Created by Dr. Zoltán Subecz, GAMF educational materials were used.
John von Neumann University, Kecskemét, GAMF Faculty of Engineering and Computer Science, Informatics Department
subecz.zoltan@nje.hu

Web-programming Seminar

Only informative text = we don't study it, only additional information

In this document, I have written many comments to the code of the exercises. You can find the solutions without explanations in the file **Solutions.zip**.

I have written the explanations after the // signs for simplicity.
These must be removed from the HTML and CSS files for proper operation.
A valid HTML comment would be: <!-- Comment -->

1 - Basics – HTML – CSS

Introduction

Recommended sites:
https://www.w3.org/
https://www.w3schools.com/
http://learn.shayhowe.com/html-css/

- **Client-side languages** (the browser "understands" these, HTML, CSS, JavaScript)
- **Server-side languages** (the browser does not understand these, the server understands these and sends their output in the response to the http request, e.g. PHP)

For HTML and CSS I will use the **Visual Studio Code** IDE in class.
At home, download the ZIP (Portable: no installation required) version from
https://code.visualstudio.com/download
You can also use other IDEs, both in class and at exams, that you find on your computer.
e.g. PHPStorm, NotePad++, Netbeans, Eclipse, Atom, Komodo Edit,

Try the examples on multiple browsers: Chrome, Firefox, MS Edge and mobile browsers

Save HTML and CSS pages to a folder and display them in a browser.

Publish your work on the Internet

Register at home with a (free) Internet provider.
What we solve in class, try on your own computer at home on the registered Internet provider!
The Homework will also have to be published on the Internet!

Chrome developer tools

Right click on the page / View page source
Right click on the page / Inspect
Load a page e.g. https://www.bbc.com/ or https://edition.cnn.com/
Use the Inspect function to analyze the structure of the page (Document Object Model, DOM).
https://hu.wikipedia.org/wiki/Document_Object_Model
Select an element in the page to inspect it function.
Element, Network, ….
The elements of the page can also be overwritten!

Markup Validation Service
Check the markup (HTML, XHTML, …) of Web documents
http://validator.w3.org/

Check this site for example: https://www.bbc.co.uk/

Structure of a basic HTML page:

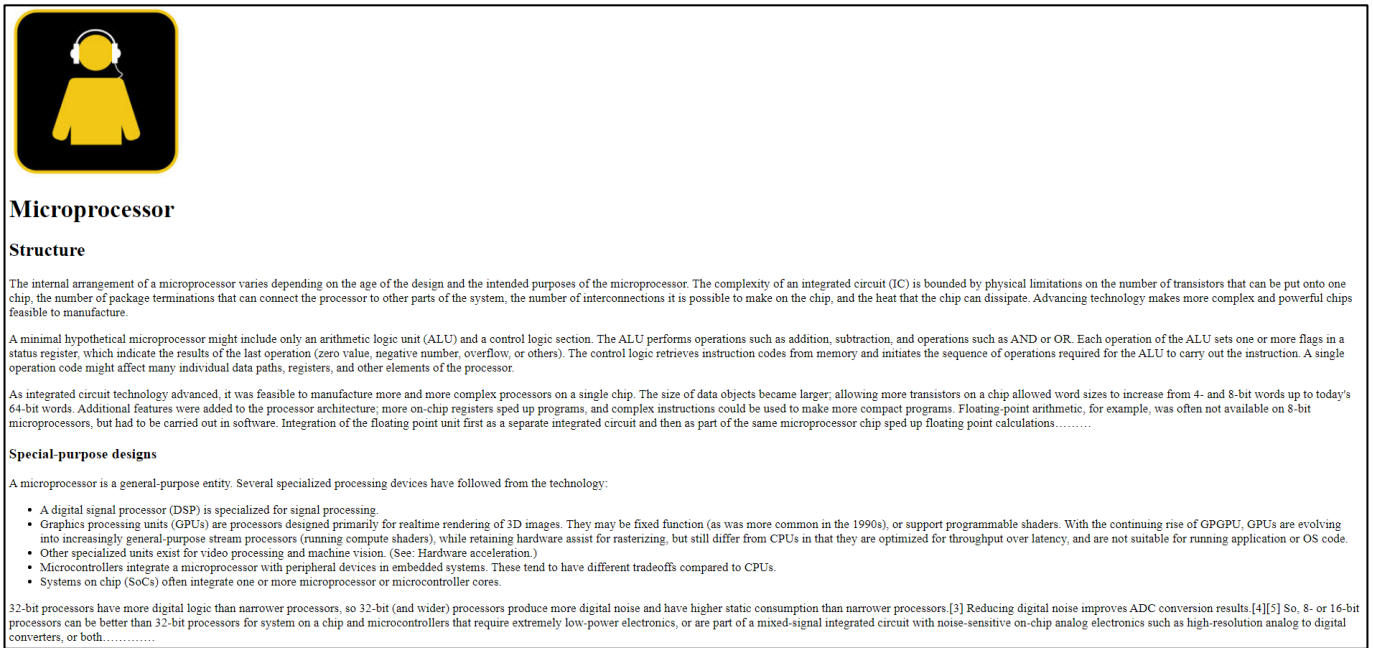
```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Page Title</title>
   </head>
   <body>
   ……….
   </body>
</html>
```

https://www.w3schools.com/html/

1st Exercise – HTML and CSS

You can find an HTML file and some image in **Web-programming-Seminar-Solutions.zip**. Copy these files to a folder, open the HTML file in a browser and open with a **Code Editor** (e.g. Visual Studio Code). We use only one image in this exercise.

1A-Exercise



Microprocessor

Structure

The internal arrangement of a microprocessor varies depending on the age of the design and the intended purposes of the microprocessor. The complexity of an integrated circuit (IC) is bounded by physical limitations on the number of transistors that can be put onto one chip, the number of package terminations that can connect the processor to other parts of the system, the number of interconnections it is possible to make on the chip, and the heat that the chip can dissipate. Advancing technology makes more complex and powerful chips feasible to manufacture.

A minimal hypothetical microprocessor might include only an arithmetic logic unit (ALU) and a control logic section. The ALU performs operations such as addition, subtraction, and operations such as AND or OR. Each operation of the ALU sets one or more flags in a status register, which indicate the results of the last operation (zero value, negative number, overflow, or others). The control logic retrieves instruction codes from memory and initiates the sequence of operations required for the ALU to carry out the instruction. A single operation code might affect many individual data paths, registers, and other elements of the processor.

As integrated circuit technology advanced, it was feasible to manufacture more and more complex processors on a single chip. The size of data objects became larger; allowing more transistors on a chip allowed word sizes to increase from 4- and 8-bit words up to today's 64-bit words. Additional features were added to the processor architecture; more on-chip registers sped up programs, and complex instructions could be used to make more compact programs. Floating-point arithmetic, for example, was often not available on 8-bit microprocessors, but had to be carried out in software. Integration of the floating point unit first as a separate integrated circuit and then as part of the same microprocessor chip sped up floating point calculations……..

Special-purpose designs

A microprocessor is a general-purpose entity. Several specialized processing devices have followed from the technology:

- A digital signal processor (DSP) is specialized for signal processing.
- Graphics processing units (GPUs) are processors designed primarily for realtime rendering of 3D images. They may be fixed function (as was more common in the 1990s), or support programmable shaders. With the continuing rise of GPGPU, GPUs are evolving into increasingly general-purpose stream processors (running compute shaders), while retaining hardware assist for rasterizing, but still differ from CPUs in that they are optimized for throughput over latency, and are not suitable for running application or OS code.
- Other specialized units exist for video processing and machine vision. (See: Hardware acceleration.)
- Microcontrollers integrate a microprocessor with peripheral devices in embedded systems. These tend to have different tradeoffs compared to CPUs.
- Systems on chip (SoCs) often integrate one or more microprocessor or microcontroller cores.

32-bit processors have more digital logic than narrower processors, so 32-bit (and wider) processors produce more digital noise and have higher static consumption than narrower processors.[3] Reducing digital noise improves ADC conversion results.[4][5] So, 8- or 16-bit processors can be better than 32-bit processors for system on a chip and microcontrollers that require extremely low-power electronics, or are part of a mixed-signal integrated circuit with noise-sensitive on-chip analog electronics such as high-resolution analog to digital converters, or both……

exercise1.html

Analyze the HTML code

Main elements on the page:
Image: <img src="image.jpg" alt="My image" title="Web-programming">
Headings: H1…H6 e.g.: <h1>……</h1>
Paragraph: ….

Define important text in a document (The content is displayed in **bold**.): ….

Unordered list:

```
<ul>
  <li>…….li>
  <li>…….li>
  <li>…….li>
</ul>
```

DIV box: ……

https://www.w3schools.com/tags/tag_div.ASP

The DIV box is used to group elements together, apply common formatting to them, and move the elements together. We'll use it a lot.

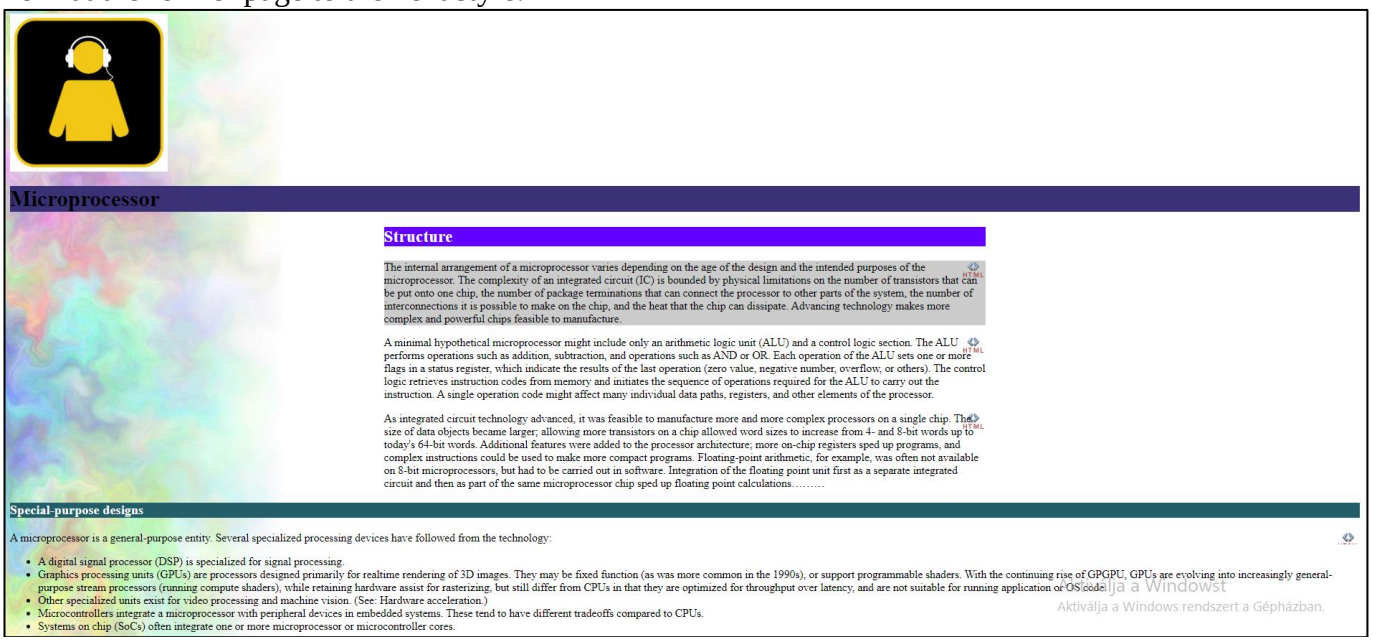
1B-Exercise - formatting with CSS

https://www.w3schools.com/css/

Use the same HTML code as in the last exercise, create only CSS style to the page. You can see that with CSS the page appearance can be very different.

Try all CSS settings!

Format the former page to the next style:



Solution

1B-Exercise\exercise1.html

Add the following **CSS styling** to the <head> section of the previous HTML page:

```
  <head>
……
    <style>
      body {          // style for the whole page
        background-image: url('plasma.jpg');      // background image for the page
        background-position: left top;      // the position of the background image
        background-repeat: repeat-y;    // Repeat the background-image vertically
      }
      h1, h2, h3{     // common style for H1, H2, H3 headings
        color: white;
```

```
        }
        h1{     style for H1
           background-color: #3B3176;        // Specify the background color with a HEX value
        }

        h2{
           background-color: rgb(255,0,255);          // Specify the background color with an RGB value
        }
        h3{
           background-color: #245E69;
        }
        // Select and style the first <p> element that are placed immediately after <H2> elements:
        h2+p {
           background-color: #ccc;    // #ccc = #cccccc;       #00f  = #0000ff
        }
        p {      // style for the paragraphs
           background-image: url(html1.gif);
           background-position: right top;
           background-repeat: no-repeat;
        }
        div {    // style for the DIV section
           width: 860px;        // The width of the  DIV section
//         https://www.w3schools.com/css/css_margin.asp
// The CSS margin properties are used to create space around elements, outside of any defined borders.
// There are properties for setting the margin for each side of an element (top, right, bottom, and left).
// You can set the margin property to auto to horizontally center the element within its container.
//         If the margin property has four values:
//                 margin: 25px 50px 75px 100px;
//                         top margin is 25px
//                         right margin is 50px
//                         bottom margin is 75px
//                         left margin is 100px
//         The element will then take up the specified width, and the remaining space will be split
//         equally between the left and right margins.
           margin: 0px auto 0px auto;
        }
     </style>
  </head>
```

2nd Exercise

Look at the solution with CSS and without it. (e.g. rename the CSS file).
Try all CSS settings: look at it with the given setting and without it.

Create the next blog-style page. It contains two blog posts. **Find these settings on the next image.**

- H1 heading (Blog page)
- H2 heading (Lorem ipsum ….)
- tag for the date (Dec 31…)
-   tag for blog post sender name (By: admin)
- tags (ul-li) (Tags: dolor onsectetur ……). Lists, but not with hyperlinks
- tag with p tags: paragraphs (The text of the blog post: Lorem ipsum ….)
- lists with hyperlinks (ul-li-a) (Add a new comment…) **Simple horizontal menu!**

Create a design that is similar to the given image.

Solution

Exercise2.html

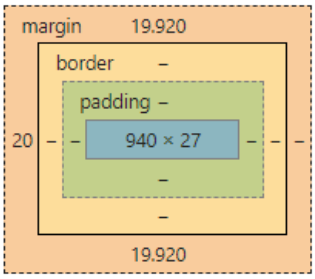
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Exercise 1</title>
<link rel="stylesheet" type="text/css" href="Exercise1.css">
</head>
<body>
// id: identifies an element on the page. There can only be one element with a given ID on a page
//        We will use it in CSS formatting. Look at Exercise2.css file
<h1 id="pagetitle">Blog page</h1>
// The <article> tag specifies independent, self-contained content.
//        Potential sources for the <article> element: Forum post, Blog post, News story, Comment
//        https://www.w3schools.com/tags/tag_article.asp
<article>        // the first blog post
// differences between id and class CSS Selectors:
//        The id selector uses the id attribute of an HTML element to select a specific element.
//        The id of an element should be unique within a page,
//         so the id selector is used to select one unique element!
//        To select an element with a specific id, write a hash (#) character, followed by the id
//        of the element.
//        The class selector selects elements with a specific class attribute.
//        So, all HTML elements with the same class attribute will have the same format and style.
//        To select elements with a specific class, write a period (.) character, followed by the name of
//        the class.
//        https://www.w3schools.com/css/css_syntax.asp
```

```
<div class="date">Dec<br><span>31</span><br>12:20</div>     // <div> tag for date
<div class="wrapper">          // the big box next to the date
<h2>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h2>          // Blog post title
// The tags are created with list, but hyperlink is not used.
          By: <div class="name">admin</div>Tags:
<ul>
<li>dolor</li>
<li>onsectetur</li>
<li>adipiscing</li>
<li>integer</li>
<li>fermentum</li>
<li>rutrum</li>
</ul>
<div>  // Blog post with 3 paragraphs
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer ex massa, fermentum a magna sit amet, hendrerit accumsan nibh. Pellentesque vel risus elit. Integer eget tempor quam. Donec dignissim fringilla nisi in ultrices. Morbi erat est, laoreet scelerisque dapibus in, mattis eget lorem. Sed faucibus mauris eget nisi ullamcorper, quis rhoncus urna aliquam. Phasellus lacinia fringilla diam, sit amet imperdiet purus molestie id.</p>
<p>Sed condimentum libero quam, ac pretium mauris feugiat nec. Etiam est orci, ornare vitae mollis vel, rutrum ac quam. Donec vitae orci dolor. Curabitur sed placerat felis. Donec ac tellus sed lectus interdum pellentesque ac nec odio. Fusce tortor dolor, ullamcorper in mattis vel, dictum at massa. Cras tempor neque nulla, sed euismod mi bibendum sed. Fusce vulputate magna ac odio accumsan, vitae pulvinar mi rutrum. Nullam convallis bibendum mi, quis aliquet nisl fringilla eget. Sed maximus luctus commodo. Curabitur interdum molestie augue, nec interdum felis bibendum quis. Proin eget volutpat purus. Maecenas suscipit dolor in ex condimentum volutpat.</p>
<p>In mattis euismod tellus nec venenatis. Quisque luctus semper arcu sed malesuada. Phasellus a rutrum purus. Nulla metus metus, tristique nec aliquet at, consectetur quis lectus. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam non iaculis felis. Etiam commodo feugiat augue, elementum mattis metus. Proin odio metus, posuere eu sagittis quis, iaculis a tellus. Proin sed malesuada arcu. Integer id magna at sapien bibendum imperdiet.</p>
</div>
<nav>  // for the two bottom hyperlinks
          // The <nav> tag defines a set of navigation links.
          // The <nav> element is intended only for major block of navigation links.
          // https://www.w3schools.com/tags/tag_nav.asp
<ul>    // with a list
<li><a href="https://www.wix.com/" target="_blank">Add a new comment</a></li>
<li><a href="https://www.tumblr.com/" target="_blank">Read more</a></li>
</ul>
</nav>
</div>
</article>
<article>          // the second blog post: similar to the first one
<div class="date">Dec<br><span>31</span><br>12:16</div>
<div class="wrapper">
<h2>Curabitur ut porta tortor, eu congue neque.</h2>
          By: <div class="name">admin</div>Tags:
<ul>
<li>porta</li>
<li>cursus</li>
<li>consectetur</li>
<li>efficitur</li>
```

```
</ul>
<div>
<p>Curabitur ut porta tortor, eu congue neque. Mauris ac cursus augue. Aenean in rhoncus ex, sed tempus massa. Etiam malesuada sed metus et luctus. Phasellus aliquet orci ac libero fermentum bibendum. Cras non dictum nibh. Morbi commodo lobortis ex in pellentesque. In tempor lobortis magna eget viverra. Sed tincidunt ante nisi, sed fermentum nisl vehicula non. Praesent lobortis nibh id malesuada placerat. Integer maximus nibh ut erat pharetra pharetra. Cras malesuada tempus turpis a sollicitudin. Nulla venenatis imperdiet velit vitae hendrerit.</p>
<p>In tristique lorem quis mauris porta placerat. Sed varius lacus a ipsum lacinia venenatis. Sed eu mattis magna. Morbi turpis orci, suscipit sit amet metus eu, auctor viverra dui. Sed in ultrices nunc, vitae finibus elit. Maecenas aliquet odio eget tristique euismod. Suspendisse eget tristique augue. Ut eu auctor justo. Aliquam erat volutpat. Morbi in risus feugiat, efficitur dolor vitae, ornare dolor. Nam sit amet purus dui. Nam id augue urna. Donec commodo vestibulum felis in aliquam. Proin eget risus nisl. In feugiat nulla at sagittis tempus. Suspendisse a purus eu velit pellentesque sollicitudin eu nec ipsum.</p>
<p>Mauris quis massa tempor, egestas tortor quis, fringilla metus. Sed consectetur scelerisque neque quis faucibus. Interdum et malesuada fames ac ante ipsum primis in faucibus. Nam arcu sapien, rhoncus vitae tempus id, finibus eget libero. Curabitur sollicitudin sodales tincidunt. Integer facilisis bibendum ex mattis consequat. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aliquam nec neque eu enim tincidunt dapibus. Duis sollicitudin posuere elit, sit amet sagittis nunc sodales id. Duis dapibus laoreet nunc a iaculis. Mauris tincidunt risus at laoreet pretium. Pellentesque vulputate pellentesque dapibus. Nullam ultrices, metus id dictum posuere, nibh dolor accumsan justo, quis posuere sem augue semper neque.</p>
<p>Ut malesuada bibendum elit eget pulvinar. Praesent vitae purus eget nulla tincidunt efficitur. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Donec quis erat rutrum, gravida arcu id, tristique nibh. Donec a risus vel risus euismod blandit ut id arcu. Vestibulum eleifend mauris a commodo condimentum. Etiam porttitor enim sed rutrum vehicula. Phasellus ut pulvinar libero. Integer ut eros rutrum felis placerat condimentum. Proin posuere velit neque, quis tristique massa vehicula sed. Pellentesque accumsan, leo sit amet tempor euismod, est enim bibendum nibh, a ultricies leo mauris et mauris. Aliquam tempor varius feugiat. Ut commodo sagittis euismod.</p>
</div>
<nav>
<ul>
<li><a href="https://www.wix.com/" target="_blank">Add a new comment</a></li>
<li><a href="https://www.tumblr.com/" target="_blank">Read more</a></li>
</ul>
</nav>
</div>
</article>
</body>
</html>
```

Exercise2.css

Difference between margin and padding: CSS Box Model
https://www.w3schools.com/css/css_boxmodel.asp
Right click on page / Inspect (Vizsgálat) / Select an element



- **Padding** - area around the content inside the border.
- **Margin** - area outside the border.

