b>Order Total: " + orderTotal.toFixed(2) + "</b></i>");

</script>
</html>
```