college CD

Scripting language II (LS2)

Lesson plan

Class 01

Revision LS1

Class 02

Revision LS1 (suite)

Class 03

Top programming languages and frameworks

- **JavaScript :**
    - The most popular language = probably due to its compatibility with all browsers
    - Preferred by most full-stack software developers nowadays
    - Can be used to develop interactive web pages
    - ALSO now used in server-side (node.js)
    - 2018 = *Stack Overflow* surveyed 100,000 developers :
      Javascript was ranked number one (followed by HTML, CSS, SQL, and others)

- **Python :**
    - Gained more in popularity than any other language (last few years)
      MOSTLY BECAUSE of the rise of Artificial intelligence
      (it is the preferred language for creating AI or machine learning based web & mobile apps)
    - ALSO because of the elimination of semicolon at the end of the statement
    - In many ways, it is similar and different from other programming languages
    - The fastest growing python package : Pandas (introduced in 2011),
      = multi-purpose language which can be used for data science + web designing
      gradually replacing Javascript as a teaching language in a growing number of institutes

- **Java :**
    - One of the most stable languages + Cross-platform + at the peak of the industry (past 20 years)
    - Widely used for building enterprise scale web applications (including Android mobile apps)
- **PHP :**
    - *Hypertext Preprocessor* = popular scripting language (introduced 1995)
    - Kept on evolving throughout these years (leaving no place to any new languages)
    - Most of the available frameworks are free + provide strong security features
    - Offers many advantages (such as libraries and modules)
      which assure dynamic software development
    - Survey by w3techs.com = the best server-side programming language
- **SQL :**
    - *Structured Query Language* =
      used for securely managing data structures and information stored in databases
    - Incredibly popular = mainly because its great features
      + allows faster retrieving of a large number of database records
    - Evans Data Corporation, 7/19 million developers in the world use SQL
- **Ruby :**
    - Developed in Japan (mid-1990s)
      = dynamic language constructed in order to simplify the programming task (making it more fun)
    - Main reason of its popularity = Ruby of rails framework (full-stack web development)
- **C/C++ :**
    - Introduced in the 1970s, = parent of many modern languages (including Java)
    - C still in the top software writing languages
    - C++ (More dynamic language than C,) = code is type-checked before it gets actually executed
    - Major difference : C does not supports encapsulation, polymorphism, and inheritance (C++ does)

- **Swift :**
    - Influenced by Python and ruby+ developed by Apple Inc (introduced in 2014)
    - Considered a beginners' friendly language
    - Best choice for the development of native iOS or MAC OS app
    - Faster and more secure than both of them

JavaScript libraries

- Billion of web pages on the Internet + many of them are interactive
- Possible : every developers wrote their own JavaScript code
  BUT a lot of them probably re-used the same code
- A programmer should re-use existing code whenever it's possible
  = no time wasted reinventing the wheel
- TO DO SO => using libraries
  = JavaScript file that containing a various functions accomplishing various useful tasks
- To know what functions are available = read the documentation
  (Most libraries have documentation, list of available functions or a real-world example)

jQuery

- Lightweight *write less do more* JavaScript library
- PURPOSE = make JavaScript easier to use on websites
- Takes a lot of common tasks requiring many lines of JavaScript code,
  and wraps them into methods that can be called using a single line of code
- ALSO simplifies a lot of the complicated tasks from JavaScript, like AJAX calls and DOM manipulation
- Contains the following features:
    - HTML/DOM manipulation
    - CSS manipulation
    - HTML event methods
    - Effects and animations
    - AJAX
    - Utilities

Why use jQuery?

