# Distributed Programming (03NQVOC)
# Distributed Programming I (03MQPOV)

*Laboratory exercise n.4*

## HTTP captures and inspection

The purpose of this exercise is to inspect HTTP requests and responses. Use the “curl” program under Linux in order to send HTTP requests and receive corresponding responses. It is possible to save separately the header of the HTTP response sent by the server by means of the –D option and the full response by means of the –o option. After each test, check the contents of the header and of the response. Also, use wireshark (the program already used in previous exercises) in order to monitor the full data exchange on the TCP channel (use interface eth0 for captures).

Here is a list of the tests to be performed:

1) Request a specific resource

```
curl -D header.txt 'http://media.polito.it' -o response.html
```

2) Send request to a search engine with parameter encoded in the URL string

```
curl -D header.txt 'http://search.yahoo.com/search?p=hello' -o response.html
```

3) Send request to a search engine with parameter in the body of the message (POST method)

```
curl -D header 'http://search.yahoo.com/search' -d 'p=hello' -o response.html
```

Repeat the previous requests in the Chrome or Chromium browser, and show the inspector window (by pressing CTRL + ALT + J) and focus attention on the Elements and Network tabs:

4) Go to `http://media.polito.it` by typing the URL in the browser bar and verify that it is possible to highlight the various elements of the page, corresponding to the various HTML elements, by simply selecting the HTML code itself. Moreover, check that using the inspect element command from the contextual menu (right mouse button) on the various elements of the page, the corresponding HTML code is shown.

5) Go to `http://search.yahoo.com/search?p=hello` by typing the URL in the browser bar. Note that in the Network tab it is possible to view the temporal diagram of the various HTTP transactions and the content of the request and response headers for each of them by clicking on it (or by clicking on View Source). This is possible even if the protocol used is HTTPS.

## Manual creation of HTML and CSS

HTML files can be written using a normal text editor (like Notepad, gedit, nano, or vi). The file name can have any of the two extensions **.htm** or **.html**.

If pages are of static type and are properly written with relative links, then it is possible to position them in any folder of the filesystem and start navigating by directly opening them with any browser.

If pages are dynamic (server-side), then it is necessary to install them on a real server web, which will be shown in the next lab.

For all the proposed exercises, always check the produced pages with at least two different browsers (e.g. IE and Firefox), changing the size of the screen (e.g. from 800x600 to 1280x1024) in order to verify the graphic quality of the built solution.

### Exercise 4.1.1 (manual creation of HTML pages)

Develop a set of personal web pages, all linked together:

- A student home page with his/her basic data (name, surname, date and place of birth)
- A page containing a brief CV (list of skills in computer science)
- A page containing names and scores of some of the passed exams (this must be implemented by using a table)

All the pages must contain, in the same position:

- A menu allowing to navigate to any one of the three pages (home, CV, scores)
- A footer identifying the author and his/her mail address

Pay particular attention to use, as much as possible, a logical formatting that is not affected by the size of the screen used to visualize the pages.

It is advisable to first build the single pages and verify their basic functionalities; after this first step, modify them to add the shared menu.

Always use (when possible) relative links and also put some links referring to external pages (e.g. to www.polito.it, or to the website of your own birthplace).

### Exercise 4.1.2 (automatic creation of HTML pages)

Re-implement at least one of the pages of the exercise 4.1.1 by using a system for automatic composition of web pages (e.g. MS-Word or OpenOffice, saving the result in HTML format). Compare the HTML code produced in this way to the code manually generated in the previous exercise in terms of code quantity and quality.

### Exercise 4.1.3 (HTML validation)

Check the correctness of all the generated pages by using the W3C validation service:

http://validator.w3.org

Correct any error signalled by the service. When pages are correct put the “valid HTML” logo into the pages by following the instructions provided in the validation page.

### Exercise 4.2.1 (graphical element)

Put in the footer of the personal web pages developed in exercise 4.1.1 the Polytechnic of Turin logo (available at the URL http://security.polito.it/img/polito.gif ) without locally downloading the file and assuring that the logo stays in the leftmost part of the footer, with the footer text written at its right.

### Exercise 4.2.2 (a common format for many pages)

Modify the personal web pages developed in exercise 4.1.1 in order to respect the following specifications:

- All the pages must use the same sans-serif font (like “Arial”, or “Helvetica”) for every text, while the footer must use a serif font, like as example “Times”, and it must be written in italic.
- The background of every page must be black and the text must be white (it is advisable to read the <body> tag definition in the HTML 4.01 manual, available online at http://www.w3.org/TR/html401)
- The table of the passed exams must have names and exam scores written in red (if the score is lower than 24) or in light green (if the score is greater or equal to 24). Scores must be centered in their column.

### Exercise 4.3.1 (HTML strict + CSS)

Modify the personal web pages developed in exercise 4.2.2 in order to respect the HTML STRICT specifications and in order to implement formatting details exclusively by using an external CSS. CSS specifications are available at the URL http://www.w3.org/TR/CSS21

### Exercise 4.3.2 (CSS layout)

Modify the personal web pages of exercise 4.3.1 in order to respect the following specifications:

- the page layout (header, menu, content, footer) must be build through CSS (for this purpose, you can use the guides and tutorials available online (e.g. http://www.w3schools.com/Css/css_float.asp and http://www.cssmastery.com/Examples.zip).
- the header must have a red border
- the footer must have a white border in its upper part

Verify that the layout is properly maintained even when the user reduces the horizontal size of the browser window.

### Exercise 4.4 (Optional)

Explore some parts of the HTML and CSS code provided with the examples of a web programming framework such as for example Bootstrap, which is available at the URL http://getbootstrap.com/getting-started/