```
#pagetitle{
   text-align:center;    // align text to center within the element
   color: yellow;        // font color yellow
   background-color: red;      // background color red
   width:800px;          // box 800px wide
   margin: 0 auto;       // center box alignment          =  margin: 0px auto 0px auto;
}
.date {            // date class
   float:left;     // The datum div should be on the left and the next part should wrap around from the right.
                   // the text would still have a margin (margin-left: 72px;),
                   // but it wouldn't start next to the date, but below it
   text-align: center;
   padding: 5px;         // padding for each side
   border: 2px solid #777;
   border-radius: 5px;           // the radius of the element's corners.
   background-color: rgb(230,230,230);
   width: 50px;          // The wrapper section is placed next to it, offset 72px from the left.
                         // The width of the <div class="wrapper"> box is 50px
                         // The margin for the <div class="wrapper"> box is 72px
   font-size: small;
   font-weight: bold;
}
.date span {     // the span tag in the <div class="date"> tag: (31)
   font-size: large;
}
.wrapper {       // the wrapper class
   margin-left: 72px;  // It takes the <div class="wrapper"> box next to the <div class="date"> box
                       // the 50px width was set in .date {…} settings
}
h2 {
   color: #006;
}
.name {          // for the <div class="name"> tag
display: inline;        // the name is in a div because of the other settings
                        // inline: it doesn't start in new line and the next part doesn't start
                        // in a new line
                        // it should be on the same line.
   color: #009;
margin-right: 25px;     // leave 25px space after the name:
                        //        with this setting you can set multiple spaces between words!
}
ul {      // list style
```

```
display: inline;          // The Tags and the following text are in the same line.
   color: #009;
   padding-left: 0px;
   list-style-type: none;          // Thre is no list-item marker
                                   // list-style-type: It specifies the type of list-item marker in a list.
}
li {
display: inline;          // The list elements are in the same line.
}
a {
margin-right: 25px;    // It sets multiple spaces between words!
   text-decoration: none;       // It removes underlines from links
}
```

Only informative text

There are several possibilities to move boxes of content on page (e.g. side by side)

Some examples:
float, display:inline, display:table-cell, display:table-row, display: table; display: table-column; display:inline-block, flex, grid overflow:auto, position:absolute, position:relative, ……

Not all browsers support each of them
We'll use some of them in the next lessons:
float, display:inline, display:table-cell, display:inline-block, display: block;

3rd exercise - Well-designed horizontal menu

In this exercise we look only at the horizontal menu, there is not much new in the others.
Important parts are highlighted in red.

Look at the solution with CSS and without it. (e.g. rename the CSS file).
Try all CSS settings: look at it with the given setting and without it.

Create the design (given on the next image). The text has not to be the same, but the text amount should be similar.

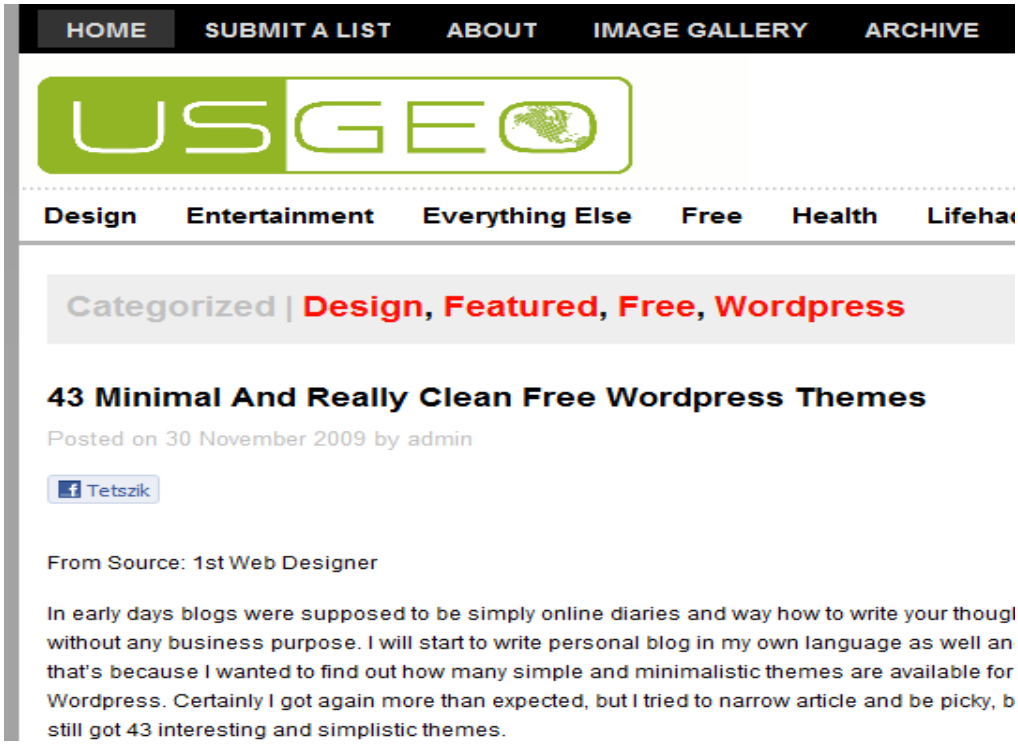
Use the next tags:

- Horizontal menu <ul> (with id), <li> and <a> tags, #000 (background color) #DDD (character color) and red (hover: select element when you mouse over them) colors
- Paragraph (<p>) with an image (logo.png)
- Horizontal menu <ul> (with id), <li> and <a> tags, #B3B3B3 (border color)
- Paragraph (<p>) with id and text, #EEE (background color) #C0C0C0 (character color) and #FF0E00 (character color)
- <h2> heading
- Paragraph (<p>) with HTML class attribute, #C0C0C0 (character color):
- Paragraph (<p>) with an image (buttons.png)), #ECEEF5 (background color) #CAD4E7 (border color) and #5C74C8 (character color)
- Two paragraphs (<p>)

The width of the content is 960px, in a centralized tag. The page background color is #A2A2A2. Create a design that is similar to the given image. Check your HTML and CSS code with a Markup Validation Service.

Two horizontal menu: HOME… Design…

Look at the images in Sources folder, especially the buttons.png image.



Solution

exercise2.html

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Exercise 2</title>
   <link rel="stylesheet" type="text/css" href="exercise2.css">
</head>
<body>
   <div>
      <nav>     // navigation: list + hyperlinks
// We put the menu items in a list because it's easier to format (CSS)
         <ul id="menu1">       // first (upper) menu
// href="#": it indicates only the hyperlink
            <li><a href="#">HOME</a></li>          // list elements
            <li><a href="#">SUBMIT A LIST</a></li>
            <li><a href="#">ABOUT</a></li>
            <li><a href="#">IMAGE GALLERY</a></li>
            <li><a href="#">ARCHIVE</a></li>
         </ul>
```

```
    </nav>
    <p id="paragraph1"><img src="logo.png"></p>
    <nav>
      <ul id="menu2">       // second menu
        <li><a href="#">Design</a></li>
        <li><a href="#">Entertainment</a></li>
        <li><a href="#">Everything else</a></li>
        <li><a href="#">Free</a></li>
        <li><a href="#">Health</a></li>
        <li><a href="#">Lifehack</a></li>
      </ul>
    </nav>
    <p id="paragraph2"><span>Categorized | </span>Design, Featured, Free, Wordpress</p>
    <article>
      <h2>43 Minimal And Really Clean Free Wordpess Themes</h2>
      <p class="paragraph3">Posted on 30 November 2009 by admin</p>
      <p><a id="button">Like</a></p>
      <p>
      Nullam non viverra magna, interdum convallis lacus. Vestibulum
      sit amet commodo quam. Proin eu nibh augue. Fusce consectetur
      fermentum accumsan. Curabitur dapibus augue ut felis lacinia,
      at pharetra turpis tincidunt. Cras viverra neque eu tellus
      aliquam, eget luctus enim aliquam. Phasellus nec gravida mi.
      Suspendisse posuere massa ut massa finibus, at ornare metus
      malesuada.
      </p>
      <p>
      Suspendisse commodo maximus mollis. Suspendisse mattis tellus
      eu euismod hendrerit. Morbi laoreet et tortor in egestas.
      Donec rhoncus auctor augue vel interdum. Morbi mollis ante
      vitae libero scelerisque malesuada. Praesent sed nisi at velit
      posuere vehicula id in magna. Nam venenatis, mauris convallis
      sagittis interdum, erat leo iaculis ex, vel consequat ex sem
      nec dolor. Cras eget ex in leo finibus viverra. In venenatis
      dapibus elit, vel faucibus quam facilisis vitae. Duis auctor
      lorem non ante tempor, sed lobortis sem volutpat. Nulla
      elementum, nisi vitae pretium commodo, felis enim malesuada
      nibh, vel tincidunt ante magna sit amet ligula. Donec ut
      vestibulum lacus, sit amet sollicitudin magna. Mauris sapien
      turpis, gravida volutpat urna non, sodales dictum lorem. Cras
      tristique aliquam elit, ut eleifend diam iaculis vel.          </p>
      </p>
    </article>
  </div>
</body>
</html>
```

exercise3.css

```
body{
  background-color: #a2a2a2;        // as specified
  margin: 0;
}
```

```
div {
   width: 960px;        // as specified
   margin: 0px auto; // align center    (top, right, bottom, left) (0px: top, bottom, auto: right, left)
   background-color: #ffffff;  // white background-color
   padding-bottom: 20px;      // padding is used to generate space around an element's content.
}

#menu1 {                first menu
   float: left;    // Wrapping. Without this, the next color setting would not take effect
   width: 100%;        // menu1 width in its container => the whole background is black
   padding: 0;
   margin: 0 0 1em 0; // em: Emphasized text, relative unit of measure
                       // e.g. 0.25em: 0.25 times the original character size
                       //There would be a white area at the top of the page
   list-style-type: none; // No list item marker is shown.
   background-color: #000; //          menu1 black background-color
}
#menu1 li { display:inline }   // with the next float:left:
                               // the menu items should be next to each other and not below each other
#menu1 a {
  float:left;
  padding: 0.25em 1em;         // It makes the bottom size larger
  margin: 0.25em 0 0.25em 1em;       // increase the distance between the menu buttons
  color: #DDD;          // menu1 text color
  text-decoration: none; // remove underlines from links
#menu1 a:hover { background-color:red} // It is used to select elements when you mouse over them.
                                       // The whole area behaves like a link.
p#paragraph1 {
   clear:left;    // Do not allow floating elements on the left side of this element
                  // https://www.w3schools.com/cssref/pr_class_clear.asp
                  // without this, there would be a small bar at the top of the page with the background
}

#menu2 {                // The second horizontal menu: similar to the first one
   float: left;
   width: 100%;
   padding: 0;
   margin: 0 0 1em 0;
   list-style-type: none;
   border-top: 0.25em dotted #b3b3b3;
   border-bottom: 0.25em solid #b3b3b3;
}
#menu2 li { display:inline }
#menu2 a {
  float:left;
  padding: 0.25em 1em;
  margin: 0.25em 0 0.25em 1em;
  color: #000;
  text-decoration: none;
  font-weight: bold;
}

p#paragraph2 {         // The text under the second menu
```

```
   clear: left;   // Do not allow floating elements on the left side of this element
                  // without this setting:
                  // There would be a narrow area between the second menu and this paragraph
   padding: 10px;       // It makes the paragraph size larger for each side of this paragraph
   color: #FF0E00;     // as specified
   font-size: x-large;
   font-weight: bold;
   background-color: #EEE;  // as specified
}
p#paragraph2 span {  // The first word of the paragraph
   color: #C0C0C0;    // as specified
}

p.paragraph3 {          // text under the H2 heading
   color: #C0C0C0;    // as specified
}

a#button {      // Like button
   color: #5C74C8;    // as specified
   padding: 3px 5px 1px 25px;         // It is used to generate space around an element's content (Like)
                                      // 25px: padding for the image
// There are four little images below each other
// 6px 6px: only the first image appears.
// #ECEEF5: background color as specified
   background: url("buttons.png") no-repeat 6px 6px #ECEEF5;
   border: 1px solid #CAD4E7;
   border-radius: 5px; // It defines the radius of the element's corners.
                       // https://www.w3schools.com/cssref/css3_pr_border-radius.asp
   text-decoration: none; // It is used to remove underlines from links
}

h2, p {                 // left margin for h2 headings and paragraphs
   margin-left: 20px;
}
```

4th Exercise – without solution files – practicing homework

Create the next design:


pixel juice
Home
About
Our Work
Contact
we squeeze the best from your website
We are pixel juice. We build creative, unique and user friendly websites and web applications.
We would love to work with you.
Work with us.
pixel juice is a team of creative designers and talented developers who are passionate about building high quality websites and web applications.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent id turpis. Suspendisse nec nibh eu lectus volutpat venenatis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Etiam lacinia. Pellentesque consequat. Maecenas tempor est eu est. Quisque non justo ut enim fringilla porttitor. Morbi turpis lorem, dictum non, condimentum ut, mattis et, risus.
Praesent faucibus lectus ac diam. Ut non felis a leo ornare lobortis. Nullam volutpat. Aliquam viverra. Duis nisi. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Duis turpis massa, placerat sed, scelerisque et, lobortis sed, eros.
Our Latest News
Nunc ac felis. Duis sed mi.
Sed egestas feugiat tortor. Praesent odio auctor nisl eget neque. Donec blandit.
Duis vitae turpis. Ut augue.
Pellentesque placerat turpis non risus. et augue bibendum facilisis. Donec cursus.
Nunc ac felis. Duis sed mi.
Sed egestas feugiat tortor. Praesent odio auctor nisl eget neque. Donec blandit.
Duis vitae turpis. Ut augue.
Pellentesque placerat turpis non risus. et augue bibendum facilisis. Donec cursus.
More News
Featured Project
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent id turpis. Suspendisse nec nibh eu lectus volutpat venenatis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere.
See more of our work
Client Login:
Username
Password
Lost password?
Go
Our Accessibility Policy:
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed leo neque, facilisis a, euismod non, eleifend quis, ligula. Nunc laoreet accumsan eros. Nam ullamcorper elit quis urna.
Lorem ipsum dolor sit amet, consectetur adipiscing elit
Contact us:
08452 246 187
Mon-Fri: 9am-5pm
Send us an Email

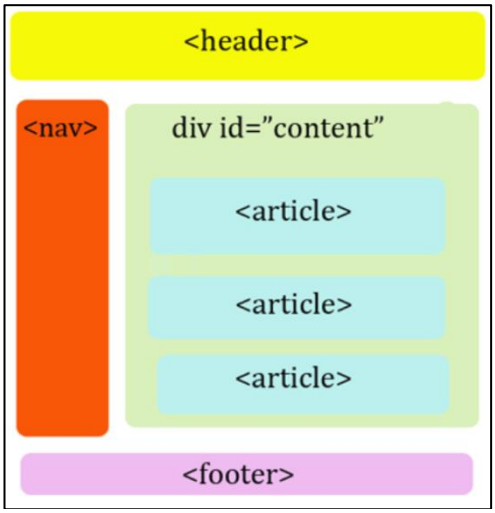
2 – HTML – CSS – header, nav, article, footer - uniformly displayed pages with menu – table, form

1st exercise – mini homepage: uniformly displayed pages with menu

Create a simple **uniformly displayed** website with 5 pages - 4 html and 1 css files – which has **menu**.
It will be a personal page. You can get the text from https://lipsum.com/

Look at the solution with CSS and without it. (e.g. rename the CSS file).
Try all CSS settings: look at it with the given setting and without it.

Uniform structure of the pages: header, footer, nav, content



The header has a logo and main title, a vertical menu on the left, the current content (with one or more <article> tags) in the middle, and a Copyright note in the footer.
Place all CSS definitions in the **style.css** file and use it consistently for the remaining pages.
The **index.html** is the home page, where a mini greeting with a photo (face.jpg) should be included. The image has border and aligned to the right.
The **contact.html** contains contact information and an embedded Google map.

We discuss these (these are ready in the solution): **index.html, style.css, contact.html,**
The rest should be done as homework.

For the two pages to be created independently:

The introduction.html contains the following:

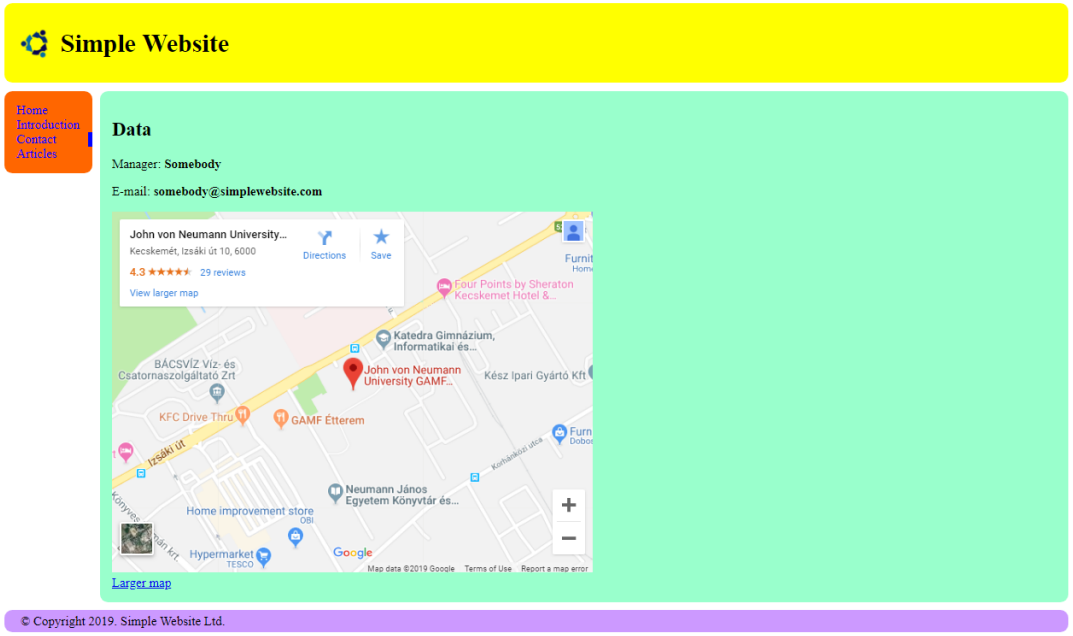
- **h1** the title 'Biography'
- **h2** the titles of the topics (e.g. Schools, Professional Knowledge)
- **strong** to highlight important words
- **abbr** or **acronym** in professional texts
- **ul, li** for linking to your school

The **articles.html** should be a page with at least 3 articles (**article** tag) and at least 2-2 paragraphs per article. Also create a mini table of contents above the articles.

In the left-hand menu, mark which menu item is current with a small blue line on the right.

Home:

Contact:



Solution only with the index.html, style.css and contact.html files. The rest must be created as practicing homework.

index.html

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
```

```
    <title>Simple Website Ltd.</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <header>
        <img src="logo.png" alt="logo">
        <h1>Simple Website</h1>
    </header>
    <div id="wrapper">          // The box (area) between the header and the footer
// aside: The <aside> tag defines some content aside from the content it is placed in.
//        The <aside> content could be placed as a sidebar in an article.
//        The <aside> tag is new in HTML5.
// https://www.w3schools.com/tags/tag_aside.asp
        <aside id="nav">          // the area to the left of <div id="content">
            <nav>           // box for the menu: list with links
// We put the menu items in a list because it will be easier to format (CSS)
                <ul>          // list with links
                    <li class="active"><a href="index.html">Home</a></li>
                    <li><a href="introduction.html">Introduction</a></li>
                    <li><a href="contact.html">Contact</a></li>
                    <li><a href="articles.html">Articles</a></li>
                </ul>
            </nav>
        </aside>
        <div id="content">        // the large green background content box
            <h2>Welcome</h2>
            <img src="face.jpg">
            <h3>What is Lorem Ipsum?</h3>
            <p>Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has
been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of
type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the
leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the
release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing
software like Aldus PageMaker including versions of Lorem Ipsum.</p>
            <h3>Where does it come from?</h3>
            <p>Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of
classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin
professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words,
consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature,
discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus
Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a
treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum,
"Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.</p>
            <p>The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those
interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also
reproduced in their exact original form, accompanied by English versions from the 1914 translation by H.
Rackham.</p>
            <h3>Why do we use it?</h3>
            <p>It is a long established fact that a reader will be distracted by the readable content of a page
when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal
distribution of letters, as opposed to using 'Content here, content here', making it look like readable
English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default
model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various
```

```
versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour
and the like).</p>
        <h3>Where can I get some?</h3>
        <p>There are many variations of passages of Lorem Ipsum available, but the majority have
suffered alteration in some form, by injected humour, or randomised words which don't look even slightly
believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything
embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat
predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of
over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum
which looks reasonable. The generated Lorem Ipsum is therefore always free from repetition, injected
humour, or non-characteristic words etc.</p>
      </div>
   </div>
   <footer>
      &copy; Copyright 2019. Simple Website Ltd.
   </footer>
</body>
</html>
```

style.css

```
header {
   border-radius: 10px;          // the radius of the element's corners.
   background-color: yellow;
   padding: 10px;       // padding on each side: increase box size
   margin: 10px;        // margin on each side: the area outside the invisible border
}
header img {   // header image
   float: left;    // It places an element (img) on the left side of its container, allowing text and inline
                   // elements to wrap around it. The "Simple Website" text is in the same line as the image
   margin: 20px 10px 0px 10px;       // margin for the image
}
div#wrapper {           // the box with "wrapper" ID.
   margin: 10px;
}
aside {           // the wide vertical area on the left side
// you can also put the boxes next to each other like this:
   display: table-cell; // Displays an element as an inline element.
                        // otherwise the next box would be under this box
   vertical-align: top;  // to the top of the left area. Otherwise it would be in the middle of the area
}
nav {   // box for the navigation
   margin: 0px 10px 0px 0px;          // top, right, bottom, left
   top: 0px;
   border-radius: 10px;          // rounded rectangle with the given radius
   background-color: #FF6600;
}
nav ul {          // The ul part of the navigation box: settings for the whole list
   list-style-type: none;          // Hide list-item marker in a list.
   margin: 0px;
   padding: 15px 0px;          // 15px: bottom and top, 0px: left and right
}
nav ul li {       // setting for the list elements in ul in the navigation
```

```
    padding: 0px 15px;;
}
// Setting for the active menu element:
// Use a thick blue right border to indicate which link is active
nav ul li.active {
    border-right: 5px solid blue;          // the active menu element right border is 5px blue
}
nav ul a {        // the link elements in ul in navigation
    text-decoration: none;        // remove underlines from links
    color: blue; // the link colour: otherwise the link/visited link/… color settings would be applied
}
// #: it's ID in the HTM.        The div box with id="content":        <div id="content">
//        the large green background content boksz
div#content {
    display: table-cell;  // otherwise this box would be under the <aside> box
    width: 100%;
    border-radius: 10px;
    background-color: #99FFCC;
    padding: 15px;
}
div#content img{        // For the img element in the <div id="content"> box
    float: right;  // Wraps the image around the text from the left: the image will be on the right
    width: 20em;          // 20em: relative unit of measurement. 20 times the original character size
    height: auto;          // the image proportions are preserved
    margin: 0 0 15px 15px;      // top, right, bottom, left
    padding: 3px;         // padding for each side: the distance between the image and the border
    border: 3px solid blue;
}
footer {
    border-radius: 10px;          // rounded rectangle with the given radius
    margin: 10px;         // margin for each side
    padding: 5px 20px;          // 5px: top, bottom      20px: right, left
    background-color: #CC99FF;
}
```

contact.html

Only the new features are highlighted. The structure of the page is the same as the previous page.

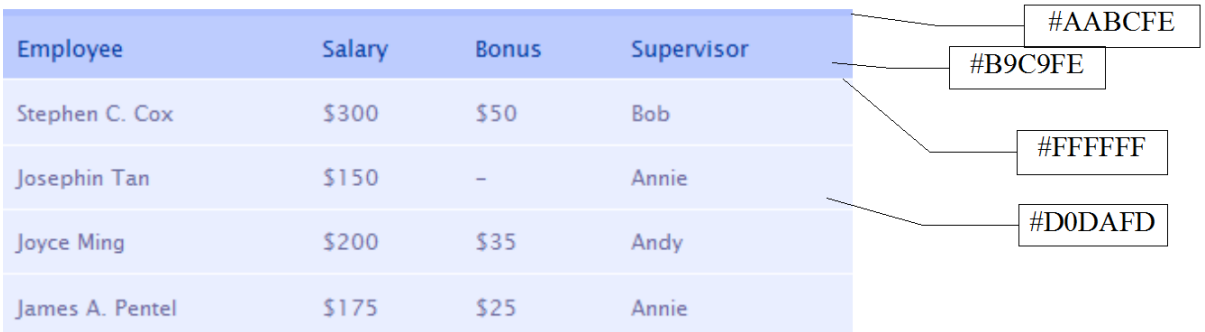
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Simple Website Ltd.</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <header>
        <img src="logo.png" alt="logo">
        <h1>Simple Website</h1>
    </header>
    <div id="wrapper">
        <aside id="nav">
            <nav>
```

```
            <ul>
               <li><a href="index.html">Home</a></li>
               <li><a href="introduction.html">Introduction</a></li>
               <li class="active"><a href="contact.html">Contact</a></li>
               <li><a href="articles.html">Articles</a></li>
            </ul>
         </nav>
      </aside>
      <div id="content">
         <h2>Data</h2>
         <p>Manager: <strong>Somebody</strong></p>
         <p>E-mail: <strong>somebody@simplewebsite.com</strong></p>
// Google map
// An inline frame is used to embed another document within the current HTML document.
// https://www.w3schools.com/tags/tag_iframe.asp
         <iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2726.3375296155727!2d19.666950
91525771!3d46.89607994478184!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x4743da7a6
c479e1d%3A0xc8292b3f6dc69e7f!2sPallasz+Ath%C3%A9n%C3%A9+Egyetem+GAMF+Kar!5e0!3m2
!1shu!2shu!4v1475753185783" width="600" height="450" frameborder="0" style="border:0"
allowfullscreen></iframe>
         <br>
// target="_blank": Open a link in a new browser window
// https://www.w3schools.com/tags/att_a_target.asp
         <a target="_blank"
href="https://www.google.hu/maps/place/Pallasz+Ath%C3%A9n%C3%A9+Egyetem+GAMF+Kar/@46.
8960799,19.6669509,17z/data=!3m1!4b1!4m5!3m4!1s0x4743da7a6c479e1d:0xc8292b3f6dc69e7f!8m2!
3d46.8960763!4d19.6691396?hl=hu">Larger map</a>
      </div>
   </div>
   <footer>
// &copy; code for this character: ©
      &copy; Copyright 2019. Simple Website Ltd.
   </footer>
</body>
</html>
```

2nd exercise - Table

Look at the solution with CSS and without it. (e.g. rename the CSS file).
Try all CSS settings: look at it with the given setting and without it.