- There are many other more recent and modern JavaScript expendable frameworks available (such as Angular, React or Vue)
- BUT many of the most important websites on the web use jQuery (Google, Microsoft, IBM, Netflix...)
- Survey by Stack Overflow (2018) = about 48% of developers still use jQuery PROBABLY BECAUSE the very popular Bootstrap framework depends on it

jQuery : the basics

Adding jQuery to Your Web Pages

Two ways to use jQuery on a web site :

- Download the jQuery library from jQuery.com (single JavaScript file used within the HTML <script> tag inside the <head> section
- Include jQuery from a CDN (hosted by both Google and Microsoft)

```
<head>
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/
        jquery.min.js"></script>
</head>
```

Downloading jQuery

There are two versions of jQuery available which can be downloaded from jQuery.com :

- **Production version :**
  This is for your live website because it has been minified and compressed.
- **Development version :**
  This is for testing and development (uncompressed and readable code)

Coding in jQuery

Using jQuery

- The codes must be placed within ***<script></script>*** tags
  (usually after the contents or in an external ***.js*** file that would be linked to the document)

- You can see jQuery as a JavaScript function which can be coded in two different ways :

  - Using the word jquery
  - Using a dollar sign

```
<script>
        jquery();
        $();
</script>
```

Targeting an element

- To target a specific element = indicate the targeted element within the function's parenthesis

- Different methods may then be used, separated by periods.

```
<script>
        $(document).method();
</script>
```

document ready handler

- Almost everything coded in jQuery needs to be contained in this handler.

- It ensures that the web document has been fully downloaded before running the jQuery codes.
  (If an elements is manipulated before it is loaded = the script would return an error

- It ensures that the code is unobtrusive, that is, it's separated from content (XHTML) and presentation (CSS).

```
<script>
        $(document).ready(function() {

         // all jQuery code goes here
});
</script>
```

DOM Manipulation

- Document Object Model (DOM) = Programming interface for HTML and XML documents
- Represents the document as *nodes* and *objects*
  so that programs can target and change the structure, the style, and specific content. The DOM represents
- DOM = object-oriented representation of the web page (can be modified with a scripting language)

Targeting and manipulating elements

- Manipulation of the DOM = allows the programmer to target and manipulate specific elements of a document
- DOM manipulation using standard JavaScript = pretty complex
  BUT jQuery makes it a lot easier with a collection of DOM related methods

jQuery selectors

```
$("div");              // selects all <div> tags
$("#myElement");       // selects tags using ID "myElement"
$(".myClass");         // selects tags using class "myClass"
$("p#myElement");      // selects <p> tags using ID "myElement"
$("p a.navigation");   // selects <a class="navigation"> nested in <p>
```

```
$("p > a");            // <a> that are direct children of paragraphs
$("input[type=text]"); //  inputs of a specified type
$("a:first");          // first <a> on the page
$("p:odd");            // all odd numbered paragraphs
$("li:first-child");   // each list item that's the first child in its list
```

```
$(":animated");        // elements currently being animated
$(":button");          // any button elements
$(":radio");           // radio buttons
$(":checkbox");        // checkboxes
$(":checked");         // checkboxes or radio buttons that are selected
$(":header");          // header elements (h1, h2, h3, etc.)
```

Entire list of jquery selectors, visit :
https://api.jquery.com/category/selectors/

Manipulating an element

- After targeting an element, it is possible to manipulate it :

Example :
To write « Hello world! » within a container using the ID « test1 ». We would first need to target the element, and then use a method to manipulate it.

```
<div id="test1"> </div>
<script type="text/javascript">
        $("#test1").text("Hello world!");
</script>
```

Method chaining

- most of the methods returns a jQuery object (can then be used to call another method)
- This allows command chaining = multiple methods performed on the same set of elements

```
<div id="test2"></div>
<script type="text/javascript">
        $("#test1").text("Hello, world!").css("color", "blue");
</script2
```

Manipulation using events

- To manipulate an element on an event
  = target the element + use a specific event handler containing the desired manipulations

```
<h1>Test 3</h1>
<button id="test3">Click here</button>
<script type="text/javascript">
$("#test3").click(function() {
        $("h1").css("color", "red");
});
</script>
```

Explanation :
The title using changes to red when the button using ID test3 is clicked.

Other event handlers : blur, focus, hover, keydown, load, mousemove, resize, scroll, submit, select.

Hiding and showing contents

hide( )

Similar to the CSS property *display:none*, hide() method hides the selected elements.

show( )
Used to show hidden content using *display:none*, for instance.

Toggle( )
Allow to alternatively show and hide content.

Note:
Both hide(), show() and toggle() methods can use parameter such as fast, slow or milli-seconds creating some sort of transition

Assignment 01

Create a page with different elements and animations.
Using buttons, manipulate elements with jQuery.

