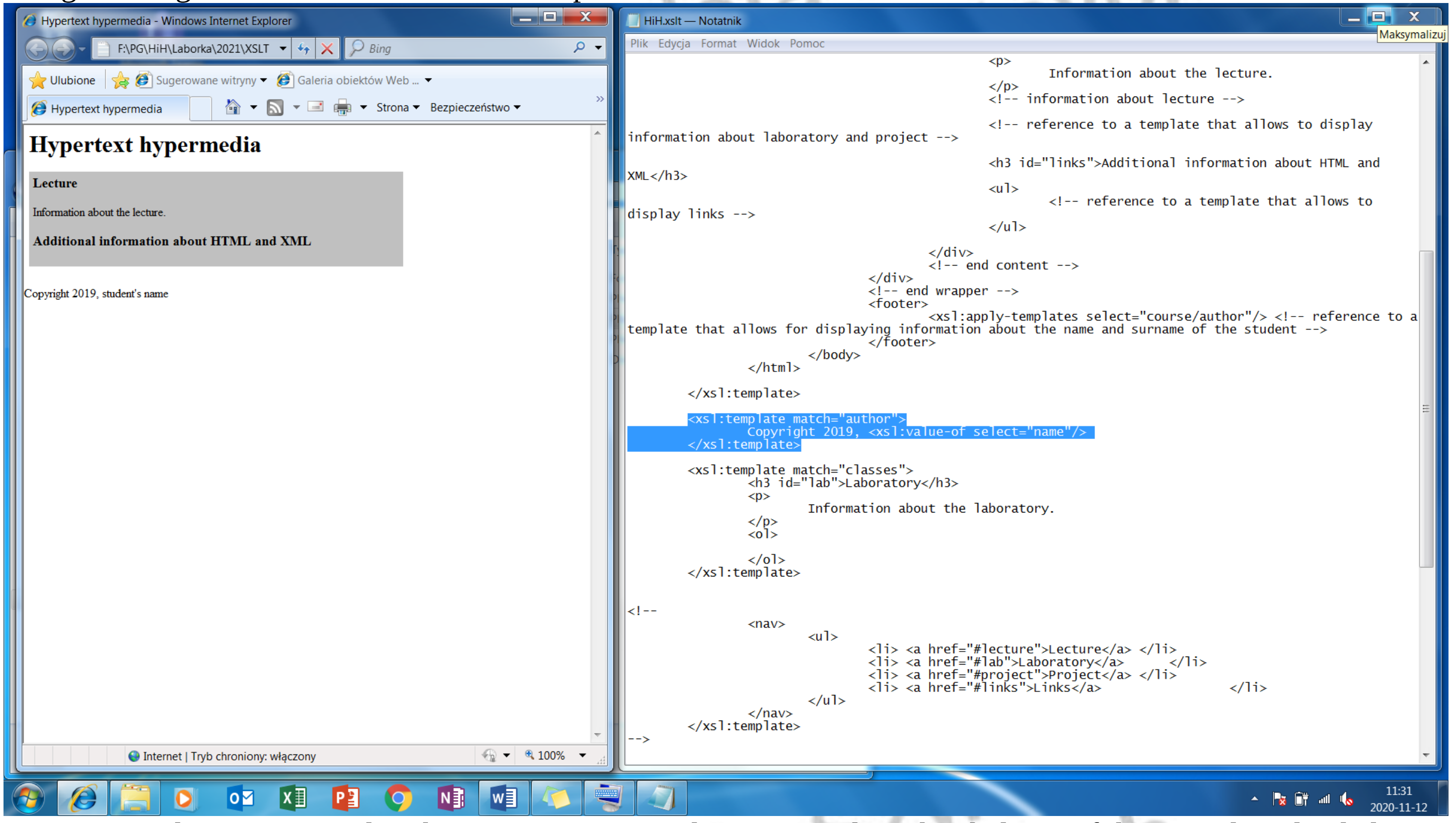
Fundamentals of XSLT

The aim of the exercise is to familiarize yourself with the XSLT standard.
Any text editor and web browser is needed during the lab.