Create the next table:



Employee	Salary	Bonus	Supervisor
Stephen C. Cox	$300	$50	Bob
Josephin Tan	$150	-	Annie
Joyce Ming	$200	$35	Andy
James A. Pentel	$175	$25	Annie

- The first row with th (table header) tag, the others with td tags (A table cell is defined with the <td> tag.)
- The text colours: #1E459E and #7274A5

https://www.w3schools.com/html/html_tables.asp

Solution

table.html

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <title>Table</title>
   <link rel="stylesheet" type="text/css" href="table.css">
</head>
<body>
<table>
   <caption>3. labor 2. feladat:<br>TABLE</caption>
   <tr>          // New row
      <th>Employee</th>      // table header cell
      <th>Salary</th>
      <th>Bonus</th>
      <th>Supervisor</th>
   </tr>
   <tr>
      <td>Stephen C. Cox</td>        // table data/cell
      <td>$300</td>
      <td>$50</td>
      <td>Bob</td>
   </tr>
   <tr>
      <td>Josephin Tan</td>
      <td>$150</td>
      <td>-</td>
      <td>Annie</td>
   </tr>
   <tr>
      <td>Joyce Ming</td>
      <td>$200</td>
      <td>$35</td>
      <td>Andy</td>
   </tr>
   <tr>
      <td>James A. Pentel</td>
      <td>$175</td>
      <td>$25</td>
      <td>Annie</td>
   </tr>
</table>
</body>
</html>
```

table.css

```
table {
// It sets whether table borders should collapse into a single border or be separated as in standard HTML.
// https://www.w3schools.com/cssref/pr_border-collapse.asp
   border-collapse: collapse;
}
th, td {
   padding: 10px 50px 10px 5px;      // top, right, bottom, left
   text-align: left;
   font-family: sans-serif;
   font-size: x-small;
}
th {      // table header cell
   color: #1E459E;
   background-color: #B9C9FE;
   border-top: 5px solid #AABCFE;
}
td {      // table data/cell
   color: #7274A5;
   background-color: #D0DAFD;
   border-top: 3px solid #FFFFFF;
}
```

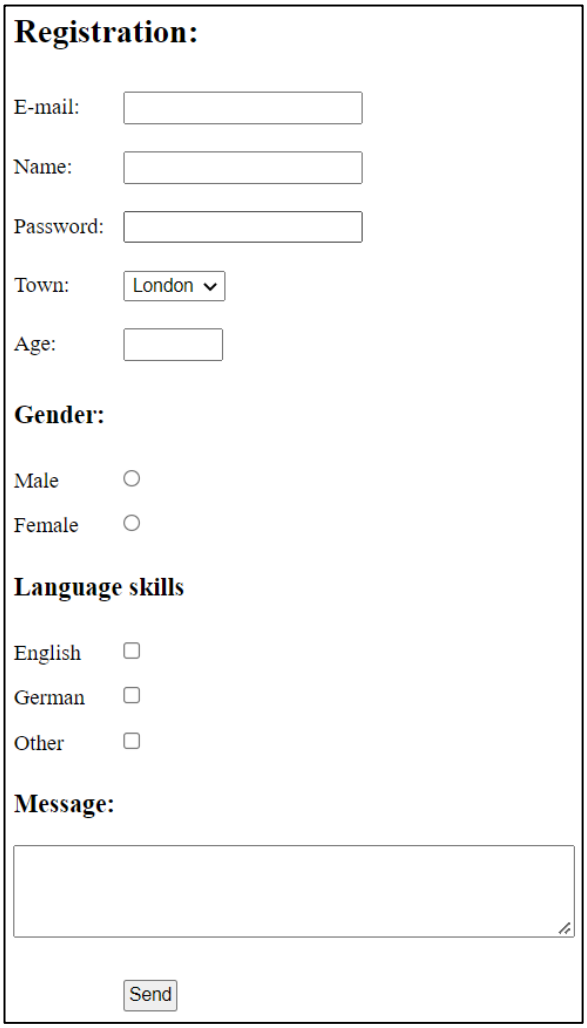
3rd exercise –Build the table into the Simple Website – Practicing Homework – Only informative text

Give another menu element (Table) to the first exercise site. When you click on this link the main content will be the table.

Practicing Homework. You can find the solution in the **Solutions.zip** file.

4th exercise - Form

Create the next Registration form:



Check the fields' content:
- E-mail: accept only email addresses that follow the standards. Required
- Name: at least 8 characters in length. Uppercase, lowercase, spaces. Required
- Password: at least 6, at most 12 characters. Required
- Town: list options: London, Paris, Berlin, Rome
- Age: only numeric input, between 10 and 120.

Style:
- label width: 80px
- for input and select elements: top, bottom margin: 10px, padding: 2px;

Clicking the **Send** button: submit the form data to a form-handler: print the form data.
We will solve it next week with server side programming.

index.html

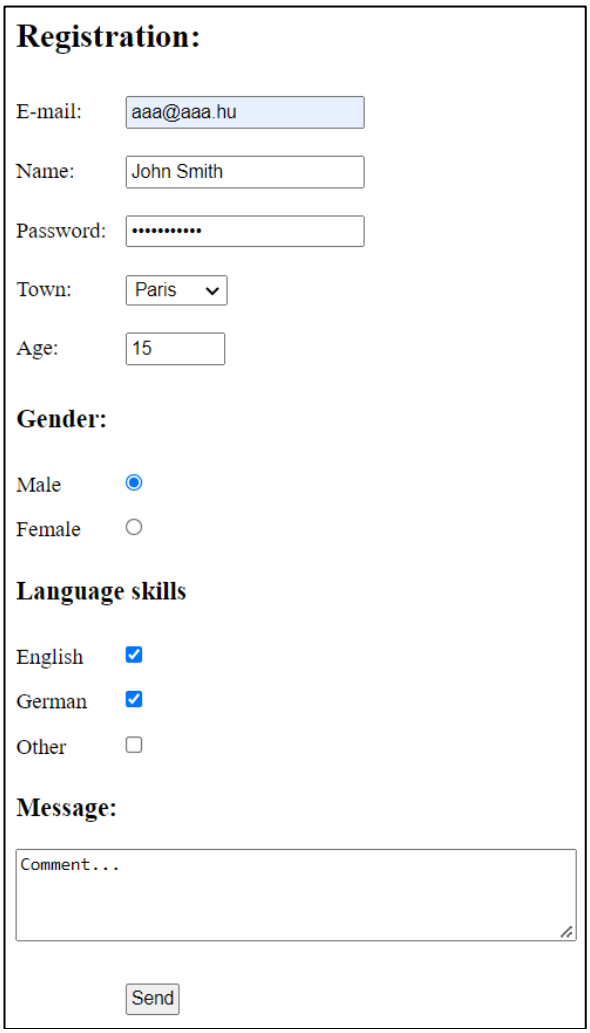
```
<!DOCTYPE html>
<html>
<head>
   <title>Form</title>
   <meta charset="utf-8">
   <link type="text/css" rel="stylesheet" href="style.css">
</head>
<body>
      <h2>Registration:</h2>
```

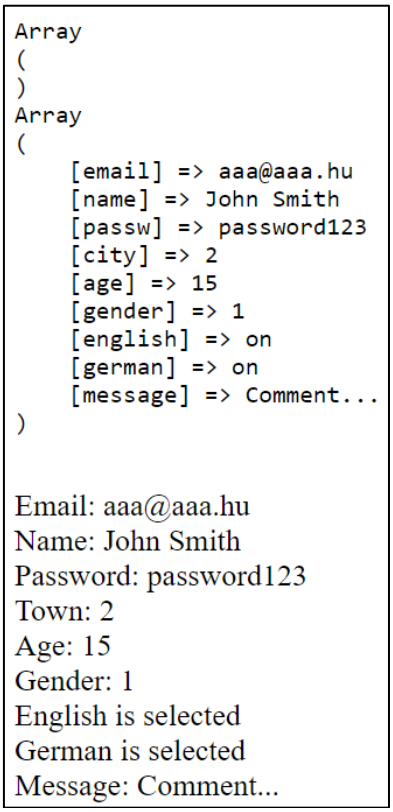
```
	<form action="process.php" method="post">
	<label>E-mail:</label><input type="email" name="email" required><br>
	<label>Name:</label><input type="text" name="name" pattern="[A-Z,a-z, ]{8}[A-Z,a-z, ]*"
required><br>
	<label>Password:</label><input type="password" name="passw" pattern=".{6,12}" required><br>
		<label>Town:</label><select name="city">
			<option value="1">London</option>
			<option value="2">Paris</option>
			<option value="3">Berlin</option>
			<option value="4">Rome</option>
		</select><br>
	<label>Age:</label><input type="number" name="age" min="10" max="120" required><br>
	  <h3>Gender:</h3>
	  <label>Male</label><input type="radio" name="gender"  value="1"><br/>
	  <label>Female</label><input type="radio" name="gender"  value="2">
	  <h3>Language skills</h3>
	  <label>English</label><input type="checkbox" name="english">
	  <br>
	  <label>German</label><input type="checkbox" name="german">
	  <br>
	  <label>Other</label><input type="checkbox" name="other">
	  <p></p>
	  <h3>Message:</h3>
	  <textarea name="message" rows="4" cols="50"></textarea>
	  <p></p>
	<label></label><input type="submit" value="Send">
	</form>
</body>
</html>
```

style.css

```
label {
   width: 80px;
   display: inline-block;
}
input, select {
   margin: 10px 0;
   padding: 2px;
}
```

Server is required to run. We'll use a server next week. Try it without server: what happens when you click on the Sent button?





process.php

```
<!DOCTYPE html>
<html>
<head>
   <title>Input data</title>
   <meta charset="utf-8">
</head>
<body>
        <pre><?php
                print_r($_GET);                 // print the $_GET global variable
                print_r($_POST);        // print the $_POST global variable
                ?>
        </pre>
        <?php
                echo "Email: ".$_POST["email"]."<br>";
                echo "Name: ".$_POST["name"]."<br>";
                echo "Password: ".$_POST["passw"]."<br>";
                echo "Town: ".$_POST["city"]."<br>";
                echo "Age: ".$_POST["age"]."<br>";
                echo "Gender: ".$_POST["gender"]."<br>";
                if(isset($_POST['english'])) echo "English is selected<br>";
                if(isset($_POST['german'])) echo "German is selected<br>";
                if(isset($_POST['other'])) echo "Other is selected<br>";
```

```
            echo "Message: ".$_POST["message"]."<br>";
        ?>
</body>
</html>
```

The <pre> tag defines preformatted text. Without the <pre> tag, it would not be formatted, but would be printed in one line. Try it!

5th exercise - Build the Form into the Simple Website – Practicing Homework – Only informative text

Give another menu element (Form) to the first exercise site. When you click on this link the main content will be the Form.
You can find the solution is **Solutions.zip** file.

6th exercise – a larger Form with more elements - Only informative text

You can find its solution is **Solutions.zip** file.

Page design using ready-made templates, themes, page sections and snippets – Only informative text

In practice, the pages do not have to be built from scratch, there are many templates and page snippets on the Internet

Templates, themes for entire pages

Free HTML, CSS themes:
Search:
free HTML CSS themes free
free HTML CSS templates free
https://www.free-css.com/free-css-templates
https://templatemo.com/
https://nicepage.com/css-templates
https://www.tooplate.com/free-templates
https://colorlib.com/wp/free-css-website-templates/
https://www.w3schools.com/w3css/w3css_templates.asp
https://mobirise.com/html-templates/
https://all-free-download.com/free-website-templates/free-html-css-templates.html
https://www.jotform.com/blog/blast-from-the-past-4-best-free-htmlcss-themes-86674/

Responsive HTML, CSS free themes
Search:
free responsive HTML CSS templates
free responsive HTML CSS themes

free responsive HTML5 CSS3 templates
free responsive HTML5 CSS3 themes
https://www.free-css.com/template-categories/responsive
https://templatemo.com/
https://www.w3schools.com/css/css_rwd_templates.asp
https://html5up.net/
https://nicepage.com/html-templates
https://speckyboy.com/free-responsive-html5-web-templates/
https://www.toocss.com/free-responsive-html-css-templates/
https://mobirise.com/html-templates/
https://nicepage.com/css-templates
https://enlear.academy/17-website-templates-html-free-download-630bf4ffeb83
https://colorlib.com/wp/free-responsive-website-templates/
https://www.mockplus.com/blog/post/free-responsive-html5-web-design-templates

Free Bootstrap themes, templates
Search:
Free responsive Bootstrap templates
Free responsive Bootstrap themes
https://bootstrapmade.com/
https://themewagon.com/themes/
https://themefisher.com/free-bootstrap-templates
https://startbootstrap.com/themes
http://visuallightbox.com/content/40-brilliant-free-bootstrap-templates-2016-86.html
https://bootstraptaste.com/
https://colorlib.com/wp/free-bootstrap-4-website-templates/
https://freshdesignweb.com/free-bootstrap-templates/
https://www.creative-tim.com/bootstrap-themes/free
https://mobirise.com/bootstrap-template/

Page sections, snippets

If we are only looking for a solution for a specific part of the page:

Search:
free HTML CSS snippets
free CSS snippets
https://www.w3schools.com/howto/default.asp
https://css-tricks.com/snippets/
https://freefrontend.com/css-code-examples/
https://codemyui.com/tag/pure-css/
https://codepen.io/jleonard/pens/popular
https://designseer.com/category/coding/

Search:
free Bootstrap snippets
https://bootsnipp.com/
https://startbootstrap.com/snippets
https://www.bootdey.com/
https://www.tutorialrepublic.com/snippets/gallery.php
https://bootstrapious.com/snippets
https://bbbootstrap.com/

https://colorlib.com/wp/category/snippets/
https://gosnippets.com/
https://mdbootstrap.com/snippets/
https://library.livecanvas.com/sections/

If only e.g. we are looking for samples on a table:

Search:
free HTML CSS table snippets
free CSS table snippets
https://colorlib.com/wp/css3-table-templates/
https://uicookies.com/css-table-templates/
https://speckyboy.com/responsive-html-table-techniques/
https://flatlogic.com/blog/37-simple-css3-html-table-templates-examples/
https://freefrontend.com/css-tables/
https://csshint.com/html-css-table/
https://www.sliderrevolution.com/resources/css-tables/
https://templatefor.net/css-tables/
https://www.web-eau.net/blog/10-examples-pricing-table-bootstrap

Search:
free Bootstrap table snippets
https://bootsnipp.com/tags/table
https://bootstrapious.com/p/tables
https://colorlib.com/wp/bootstrap-tables/
https://freefrontend.com/bootstrap-tables/
https://www.bootdey.com/snippets/tagged/table
https://www.tutorialrepublic.com/snippets/gallery.php?tag=table
https://getbootstrap.com/docs/4.1/content/tables/
https://startbootstrap.com/snippets
https://mdbootstrap.com/docs/standard/data/tables/examples-and-customization/

3 - Server side programming (PHP) - multiplication table, form generation, greatest common divisor, calculator

Introduction

We used only the browser so far to solve the tasks, because the browser understands the HTML and CSS codes. The browser doesn't understand the PHP codes, so we need a **server** that understands the PHP instructions and sends the HTML and CSS codes to the client (browser).
PHP: server side programming language.
HTML, CSS, Javascript run in the browser, PHP on the server.

Create a folder, e.g. **c:\exercise**. It'll be the **work-folder**, copy the files here.

PHP with Development Serverrel – We use it for a few weeks.

It's a weaker solution than the one we'll use in a few weeks, but it's still useful to know. e.g. simple server, usage of ports.

In a few weeks, we'll use the Apache web server in XAMPP, which has more features and more configuration options. For example, the use of the .htaccess file.

This web server is designed to aid application development. It may also be useful for testing purposes or for application demonstrations that are run in controlled environments.
https://www.php.net/manual/en/features.commandline.webserver.php

A,
With the PHP package in the c:\xampp\php folder – We use it, because PHP is already on the computer with XAMPP, you don't need to install it.

Start PHP Development Server in Command line. You can access the Command line:

- **Visual Studio Code**
  => File menu / Open folder: open c:\exercise work-folder
  => Terminal menu / New Terminal: It opens the Command prompt in VS Code.

  or
- **Windows: Start / Windows system / Command prompt**

  Goto folder e.g. cd exercise
  Go out of folder: cd ..

A, without PATH: simpler solution
\xampp\php\php -S localhost:8000 (8000-es port)

Development Server (http://localhost:8000) started

B, Set the \xampp\php folder to PATH, it it isn't there, and start it:
path (prints the contents of PATH)

```
set PATH=%PATH%;C:\xampp\php
php -S localhost:8000
```

B,
Downloading the PHP package from the Internet – more preparation, we don't use it

Download the PHP Zip version:
https://windows.php.net/download/
Unpack its contents into e.g. the c:\php folder

Starting PHP Development Server from the command line
\php\php -S localhost:8000

If you type to browser: http://localhost:8000/ => error message: The requested resource / was not found on this server.

Create an **index.html** or **index.php** file with the following content: **Hello**
then for http://localhost:8000/ it will print: Hello

Download and install XAMPP at home: c:\xampp
https://www.apachefriends.org/download.html
XAMPP is a free, easy-to-install **Apache** distribution including **MariaDB**, **PHP** and Perl components.

Introductory task - phpinfo()

Not in Solutions.zip

exercise.php

```
<?php
phpinfo();
```

In Browser:
http://localhost:8000/exercise.php
It prints a lot of information about the running PHP.

When using the PHP Development server, you should pay attention to the following:
At the command line: clicking on the panel will switch to SELECT mode, which will temporarily stop the server!
This is also written in the top line: Select …..
Try with the previous task!

In this case, you must exit SELECT mode with ESC!

This is true for all command line programs (e.g. NodeJS, Laravel, ...), not only for PHP.

https://forums.pmmp.io/threads/powershell-cmd-select-mode-pauses-server.6666/
https://stackoverflow.com/questions/33883530/why-is-my-command-prompt-freezing-on-windows-10

exercise.php/aaa/bbb/ccc => ignores the part after exercise.php

http://localhost:8000/exercise.php/aaa/bbb/ccc
It gives the same result as the previous one:
http://localhost:8000/exercise.php

1st exercise - Multiplication table

Create a Multiplication table in PHP.

First version

1 * 1 = 1	2 * 1 = 2	3 * 1 = 3	4 * 1 = 4	5 * 1 = 5	6 * 1 = 6	7 * 1 = 7	8 * 1 = 8	9 * 1 = 9	10 * 1 = 10
1 * 2 = 2	2 * 2 = 4	3 * 2 = 6	4 * 2 = 8	5 * 2 = 10	6 * 2 = 12	7 * 2 = 14	8 * 2 = 16	9 * 2 = 18	10 * 2 = 20
1 * 3 = 3	2 * 3 = 6	3 * 3 = 9	4 * 3 = 12	5 * 3 = 15	6 * 3 = 18	7 * 3 = 21	8 * 3 = 24	9 * 3 = 27	10 * 3 = 30
1 * 4 = 4	2 * 4 = 8	3 * 4 = 12	4 * 4 = 16	5 * 4 = 20	6 * 4 = 24	7 * 4 = 28	8 * 4 = 32	9 * 4 = 36	10 * 4 = 40
1 * 5 = 5	2 * 5 = 10	3 * 5 = 15	4 * 5 = 20	5 * 5 = 25	6 * 5 = 30	7 * 5 = 35	8 * 5 = 40	9 * 5 = 45	10 * 5 = 50
1 * 6 = 6	2 * 6 = 12	3 * 6 = 18	4 * 6 = 24	5 * 6 = 30	6 * 6 = 36	7 * 6 = 42	8 * 6 = 48	9 * 6 = 54	10 * 6 = 60
1 * 7 = 7	2 * 7 = 14	3 * 7 = 21	4 * 7 = 28	5 * 7 = 35	6 * 7 = 42	7 * 7 = 49	8 * 7 = 56	9 * 7 = 63	10 * 7 = 70
1 * 8 = 8	2 * 8 = 16	3 * 8 = 24	4 * 8 = 32	5 * 8 = 40	6 * 8 = 48	7 * 8 = 56	8 * 8 = 64	9 * 8 = 72	10 * 8 = 80
1 * 9 = 9	2 * 9 = 18	3 * 9 = 27	4 * 9 = 36	5 * 9 = 45	6 * 9 = 54	7 * 9 = 63	8 * 9 = 72	9 * 9 = 81	10 * 9 = 90
1 * 10 = 10	2 * 10 = 20	3 * 10 = 30	4 * 10 = 40	5 * 10 = 50	6 * 10 = 60	7 * 10 = 70	8 * 10 = 80	9 * 10 = 90	10 * 10 = 100

This table contains 10 cells.

Second version

1 * 1 = 1	1 * 2 = 2	1 * 3 = 3	1 * 4 = 4	1 * 5 = 5	1 * 6 = 6	1 * 7 = 7	1 * 8 = 8	1 * 9 = 9	1 * 10 = 10
2 * 1 = 2	2 * 2 = 4	2 * 3 = 6	2 * 4 = 8	2 * 5 = 10	2 * 6 = 12	2 * 7 = 14	2 * 8 = 16	2 * 9 = 18	2 * 10 = 20
3 * 1 = 3	3 * 2 = 6	3 * 3 = 9	3 * 4 = 12	3 * 5 = 15	3 * 6 = 18	3 * 7 = 21	3 * 8 = 24	3 * 9 = 27	3 * 10 = 30
4 * 1 = 4	4 * 2 = 8	4 * 3 = 12	4 * 4 = 16	4 * 5 = 20	4 * 6 = 24	4 * 7 = 28	4 * 8 = 32	4 * 9 = 36	4 * 10 = 40
5 * 1 = 5	5 * 2 = 10	5 * 3 = 15	5 * 4 = 20	5 * 5 = 25	5 * 6 = 30	5 * 7 = 35	5 * 8 = 40	5 * 9 = 45	5 * 10 = 50
6 * 1 = 6	6 * 2 = 12	6 * 3 = 18	6 * 4 = 24	6 * 5 = 30	6 * 6 = 36	6 * 7 = 42	6 * 8 = 48	6 * 9 = 54	6 * 10 = 60
7 * 1 = 7	7 * 2 = 14	7 * 3 = 21	7 * 4 = 28	7 * 5 = 35	7 * 6 = 42	7 * 7 = 49	7 * 8 = 56	7 * 9 = 63	7 * 10 = 70
8 * 1 = 8	8 * 2 = 16	8 * 3 = 24	8 * 4 = 32	8 * 5 = 40	8 * 6 = 48	8 * 7 = 56	8 * 8 = 64	8 * 9 = 72	8 * 10 = 80
9 * 1 = 9	9 * 2 = 18	9 * 3 = 27	9 * 4 = 36	9 * 5 = 45	9 * 6 = 54	9 * 7 = 63	9 * 8 = 72	9 * 9 = 81	9 * 10 = 90
10 * 1 = 10	10 * 2 = 20	10 * 3 = 30	10 * 4 = 40	10 * 5 = 50	10 * 6 = 60	10 * 7 = 70	10 * 8 = 80	10 * 9 = 90	10 * 10 = 100

This table contains 10*10 = 100 cells.

Solution
Both versions are in one file.
We generate the HTML code with PHP =>
View the source of the page in a browser! (no PHP, just HTML)

exercise1.php

```
<!DOCTYPE HTML>
<html>
   <head>
      <meta charset="utf-8">
      <title>Multiplication table</title>
   </head>
   <body>
      <h2>First version</h2>
      <table border="1">
```

```
        <tbody>
        <tr>    // table row
// HTML and PHP instructions in one file:
// PHP part:    <?php …….  ?>
            <?php for ($cell=1; $cell<11; $cell++) { ?>        // for loop
              <td>       // new table data cell
                <?php
// 10 rows in each cell
                for ($row=1; $row<11; $row++) {
                  echo "$cell * $row = ";     // e.g.   5*7 =
                  echo $cell*$row;    // the multiplication of variables
                  echo '<br>';
                }
                ?>
              </td>
            <?php } ?>
          </tr>
          </tbody>
      </table>

      <h2>Second version</h2>                  // There is nothing new in it – Only informative text
      <table border="1">
        <tbody>
        <?php for ($row=1; $row<11; $row++) { ?>
          <tr>
            <?php
            for ($cell=1; $cell<11; $cell++)
              echo "<td>$row * $cell = " . $cell*$row. "</td>";
            ?>
          </tr>
        <?php } ?>
        </tbody>
      </table>
  </body>
</html>
```

http://localhost:8000/exercise1.php
prints the multiplication tables.

The default port for HTTP is 80 – If you use this, you don't need to specify the port in the URL

> if there is no port assigned to an HTTP connection, Port 80 should be used.
> https://www.techopedia.com/definition/15709/port-80
>
> It is the default port for HTTP (Hypertext Transfer Protocol) network traffic. When you type a website URL into your browser, it connects to the web server on port 80 to request the web page.
> https://sslinsights.com/what-is-port-80/

(For HTTPS, the default port is 443)

Try the previous task with port 80

Starting the server
\xampp\php\php -S localhost:80

Both are good:
http://localhost/exercise1.php
http://localhost:80/exercise1.php

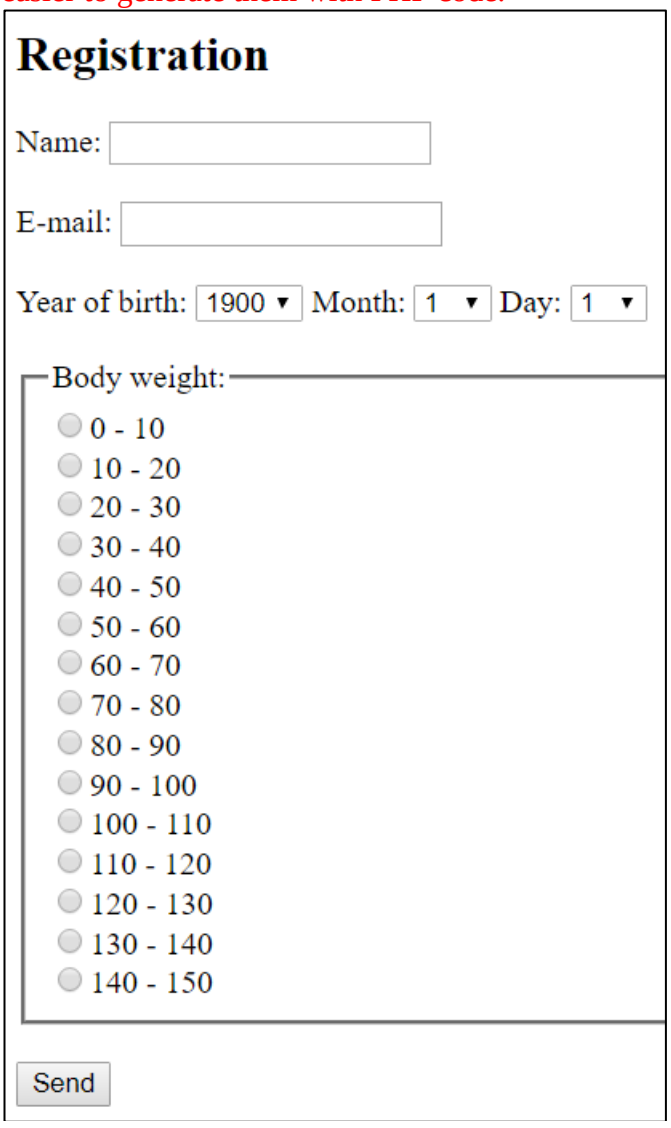
From here we use port 80.

2nd exercise: Form generation

Create a registration form :

- Name and email: text
- year of birth: 1900-2010, month: 1-12, day: 1-31 in drop-down lists (use PHP for loops)
- weight: 0-10, 10-20, … 140-150 with Radio Buttons (use for loop)

We could create the form with HTML code, but due to the many (111) list items and radio buttons, it is easier to generate them with PHP code.



Solution

exercise2.php