Class 04

Coding in jQuery (suite)

Accessing and manipulating classes

Using jQuery you can:

- addClass
- removeClass
- toggleClass

```
<script>
$("div").addClass("content");          // adds class "content" to all <div>
$("div").removeClass("content");       // removes class "content" from all <div>
$("div").toggleClass("content");       // toggles the class "content" on all<div>
</script>
```

Note:
With t*oggleClass*, the class element will be added if it doesn't exist, and removed it if it does.

Explanation:
The class *textblue* is added to <h1> on a click event.

CSS:

```
.textblue {
        color: blue;
}
```

HTML:

```
<h1>Test</h1>
<button id="changetext">Change text color</button>
```

jQuery:

```
$("#changetext").click(function(){
        $("h1").addClass("textblue");
});
```

Using conditional structure

```
<script>
if ($("#myElement").hasClass("content")) {
        // do something here
}
</script>
```

Showing and hiding elements

```
<script>
$("#myElement").hide("slow", function() {
        // do something once the element is hidden
});

$("#myElement").show("fast", function() {
        // do something once the element is shown
});
$("#myElement").toggle(1000, function() {
        // do something once the element is shown/hidden
});
</script>
```

```
<script>
<section id="hidetest">
        <div>Hide test</div>
</section>
<script>
$("#hideshow").click(function(){
        $("section#hidetest div").toggle(3000);
});
</script>
```

Explanation :
When <section> is clicked, the it contains disappear in 3 seconds (3000 milliseconds).

Because it uses *.toggle* (and not *.hide*), the reappears if <section> is clicked again.

FadeIn and fadeOut

```
$("#myElement").fadeOut("slow", function() {
        // do something when fade out finished
}

$("#myElement").fadeIn("fast", function() {
        // do something when fade in finished
}
```

To fade an element only partially (in or out):

```
$("#myElement").fadeTo(2000, 0.4, function() {
        // do something when fade is finished
}
```

Explanations:
(2000) = duration of the animation.
(0.4) = opacity level

Exercise

Create a page with different elements and animations.
Using different events, manipulate the classes and fade elements with jQuery.

jQuery Animations and Effects

Sliding elements up or down

```
$("#myElement").slideDown("fast", function() {
        // do something when slide down is finished
}

$("#myElement").slideUp("slow", function() {
        // do something when slide up is finished
}

$("#myElement").slideToggle(1000, function() {
        // do something when slide up/down is finished
}
```

Animating CSS manipulation

```
$("#myElement").animate(
        {
                opacity: .3,
                width: "500px",
                height: "700px"
        }, 2000, function() {
                // optional callback after animation completes
        }
);
```

Explanations :
The method ***.animate*** is used containing the CSS to manipulate.

NOTE :
jQuery animation has its quirks (for example, animating colors requires a special plugin).

Exercice

Create a page with different elements and animations.
Using different events, manipulate the classes and fade elements as well as animation and slide in/out with jQuery.

Assignment 02

Create an accordion (see jquery-04/accordion)

Class 05

Coding jQuery (suite)

Managing content

html()

Gets the HTML of any element (like innerHTML in JavaScript)
It can either return the content of an element or set its content (replacing existing content).

```
<button>Replace existing content</button>
<p>This is paragraph #1</p>
<p>This is paragraph #2</p>

<script>
$("button").click(function(){
        $("p").html("<h1>Hello world!</h1>");
});
</script>
```

text()

Works just like html() BUT returns text element only (HTML tags would be treated as text)

append() and prepend

Works just like html() = appends text + HTML tags before or after the existing content of the selected element

```
<button>Append content using: append()</button>
<h1>Existing content</h1>

<script>
$("button").click(function(){
        $("h1").append(" / added content");
});
</script>
```

after() and before()

Works just like html() = inserts text and HTML contents after or before the selected element.