- At the end of the class, you should pack your catalog into a ZIP file named `Surname_Name_Index_XSLT.ZIP` (without Polish letters) and send it to Moodle.
- The submitted catalogue is also to include a class report. The report file is to be called `Surname_Name_Index_XSLT` and it should contain screenshots containing both the code running in the browser and the code view. The screenshot should show the date and time. The code responsible for the given stage should be marked. An example screenshot should look like this:



Hypertext hypermedia

Lecture

Information about the lecture.

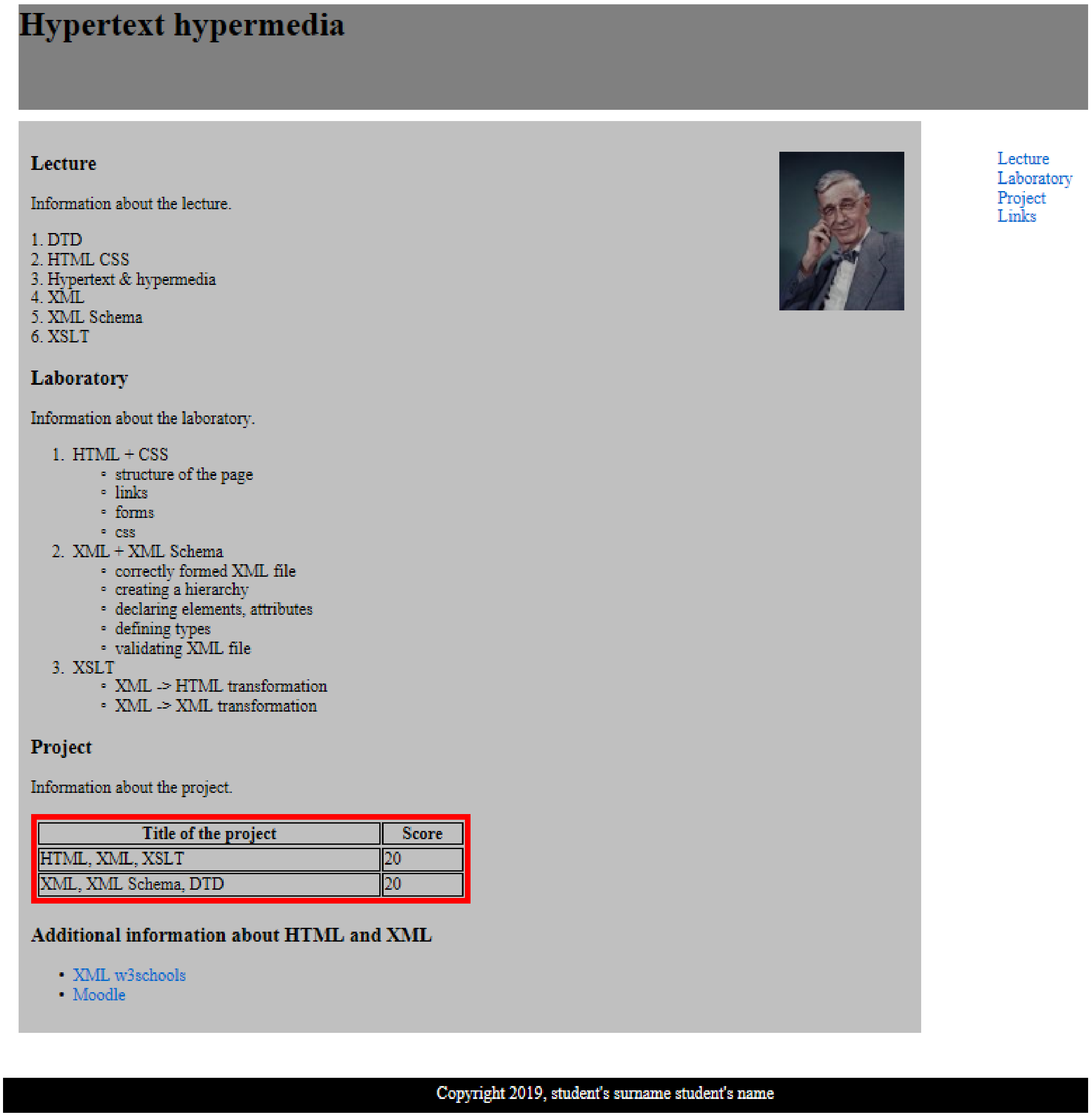
Additional information about HTML and XML