```
<!DOCTYPE HTML>
<html>
   <head>
      <meta charset="utf-8">
      <title>Form</title>
   </head>
   <body>
      <h2>Registration</h2>
      <form action="registration.php">
         <label>Name: <input name="name" type="text"></label><br><br>
         <label>E-mail: <input name="email" type="text"></label><br><br>
         <label>Year of birth: <select name="year">          // drop-down list
         // We have to create this HTML code with PHP:
         //       <option value="1900">1900</option>
         //       <option value="1901">1901</option>
               <?php for ($year=1900; $year<=2010; $year++) { ?>
                  <option value="<?php echo $year ?>"><?php echo $year ?></option>
               <?php } ?>
            </select></label>
         <label>Month: <select name="month">// The other two drop-down lists are similar to the first one
               <?php for ($month=1; $month<=12; $month++) { ?>
                  <option value="<?php echo $month ?>"><?php echo $month ?></option>
               <?php } ?>
            </select></label>
         <label>Day: <select name="day">
               <?php for ($day=1; $day<=31; $day++) { ?>
                  <option value="<?php echo $day ?>"><?php echo $day ?></option>
               <?php } ?>
            </select></label>
         <br><br>
// <fieldset>: to group elements in the form
//       Group the radio buttons: one of them can be selected
         <fieldset>
            <legend>Body weight:</legend>
// We have to create this HTML code with PHP:
//       <label><input name="weight" value="0" type="radio">0 - 10<br></label>
//       <label><input name="suly" value="10" type="radio">10 - 20<br></label>
            <?php for ($weight=0; $weight<=140; $weight+=10) { ?>
               <label><input name="weight" value="<?php echo $weight ?>" type="radio"><?php echo
$weight . ' - ' . ($weight+10); ?><br></label>                    <?php } ?>
         </fieldset>
         <br>
         <input value="Send" type="submit">
      </form>
   </body>
</html>
```

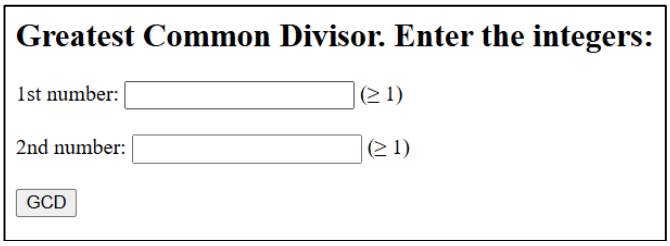
registration.php

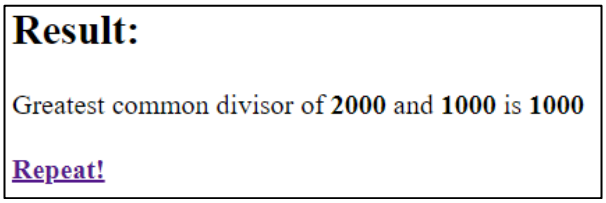
```
<?php
   print_r($_GET);
   print_r($_POST);
?>
```

What happens after clicking the button?
- what does it print?
- what will be the URL?
    => We did not specify the method (Get, Post) for Form: default setting: Get

3rd exercise - Greatest common divisor

Calculate the greatest common divisor of two positive integers. When entering the numbers, we check in the HTML code that they are positive integers.
Greatest common divisor: the largest number that divides both of them without leaving a remainder.





Solution-A - with two files (exercise3.html, exercise3.php)

exercise3.html

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Greatest Common Divisor</title>
   </head>
   <body>
      <h2>Greatest Common Divisor. Write the integers:</h2>
      <form action="exercise3.php" method="POST">
// At least one digit from 1-9. It can also be followed by digits from 0-9:
//       "[1-9][0-9]*":          first digit: 1-9;          [0-9]*: zero or more digits 0-9
//                                                                  &ge;   greater or equal character: ≥
         1st number: <input type="text" name="num1" pattern="[1-9][0-9]*" required> (&ge; 1)<br><br>
         2nd number: <input type="text" name="num2" pattern="[1-9][0-9]*" required> (&ge; 1)<br><br>
         <input type="submit" value="GCD">
      </form>
   </body>
</html>
```

exercise3.php

Solution A/1 – With Linear Search

In a loop, we move down from the smaller number in steps of 1 until both numbers are divisible by the loop variable. If they are divisible, then the loop variable is the common divisor. When we have found the first such number, we exit the loop: this will be the greatest common divisor.

```
<?php
// If there are both "num1" and "num2" keys in the $_POST array:
//        e.g. if task3.php is called first: the $_POST array is empty!
   if(isset($_POST["num1"]) && isset($_POST["num2"]))
   {
// we define two variables so that we don't have to use this long one for every step: $_POST["num1"]
      $a = $_POST["num1"];
      $b = $_POST["num2"];
      if($a<=$b) $smaller = $a;
         else $smaller = $b;
      for($i=$smaller;$i>=1;$i--)
         if($a%$i==0 && $b%$i==0){
            $gcd=$i;
            break;
         }
// $output: we print it in the HTML section.
      $output = "<h2>Result:</h2> Greatest common divisor of <strong>".$a."</strong> and
<strong>".$b."</strong> is <strong>".$gcd."<strong>";
   }
   else
// When do we get here? => call exercise3.php first: the $_POST array is empty!
      $output  = "<h2>Incorrect call: There is no data to process!</h2>";
?>
```

// It would work without the red codes, but exercise3.php would produce a non-standard HTML page:

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Greatest common divisor</title>
   </head>
   <body>
      <?php echo $output; ?>
      <br><br>
      <a href="exercise3.html">Repeat!</a>
   </body>
</html>
```

A/2. solution – With Euclidean algorithm

more efficient, faster (consists of fewer steps)
This method is not as clear as the previous one, but it has been proven to work well

https://en.wikipedia.org/wiki/Euclidean_algorithm

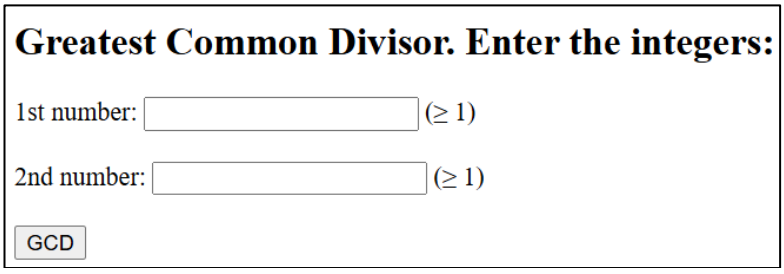
We check which of the two numbers is larger, and subtract the smaller one from it. We repeat this until the two numbers are equal. At this point, we get the greatest common divisor we are looking for.

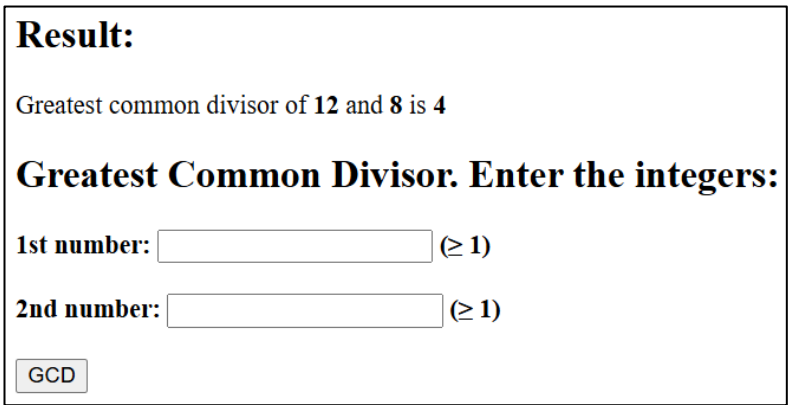
Just briefly, discussing the new features:

exercise3.php

```
<?php
   if(isset($_POST["num1"]) && isset($_POST["num2"]))
   {
// We will use the $_POST values for the output.
      $a = $_POST["num1"];
      $b = $_POST["num2"];
      while($a != $b)
         if($a < $b)
            $b -= $a;
         else
            $a -= $b;
// We get the lowest common divisor in the variable $a (it will also be the same in $b, because $b=$a)
      $output = "<h2>Result:</h2> Greatest common divisor of <strong>".$_POST["num1"]."</strong>
and <strong>".$_POST["num2"]."</strong> is <strong>".$a."<strong>";
   }
   else
      $output  = "<h2>Incorrect call: There is no data to process!</h2>";
?>

<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Greatest common divisor</title>
   </head>
   <body>
      <?php echo $output; ?>
      <br><br>
      <a href="exercise3.html">Repeat!</a>
   </body>
</html>
```

B, Solution– with 1 file (exercise31.php)





Solution:

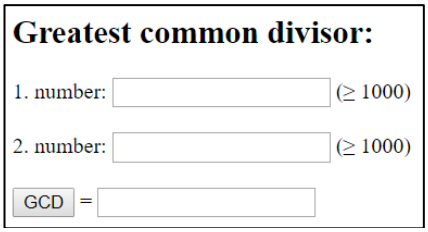
The Euclidean algorithm presented in the A/2 solution is used.

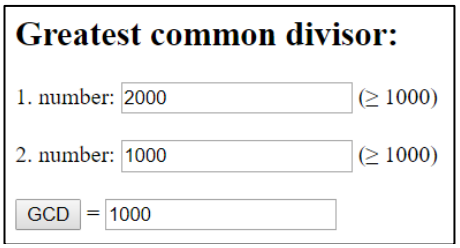
exercise31.php

```
<?php
// At the first time the $_POST array is empty, so this block will be skipped:
   if(isset($_POST["num1"]) && isset($_POST["num2"]))
   {
// If we have already sent data with the form: calculates the greatest common divisor of the two numbers:
      $a = $_POST["num1"];
      $b = $_POST["num2"];
      while($a != $b)
         if($a < $b)
            $b -= $a;
         else
            $a -= $b;
   }
?>
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Greatest common divisor</title>
   </head>
   <body>
// On first run, the $a variable does not exist, so the part in the next IF is not executed
// On second run, it prints the result.
      <?php if(isset($a)) echo "<h2>Result:</h2> Greatest common divisor of
<strong>".$_POST["num1"]."</strong> and <strong>".$_POST["num2"]."</strong> is
<strong>".$a."<strong>"; ?>
// It is executed on the first and every subsequent run:
      <h2>Greatest Common Divisor. Write the integers:</h2>
      <form method="POST">
         1. number: <input type="text" name="num1" pattern="[1-9][0-9]*" required> (&ge; 1)<br><br>
         2. number: <input type="text" name="num2" pattern="[1-9][0-9]*" required> (&ge; 1)<br><br>
         <input type="submit" value="GCD">
      </form>
   </body>
</html>
```

C, Solution – with one file, slightly different page layout – Only informative text

In this solution the page structure is almost the same in the first and second state.
The result is written in a read-only textbox.





Solution

exercise32.php

```
<?php
   if(isset($_POST["num1"]) && isset($_POST["num2"]))
   {
      $num1 = $_POST["num1"];
      $num2 = $_POST["num2"];

      $a = $num1;
      $b = $num2;

      while($a != $b)
// condition ? instruction1 : instruction2       means:
// if(condition) instruction1; else instruction2;
         $a < $b ? $b -= $a : $a -= $b;
   }
?>
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Greatest common divisor</title>
   </head>
   <body>
   <h2>Greatest common divisor:</h2>
      <form method="POST">
// condition ? instruction1 : instruction2       means:
// if(condition) instruction1; else instruction2;
```

```
        1. number: <input type="number" name="num1" value=<?php echo(isset($_POST["num1"]) ?
"\"".$_POST["num1"]."\"" : "\"\""); ?> min="1000" required> (&amp;ge; 1000)<br><br>
        2. number: <input type="number" name="num2" value=<?php echo(isset($_POST["num2"]) ?
"\"".$_POST["num2"]."\"" : "\"\""); ?> min="1000" required> (&amp;ge; 1000)<br><br>
        <input type="submit" value="GCD"> = <input type="text" value=<?php echo(isset($a) ?
"\"".$a."\"" : "\"\""); ?> readonly>
    </form>
  </body>
</html>
```

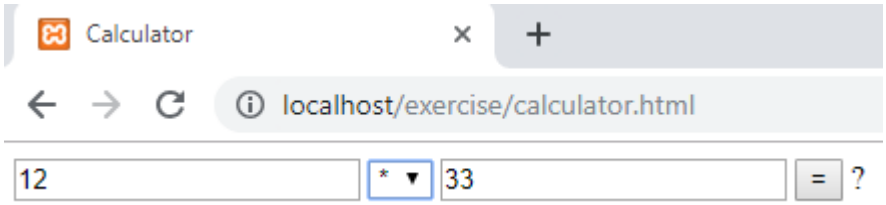
4th exercise – Calculator – Similar to the previous exercise - Practicing homework – Only informative text

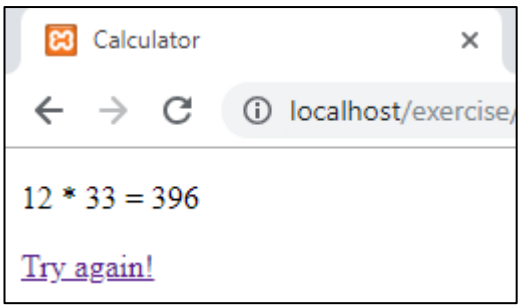
Create a simple 4 basic operations calculator

Version-1 **(with 2 files)**

The **calculator.html** contains the next input elements:

operand1	(text)
operation	(list)
operand2	(text)
calculate	(button)





The solution can be found in Solutions.zip

You can also send data to the a php file from a Form on a remote machine! – Only informative text

probe.html – It can be in any folder

```
<!DOCTYPE html>
<html>
<head>
        <title>Form</title>
```

```
        <meta charset="utf-8">
</head>
<body>
        <form action="http://localhost/exercise/process.php" method="post">
                Name: <input type="text" name="name"><br>
                <input type="submit" value="Send">
        </form>
</body>
</html>
```

c:\xampp\htdocs\exercise**process.php**

```
<?php
        print_r($_POST);
?>
```

After filling out the form, it calls the **process.php** file, which prints the submitted data.
It works the same way with Internet hosting instead of localhost.

4 - PHP – Database access – Login / Registration

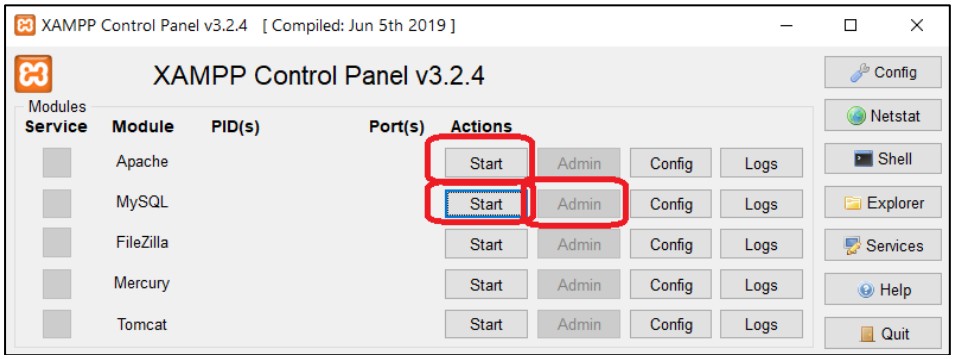
Similar to the previous lesson:

- Create a working folder, e.g. c:**exercise** **Copy here the solution files**
- Visual Studio Code
  => File menu / Open folder: open c:\exercise work-folder
  => Terminal menu / New Terminal: It opens the Command prompt in VS Code.
  OR
  Windows: Start / Windows system / Command prompt
- Start the Development Server with e.g. the 8000 port
  Don't start it with port 80 because the Apache server also uses it and they block each other!!! (see the figure below)
  \xampp\php\php -S localhost:8000

What you learned in the Databases lesson is used here, we will not learn it again.

With a MySQL database in XAMPP

Starting the XAMPP Control Panel
Apache Start, MySQL Start



Proper shutdown: MySQL Stop, Apache Stop,

Type in browser: localhost

Importing

Create the **databaselesson database** and **the users table** with importing the **databaselesson.sql** file from **Sources.zip** with **phpMyAdmin**.
Look at the imported **users** table in the database.
The password is encrypted with SHA-1algorithm. In cryptography, **SHA-1** (Secure Hash Algorithm 1) is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value known as a message digest – typically rendered as a hexadecimal number, 40 digits long.
https://en.wikipedia.org/wiki/SHA-1
https://en.wikipedia.org/wiki/Secure_Hash_Algorithms

1 – Initial tasks – in 20 minutes

Print ID and first name where ID<10

a.php - without HTML section

```
<?php
	try {
	// Connecting to the database		https://www.php.net/manual/en/book.pdo.php
	// root: user in the database (created when installing XAMPP)	'': password: empty string
		$pdo = new PDO('mysql:host=localhost;dbname=databaselesson', 'root',
'',array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
		$pdo->query('SET NAMES utf8 COLLATE utf8_general_ci');
		$statement = "Select id, first_name From users Where id<10";
		$result = $pdo->query($statement);
		foreach ($result as $row)
			print $row['id'] . " " .$row['first_name'] . "<br>";
	}
	catch (PDOException $e) {
		echo "Error: ".$e->getMessage();
	}
?>
```

http://localhost/a.php

b.php- with HTML section

In the PHP section we create the $result array. In the HTML section with a PHP script we print the results row-by-row.

```
<?php
	try {
		// Connecting
		$pdo = new PDO('mysql:host=localhost;dbname=databaselesson', 'root',
'',array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
		$pdo->query('SET NAMES utf8 COLLATE utf8_general_ci');
		$statement = "Select id, first_name From users Where id<10";
		$result = $pdo->query($statement);
	}
	catch (PDOException $e) {
		echo "Error: ".$e->getMessage();
	}
?>
<!DOCTYPE html>
<html>
  <head>
    <title>2/B exercise</title>
    <meta charset="utf-8">
  </head>
  <body>
	<h1>2/B exercise</h1>
	<?php
		foreach ($result as $row)
			print $row['id'] . " " .$row['first_name'] . "<br>";
	?>
```

```
   </body>
</html>
```

http://localhost/b.php

c.php - with HTML section - into a table - There is nothing new in it – Only informative text

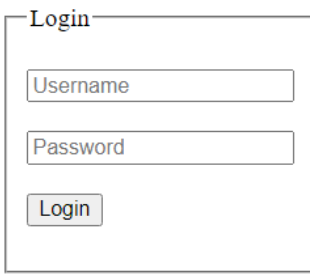
1	FirstName_1
2	FirstName_2
3	FirstName_3
4	FirstName_4
5	FirstName_5
6	FirstName_6
7	FirstName_7
8	FirstName_8
9	FirstName_9

```
<?php
        try {
                // Connecting
                $pdo = new PDO('mysql:host=localhost;dbname=databaselesson', 'root',
'',array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
                $pdo->query('SET NAMES utf8 COLLATE utf8_general_ci');
                $statement = "Select id, first_name From users Where id<10";
                $result = $pdo->query($statement);
        }
        catch (PDOException $e) {
                echo "Error: ".$e->getMessage();
        }
?>

<!DOCTYPE html>
<html>
   <head>
      <title>2/C exercise</title>
      <meta charset="utf-8">
                <style>
                        table, td, tr {
                                border: 1px  solid black;
                        }
                </style>
   </head>
   <body>
        <h1>2/C exercise</h1>
        <table>
        <?php foreach ($result as $row)
                print "<tr><td>" . $row['id'] . "</td>"  . " <td>" .$row['first_name'] . "</td></tr>";
        ?>
        </table>
   </body>
</html>
```

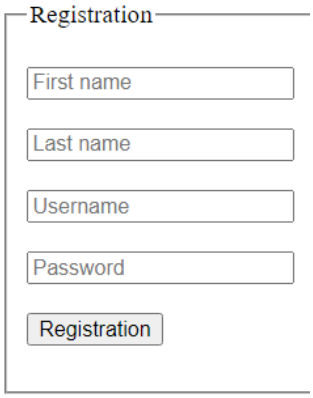
http://localhost/c.php

2nd exercise

A, Create a Registration/Login page



Register for login!



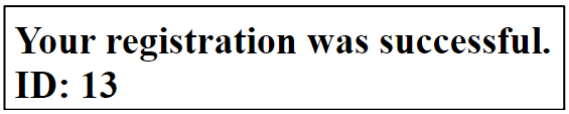
B, Create the **login.php** script, which is responsible for login process:

a) If the login was successful:

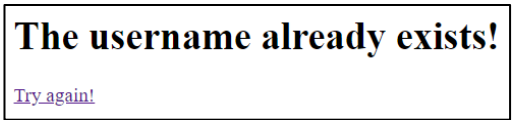
b) if the login wasn't successful:

C, Create the **registration.php** script, which is responsible for the registration process:

a) If the registration was successful:



b) If the registration wasn't successful:



Solution

exercise.html

There is no too much novelty in it.

We create two forms: one for login and one for registration.
The first form's data is submitted to login.php the second to registration.php.

```
<!DOCTYPE html>
<html>
  <head>
    <title>MySql</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form action = "login.php" method = "post">
      <fieldset>
        <legend>Login</legend>
        <br>
// The placeholder attribute specifies a short hint that describes the expected value of an input field
        <input type="text" name="username" placeholder="Username" required><br><br>
        <input type="password" name="password" placeholder="Password" required><br><br>
        <input type="submit" name="login" value="Login">
        <br> 
      </fieldset>
    </form>
    <h3>Register for login!</h2>
    <form action = "registration.php" method = "post">
      <fieldset>
        <legend>Registration</legend>
        <br>
        <input type="text" name="firstname" placeholder="First name" required><br><br>
        <input type="text" name="lastname" placeholder="Last name" required><br><br>
        <input type="text" name="username" placeholder="Username" required><br><br>
        <input type="password" name="password" placeholder="Password" required><br><br>
        <input type="submit" name="registration" value="Registration">
        <br> 
      </fieldset>
    </form>
  </body>
</html>
```

login.php

```
<?php
// Did we come from the form?:
    if(isset($_POST['username']) && isset($_POST['password'])) {
        try {
            // Connecting to the database
            $dbh = new PDO('mysql:host=localhost;dbname=databaselesson', 'root',
'',array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
            $dbh->query('SET NAMES utf8 COLLATE utf8_general_ci');
// We use a prepared statement              Advantages of a prepared statement: see below
// variables: :login, :password =>      they will get values at the execute statement.
```

```
        $sqlSelect = "select id, first_name, last_name from users where user_name = :username and
password = sha1(:password)";
//  PDO::prepare — Prepares a statement for execution and returns a PDOStatement object
//        http://php.net/manual/en/pdo.prepare.php
        $sth = $dbh->prepare($sqlSelect);
// The prepared SQL statement is executed using the execute method, substituting the two read values
//        The substitution is performed using an array (key-value pairs)
        $sth->execute(array(':username' => $_POST['username'], ':password' => $_POST['password']));
// https://www.php.net/manual/en/pdostatement.fetch.php
//        Fetches a row from a result set associated with a PDOStatement object.
//        In this case the result set contains only one row
//                If the result set contains several rows then use the fetchAll method.
/                 http://php.net/manual/en/pdostatement.fetchall.php
        $row = $sth->fetch(PDO::FETCH_ASSOC);
//        $row is an associative array. We use it in the next HTML section.
      }
      catch (PDOException $e) {      // exception handling. e.g. if the database cannot be found
        echo "Error: ".$e->getMessage();
      }
   }
?>

// Creates the html page using the data in the $row array.
// The $row array exists if the previous block has been executed.
<!DOCTYPE html>
<html>
   <head>
      <title>Login</title>
      <meta charset="utf-8">
   </head>
   <body>
      <?php if(isset($row)) { ?>        // if there is $row array
         <?php if($row) { ?>            // if the $row array is not empty
            <h1>Logged in:</h1>
// 'id': table field name from the SQL statement:
            ID: <strong><?= $row['id'] ?></strong><br><br>
            Name: <strong><?= $row['first_name']." ".$row['last_name'] ?></strong>
         <?php } else { ?>              // if the $row array is empty
            <h1>Sorry, we couldn't sign you in.</h1>
            <a href="exercise.html" >Try again!</a>
         <?php } ?>
      <?php } ?>
   </body>
</html>
```

Advantages of prepared statements

User data is included in the statement after compilation.

http://php.net/manual/en/pdo.prepare.php

- If we want **to execute the statement many times with different parameters**, we first prepare the statement skeleton and then replace it with the actual parameters in the marked places when executing it. This allows for much faster execution.
- Protection against **SQL injection attacks**: hackers use a special string written in text input fields to force the application to execute a data-extracting, destructive SQL statement that is

different from the one originally intended. In the last chapter, we will study examples of SQL injection attacks.

registration.php

It is similar to login.php. We study only the new instructions.