Note : it is important to alternate the use of single and double quotes, just like with JavaScript.

```
<button>Inserting new content using: after()</button>
<div>Existing content</div>

<script>
$("button").click(function(){
        $("div").after("<h1 style='color: red;'>New content</h1>");
});
</script>
```

remove(), empty() and detach()

Used to remove the selected element.

remove()

This method removes the selected elements, including all text and child nodes.

empty()

This method removes only the content from the selected elements.

detach()

This method removes the elements without removing data and events.

```
<button>Removing existing content using: remove()</button>
<div><h1>Existing content</h1></div>

<script>
$("button").click(function(){
        $("div").remove();
});
</script>
```

Other content manipulation methods (selected)

appendTo()	Inserts HTML elements at the end of selected elements.
attr()	Sets or returns attributes/values of selected elements.
clone()	Makes a copy of selected elements.
children()	Returns all direct children of the selected element.
contents()	Returns the text nodes of the selected element.
find()	Returns the descendants of the selected element.
first()	Returns the first occurrence of the selected element - last() does the opposite.
hasClass()	Checks if any of the selected elements have a specified class name.
height()	Sets or returns the height of selected elements.
innerHeight()	Returns the height of an element (includes padding, but not border).
innerWidth()	Returns the width of an element (includes padding, but not border).
insertAfter()	Inserts HTML elements after selected elements.
insertBefore()	Inserts HTML elements before selected elements.
offset()	Sets or returns the offset coordinates for selected elements (relative to the document).
offsetParent()	Returns the first positioned parent element.
outerHeight()	Returns the height of an element (includes padding and border).
outerWidth()	Returns the width of an element (includes padding and border).
next()	Return the next sibling of the selected element.
not()	Returns elements that do not match a certain criteria.
position()	Returns the position (relative to the parent element) of an element.
prependTo()	Inserts HTML elements at the beginning of selected elements.
prop()	Sets or returns properties/values of selected elements.
removeAttr()	Removes one or more attributes from selected elements.
removeProp()	Removes a property set by the prop() method.
replaceAll()	Replaces selected elements with new HTML elements.
replaceWith()	Replaces selected elements with new content.
scrollLeft()	Sets or returns the horizontal scrollbar position of selected elements.
scrollTop()	Sets or returns the vertical scrollbar position of selected elements.
siblings()	Returns all siblings of the selected element.
unwrap()	Removes the parent element of the selected elements.
val()	Sets or returns the value attribute of the selected elements (for form elements).
width()	Sets or returns the width of selected elements.
wrap()	Wraps HTML element(s) around each selected element.
wrapAll()	Wraps HTML element(s) around all selected elements.
wrapInner()	Wraps HTML element(s) around the content of each selected element.

Getting values from a form

It is necessary to extract values using the method **val()** while selecting the proper fields using their ID.

```
<form action="">
        <label>Input :</label>
        <input type="text" id="text" name="name" value="" />
        <input type="button" id="text_value" value="Print Value"/>
</form>

<div> </div>

<script>
$('#text_value').click(function() {
        var text_extract = $("#text").val();
        $("div").html("<h1>" + text_extract + "<h1>");
});
</script>
```

Explanations :
In the example above, the text written in the field by the user is added to the DIV wrapped in H1 tags. A variable is first declared (text_extract) to store the value extracted from the field using ID text. The variable is then used, along with the method html(), to transfer the content to the DIV.

Assignment 03

Create a page with a form section and a display section.
Use the form to gather different information from the user and
use the collected value to create a nice presentation in the display section.

Class 06

Coding in jQuery (suite)

mouseover() and mouseout()

- indicate what manipulation to operate when the mouse pointer hovers or not an element
- *CSS :hover* effect = both methods needs to be used

```
<h2>Add a yellow background to the title below by mouseover</h2>
<h1>Content affected by mouseover() and mouseout()</h1>

<script>
$("h2").mouseover(function(){
        $("h1").css("background-color", "yellow");
});
$("h2").mouseout(function(){
        $("h1").css("background-color", "transparent");
});
</script>
```

Other mouse event (selection) :

.dblclick()
Triggers an action on double-click.

.hover()
Binds one or two handlers to the selected elements, to be executed when the mouse pointer enters and leaves the elements.

.mousedown() / .mouseup()
Triggers an action when the mouse button is down or up.

.mouseenter() / .mouseleave()
Triggers an action when the mouse pointer enters or leave an element.

.mousemove()
Triggers an action when the mouse pointer moves.