Copyright 2019, student's name

```
                                                <p>
                                                        Information about the lecture.
                                                </p>
                                                <!-- information about lecture -->

                                                <!-- reference to a template that allows to display
information about laboratory and project -->

                                                <h3 id="links">Additional information about HTML and
XML</h3>
                                                <ul>
                                                        <!-- reference to a template that allows to
display links -->
                                                </ul>

                                        </div>
                                        <!-- end content -->
                                </div>
                                <!-- end wrapper -->
                                <footer>
                                        <xsl:apply-templates select="course/author"/> <!-- reference to a
template that allows for displaying information about the name and surname of the student -->
                                </footer>
                        </body>
                </html>

        </xsl:template>

        <xsl:template match="author">
                Copyright 2019, <xsl:value-of select="name"/>
        </xsl:template>

        <xsl:template match="classes">
                <h3 id="lab">Laboratory</h3>
                <p>
                        Information about the laboratory.
                </p>
                <ol>

                </ol>
        </xsl:template>


<!--
                <nav>
                        <ul>
                                <li> <a href="#lecture">Lecture</a> </li>
                                <li> <a href="#lab">Laboratory</a>      </li>
                                <li> <a href="#project">Project</a> </li>
                                <li> <a href="#links">Links</a>                 </li>
                        </ul>
                </nav>
        </xsl:template>
-->
```

1. Create a directory named with your own name and surname. Place the skeleton of the page downloaded from Moodla in it. You can use the Visual environment or an ordinary notebook to work with files. Remember to periodically save your work results.
2. Familiarize yourself with the downloaded files. Analyze the HH.xml file, pay attention to the structure of the document, the tags and the link to the XSLT document. Analyze HH.xsl file, pay attention to the template for the root and how to create an HTML document. At the end of the file there are some useful HTML code fragments in the comments. **You must use a <template> with match attribute during the lab. <for-each> should be used only if it is clearly written in the instruction.**
3. (1pt) Add your name and surname in the `name` and `surname` element in the XML file. In the XSLT file, extract your name from an XML document and display it in the footer of the page. Use the existing template matching the `author` element, complete it with the appropriate components.
4. **Place a screenshot in the report file.**
5. (1pt) Create named `template` (`template` with `name` attribute). In this template, place the menu to navigate within the page. Call the template in the right place in the root template.
6. **Place a screenshot in the report file.**
7. (1,5pt) Display images with Vannewar Bush. Create template with match attribute and apply it in proper place. For the image use class="right" to set the image on the right. When you hover over the image, the text from the "image" element (here: Vannevar Bush) should be displayed.
8. **Place a screenshot in the report file.**

9. (1,5pt) Display running links. Create a template for the `link` element. In this template an html element that allows you to get a working link should be created. At the appropriate place, apply the template. Only the last two links from the XML file should be displayed.
10. **Place a screenshot in the report file.**
11. (2pt) Display information about lecture themes, sort them (<xsl:sort>) and number (<xsl:number>). To make the numbering correct, use the value attribute and the position() function. To process all the necessary nodes, use a loop (<for-each>). Keep in mind that predicates in XPath allow you to specify what set of nodes to use (here you should select `theme` elements for which the attribute `kind` takes the value of `lecture`).
12. **Place a screenshot in the report file.**
13. (2 pt) Display information about laboratory. Complete the template for the `classes` item. In this template, use the conditional instruction (<xsl:if > or < xsl:choose >) to perform the relevant instructions for laboratory items only (test the value of the `kind` attribute). Create appropriate templates for `component` and `theme` elements and apply them accordingly.
14. **Place a screenshot in the report file.**
15. (1 pt) Display information about project. Create appropriate templates and apply them.
16. **Place a screenshot in the report file.**



Hypertext hypermedia

Lecture

Information about the lecture.

1. DTD
2. HTML CSS
3. Hypertext & hypermedia
4. XML
5. XML Schema
6. XSLT

Laboratory

Information about the laboratory.