```
<?php
   if(isset($_POST['username']) && isset($_POST['password']) && isset($_POST['firstname']) &&
isset($_POST['lastname'])) {
      try {
         $dbh = new PDO('mysql:host=localhost;dbname=databaselesson', 'root', '',
array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
         $dbh->query('SET NAMES utf8 COLLATE utf8_general_ci');
         // Does the username already exist?          You cannot register with the same.
         // With prepared statement:
         $sqlSelect = "select id from users where user_name = :username";
         $sth = $dbh->prepare($sqlSelect);
         $sth->execute(array(':username' => $_POST['username']));
         // If the username already exists:
         if($row = $sth->fetch(PDO::FETCH_ASSOC)) {
         // we will use the $message and $again variables in the HTML section.
            $message = "The username already exists!";
            $again = "true";
         }
         else {
// Register if the username doesn't exist.
            $sqlInsert = "insert into users(id, first_name, last_name, user_name, password)
                  values(0, :firstname, :lastname, :username, :password)";
            $stmt = $dbh->prepare($sqlInsert);
            $stmt->execute(array(':firstname' => $_POST['firstname'], ':lastname' => $_POST['lastname'],
                  ':username' => $_POST['username'], ':password' => sha1($_POST['password'])));
// rowCount: returns the number of rows affected by the last DELETE, INSERT, or UPDATE statement
//       executed by the corresponding PDOStatement object.
//       http://php.net/manual/en/pdostatement.rowcount.php
            if($count = $stmt->rowCount()) {
//       If $count is greater than 0 it means that the registration was successful
// lastInsertId: Returns the ID of the last inserted row or sequence value
               $newid = $dbh->lastInsertId();
               $message = "Your registration was successful.<br>ID: {$newid}";
               $again = false;
            }
            else {
               $message = "Your registration wasn't successful.";
               $again = true;
            }
         }
      }
      catch (PDOException $e) {      // exception handling. e.g. if the database cannot be found
         echo "Error: ".$e->getMessage();
      }
   }

// Print the registration result on the page in the form specified at the beginning of the task.
```

```
?>
<!DOCTYPE html>
<html>
   <head>
      <title>Registration</title>
      <meta charset="utf-8">
   </head>
   <body>
// if we called the registration.php file from the exercise.html page
//       and the database operations were executed, then the $message variable exists:
      <?php if(isset($message)) { ?>
         <h1><?= $message ?></h1>
         <?php if($again) { ?>
            <a href="exercise.html">Try again!</a>
         <?php } ?>
      <?php } ?>
   </body>
</html>
```

Instead of the variables $message and $return, it can also be solved with only one variable. HW: solve with one variable

Database access settings for Internet implementation – Must also be used in Homework

After registering for the Internet hosting, you will either receive the following data marked in red, or you can create a database there yourself.

Change these settings in the PHP files:

```
$dbh = new PDO('mysql:host=localhost;dbname=dbname1', 'username1', '****',
                array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
```

localhost: use this hostname here too
dbname1: the database name at host
username1: the username to the database
'****': password to the database

Additional exercises for practicing the database instructions - If the SQL statement returns multiple records – If there is still time at the end of the class

3rd exercise
Print **all fields** of the user table where id<10

The SQL statement: Select * From users Where id<10

3/A - without prepared statement

3a.php

```
<?php
        try {
                // Connecting
```

```
		$pdo = new PDO('mysql:host=localhost;dbname=databaselesson', 'root',
'',array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
		$pdo->query('SET NAMES utf8 COLLATE utf8_general_ci');
		$statement = "Select * From users Where id<10";
		$result = $pdo->query($statement);
		foreach ($result as $row){	// users
			for( $i= 0 ; $i < count($row)/2 ; $i++)		// fields
		// The $row array contains the data duplicated:
		//		( [id] => 1, [0] => 1, [first_name] => FirstName_1, [1] => FirstName_1, …..)
				print $row[$i] . " " ;
			print "<br>";
		}
	}
	catch (PDOException $e) {
		echo "Error: ".$e->getMessage();
	}
?>
```

3/B - with prepared statement

3b.php

```
<?php
	try {
		// Connecting
		$pdo = new PDO('mysql:host=localhost;dbname=databaselesson', 'root',
'',array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
		$pdo->query('SET NAMES utf8 COLLATE utf8_general_ci');
		$statement = "Select * From users Where id<10";
		$sth = $pdo->prepare($statement);
		$sth->execute();
		$result = $sth->fetchAll(PDO::FETCH_ASSOC);
		foreach( $result as $row ) {
			foreach( $row as $field )
				print $field . " ";
			print "<br>";
		}
	}
	catch (PDOException $e) {
		echo "Error: ".$e->getMessage();
	}
?>
```

With SQLite database – only informative text

The entire database is a single file, e.g. task.db, which contains the tables and all the data.
We will not use it in the context of this subject, but it is useful in simpler, single-user environments, e.g. in a desktop or mobile application.

5- Exam-1 – From the topics of 1-3 lessons - with computer based exercises

6- Methods that we also use in the Front controller exercise – Magyar alapján újra átdolgozni

Executing PHP with XAMPP's built-in PHP and web server - We'll use this from now on

It is better than the Development Server solution we have been using so far, because XAMPP also contains the Apache web server, which provides additional services and more configuration options, such as the use of the **.htaccess** file, which we will use later.

XAMPP Control panel => Starting Apache and MySQL servers

It also uses the default port 80:

Working-directory: **C:\xampp\htdocs**

<u>In XAMPP, you can change the working directory if you don't want to use it:</u>
C:\xampp\htdocs
in C:\xampp\apache\conf\httpd.conf file e.g.:
DocumentRoot "c:/Munka/www"
and
<Directory "c:/Munka/www">
In this case, the working folder will be the specified folder.

Create a folder inside c:\xampp\htdocs called task, for example: **c:\xampp\htdocs\exercise**
This will be our working-directory, we will copy the source files here.

Passing data in the URL – 10 minutes

A, with $_GET array – in this format: try.php?key1=value1&key2=value2

Create **try.php** file in the working-directory:

```
<?php
	var_dump($_GET);
	echo "<br>" . $_GET["key1"].", ".$_GET["key2"];
?>
```

In the URL, we pass data in the form of **key-value pairs**:

http://localhost/exercise/try.php?key1=value1&key2=value2&key1=value3

Prints:
array(2) { ["key1"]=> string(6) "value3" ["key2"]=> string(6) "value2" }
value3, value2

In an associative array, the keys must be unique!
Note that the & and = characters must be used for this separation in the URL.

B, with $_SERVER array – in try.php?aaa/bbb/ccc format

Create **try.php** file in the working-directory:

```
<?php
   echo $_SERVER['QUERY_STRING']."<br>";
// explode: splits the string along "/" characters and puts the parts into the $array array:
//        https://www.php.net/manual/en/function.explode.php
   $array=explode("/", $_SERVER['QUERY_STRING']);
   echo $array[0];
?>
```

http://localhost/exercise/try.php?aaa/bbb/ccc
Prints:
aaa/bbb/ccc
aaa

We studied that in the case of index.php, we don't need to specify the file name - this is also true here

index.php

```
<?php
        var_dump($_GET);
?>
```

http://localhost/exercise/?key1=value1&key2=value2&key1=value3
Prints: array(2) { ["key1"]=> string(6) "value3" ["key2"]=> string(6) "value2" }

---

index.php

```
<?php
   echo $_SERVER['QUERY_STRING'];
?>
```

http://localhost/exercise/?aaa/bbb/ccc
Prints: aaa/bbb/ccc

Delete index.php.

with .htaccess file - not only index.php can be left out but also try.php and the ? is not needed – in http://localhost/exercise/aaa/bbb/ccc format

Redirect URL parameters to try.php

Create **.htaccess** file:

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ try.php?$1 [L,QSA]
```

Jelentése:
Ha nem fájl és nem mappa, akkor hívja meg a proba.php-t az URL-ben lévő paraméterrel, a következő formában: proba.php**?**paraméter
A **?** azért kell a **proba.php?** végére, mert láttuk, hogy a **$_SERVER['QUERY_STRING']**-el a **?** utáni részt olvassuk ki.

try.php

```
<?php
   echo $_SERVER['QUERY_STRING'];
?>
```

http://localhost/exercise/aaa/bbb/ccc
Prints: aaa/bbb/ccc

Ezt Router-ként tudjuk használni (majd pl. a Front controlleres feladatban is)

http://localhost/exercise/introduction
http://localhost/exercise/articles
http://localhost/exercise/contact

The .htaccess file cannot be used with the Development Server we have been using in the previous weeks!

It is not possible to handle .htaccess using PHP's built-in webserver (it is not relying on **Apache**, it is implemented entirely in PHP's core)
https://stackoverflow.com/questions/27381520/php-built-in-server-and-htaccess-mod-rewrites

We review the 3 main new elements used in the next lesson exercise (Front Controller design pattern).

Exercise-1- Writing the permanent parts of the pages in 1-1 files and inserting them with INCLUDE

In the 2nd lesson of the seminar, we learned about an application, where the pages have common parts (e.g. header, footer, left aside). At that lesson the common parts were copied to the pages, so several parts were stored redundantly. That redundant storage cause problems. For example if we want to change something in the common part, then we have to change it in all files. This problem can be solved with the PHP **include** statement.
https://www.php.net/manual/en/function.include.php
The common parts will be stored in separated files (**header.html, footer.html, aside.html**) and will be included into the pages e.g. **include("header.html")**; So if we want to alter something in the common part, we have to do it only in one file.

Solution

We will only discuss the new items marked in red.

The common parts:

header.html

```
<header>
   <img src="logo.png" alt="logo">
   <h1>Simple Website</h1>
</header>
```

footer.html

```
<footer>
   &copy; Copyright 2019. Simple Website Ltd.
</footer>
```

aside.html

```
<aside id="nav">
   <nav>
      <ul>
         <li class="active"><a href="index.php">Home</a></li>
         <li><a href="introduction.php">Introduction</a></li>
         <li><a href="contact.php">Contact</a></li>
         <li><a href="articles.php">Articles</a></li>
      </ul>
   </nav>
</aside>
```

Pages appearing in the navigation:

index.php

```
<!DOCTYPE html>
<html>
```

```
<head>
   <meta charset="utf-8">
   <title>Simple Website Ltd.</title>
   <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
   <?php include("header.html"); ?>
   <div id="wrapper">
      <?php include("aside.html"); ?>
      <div id="content">
         <h2>Welcome</h2>
         <img src="face.jpg">
         <h3>What is Lorem Ipsum?</h3>
         <p>Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularized in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.</p>
         <h3>Where does it come from?</h3>
         <p>Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.</p>
         <p>The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.</p>
         <h3>Why do we use it?</h3>
         <p>It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).</p>
         <h3>Where can I get some?</h3>
         <p>There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable. The generated Lorem Ipsum is therefore always free from repetition, injected humour, or non-characteristic words etc.</p>
      </div>
   </div>
   <?php include("footer.html"); ?>
</body>
```

```
</html>
```

contact.php

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <title>Simple Website Ltd.</title>
   <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
   <?php include("header.html"); ?>
   <div id="wrapper">
      <?php include("aside.html"); ?>
      <div id="content">
         <h2>Data</h2>
         <p>Manager: <strong>Somebody</strong></p>
         <p>E-mail: <strong>somebody@simplewebsite.com</strong></p>
         <iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2726.3375296155727!2d19.666950
91525771!3d46.89607994478184!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x4743da7a6
c479e1d%3A0xc8292b3f6dc69e7f!2sPallasz+Ath%C3%A9n%C3%A9+Egyetem+GAMF+Kar!5e0!3m2
!1shu!2shu!4v1475753185783" width="600" height="450" frameborder="0" style="border:0"
allowfullscreen></iframe>
         <br>
         <a target="_blank"
href="https://www.google.hu/maps/place/Pallasz+Ath%C3%A9n%C3%A9+Egyetem+GAMF+Kar/@46.
8960799,19.6669509,17z/data=!3m1!4b1!4m5!3m4!1s0x4743da7a6c479e1d:0xc8292b3f6dc69e7f!8m2!
3d46.8960763!4d19.6691396?hl=hu">Larger map</a>
      </div>
   </div>
   <?php include("footer.html"); ?>
</body>
</html>
```

The CSS file is the same as in the original exercise.
Create also the other pages.

Some advantages of INCLUDE:

- the application can be structured better
- code reuse: write once and use it in many places
- The user does not know from which files the page content is read. It is also not necessary that these content files are directly accessible to the user, they can be in hidden files or hidden folders, e.g. passwords.

There is a little problem: the Home menu element is indicated as Active in each cases. We'll solve this problem later.

Exercise-2- For each menu item, we call index.php (front controller) with different parameters

Solve the previous exercise with the next method: Call always the same file (index.php), if the user select whatsoever menu element. A parameter will be used to indicate the required page for loading into the content area (with green background) (<**div id="content">….. **).

A, with $_SERVER array and .htaccess file

URLs:
http://localhost/exercise
http://localhost/exercise/introduction
http://localhost/exercise/contact
http://localhost/exercise/articles

Only the Home and Contact pages are implemented. The rest can be implemented as a **practice homework**.

We can read the parameters from the $_SERVER global array.

- We also save the content of the pages in files (**mainpage_content.html, contact_content.html**)
- Here we enter the parameter (aside.html) in the menu link in a similar way, e.g.: <a **href="contact"**>Contact
- we read the parameter from the $_SERVER array and load the content of the selected page into the variable section (green background): ** ….. **.

By clicking on a given menu item, we pass the parameter in the URL address of the link

Solution
We discussed the basics in the previous examples. Here we will only discuss the new features, the parts marked in red.

.htaccess
```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?$1 [L,QSA]
```

If it is not a file and not a folder, then call **index.ph**p? with the parameter in the URL.

Files taken from the previous exercise:

header.html
```
<header>
   <img src="logo.png" alt="logo">
   <h1>Simple Website</h1>
</header>
```

footer.html
```
<footer>
   &copy; Copyright 2019. Simple Website Ltd.
</footer>
```

The new files:

aside.html
```
<aside id="nav">
   <nav>
```

```
      <ul>
        <li class="active"><a href=".">Home</a></li>
        <li><a href="introduction">Introduction</a></li>
        <li><a href="contact">Contact</a></li>
        <li><a href="articles">Articles</a></li>
      </ul>
    </nav>
</aside>
```

mainpage_content.html

```
        <h2>Welcome</h2>
        <img src="face.jpg">
        <h3>What is Lorem Ipsum?</h3>
        <p>Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularized in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.</p>
        <h3>Where does it come from?</h3>
        <p>Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.</p>
        <p>The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.</p>
        <h3>Why do we use it?</h3>
        <p>It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).</p>
        <h3>Where can I get some?</h3>
        <p>There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable. The generated Lorem Ipsum is therefore always free from repetition, injected humour, or non-characteristic words etc.</p>
```

contact_content.html

```
        <h2>Data</h2>
        <p>Manager: <strong>Somebody</strong></p>
```

```
        <p>E-mail: <strong>somebody@simplewebsite.com</strong></p>
        <iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2726.3375296155727!2d19.666950
91525771!3d46.89607994478184!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x4743da7a6
c479e1d%3A0xc8292b3f6dc69e7f!2sPallasz+Ath%C3%A9n%C3%A9+Egyetem+GAMF+Kar!5e0!3m2
!1shu!2shu!4v1475753185783" width="600" height="450" frameborder="0" style="border:0"
allowfullscreen></iframe>
        <br>
        <a target="_blank"
href="https://www.google.hu/maps/place/Pallasz+Ath%C3%A9n%C3%A9+Egyetem+GAMF+Kar/@46.
8960799,19.6669509,17z/data=!3m1!4b1!4m5!3m4!1s0x4743da7a6c479e1d:0xc8292b3f6dc69e7f!8m2!
3d46.8960763!4d19.6691396?hl=hu">Larger map</a>
```

index.php

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <title>Simple Website Ltd.</title>
   <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
   <?php include("header.html"); ?>
   <div id="wrapper">
      <?php include("aside.html"); ?>
      <div id="content">
         <?php
            $page = $_SERVER['QUERY_STRING'];
            if($page=="") include("mainpage_content.html");
            if($page=="introduction") include("introduction_content.html");
            if($page=="contact") include("contact_content.html");
            if($page=="articles") include("articles_content.html");
         ?>
      </div>
   </div>
   <?php include("footer.html"); ?>
</body>
</html>
```

The CSS file is the same as in the original exercise.
Create also the missing pages.

B, with $_GET array – Only informative text

You can find it in **Solutions.zip**.

3rd - User management: Login/Logout, Session

3A - Introductory task

session.php

```
<?php
   session_start();
   echo "SessionID: ".session_id()."<br>";
   if(isset($_SESSION['views']))
      $_SESSION['views']=$_SESSION['views']+1;
   else
      $_SESSION['views']=1;
   $_SESSION['Hello1']="Hello2";
   if(isset($a))
      $a=$a+1;
   else
      $a=1;
?>
<html>
<!DOCTYPE html>
<body>
   <?php
      echo $a;
      echo "<p></p>";
      print_r($_SESSION);
      echo "<p></p>";
      echo "Pageviews=". $_SESSION['views'];
   ?>
</body>
</html>
```

http://localhost/exercise/session.php
Reload the page several times!

```
SessionID: 99dj1scg6c4c5i7klb80djbsbp
1

Array ( [views] => 12 [Hello1] => Hello2 )

Pageviews=12
```

- **Open in other browser tab**: It continues from the actual number.
  If we are logged in in a web-page and we use the page on another browser tab: the browser knows that we are logged in.
  - **If we use a simple variable ($a)** for it, it always starts from 1.
- **Open in onather browser**: it starts from 1
- If we are logged in in a web-page and we use the page in another browser: the browser thinkss that we are NOT logged in.
- We printed SessionID as well.

Sessions

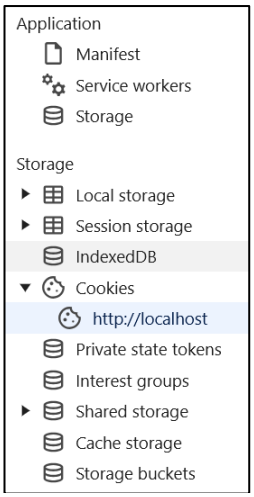
Session variables store user information to be used across multiple pages (e.g. username, favorite color, etc). By default, session variables last until the user closes the browser.
https://www.w3schools.com/php/php_sessions.asp

https://en.wikipedia.org/wiki/Session_(computer_science)
n computer science and networking in particular, a session is a time-delimited two-way link, a practical (relatively high) layer in the tcp/ip protocol enabling interactive expression and information exchange between two or more communication devices or ends – be they computers, automated systems, or live active users (see login session). A session is established at a certain point in time, and then 'torn down' - brought to an end - at some later point. An established communication session may involve more than one message in each direction. A session is typically stateful, meaning that at least one of the communicating parties needs to hold current state information and save information about the session history to be able to communicate, as opposed to stateless communication, where the communication consists of independent requests with responses.

A session is started with the **session_start()** function.
Session variables are set with the PHP global variable: **$_SESSION**.
A **PHP session** stores data on the server. Every user is identified through a unique number called session identifier or SID. The session IDs are randomly generated by the PHP engine.
You must call the session_start() function **at the beginning of the page** i.e. **before any output generated** by your script in the browser.

The SessionID is stored in the Cookie by the browser.

A, View in browser:
Chrome: right click on page / Inspect / Application / Storage / Cookies /



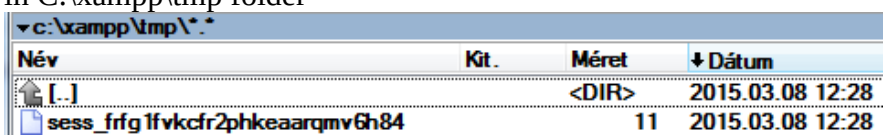
Name	Value	Domain	Path	Expires / M...	Size	Http...
PHPSESSID	imiu04okfk2b6b6qjnofhm5p7v	localhost	/	Session	35	

B, Print in PHP

Type after session_start();
echo session_id()."
";

C, Check in the server:

in C:\xampp\tmp folder



3B - Main task

Import the **databaselesson.sql** file into the database if it is not already imported.
The login details can be read from the file.

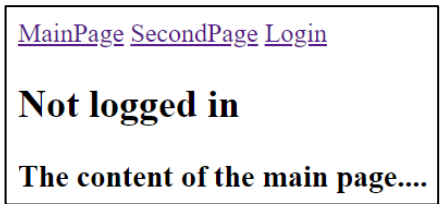
We note whether the user of the page is logged in or not, as well as their last name and first name. We keep track of this as we move through the menu items.

Display **at the top of the page**:
Menu bar
whether the user is logged in or not. If the user is logged in, display ID, first name and last name.

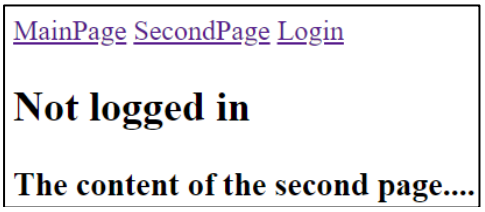
http://localhost/exercise/mainpage.php

A, If not logged in:

Main Page

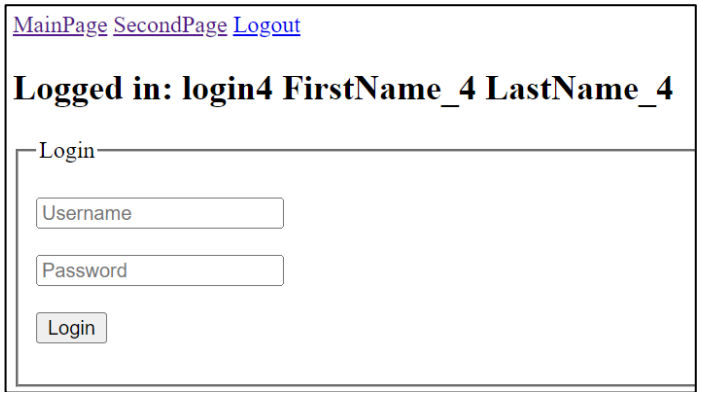


Second page



B, If logged in:

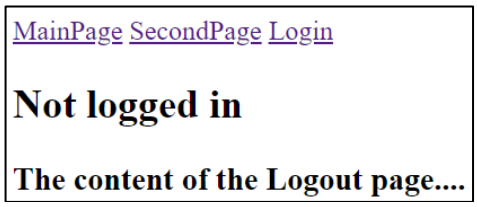
After login:



Main Page

Second page

Logout age



Since the top part is the same in all cases, we write it in a separate file: **top.php** and load it into the pages (INCLUDE)

Solution

mainpage.php

```
<?php
   session_start();
   include("top.php");
   echo "<H3>The content of the main page.... </H3>";
?>
```

top.php

```
<a href="mainpage.php">MainPage</a> <a href="secondpage.php">SecondPage</a>
<?php
   if(!isset($_SESSION['user'])) echo '<a href="login.php">Login</a> ';
   else echo '<a href="logout.php">Logout</a>';
   //print_r($_SESSION);
   echo "<h2>";
   if(isset($_SESSION['user']))
      echo "Logged in: ".$_SESSION['user']." ".$_SESSION['fn']." ".$_SESSION['ln'];
   else
      echo "Not logged in";
   echo "</h2>";
?>
```

In this way, it is also possible to resolve that certain parts of a page can only be seen by logged-in users!!!

if (isset($_SESSION['……..'])) { … part of the page to be displayed …}

secondpage.php

```
<?php
   session_start();
   include("top.php");
   echo "<H3>The content of the second page.... </H3>";
?>
```

login.php

```
<?php
   session_start();
   if(isset($_POST['username']) && isset($_POST['password'])) {
      try {
         // Connection
         $dbh = new PDO('mysql:host=localhost;dbname=databaselesson', 'root', '',
                  array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
         $dbh->query('SET NAMES utf8 COLLATE utf8_general_ci');
```

```
        // Search for a user
        $sqlSelect = "select id, first_name, last_name from users where user_name = :usern and password
= sha1(:pwd)";
        $sth = $dbh->prepare($sqlSelect);
        $sth->execute(array(':usern' => $_POST['username'], ':pwd' => $_POST['password']));
        $row = $sth->fetch(PDO::FETCH_ASSOC);

        if($row) {
          $_SESSION['fn'] = $row['first_name'];
          $_SESSION['ln'] = $row['last_name'];
          $_SESSION['user'] = $_POST['username'];
        }
        if(isset($row) && !$row){
          echo "<h2>Login failed!</h2>";
          session_unset();
        }

      }
      catch (PDOException $e) {
         echo "Error: ".$e->getMessage();
      }
   }
?>
<!DOCTYPE html>
<html>
  <head>
    <title>MySql</title>
    <meta charset="utf-8">
  </head>
  <body>
    <?php include("top.php"); ?>
    <form method = "post">
      <fieldset>
        <legend>Login</legend>
        <br>
        <input type="text" name="username" placeholder="Username" required><br><br>
        <input type="password" name="password" placeholder="Password" required><br><br>
        <input type="submit" name="login" value="Login">
        <br> 
      </fieldset>
    </form>
  </body>
</html>
```

logout.php

```
<?php
   session_start();
   session_unset();
   include("top.php");
   echo "<H3>The content of the Logout page.... </H3>";
?>
```

7 - PHP - Front Controller design pattern – ONLY FOR HOMEWORK

Exercise

Develop a website using the **dynamic front-controller design pattern**.