Mouse pointer coordinates

- ***event.pageX*** and ***event.pageY***
  = get the coordinate of the mouse pointer from the left and the top borders

- To get the coordinate of a click, simply change *mousemove* for *click* in the example below

```
<div id="container"> </div>

<script>
$(document).on("mousemove", function(event){
        $("#container" ).text( "X: " + event.pageX + ", Y: " + event.pageY);
});
</script>
```

Assignment 04

Using ***event.pageX*** and ***event.pageY***, make it so the background image changes based on X or Y mouse position (minimum 3 different background)

Class 07

Revision

Workshop

Class 08

Mid-term exam

Class 09

AJAX requests

load()

- LIKE PHP *file_get_contents()*

- Allows to retrieve the content of a specific external file
  ALSO to retrieve specific content from a given file

- USUALLY requires for the file to be hosted
  (use a virtual server such as MAMP to try this function)

Syntax

$(selector).load(URL, data, callback);

Retrieving the entire content of a file (EXAMPLES ON MAMP 09-01)

- load() retrieves the entire content of the file
- Can use different types of files such(HTML or TXT...) including markup tags

```
<h1> </h1>

<script>
$("h1").load("mytext.txt");
</script>
```

Retrieving specific element from a file's content using ID (idem)

- Specify an ID or a TAG after the file name
  TO TARGET specific content of a file = All contents marked with ID/TAG will be displayed

```
<h1> </h1>

<script>
$("h1").load("mytext.html #first");
</script>
```

Using status messages (EXAMPLES ON MAMP 09-02)

POSSIBLE to use the other parameters with load()
TO specify actions to be taken upon successful or unsuccessful result of loading an external file's content

```
<h1> </h1>
<button>Test</button>

<script>
$("button").click(function(){

        $("h1").load("mytext.txt", function(response, status, http){
                if(status == "success")
                        alert("This worked successfully!");

                if(status == "error")
                        alert("Error: " + http.status + ": " + http.statusText);
         });
});
</script>
```

NB: Make voluntary error in code to show error message

Using load() on inline events

Note: HTML tags within txt document wouldn't work
use a HTML document in such a situation

```
<button onclick=$("#container").load("myfile.txt");>Test</button>

<div id="container"> </div>
```

Assignment 05

Using the load() function, create a complete homepage using load() like we would use PHP includes and create a container displaying different contents using tabs with inline events.

Class 10

JSON

- Standing for *JavaScript Object Notation*
- JSON is a human readable format for structuring data
- Primarily used to transmit data between a server and web application (alternative to XML)

Keys and Values

- JSON data consists in pairs of keys (equivalent of properties) and values separated by colons + placed between double quote marks in an object
  They make a ***key/value pair.***

Key: A key is always a string enclosed in quotation marks.
Value: A value can be a string, number, boolean expression, array, or object.

```
"car" : "Mazda"
```

Syntax and structure of JSON files

- Braces are used to define an object
- Objects may contain many key/value pairs
- NOTA : elements of JSON's objects can be : strings, numbers, arrays, objects, booleans, null, etc.

{...} Brace brackets are used to define an object.
[...] Brackets are used to define an array.
N.B.: Commas are used to separate elements of an object
(no comma after the last element or the element will not be valid).

```
{
"name" : "John",
"age" : 30,
"city" : ["Montreal", "New York", "Paris"]
}

"user" : {
            "name" : "John",
            "age" : 30,
            "city" : "New York"
}
```

Retrieving data from a JSON files

- In JSON files = data consist in a string of variable length (can be enormous).

- Testing JSON with external JSON files
  = only works with hosted files (some virtual servers may allow testing locally).

Parsing the JSON data

- To be able to access the data of JSON files = needs to be parsed
- Can be done using the function ***JSON.parse()***.

Retrieving data (Example : 10/json-01.html)

- Once an object contain the parsed JSON data has been created
  it is possible to retrieve specific values (associated with the different keys)
  Syntax : *object.key*

(Example : 10/json-01.html)

Retrieving specific data from JSON

```
<h2> </h2>

<script>
let myJson = '{"name":"John", "age":30, "city":"New York"}';

let myObject = JSON.parse(myJson);

$("h2").html(myObject.name + ", " + myObject.age + ", " + myObject.city);
</script>
```

Viewing and validating

- Especially at the beginning = useful to be able to visualize and validate the source code

- Many online editors / validators :
  http://jsonviewer.stack.hu/
  https://jsonformatter.curiousconcept.com/
  https://jsoneditoronline.org/

- Simply copy and paste your code in the application's "Text" window.
  The validator the indicates if errors are found and the viewer window shows the tree structure of the code.