1. HTML + CSS
   - structure of the page
   - links
   - forms
   - css
2. XML + XML Schema
   - correctly formed XML file
   - creating a hierarchy
   - declaring elements, attributes
   - defining types
   - validating XML file
3. XSLT
   - XML -> HTML transformation
   - XML -> XML transformation

Project

Information about the project.

Title of the project	Score
HTML, XML, XSLT	20
XML, XML Schema, DTD	20

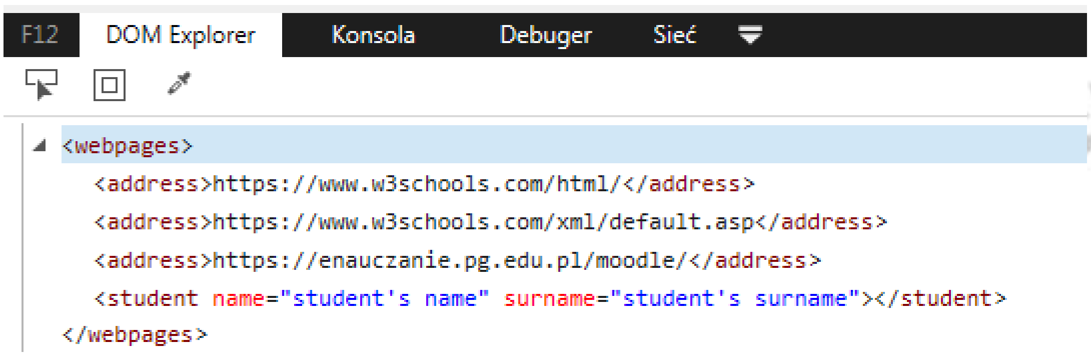
Additional information about HTML and XML

- XML w3schools
- Moodle

Lecture
Laboratory
Project
Links

Copyright 2019, student's surname student's name

17. (1 pt - an additional task, additional point) Familiarize yourself with the file HiH2.xslt. The file allows you to create a new XML file containing only information about the links. Become familiar with the way you create new elements. We can see the effect, for example, by displaying the HiH.xml file in the browser (in the HiH.xml file, you must rename the xslt file in instruction <?xml-stylesheet>). The structure of the new XML file can be viewed by selecting the F12 development tools. (Alternatively, we can transform XML into Visual Studio). Add an element named `student` with the `name` and `surname` attributes to the output file. Attribute values should be taken from an XML file.



```
<webpages>
  <address>https://www.w3schools.com/html/</address>
  <address>https://www.w3schools.com/xml/default.asp</address>
  <address>https://enauczanie.pg.edu.pl/moodle/</address>
  <student name="student's name" surname="student's surname"></student>
</webpages>
```

18. **Place a screenshot in the report file.**

XSLT some information

1. **Templates describe what is to be done**
   <xsl:template match="XPath expression" name="name" mode="mode">
   ...
   </ xsl:template>
2. **applying the template defined with the attribute match**
   <xsl:apply-templates select=" XPath expression"/>
3. **calling the template defined with the attribute name**
   <xsl:call-template name="name"/>
4. **selecting the value**
   <xsl:value-of select=" XPath expression "/>

"/"	root
"element's_name"	element
"@attribute's_name"	attribute
"."	current node

5. **conditional instructions**
   <xsl:if test="expresion">
   ...
   </xsl:if>
6. **conditional instructions**
   <xsl:choose>
   <xsl:when test="expression1">
   ...
   </xsl:when>
   <xsl:when test=" expresion2">
   ...

```
        </xsl:when>
        ...
        <xsl:otherwise>
                ...
        </xsl:otherwise >
</xsl:choose>
```

7. **loop**

```
<xsl:for-each select="XPath expression">
        ...
</xsl:for-each>
```

8. **sorting**

```
<xsl:sort select=" XPath expression" order="sort order"/>
ascending
descending
```

9. **numbering**

```
<xsl:number format="number format"/>
```

Numer format: 1) a) A. ...

10. **Creating new elements and attributes**

```
<xsl:element name="name of the element"></xsl:element>
<xsl:attribute name="name of the attribute"></xsl:attribute>
```