In the previous chapter, we reviewed the main parts of the task:

- Writing the common parts on the pages in 1-1 files and inserting them with INCLUDE
- We transmit in the URL which part of the content of the page to load into the large green background area.
- We call index.php for each menu item (front controller)
- User management: Login/Logout, Sessions

In addition to these, we use another method:
Create the application so that **the menu items can be changed**: the names of the menu items and the corresponding pages to be loaded can be set in a **config file**. This way we create a flexible little framework.

front controller design pattern – Only informative text

https://martinfowler.com/eaaCatalog/frontController.html
https://www.geeksforgeeks.org/front-controller-design-pattern/
https://www.tutorialspoint.com/design_pattern/front_controller_pattern.htm
https://en.wikipedia.org/wiki/Front_controller

At the **front controller design pattern** you have a main controller that handles every request for a website. It is a commonly used design pattern for many MVC based web applications. To use the

front controller pattern, you need to have a single part of your website that is fully responsible for handling all incoming requests to your site/application. The front controller: index.php
Front Controller refers to a design pattern where a single component in your application is responsible for handling all requests to other parts of an application. It centralizes common functionality needed by the rest of your application. Templating, routing, and security are common examples of Front Controller functionality. The benefit to using this design pattern is that when the behavior of these functions need to change, only a small part of the application needs to be modified. All requests for a domain are handled by a single point of entry (the front controller).

1/A Solution – with $_SERVER array

On the left side is the **menu bar**, which is set in the **templates/index.tpl.php** file. For each link, we call the same page (**index.php**). Depending on which link we clicked, the content of the linked page is loaded into the large green box in the middle.
In each case, the header, footer and the left menu bar are the same. The small difference in the menu bar is that it indicates with a small blue rectangle which is the active menu and thus the active page.

The links in the left menu will be these for the exercise working folder:
Home: http://localhost/exercise/
Introduction: http://localhost/exercise/introduction
Contact: http://localhost/exercise/contact
Articles: http://localhost/exercise/articles
Table: http://localhost/exercise/table

In the previous chapter, we saw that the **.htaccess** file places the part of the URL after http://localhost/exercise/ after index.php? and thus calls the index.php file. e.g. index.php?contact

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?$1 [L,QSA]
```

If it is not a file and not a folder, then call **index.php?** with the parameter in the URL.

We organize files into folders according to functions:

Main folder: index.php
images: face.jpg, logo.png
styles: style.css, table.css
includes: config.inc.php
templates: index.tpl.php The code for the common parts of the page.
templates/pages: The content of the given pages
home.tpl.php:
contact.tpl.php:
table.tpl.php:
404.tpl.php:

In the Homework, you don't need to know the application code in detail, you just need to use it.
In this bordered section, I have written down the minimum you need to know in order to use this small "framework" to create your own web application, for example for the Homework.
- The menu items and the main properties of the page must be set in the **includes/config.inc.php** file.
The **menu items** are set in the array $pages = array(…).
For example:
'**contact**' => array('file' => '**contact**', 'text' => '**Contact**'),
here the first **'contact'** string gives the URL address parameter of the menu item, for example:

http://localhost/exercise/contact

In the case of **'file' => 'contact'**, the 'contact' string is the name of the file from which the page's content part must be loaded into the large green background part. for example:

templates/pages/**contact**.tpl.php

The **'Contact'** string gives what the menu item name should be in the menu system:

- The common parts of the pages should be created in the **templates/index.tpl.php** file.
- The **templates/pages/** folder contains the files from which the content of the page should be loaded into the large green background section

For the Homework, you should start from Solution 2. This includes user management: Login/Logout/Registration

Since we include the page parts with PHP, if the user does not know the folder structure, he cannot view the files. If he knows the folder structure, he can view them, this small application does not protect against this. e.g. check:
http://localhost/exercise/styles/
http://localhost/exercise/styles/style.css

In the previous chapter, we saw that the redirection in the **.htaccess** file is not allowed for folder and file names.

1/B Solution - with $_GET array – Only informative text

You can find it in **Solution.zip**.

2nd Solution – with User Management – A - with $_SERVER array

Import the **databaselesson.sql** file into the database, if it is not already imported.

Extend the website with the “Login” and “Logout” menu items as follows:

a) The “Login” menu item is visible if the user is not logged in.
b) The “Logout” menu item is visible if the user is logged in.
c) Clicking on the “Login” menu item will open a page where you can log in or register.
d) The user is not automatically logged in after registration.
e) The system should display the logged in user, if any, in the following form: Logged in: Last_name First_name (Login_name)
e) Display the logged-in user information in the page header in the following format:
Logged in: Last_name First_name (Login_name)

Methods learned in previous exercises that we will use here

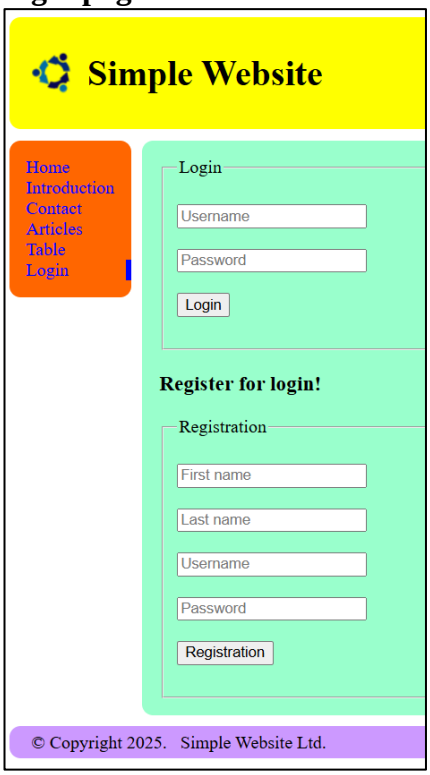
- Login/Logout, with database
- Sessions

Folder structure

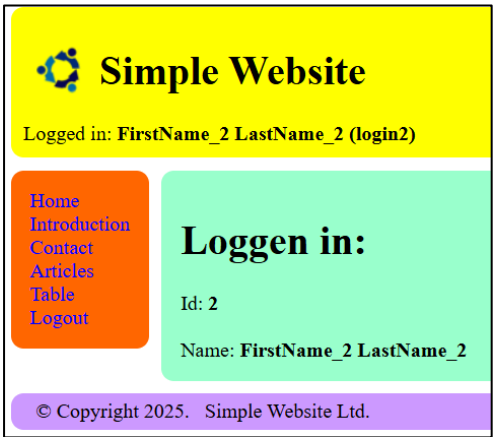
Main folder: index.php

images: face.jpg, logo.png
styles: styles.css, table.css
includes: config.inc.php
logicals: login2.php, logout.php, register.php
templates: index.tpl.php
templates/pages:
home.tpl.php:
contact.tpl.php:
table.tpl.php:
404.tpl.php:
login.tpl.php
login2.tpl.php
logout.tpl.php
register.tpl.php

Login page:

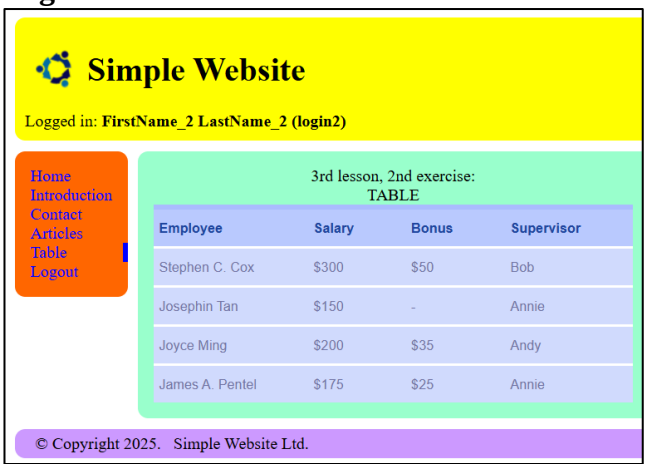


After login:



With logged is user:

Logout:



This is also true here:
In the Homework, you don't need to know the application code in detail, you just need to use it.

includes / config.inc.php
In the $pages array, we add an array to each entry: 'menun' => array(1,1)
'introduction' => array('file' => 'introduction', 'text' => 'Introduction', **'menun' => array(1,1))**,
Here we use a 2-element array to manage which menu should appear and when (1=yes, 0=no):
Number 1: if the user is logged in
Number 2: if the user is not logged in

…….
```
$pages = array(
   '/' => array('file' => 'home', 'text' => 'Home', 'menun' => array(1,1)),
   'introduction' => array('file' => 'introduction', 'text' => 'Introduction', 'menun' => array(1,1)),
   'contact' => array('file' => 'contact', 'text' => 'Contact', 'menun' => array(1,1)),
   'articles' => array('file' => 'articles', 'text' => 'Articles', 'menun' => array(1,1)),
   'table' => array('file' => 'table', 'text' => 'Table', 'menun' => array(1,1)),
   'login' => array('file' => 'login', 'text' => 'Login', 'menun' => array(1,0)),
   'login2' => array('file' => 'login2', 'text' => '', 'menun' => array(0,0)),
   'logout' => array('file' => 'logout', 'text' => 'Logout', 'menun' => array(0,1)),
   'register' => array('file' => 'register', 'text' => '', 'menun' => array(0,0))
);
```
……..

Login and register do not appear in the menu, but they are also needed:
We use them in the action=… section for the forms in the templates\pages\login.tpl.php file.

2nd Solution – with User Management – A - with $_GET array – Only informative text

You can find the solution in **Solutions.zip**.

8 – Dropdown menus - Bootstrap basic, Responsive design with Bootstrap

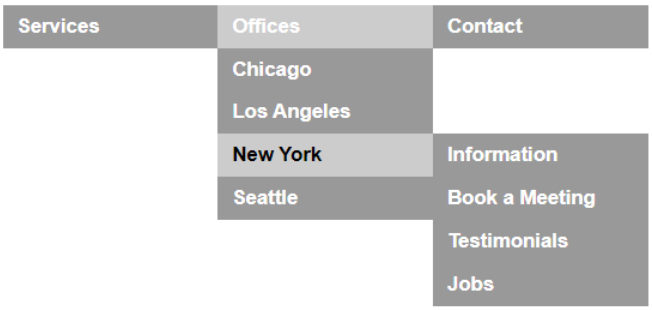
For today's class tasks, we don't need XAMPP or PHP, we only use HTML, CSS, and JavaScript, which the browser understands.

A, Dropdown menus – Only informative text – Useful for homework – in 2 minutes

A kódot nem nézzük meg, csak böngészőben az oldalt. A **Megoldások.zip-ben** megtalálják a többszintű menük HTML, CSS forrását, amit a házifeladatban is felhasználhatnak.

Dropdown-menu1.html
2 level menu

Dropdown-menu1.html
3 leve menu:



More multi-level menu examples
https://www.w3schools.com/css/css_dropdowns.asp
https://www.w3schools.com/howto/howto_css_dropdown.asp
https://www.w3schools.com/howto/howto_css_mega_menu.asp
https://freefrontend.com/css-dropdown-menus/

B, Bootstrap (HTML, CSS and JavaScript) framework- Responsive design

So far, we have written CSS to style ID, CLASS and other elements. Now, a lot of pre-made CSS is already written and named, e.g. . Here, we just use these in the HTML code.

Some descriptions about Bootstrap:

https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/

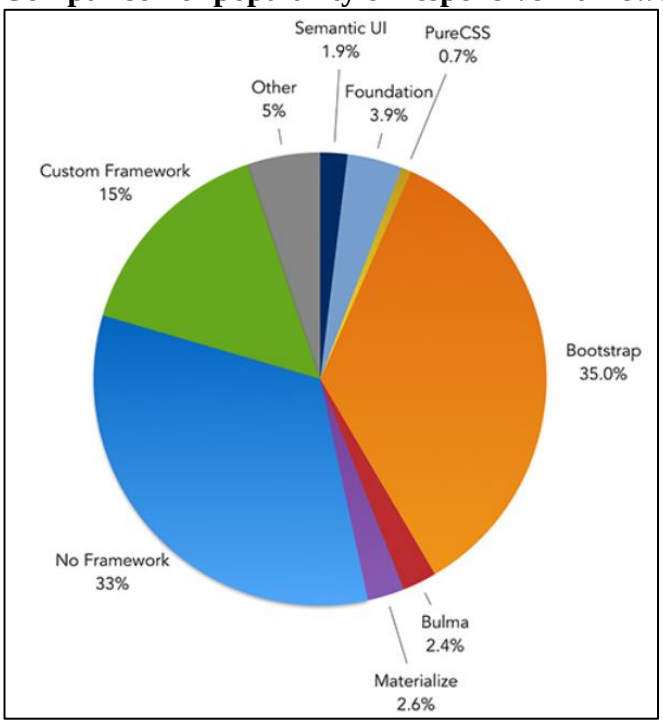
> **Bootstrap is the most popular and powerful** front-end (HTML, CSS, and JavaScript) framework for faster and easier responsive web development.

https://www.w3schools.com/bootstrap4/default.asp
https://www.w3schools.com/bootstrap4/bootstrap_get_started.asp
https://websitesetup.org/bootstrap-tutorial-for-beginners/
https://getbootstrap.com/docs/4.4/getting-started/introduction/
https://www.tutorialspoint.com/bootstrap4/index.htm
https://www.freecodecamp.org/news/learn-bootstrap-4-in-30-minute-by-building-a-landing-page-website-guide-for-beginners-f64e03833f33/
https://www.bootstrapdash.com/bootstrap-4-tutorial/introduction/

There are other responsive frameworks as well.

> **Search:** best responsive frameworks
> https://www.mockplus.com/blog/post/css-framework
> https://geekflare.com/best-css-frameworks/
> https://blog.templatetoaster.com/best-responsive-web-design-frameworks/
> https://www.creativebloq.com/features/best-css-frameworks
> https://www.skysilk.com/blog/2018/6-best-css-frameworks-2019/
> https://www.themexpert.com/blog/top-5-responsive-css-framework
> https://www.sitepoint.com/most-popular-frontend-frameworks-compared/
> https://www.webfx.com/blog/web-design/html5-frameworks/
> https://devrix.com/tutorial/35-best-html5-and-css3-responsive-frameworks/
> https://catswhocode.com/css-frameworks/

Comparison of popularity of responsive frameworks



Introductory task – in 5 minutes

Print the text **Hello Bootstrap!**: a centered paragraph with a width of 50%, centered blue bold text, yellow background, H1 heading size.
For this task, we will use the **pre-built classes of the Bootstrap framework**.

exercise-1.html

```
<!doctype html>
<html>
  <head>
    <title>My Website</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  </head>
  <body>
    <p class="w-50 text-primary text-center bg-warning font-weight-bold mx-auto display-1">Hello
Bootstrap!</p>
  </body>
</html>
```

w-50: 50% width — https://getbootstrap.com/docs/4.0/utilities/sizing/
text-center: középre igazított szöveg — https://getbootstrap.com/docs/4.0/utilities/text/
text-primary: blue color — https://getbootstrap.com/docs/4.0/utilities/colors/
bg-warning: yellow background
font-weight-bold: bold text
mx-auto: centralized paragraph (left and right margin: auto)
https://getbootstrap.com/docs/4.0/utilities/spacing/
display-1: Heading-1 (H1) text — https://getbootstrap.com/docs/4.0/content/typography/

The settings for each name can be found in the **bootstrap.min.css** file.

Open with browser:
https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css
Search for: **text-primary**
You can find it: **.text-primary{color:#007bff!important}**

Responsive design

Responsive web design or responsive design is an approach to web design that aims to make web pages render well on a variety of devices and window or screen sizes from minimum to maximum display size to ensure usability and satisfaction.
Responsive web design, or RWD, is a design approach that addresses the range of devices and device sizes, enabling automatic adaption to the screen, whether the content is viewed on a tablet, phone, television, or watch.

Free Download Responsive HTML, CSS Themes

Search:
free responsive html css
free responsive html5 css3
pl.
https://html5up.net/

https://medium.com/level-up-web/top-free-responsive-html5-css3-website-templates-2018-edition-5b3fb3d96cc9
https://w3layouts.com/free-responsive-html5-css3-website-templates/
https://themewagon.com/theme_tag/free/

Responsive design with Bootstrap

We study two main subject in this lesson, which can be used in Homework:

- The DIV boxes are placed below each other (mobile device) or next to each other (desktop computer).
- The menu is different in desktop computer and mobile device. On desktop computer screen the menu elements are next to each other, on mobile device with a collapsed menu.

At the end, we study a **complex example** – it can be used well for **homework**.

A, Grid System

https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-grid-system.php

We can use predefined grid classes for different devices:
mobile, tablet, laptop, desktop
We can handle a 12 column grid system.

With the Grid system we can create layouts like this. We can specify how many DIV boxes should be in a column on each device and in what layout:

Important properties of the grid system. We use this table in the exercises:

Features Bootstrap 4 Grid System	**Extra small** <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**Extra large** ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Ideal for	Mobile (Portrait)	Mobile (Landscape)	Tablets	Laptops	Laptops & Desktops
Class prefix	`.col-`	`.col-sm-`	`.col-md-`	`.col-lg-`	`.col-xl-`
Number of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

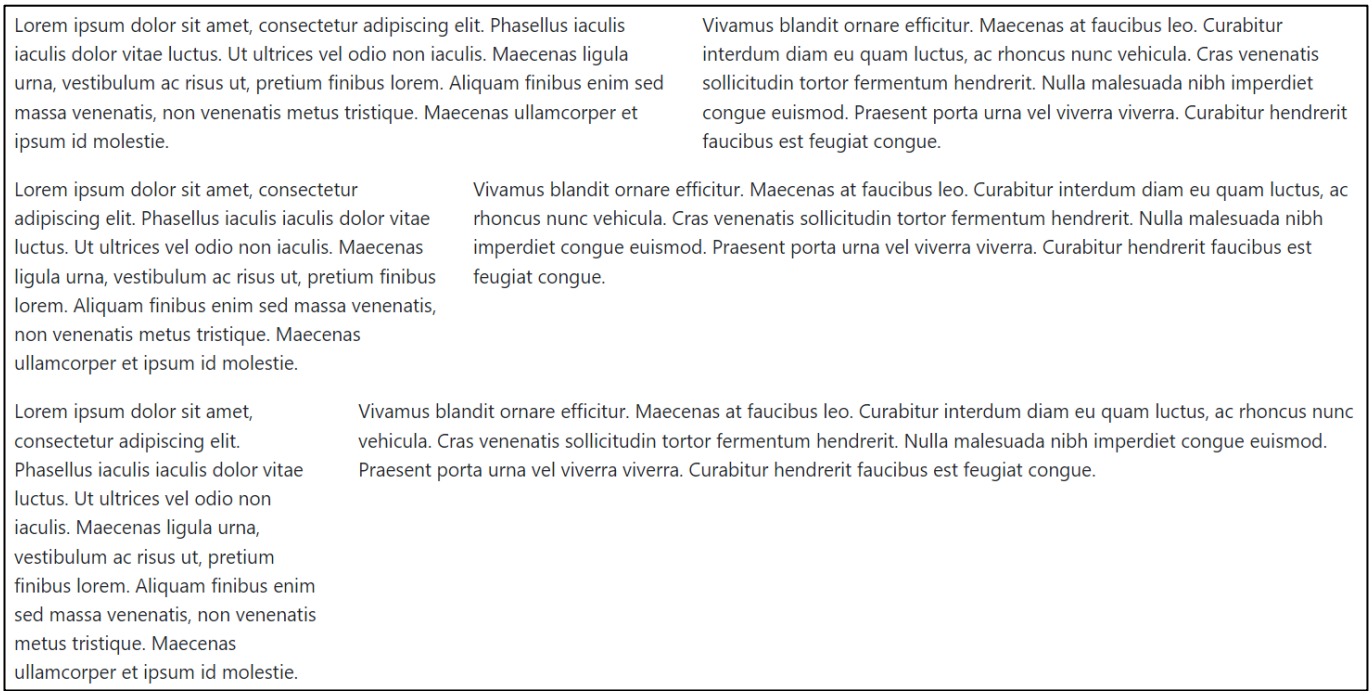
** **: This is the basis of the 12-column responsive system. This includes the rows and columns.
** **: we create rows inside the container
Creating columns based on the previous table, e.g..

page-02.html - Two Column Layouts
Try without Bootstrap!

For medium devices, create 2-column layouts in 3 rows.

Medium (≥768px): medium, large and extra large devices like tables, laptops and desktops

In the first row, in two equal (6:6) columns (6:6 = 1:1) (6+6=12)
In the second row, in columns with a 4:8 ratio (4:8 = 1:2) (4+8=12)
In the second row, in columns with a 3:9 ratio (3:9 = 1:3) (3+9=12)



Each part contains one DIV box!

For devices smaller than 768px, all 6 boxes will be placed one below the other:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus iaculis iaculis dolor vitae luctus. Ut ultrices vel odio non iaculis. Maecenas ligula urna, vestibulum ac risus ut, pretium finibus lorem. Aliquam finibus enim sed massa venenatis, non venenatis metus tristique. Maecenas ullamcorper et ipsum id molestie.

Vivamus blandit ornare efficitur. Maecenas at faucibus leo. Curabitur interdum diam eu quam luctus, ac rhoncus nunc vehicula. Cras venenatis sollicitudin tortor fermentum hendrerit. Nulla malesuada nibh imperdiet congue euismod. Praesent porta urna vel viverra viverra. Curabitur hendrerit faucibus est feugiat congue.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus iaculis iaculis dolor vitae luctus. Ut ultrices vel odio non iaculis. Maecenas ligula urna, vestibulum ac risus ut, pretium finibus lorem. Aliquam finibus enim sed massa venenatis, non venenatis metus tristique. Maecenas ullamcorper et ipsum id molestie.

Vivamus blandit ornare efficitur. Maecenas at faucibus leo. Curabitur interdum diam eu quam luctus, ac rhoncus nunc vehicula. Cras venenatis sollicitudin tortor fermentum hendrerit. Nulla malesuada nibh imperdiet congue euismod. Praesent porta urna vel viverra viverra. Curabitur hendrerit faucibus est feugiat congue.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus iaculis iaculis dolor vitae luctus. Ut ultrices vel odio non iaculis. Maecenas ligula urna, vestibulum ac risus ut, pretium finibus lorem. Aliquam finibus enim

This represents the real-world case where boxes are placed one below the other on mobile, and next to each other on a larger screen.

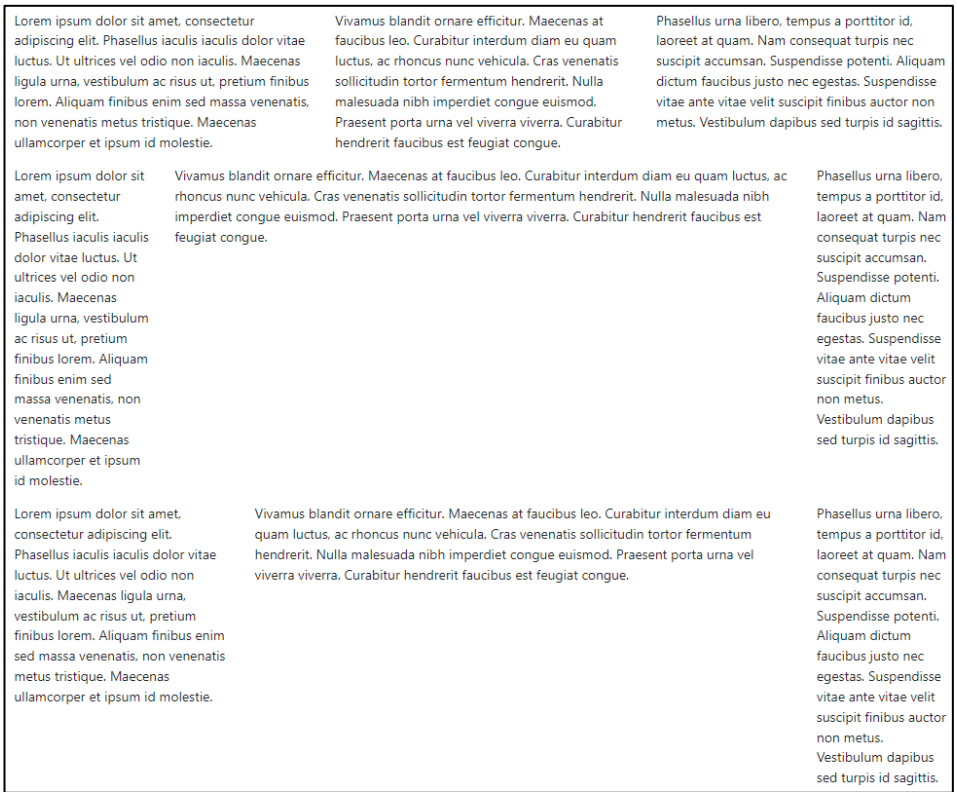
DIV boxes are block-level HTML elements: by default (without Bootstrap), the boxes would be placed one below the other. We use CCS to specify that they should be placed next to each other, not below each other.

In the examples, we only look at the parts between <body>…</body>, the parts outside that are the same in all examples.

