# XML Practical 1

The purpose of this practical is to introduce you to Java XML parsers (DOM and SAX) and make you familiar with the different approaches for each type of parser. You will be given simple java programs which you are expected to modify to serve their purpose.

**To pass the checkpoint**

1. Submit your code, either as a link to your GitHub repository or as a zip file. It must contain modified versions for either the SAX or DOM solution (or optionally both):
    a. The SAX solution should contain:
        i. SAXMenu.java: The SAX parser,
        ii. MenuHandler.java: The event handler for the SAX parser
    b. The DOM solution should contain:
        i. DOMMenu.java: The DOM parser
2. Provide a single peer-review of another student's solution

## The premise

You are a group of entrepreneurs that are about to launch a website for restaurants (or restaurant type establishments). The website allows visitors to browse menus from multiple restaurants in a standard environment where they can see various information about menu items (e.g., nutrition, price, ingredients, etc.). Each restaurant will upload their own menu to an automatic script which will parse it into your website. You have decided that the restaurants will have to upload their menus as a complete XML file, which can be directly passed to the interface without any manual post processing.
Having designed the XML template, your current task is to code the validator and parser.

# The task

1. Obtain the starter code and set up your GitHub repository. You should make it public at least until you have received your peer-review. Upload the starter code to your repository and remember to update when you have finished the tasks below. Alternatively (if you choose not to use GitHub), then just download the code and start working on the tasks below. The starter code can be found ...
    a. in this repository: https://github.com/saemundur-haraldsson/CSCU9T4-F_XMLpractical (fork it to your own before checking out)
    b. as a zip file on Canvas (where you found these instructions)

Solve the following tasks by using SAX or DOM (or both if you are feeling ambitious).

2. With your preferred IDE or editor open, explore, and run your chosen and provided java program. The program accepts a single command line argument which is the path to the xml file.
   If you run it directly from your IDE then you'll have to add the reference to the xml file in the run configurations.

3. Modify either `DomMenu.java` or `MenuHandler.java` so that it validates the menu against a given schema and then prints out 3 information about each menu item (e.g., name, price and calories) if it validates otherwise it should print out a useful error message.
   The path to the schema file should be an additional command line argument.
   For a validated menu, the output should look something like this:

```
Water       £0.90       200 kcal
Burger      £8.10       600 kcal
Fries       £2.20       300 kcal
....etc.
```

   For this you will have to make the following modifications to either `DomMenu.java`, `SAXMenu.java` and/or `MenuHandler.java`:

    a. Add a validation to a sensible location in the process using the `validateDocument` method and the input arguments. Only the error should be displayed if the document is not validated. *Hint: It should be used after you load the document.*

    b. Modify the validation method such that it prints a helpful message to the user telling them what was wrong with their xml. Test your implementation with a valid xml and multiple variations of invalid xml files. You can compare the output of your validation to any online validator. *Hint: `Validator.validate` throws `SAXParseException` which might contain helpful information*.

    c. Modify printNodes to print out the information, using appropriate methods. *Hint: for DOM use `getTextContent()` and iterate over* `NodeList` *which you can get with* `document.getElementsByTagName("*")`

***Optional extra*:** Using Xpath, print a couple of menu combos and their combined info. Make the program do the calculations for you and don‘t just add an extra menu item into the xml. For example print out:

```
Burger + Fries          £10.30     900 kcal
```

4. ***Optional extra (if you are attempting SAX solution):*** Add a validation method to the handler which does not rely on loading the whole document into memory. *Hint: The implementation is not far from the DOM example*