Retrieving data from external JSON files

Online examples : http://www.monamijean.com/cdi/json-01.html

- Retrieving data = basically the same as before (remember : external JSON files needs to be tested online)

Loading data from external JSON files (USE MAMP)

- for the data to be available in the current document
  = a http request needs to be made to store the data in an object (a variable)
  USING the function *XMLHttpRequest()* (complicated at first but we'll see another option)

- The open() and send() function will also be used

Retrieving data from an external JSON file using a http request

```
<h2></h2>

<script>
var xmlhttp = new XMLHttpRequest();

xmlhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
                var obj = JSON.parse(this.responseText);
                $("h2").html(obj.prenom + " " + obj.nom + ", " + obj.age);
        }
};

xmlhttp.open("GET", "http://www.monamijean.com/cdi/mytest.json", true);
xmlhttp.send();
</script>
```

Retrieving data from an external JSON file using getJSON() function

```
<h1></h1>

<script>
$.getJSON('http://www.monamijean.com/cdi/mytest.json', function (data) {
        $("h1").append("You are " + data.age + " year old.");
});
</script>
```

Assignment 06

Using the external file supplied by the teacher, create a weather report homepage using JSON, JavaScript and jQuery.

http://www.monamijean.com/cdi/meteo.json

Class 11

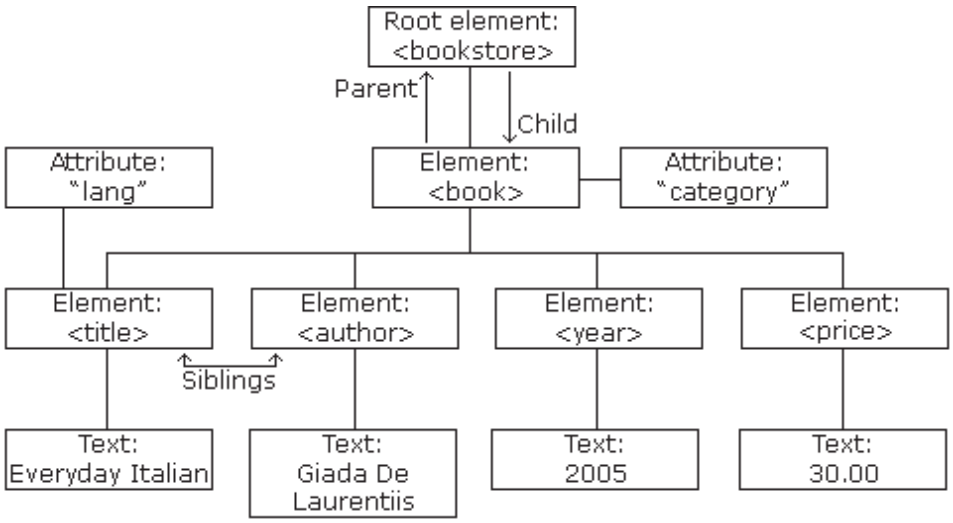
XML

- Standing for *eXtensible Markup Language*
- Markup language used to store and transport data
- Designed to be both human and machine readable.
- Less popular in the last couple of years = being replaced with JSON)
- Still plays an important role in many different IT systems
  (used in many aspects of web development)
- Doesn’t depend on platform nor software nor programming language
  = it is possible to write a program in any language on any platform to send, receive or store data using XML
- Llike PHP includes = XML is a complement to HTML
  used to separate data from presentation
- XML doesn't actually do anything
  Simply used to structure data. = XML file can then be used in a program to display the data