```
……
   <div class="container">
      <!--Row with two equal columns-->
      <div class="row">
// by default (without Bootstrap) the next two DIV boxes would be placed one below the other.
// But for Medium or larger (≥768px) devices they will be placed next to each other with a 6:6 ratio
         <div class="col-md-6">Column left</div>
         <div class="col-md-6">Column right</div>
      </div>

      <!--Row with two columns divided in 1:2 ratio-->
      <div class="row">
         <div class="col-md-4">Column left</div>
         <div class="col-md-8">Column right</div>
      </div>

      <!--Row with two columns divided in 1:3 ratio-->
      <div class="row">
         <div class="col-md-3">Column left</div>
```

```
        <div class="col-md-9">Column right</div>
    </div>
</div> ……..
```

page-03.html
Create a 3-column layout for laptops and desktops (Large ≥992px).
lg: large ≥992px Laptops



For other displays, there should still be columns below each other.

```
<div class="container">
    <!--Row with three equal columns-->
    <div class="row">
        <div class="col-lg-4">Column left</div>
        <div class="col-lg-4">Column middle</div>
        <div class="col-lg-4">Column right</div>
    </div>

    <!--Row with three columns divided in 1:4:1 ratio-->
    <div class="row">
        <div class="col-lg-2">Column left</div>
        <div class="col-lg-8">Column middle</div>
        <div class="col-lg-2">Column right</div>
    </div>

    <!--Row with three columns divided unevenly-->
    <div class="row">
        <div class="col-lg-3">Column left</div>
        <div class="col-lg-7">Column middle</div>
```

```
      <div class="col-lg-2">Column right</div>
   </div>
</div>
```

page-04.html
In the following example, there will be only one row on large displays in 3 columns with a 3:6:3 layout. On medium displays, there will be two columns next to each other with a 4:8 layout, with the 3rd column below them.
On smaller displays, the DIV boxes should be placed one under the other
md: medium ≥768px Tablets
lg: large ≥992px Laptops

This represents the real-world case where DIV boxes are placed one under the other on mobile, on medium displays two DIV boxes are placed next to each other, and the third one is placed below them. On large displays, all three DIV boxes are placed next to each other.

```
<div class="container">
   <div class="row">
      <div class="col-md-4 col-lg-3">Column one</div>
      <div class="col-md-8 col-lg-6">Column two</div>
      <div class="col-md-12 col-lg-3">Column three</div>
   </div>
</div>
```

Since in the medium case 4+8+12=24>12, the third DIV box is placed on a new line.

page-05.html
On large devices place 12 DIV boxes in 4 rows, in each row put 3 boxes. On smaller screens the 12 boxes be below each other.
lg: large ≥992px Laptops

```
<div class="container">
   <div class="row">
      <div class="col-lg-4"><p>Box 1</p></div>
      <div class="col-lg-4"><p>Box 2</p></div>
      <div class="col-lg-4"><p>Box 3</p></div>
      <div class="col-lg-4"><p>Box 4</p></div>
      <div class="col-lg-4"><p>Box 5</p></div>
      <div class="col-lg-4"><p>Box 6</p></div>
      <div class="col-lg-4"><p>Box 7</p></div>
      <div class="col-lg-4"><p>Box 8</p></div>
      <div class="col-lg-4"><p>Box 9</p></div>
      <div class="col-lg-4"><p>Box 10</p></div>
      <div class="col-lg-4"><p>Box 11</p></div>
      <div class="col-lg-4"><p>Box 12</p></div>
   </div>
</div>
```

page-06.html
The goal is a layout like this:

We place 12 DIV boxes:
For Extra Large displays, 4 boxes should be next to each other, for Large displays, 3 boxes, for Medium displays, 2 boxes, and for smaller displays, the boxes should be placed one under the other.
We do not specify the rows in advance, so the number of boxes determines how many rows there will be.

```
<div class="container">
  <div class="row">
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 1</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 2</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 3</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 4</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 5</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 6</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 7</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 8</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 9</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 10</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 11</p></div>
    <div class="col-lg-4 col-md-6 col-xl-3"><p>Box 12</p></div>
  </div>
</div>
```

Nesting of Grid Columns
The Bootstrap grid columns are also nestable, that means you can put rows and columns inside an existing column. However, the formula for placing the columns will be the same, i.e. the sum of column numbers should be equal to 12 or less within a single row.

```
<div class="container">
  <div class="row">
    <div class="col-sm-8">Column left</div>
    <div class="col-sm-4">
      <div class="row">
        <div class="col-12"></div>
      </div>
      <div class="row">
        <div class="col-6"></div>
        <div class="col-6"></div>
      </div>
    </div>
  </div>
</div>
```

All Bootstrap classes
https://bootstrapshuffle.com/classes
https://www.w3schools.com/bootstrap4/bootstrap_ref_all_classes.asp

B, The menu layout should be different on each display

With menu items appearing side by side on desktop. On mobile, the **collapsible menu** should have menu items below each other.

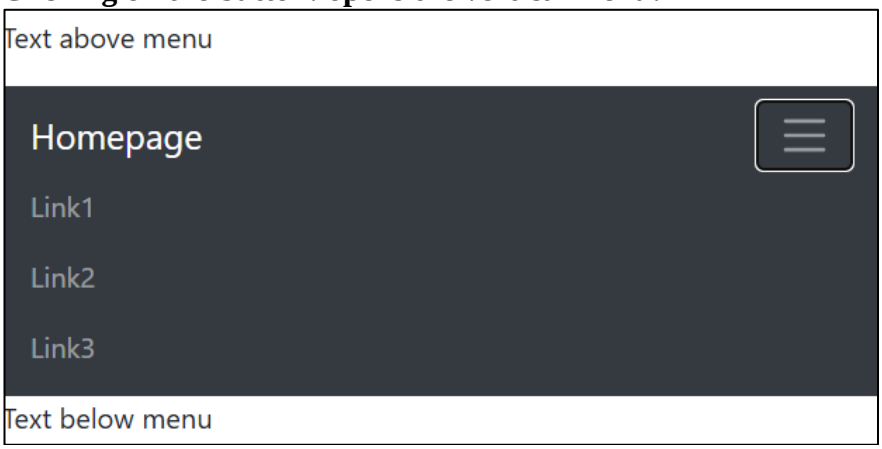
page-07.html
Try without Bootstrap!

For small screens (>=576px) or larger:

For smaller screens:

Menu with Hamburger button: 

Clicking on the button: opens the vertical menu:



Solution
We create the menu system here as before: list + links. (ul, li, and a elements)
And we create a (horizontal) menu from it with CSS

Explanation after the solution.

……

```
<body>
  <p>Text above menu</p>
  <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
    <a class="navbar-brand" href="#">Homepage</a>
```

```
   <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#collapsibleNavbar">
     <span class="navbar-toggler-icon"></span>
   </button>
   <div class="collapse navbar-collapse" id="collapsibleNavbar">
    <ul class="navbar-nav">
      <li class="nav-item">
        <a class="nav-link" href="https://index.hu/">Link1</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="https://www.origo.hu/index.html">Link2</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="https://mtva.hu/">Link3</a>
      </li>
    </ul>
   </div>
  </nav>
  <p>Text below menu</p>
</body>
……..
```

Explanation

A standard navigation bar is created with the **.navbar** class.

navbar-expand-sm: A horizontal navbar that becomes vertical on small screens

sm: small ≥576px Mobile (Landscape): the menu bar will open if larger

An alternative size to sm is: **navbar-expand-xl|lg|md|sm**

To add links inside the navbar, use a **<ul>** element with class="**navbar-nav**". Then add **<li>** elements with a **.nav-item** class followed by an **<a>** element with a **.nav-link** class

Explanation of the remaining parts:

bg-dark: dark background with a wide stripe.

navbar-dark: white font menu items

navbar-brand: highlight the first menu item (Homepage)

We need these for the collapsible menu:

navbar-toggler, data-toggle="collapse" data-target="#collapsibleNavbar"
collapse navbar-collapse id="collapsibleNavbar"

To create a collapsible navigation bar, use a button with class="**navbar-toggler**", **data-toggle="collapse"** and **data-target="#thetarget"**. Then wrap the navbar content (links, etc) inside a div element with class="**collapse navbar-collapse**", followed by an id that matches the data-target of the button: "**thetarget**".

collapse: Collapsibles are useful when you want to hide and show large amount of content

navbar-collapse: the navigation bar is replaced by a button in the top right corner.

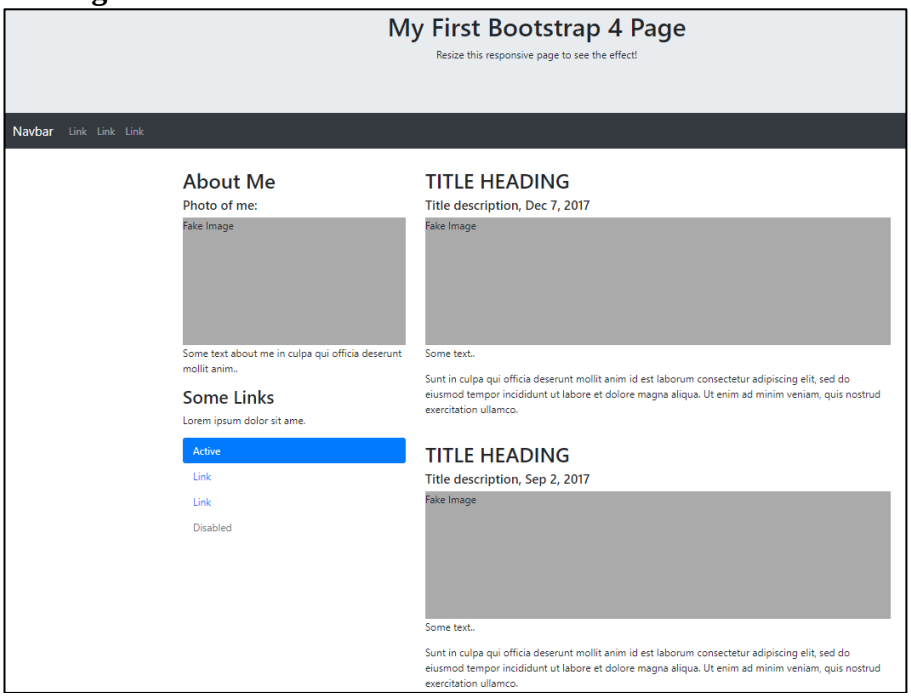
navbar-nav: for full-height and lightweight navigation (including support for dropdowns).

navbar-toggler-icon: button shape 

A complex exercise – we use the elements we have learned so far

https://www.w3schools.com/bootstrap4/default.asp

On large devices:



There's not much new in it => **Solutions.zip**

9 – Javascript

The XAMPP usage isn't necessary, because the browser understands the JavaScript.

1st Exercise - Filling an array, displaying it in a table

Create an HTML file (exercise.html) that loads a CSS file (style.css) and a JavaScript file (exercise.js) in its HEAD.

Specification for the JavaScript file:

1. Define an empty array
2. Get the array's length (prompt box) until the user confirms the given number
3. Fill the array with random numbers
4. List the array in a table
5. Display an alert box to inform that the task is solved

Specification for the CSS file:

1. 10 cells in each row
2. The width of the cells is 50px
3. Margin: 5px
4. Text color: navy
5. Odd rows' background color: #aaaaaa
6. Even rows' background color: #dddddd
7. Column separator: 3px, black

The array content:

24	3	1	68	86	99	32	89	78	61
59	78	20	43	96	26	31	44	23	44
82	97	22	2	12	61	91	78	32	15
90	80	31	9	24	26	90	24	72	90
95	42	36	87	27	73	82	31	91	26
85	79	43	53	18					

Solution – with separated Javascript file.

We can see that the Javascript code is executed first (it is in the head).

To compare: right click on the page

View page source
Inspect

exercise.html

```
<!DOCTYPE html>
<html>
<head>
```

```
    <title>JavaScript</title>
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="style.css">
    <script type="text/javascript" src="exercise.js">
    </script>
</head>
<body>
        The first row of the page.
</body>
</html>
```

exercise.js

```
// 1. Define an empty array
var array = new Array();
var arrayLength;

// 2. Get the array's length (prompt box) until the user confirms the given number
do
   arrayLength = prompt("Enter the array's length!");
while (! confirm("Are you sure that the array's length is "+arrayLength+"?"));

// 3. Fill the array with random numbers
for(var i= 0; i<arrayLength; i++)
// Return a random number between 0 (inclusive) and 1 (exclusive)
// https://www.w3schools.com/jsref/jsref_random.asp
   array[i] = Math.round(Math.random() * 100);
// 4. List the array in a table
document.writeln("<h1>The array's content:</h1>");
document.writeln("<table>");
for(var i=0; i<arrayLength; i++) {
   if (i%10==0)         // if  "i" variable is divisible by 10
   // condition ? instruction1 : instruction2   means:
   // if(condition) instruction1; else instruction2;
// even, odd => style.css
//         table new row: <tr class = "even">   vagy    <tr class = "odd">
//         i/10%2==0: for the even rows
      document.write("<tr class=\""+(i/10%2==0 ? "even" : "odd")+"\">");
   document.writeln("<td>");          // new cell
   document.writeln(array[i]);
   document.writeln("</td>");
   if (i % 10 == 9 || i == (arrayLength-1))
      document.writeln("</tr>");       // the end of table row
}
document.writeln("</table>");

// 5. Display an alert box to inform that the task is solved
alert("Ready!");
```

style.css

```
// We don't deal with CSS in detail after the first exam.
// It sets whether table borders should collapse into a single border or be separated as in standard HTML:
table {
```

```
    border-collapse: collapse;
}
tr {
    border-left:  3px solid #000000;    // Set the style of the left border
}
tr.even {
    background-color: #dddddd;          // even rows' background color
}
tr.odd {
    background-color: #aaaaaa;
}
td {
    width: 50px;
    padding: 5px;
    text-align: right;
    color: navy;
    border-right: 3px solid #000000;   // cell's right border
}
```

JavaScript is cached by the browser, so if we change the JavaScript code and refresh the browser, we often do not see the effect of the change.
Temporarily disable caching in Chrome (while the Inspector is open):
right click on the page / Inspector / Network / Disable cache (while DevTools is open)

JavaScript is cached by the browser, so if we change the JavaScript code and refresh the browser, we often do not see the effect of the change.
Temporarily disable caching in Chrome (while the Inspector is open):
right click on the page / Inspector / Network / Disable cache (while DevTools is open)

Document Object Model (DOM)

The DOM is a standard object model that HTML and XML are also built on. The model is a system of objects with a parent-child relationship. The DOM allows us to access the elements of our HTML document, as well as browser events such as clicking or scrolling. In the DOM, elements are called nodes.

We can modify/access the page with JavaScript if it is run after the DOM of the HTML document has been created!
=> window.onload = … runs when the page loads => next exercise
Accessing page elements, e.g.: getElementById(…), getElementsByClassName(…)

Short exercise

The page without JavaScript
First text.

With JavaScript, after the page loaded
First text. Second text.

After clicking the text
First text. Second text. Third text.

Solution

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>exercise</title>
  <style type="text/css">
    .red {
      color: red;
    }
    .blue {
      color: blue;
    }
    .green {
      color: green;
    }
  </style>
  <script type="text/javascript">
                function click() {
                        this.classList.remove("blue");
                        this.classList.add("green");
                        this.innerHTML = this.innerHTML + "  Third text.";
                };
                window.onload = function() {
                        var element = document.getElementById('paragraph');
                        element.classList.remove("red");
                        element.classList.add("blue");
                        element.innerHTML = element.innerHTML + "  Second text.";
                        element.onclick = click;
                }
  </script>
</head>
<body>
  <p id="paragraph" class="red">First text.</p>
</body>
</html>
```

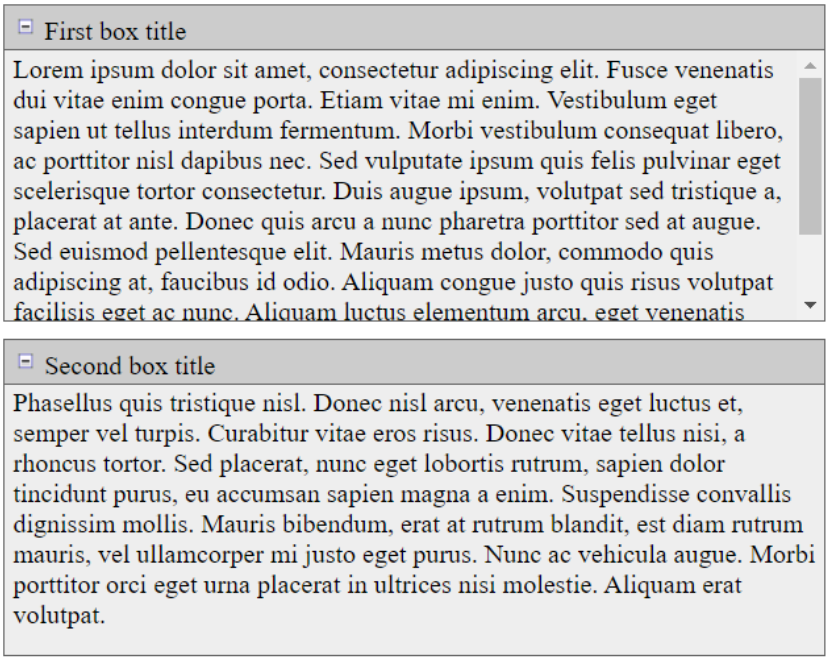
2nd Exercise – openable-closable boxes (divs) with JavaScript

Modify the ***exercise3.html*** script (in Web-programming-Sources.zip) with JavaScript.

Create boxes (divs):
- The boxes have titles,
- The boxes have icons next to the title: by clicking the icon the boxes open or close.
- First the boxes are opened (so the content of them are readable with disabled JavaScript).
  Close the boxes when the web page is loaded.
- Set the mouse cursor to "pointer" when the mouse cursor moving over the icon.
- First the title for the icons are: "not working" (with disabled JavaScript). With JavaScript change this title: "show details" <=> "hide details".
- There are two big boxes. Each of them contains two little boxes.

Use the block-close.gif and block-open.gif images from the images folder.

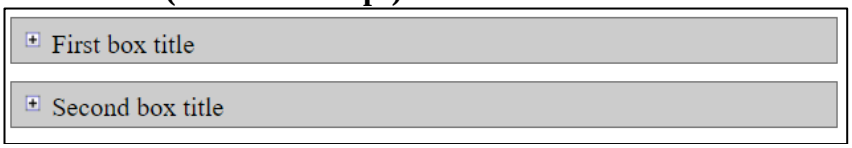
Initial state (without JavaScript):



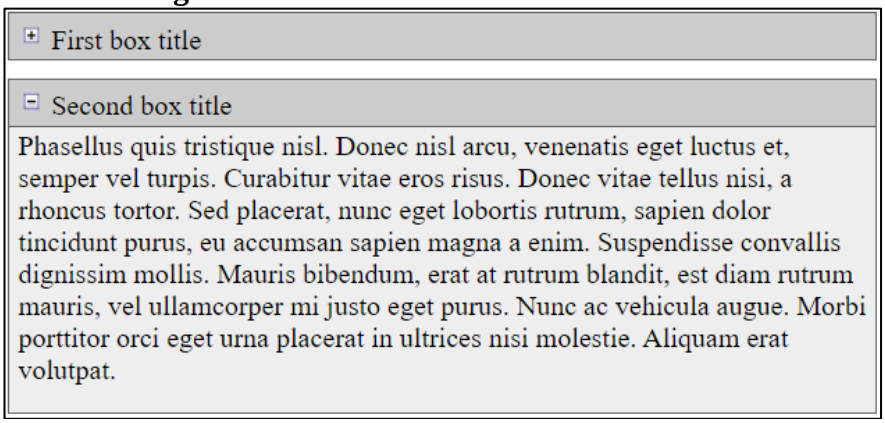
The user can read the whole content if the browser doesn't run the JavaScript.

After downloading the page:
- close the boxes
- change the icon to + type
- set the icon title to "show details"
- Set the mouse cursor to "pointer" when the mouse cursor moving over the icon.

Initial state (with JavaScript):



After clicking the icon:



Assign the **boxCloseOpen()** function to the **icon's click event**!
To make the **boxCloseOpen()** work, we take advantage of the fact that icons with the image class and div elements with the boxbody class have ids ending in the same sequence number. We read the id of the image that received the click event into the id variable.

Using the substr() method of the id variable, we copy the part of the image's identifier starting with the 3rd character, which is the image's sequence number in HTML. The identifier of the associated boxbody class element is body and the same sequence number, so we examine the value of the display property of this element. If none, then the body of the box is hidden, so it must be "opened", otherwise it must be "closed".
To "open", we set the display property of the corresponding box body to block, then change the image back to a - sign, and change the tooltip. During “closing” we do the opposite.

Solution

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Exercise 3</title>
  <style type="text/css">                    // CSS => first exam topic
    div.box {
      width: 500px;
      margin-bottom: 10px;
    }
    div.boxhead {
      padding: 2px 5px;
      border: 1px solid #666666;
      background-color: #cccccc;
    }
    div.boxbody {
      padding: 2px 5px;
      border: 1px solid #666666;
      border-top: 0;
      background-color: #eeeeee;
      height: 160px;
      overflow: auto;
    }
  </style>
  <script type="text/javascript">
        function boxCloseOpen() {
        // Get the clicked image’s identifier:
              var id  = this.getAttribute("id");
        // The identifiers of the images: id="image1" or "image2"
        // Get the identifier number: "1" or "2":
              var number = id.substr(5);     // 5: the index of the "1" substring in "image1" string
        // "image1" => "body1"         "image2" => "body2"
        // Get the "body1" or "body2" DIV box. It’s the DIV box belonging to the clicked image:
              var      boxbody = document.getElementById("body"+number);
        // If the selected boxbody DIV is closed, then open it:
              if(boxbody.style.display=="none"){
                    // open the box:
                    boxbody.style.display="block";
                    // change the icon to - type
                    this.setAttribute("src", "images/block-close.gif");
                    // set the icon title to "hide details":
                    this.setAttribute("title", "hide details");
              }
```