XML structure

- XML resembles HTML = uses opening and endings tags
  BUT self-describing syntax = no predefined tags
- In XML = tags are created accordingly to the database needs
- Fields are grouped withing entries,
  entries are grouped within a table in a child-sibling relationship

```
<root>
        <child>
                <subchild>.....</subchild>
        </child>
</root>
```




```
<?xml version="1.0" encoding="UTF-8"?>          // XML prolog
<booklist>                                      // Table
        <book>                                  // Entry
                <title>Harry Potter</title>     // Field
                <author>J K. Rowling</author>
                <year>2005</year>
                <price>29.99</price>
        </book>
        <book>
                <title>The flowers of evil</title>
                <author>Charles Baudelaire</author>
                <year>1857</year>
                <price>39,95</price>
        </book>
</booklist>
```

Viewing XML files

- XML document must be saved using XML extension
- Can be opened in a browser (tree view)

Styling XML files using CSS

- NOT the most proper way to display XML data
  BUT possible to use a CSS file to style an XML document's content

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="mystyle.css"?>
<booklist>
        <book>
```

Class 12

Cookies

- File created on client-side to store data received by a computer
  THEN sent back without changing it

- When a user visits a website a cookie to his computer
  The computer stores it inside the web browser

Cookies purposes

- Contrarily to popular beliefs, cookies aren't always a bad thing
  Help keeping track of a user's visits and activity on a website

- EXAMPLE:
  A cookie maybe used to keep track of a shopping cart's items as the user keep on shopping.
  Without cookies, the user's shopping cart would be deleted every time a new link would be clicked

- EXAMPLE:
  May also use cookies to keep a record of a user's most recent visit, name, or to record login information

Types of cookies

Session cookies

- Strictly used during a user's current navigation on a website
- The cookie is deleted as soon as the session is over.

Tracking cookies (persistent cookies)

- Used to create long-term records of a user's visits to a given website

Authentication cookies

- Tracks if a user is logged in or not, and if so, under what name

Potential treats

- Cookies do not transfer viruses or malware (have no way to affect how a computer)
- BUT, some viruses and malwares can disguise themselves as cookies
- READ ABOUT *supercookies* and *zombie cookies* which recreate itself after being deleted
- Tracking cookies may also cause security
  BECAUSE they make it easier to watch where a user is going and what he or she is doing online
-
-

Setting a simple cookie

```
<script type="text/javascript">
        document.cookie = "name=John Doe";
</script>
```

Reading existent cookies (Notes: cookies-1.html)

```
alert(document.cookie);

OR

<h2> </h2>

<script type="text/javascript">
        $("h2").html(document.cookie);
</script>
```

Setting a cookie

- When resetting a cookie using the same name = the cookie is updated
  the new value replaces the existing one

- When a new cookie is set = existing cookies remain
  AND each new cookie is added to the browser cookie folder

Expiry date/time

```
<script type="text/javascript">
        document.cookie = "userName=John Doe;
                           expires=sat, 31 Dec 2033 12:00:00 UTC;  OR max-age=3600;
                           path=/; domain=mysite.com";
</script>
```

Domain

- Hosts the cookie will be sent to (defaults: current document location (not including subdomains)
- If a domain is specified, subdomains are always included.

Path

- URL path that must exist in the requested resource before sending the Cookie header.
- The "/" character is interpreted as a directory separator
  sub directories will be matched as well (e.g. path=/docs, "/docs", "/docs/Web/", or "/docs/Web/HTTP").

js-cookie

- JS-COOKIE is simple, lightweight JavaScript application = allows handling cookies easily.
- It works with all browsers and has been heavily tested
- Can be downloaded to be installed within a website's structure or use a simple CDN

js-cookie CDN

https://cdn.jsdelivr.net/npm/js-cookie@2/src/js.cookie.min.js

Setting a cookie

Valid across an entire site:

Cookies.set('name', 'value')

Expires 7 days from now, valid across the entire site:

Cookies.set('name', 'value', **{ expires: 7 }**)

Expiring cookie valid to the path of the current page:

Cookies.set('name', 'value', { expires: 7, **path:' '** })

Reading a cookie

Read cookie:

Cookies.get('name'); // Would show associated value

Read all visible cookies:

Cookies.get();

Removing a cookie

Cookies.remove('name');

IMPORTANT! To delete a cookie which isn't relying on the default attributes, you must pass the exact same path and domain attributes used to set the cookie:

Cookies.remove(**'name', { path: '', domain: '.yourdomain.com'** })