```
        // If the selected boxbody DIV is opened, then close it:
                else {
                        // open the box:
                        boxbody.style.display="none";
                        // change the icon to + type
                        this.setAttribute("src", "images/block-open.gif");
                        // set the icon title to "show details":
                        this.setAttribute("title", "show details");
                }
        }
// The load event fires at the end of the document loading process. At this point, all of the objects in the
// document are in the DOM, and all the images, scripts, links and sub-frames have finished loading.
        window.onload = function() {
        // Close the DIV boxes:
                // Get the DIVs with ClassName("boxbody").
                var elements = document.getElementsByClassName("boxbody")
                for (var i=0; i < elements.length; i++)
                        elements[i].style.display="none";
        // Initial settings for the images:
                elements = document.getElementsByClassName("image")
                for (var i=0; i < elements.length; i++){
                        // change the icon to + type:
                        elements[i].setAttribute("src", "images/block-open.gif");
                        // set the icon title to "show details":
                        elements[i].setAttribute("title", "show details");
                        //Set the mouse cursor to "pointer" when the mouse cursor moving over the icon:
                        elements[i].style.cursor="pointer";
                        // Assign boxCloseOpen() function to the icon click event:
                        elements[i].onclick = boxCloseOpen;
                        // It's also good: elements[i].addEventListener('click', boxCloseOpen);
                }
        }
  </script>
</head>
<body>
  <div class="box">
    <div class="boxhead">
// In the first div box the ids have number 1:       "image1" => "body1":
      <img src="images/block-close.gif" class="image" id="image1" title="not working">
      <span class="boxtitle">First box title</span>
    </div>
    <div class="boxbody" id="body1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce venenatis
      dui vitae enim congue porta. Etiam vitae mi enim. Vestibulum eget sapien
      ut tellus interdum fermentum. Morbi vestibulum consequat libero, ac porttitor
      nisl dapibus nec. Sed vulputate ipsum quis felis pulvinar eget scelerisque
      tortor consectetur. Duis augue ipsum, volutpat sed tristique a, placerat
      at ante. Donec quis arcu a nunc pharetra porttitor sed at augue. Sed euismod
      pellentesque elit. Mauris metus dolor, commodo quis adipiscing at, faucibus
      id odio. Aliquam congue justo quis risus volutpat facilisis eget ac nunc.
      Aliquam luctus elementum arcu, eget venenatis sapien ultricies eget. Nunc
      lobortis, magna et lobortis elementum, sapien turpis adipiscing nibh, sit
      amet varius odio odio quis nunc. Proin eget orci in felis eleifend malesuada
```

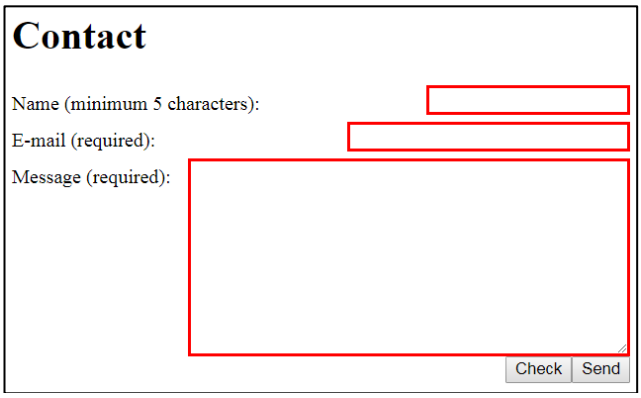
```
    vel nec elit.
   </div>
  </div>
  <div class="box">
   <div class="boxhead">
// In the second div box the ids have number 2: "image2" => "body2":
    <img src="images/block-close.gif" class="image" id="image2" title="not working">
    <span class="boxtitle">Second box title</span>
   </div>
   <div class="boxbody" id="body2">
    Phasellus quis tristique nisl. Donec nisl arcu, venenatis eget luctus et,
    semper vel turpis. Curabitur vitae eros risus. Donec vitae tellus nisi, a
    rhoncus tortor. Sed placerat, nunc eget lobortis rutrum, sapien dolor
    tincidunt purus, eu accumsan sapien magna a enim. Suspendisse convallis
    dignissim mollis. Mauris bibendum, erat at rutrum blandit, est diam rutrum
    mauris, vel ullamcorper mi justo eget purus. Nunc ac vehicula augue. Morbi
    porttitor orci eget urna placerat in ultrices nisi molestie. Aliquam erat
    volutpat.
   </div>
  </div>
</body>
</html>
```

10 – PHP, JavaScript

1st exercise

HTML, CSS part

Create the next Form.

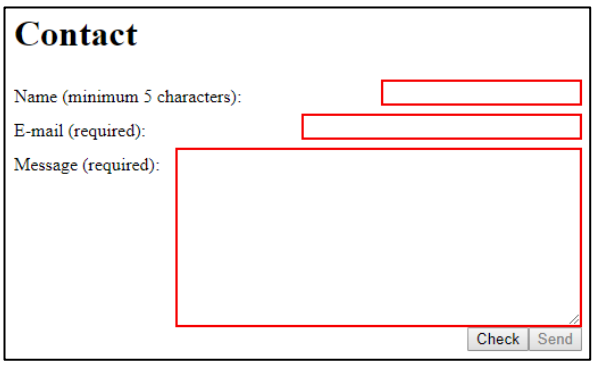


JavaScript part

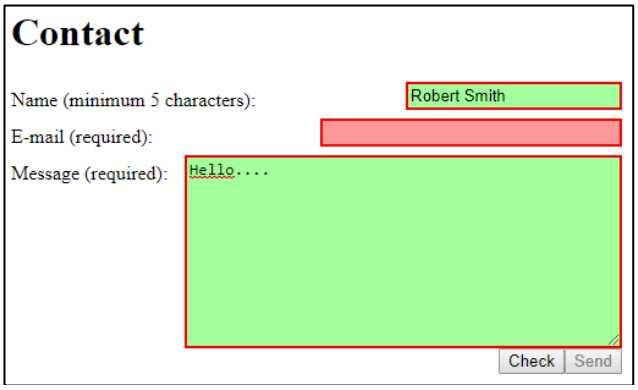
Disable the **Send button** immediately after the page has been loaded.
With the **Check button** the user can check the Form data.

- Set the background color of the correctly completed form elements to light green, the others light red!
- Use the next regular expression for email checking:
  /^([A-Za-z0-9_\-\.])+\@([A-Za-z0-9_\-\.])+\.([A-Za-z]{2,4})$/;
- Enable the Send button if all form elements are correct!
  Otherwise give focus to the first incorrect element.
- Before sending the form data check again the form elements. The form data can only be sent if the values are correct.

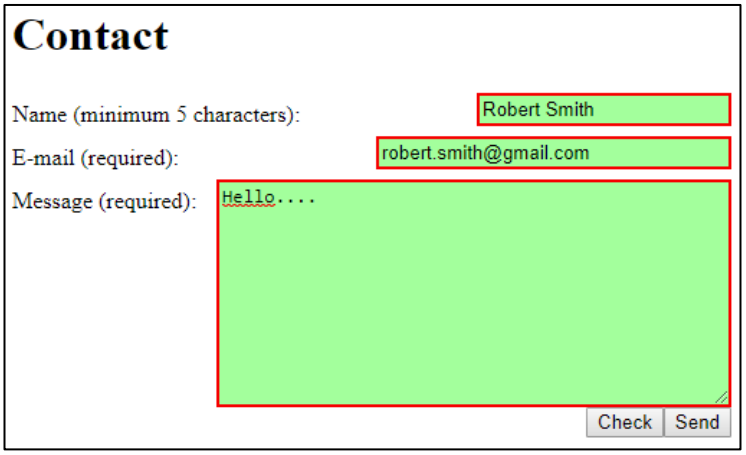
The **Send button** is disabled immediately after the page has been loaded:



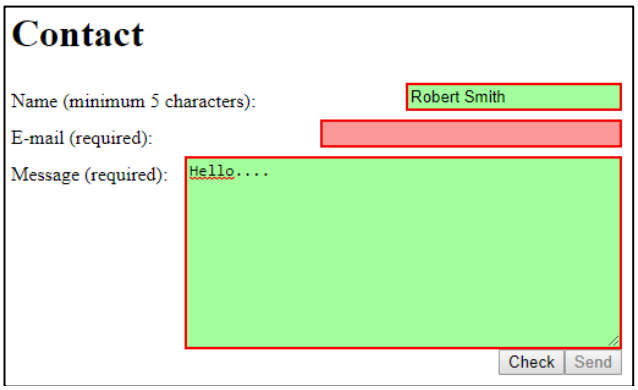
After checking:
In the case of incorrect values: the Send button is still disabled



In the case of correct values: the Send button is enabled



If we make something wrong again: the Send button is disabled again:



Try the solution without the js file!

Do not validate the form data in the HTML code now, so that the JavaScript validation effect can take effect if the form is filled out incorrectly. We validated the form data on the client side with JavaScript.

PHP part

In the case of correct values. The data is sent to the **contact.php** script. Also perform server-side validation in your PHP script! To test server-side validation, try the application without JavaScript.

In the case of correct values the result:

The received results:

```
array(3) {
  ["name"]=>
  string(12) "Robert Smith"
  ["email"]=>
  string(22) "robert.smith@gmail.com"
  ["textarea"]=>
  string(9) "Hello...."
}
```

Solution

Client-side Validation

Before submitting the form, we can check whether it is filled out correctly. The check is performed in two places on the client side:

- when clicking the Verify button
- when clicking the Submit button before sending.

The purpose of client-side verification is to ensure that only correct data is sent to the server. This reduces the load on the network and the server.

Server-Side Validation

We need to check form validation on **server side** too, because there is no guarantee that the input given by the user is always correct. The server must be protected from attacks and incorrect data.

contact.html

```
// The check() method is called in two places in the html file.

<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <title>Contact</title>
   <link rel="stylesheet" type="text/css" href="css/main.css">
   <script type="text/javascript" src="js/main.js"></script>          // First the javaScript is executed
</head>
<body>
   <h1>Contact</h1>
// If the Send button is enabled and we click it, then first the check() JavaScript method is executed.
// If it returns true then the data is sent to the contact.php file with POST method.
   <form name="contact" action="php/contact.php" onsubmit="return check();" method="post">
      <div>
         <label><input type="text" id="name" name="name" size="20" maxlength="40">Name
(minimum 5 characters): </label>
         <br/>
         <label><input type="text" id="email" name="email" size="30" maxlength="40">E-mail
(required): </label>
         <br/>
         <label> <textarea id="textarea" name="textarea" cols="40" rows="10"></textarea> Message
(required): </label>
         <br/>
// The next two tags define buttons:
```

```
// The <INPUT…> button belongs to the Form. It submits all form values
// The <BUTTON…> element is independent from the FORM. It can be anywhere on the page.
//        We can assign any event handling to it.
        <input id="send" type="submit" value="Send">
// when the user clicks the Check button: the check() JavaScript method is executed.
        <button onclick="check();" type="button">Check</button>
      </div>
   </form>
</body>
</html>
```

main.js

```
window.onload = function() {
        var send = document.getElementById("send");
        if (send)                   // If the send object exist on the page
                send.disabled = true;           // A disabled element is unusable and un-clickable.
};

// Client side validation:
function check() {      // It is called in two different cases in the HTML file
// Instead of the next two variables, it can be solved with only one variable.
//        HW: Solve it with one variable
// At the end of this method (return correct) the value of the correct variable is returned.
// We set it true here. If we find something is wrong during the validation, we set it to false.
        var correct = true;
// If an error occurs then give focus to the first incorrect element [focus() function]
// If there is no error then the focus variable remains null:
        var focus = null;
// Why we check first the last element (textarea) on the page?          Message, E-mail, Name
        var textarea = document.getElementById("textarea");
        if (textarea) {             // If the textarea exist on the page:
                if (textarea.value.length==0) {         // If the textarea is empty
                        correct = false;
                        textarea.style.background = '#f99';    // set background color to red
                        focus = textarea;       // The focus object will be the textarea object
                } else
                        textarea.style.background = '#9f9';    // set background color to green
        }
        var email = document.getElementById("email");
        if (email) {
// regular expression for the email:    https://regex101.com/
                var checkPattern = /^([A-Za-z0-9_\-\.])+\@([A-Za-z0-9_\-\.])+\.([A-Za-z]{2,4})$/;
// If the email is not correct:
                if (!checkPattern.test(email.value)) {
                        correct = false;
                        email.style.background = '#f99';
                        focus = email;
                } else
                        email.style.background = '#9f9';
        }
        var name = document.getElementById("name");
        if (name) {
```

```
                if (name.value.length<5) {     // if the name's length is less than 5
                        correct = false;
                        name.style.background = '#f99';
                        focus = name;
                } else
                        name.style.background = '#9f9';
        }

// originally the focus was set to null. If we set the focus during the validation:
        if (focus)
// Sets the focus to the object indicated by the focus:
                focus.focus();
        var send = document.getElementById("send");
        if (send)

// Ha minden adat jó volt, akkor rendben=true => !rendben=false => kuld.disabled = false
// If all the data was not good, then rendben=false => !rendben=true => kuld.disabled = true

// If every element is correct: correct = true => !correct = false => send.disabled = false
// If all the data was not correct, then correct = false => !correct = true => send.disabled = true
                send.disabled = !correct
        return correct;
}
```

main.css **We don't deal with the CSS in this lesson. It's the theme of the first lessons.**

```
div {
        width: 500px;
        }
label {
        line-height: 30px;

        }
label input, label textarea {
        /*position: relative;
        left: 200px;*/
        float: right;
        border: 2px red solid;
        padding: 2px;
        }
textarea {
        width: 350px;
        }
input[type="submit"] {
        float: right;
        clear: both;
        }
button {
        float: right;
        }
```

contact.php

If all fields were filled in correctly, then call this PHP script.

Try the app without JavaScript.

```
<!DOCTYPE html>
<html>
	<head>
		<meta charset="utf-8">
	</head>
	<body>
<?php
// server side validation
// If there is no 'name' key in the $_POST array OR its length is less than 5:
	if(!isset($_POST['name']) || strlen($_POST['name']) < 5)
	{
		exit("Wrong name: ".$_POST['name']);	// It exits the program
	}
// email checking with the given regular expression:
	$re = '/^([A-Za-z0-9_\-\.])+\@([A-Za-z0-9_\-\.])+\.([A-Za-z]{2,4})$/';
	if(!isset($_POST['email']) || !preg_match($re,$_POST['email']))
	{
		exit("Wrong email: ".$_POST['email']);
	}
// If there is no Message textarea or its data is empty
	if(!isset($_POST['textarea']) || empty($_POST['textarea']))
	{
		exit("Wrong text: ".$_POST['textarea']);
	}

	echo "<h3>The received results:</h3>";
// Prints the data of the $_POST array in a structured way.
	echo "<pre>";
	var_dump($_POST);
	echo "</pre>";
?>
	</body>
</html>
```

Practicing Homework

Try the task without the Submit button being grayed out. The button will not be grayed out, but everything else will be the same.

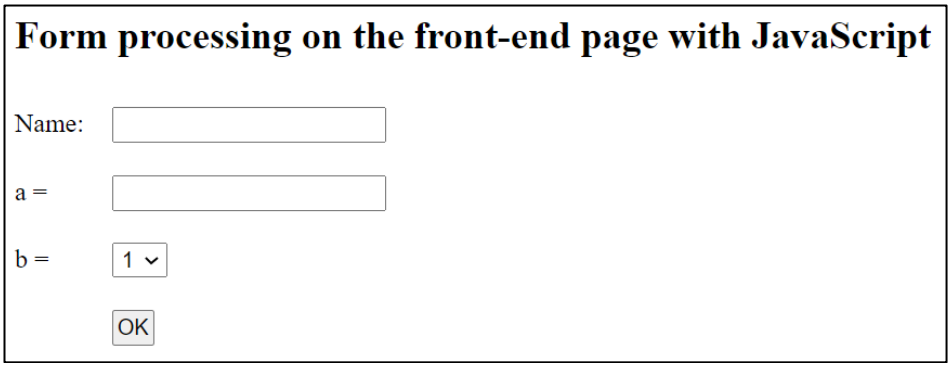
Modify the JavaScript file:

window.onload = function() {…}	delete the function
kuld.disabled = !rendben;	delete the row

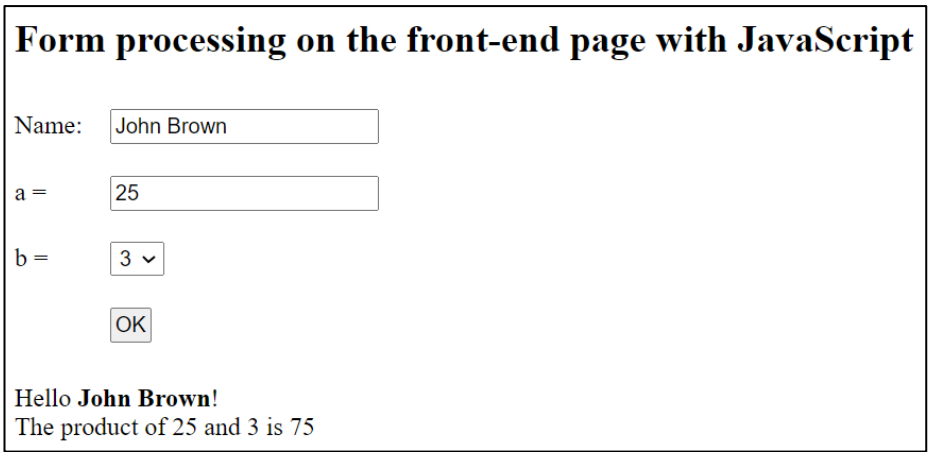
2nd exercise - Form processing on front-end with JavaScript – 10 min

Complete the following tasks:

- The user must enter a name in the first text box.
- You can only enter a number in the second text box
- You can choose from numbers 1-5 in the drop-down list.



After filling the form and clicking the button:



Solution

form-processing.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>form processing</title>
  <style type="text/css">                    // CSS => the topic of the first exam
        label {
              width: 60px;
              display: inline-block;
        }
        input, select {
               margin: 10px 0;
               padding: 2px;
        }
  </style>
  <script type="text/javascript">
        function send() {
              var res = document.getElementById('result');
              res.innerHTML = "Hello <strong>" + document.getElementById('name').value +
"</strong>!<br>";
              var number1 = document.getElementById('num1').value;
              var number2 = document.getElementById('num2').value;
```

```
                res.innerHTML += ("The product of " + number1 + " and " + number2 + " is " +
number1*number2);
        }
  </script>
</head>
<body>
        <h2>Form processing on the front-end page with JavaScript</h2>
        <label>Name:</label><input type="text" id="name"><br>
        <label>a =</label><input type="text" id="num1"><br>
        <label>b =</label><select id="num2">
                <option value="1">1</option>
                <option value="2">2</option>
                <option value="3">3</option>
                <option value="4">4</option>
                <option value="5">5</option>
        </select><br>
        <label></label><input type="submit" onclick="send()" value="OK">
        <p id="result"></p>
</body>
</html>
```

If we entered restrictive expressions into the INPUT elements, e.g.:
<input type="text" id="num1" pattern="[1-9][0-9]*" required>
they would have no effect here, because the content is not checked when the button is clicked, the onclick="send()" method is executed immediately.
We can do these restrictions with JavaScript on the client side => this is the topic of the next chapter

3rd exercise – JavaScript is asynchronous – 5 min

https://www.w3schools.com/js/js_asynchronous.asp
https://www.geeksforgeeks.org/asynchronous-javascript/

exercise.html

```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript</title>
   <meta charset="utf-8">
        <script type="text/javascript">
                function print2() {
                        document.writeln("Hello2<br>");
                }
                function print3() {
                        // after a 1 second delay print:
                        setTimeout(function(){
                                document.writeln("Hello3<br>");
                        }, 1000);
                }
                window.onload = function() {
                        document.writeln("Hello1<br>");
                        print2();
```

```
                print3();
                document.writeln("Hello4<br>");
            }
        </script>
</head>
<body>
</body>
</html>
```

It prints in this order:
Hello1
Hello2
Hello4
Hello3

4th exercise – Running JavaScript without a browser: in Command line – Node.js is required – 5 min

Install the latest LTS version of Node.js if it is not on your computer
https://nodejs.org/en/

Create the exercise.js file in any folder:

```
for (var i=0; i<3; i++) {
  console.log("Hello world!");
}
```

in Command line
node exercise.js

It prints:
Hello world!
Hello world!
Hello world!

11 - PHP – Images, Gallery, Image uploading – Web apps security

1- Display galery – 30 min

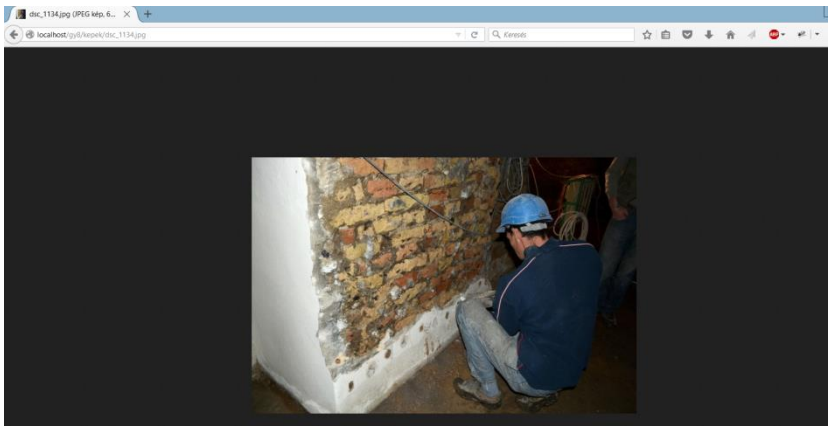
Create an **image** folder in the document directory of the application. Copy the images from the Sources file to this folder.

There are images in the **images** folder, dislpay them in a gallery.
You have to use the images in the **uploading** folder in the 2nd exercise.

index.php

- Read all files from the images folder.
- Select the image files (jpg or png)
- Create a simple gallery with these images. Display the images with their names and last modification time. Sort the images based on last modification time.
- The image with 200 pixels, centralized, maximum three images in a row.

Use the images as links: if the user clicks an image, it is shown in the original size in browser.



Solution

Right click on page => Show page source. You have to create such rows:

<img src="images/DSC_1134.JPG">

It sets the image with the hyperlink.

We use two files for this application:

- index.php: the main file
- config.inc.php: setting some variables
  We'll use them in the 2nd exercise too.

config.inc.php including in index.php: include('config.inc.php');

```
<?php
   $FOLDER = './images/';
   $TYPES = array ('.jpg', '.png');     // image types
   $MEDIATYPES = array('image/jpeg', 'image/png');
   $DATEFORMAT = "m/d/Y H:i";          // for printing the date in this format: 0/22/2016 11:17
   $MAXSIZE = 500*1024;                // we'll use in the 2nd exercise          500kB
?>
```

index.php

```
<?php     // Application logic:
   include('config.inc.php');

// Store the images data in an array in key-value pairs. key: filename, value: last modification time
   $images = array();
   $reader = opendir($FOLDER);
   while (($file = readdir($reader)) !== false) {
      if (is_file($FOLDER.$file)) {    // the subfolders are not selected, only the files
// getting the filename extension (file type) (last 3 characters) with converting it to lowercase letters
// substr fuction: 2 parameters: 1st: input string, 2nd: start position
         $end = strtolower(substr($file, strlen($file)-4));
         if (in_array($end, $TYPES)) {        // only jpg and png types are allowed.
               // $FOLDER.$file: string concatenation e.g. images/DSC_1099.JPG
               // filemtime: Gets modification time of the file, given in UNIX time. e.g. 1477118030:
               //       the number of seconds that have elapsed since 00:00:00 UTC on 1 January 1970.
               //       https://www.php.net/filemtime
               //       https://en.wikipedia.org/wiki/Unix_time
```

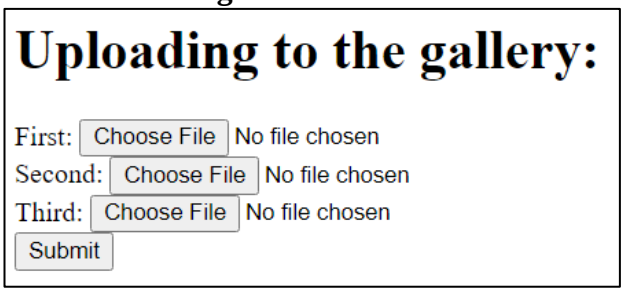
```
            $images[$file] = filemtime($FOLDER.$file);  // key: filename, value: last modification time
        }
      }
   }
   closedir($reader);

// Visualization logic:
?><!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <title>Gallery</title>
   <style type="text/css">     // CSS setting
      div#gallery {margin: 0 auto; width: 620px;}    // Display gallery DIV centralized, 620px with
                                                     // 0: top, bottom, auto: right, left: centralizing
// The former DIV box with is set to 620 pixels and the image with is 200 pixels =>
//        3 images are in a row.
      div.image { display: inline-block; }      // Only one image would be shown without it.
      div.image img { width: 200px; }           // image width: 200 pixels
   </style>
</head>
<body>
   <div id="gallery">
   <h1>Gallery</h1>
   <?php
// Sort an associative array in descending order, according to the value (date)
// https://www.w3schools.com/php/func_array_arsort.asp
//        PHP has several functions that deal with sorting arrays
//        https://www.php.net/manual/en/array.sorting.php
   arsort($images);
// Loop through the associative array          Store key => $file, value => $date
   foreach($images as $file => $date)
   {
   ?>
      <div class="image">
// sets the image with hyperlink
// e.g. <a href="images/DSC_1134.JPG"><img src="images/DSC_1134.JPG"></a>
         <a href="<?php echo $FOLDER.$file ?>">
            <img src="<?php echo $FOLDER.$file ?>">
         </a>
// Prints the filename
         <p>Name:  <?php echo $file; ?></p>
// Prints the last modification time:
// The date() function formats a local date and time, and returns the formatted date string.
//        https://www.w3schools.com/php/func_date_date.asp
         <p>Date:  <?php echo date($DATEFORMAT, $date); ?></p>
      </div>
   <?php
   }
   ?>
   </div>
</body>
</html>
```

2 – Image upload – 30 min

Complete the gallery with **upload.php**:

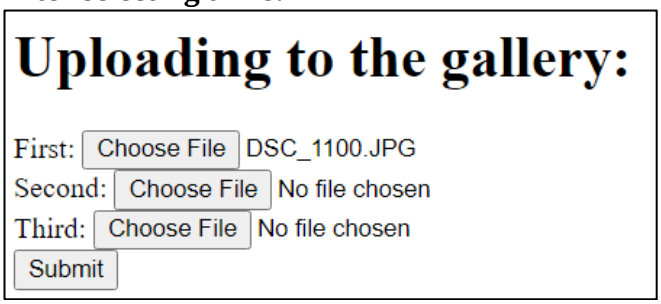
- Create a file uploader form for uploading at most 3 images (**jpg or png**) with maximum 500Kb sizes to the **images** folder.
- Upload the given images to the gallery with this PHP script. The images can be in any folder on the computer. You can find files in the **Uploading** folder.

The form arrangement:

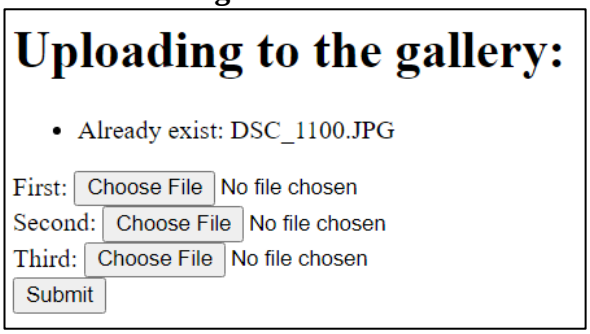


Maximum 3 images can be uploaded

After selecting a file:



After submitting the form:



We uploaded 1 file

https://www.tutorialspoint.com/php/php_superglobals_files.htm
https://www.w3schools.com/php/php_file_upload.asp
https://www.geeksforgeeks.org/php-_files-array-http-file-upload-variables/

We use a form for file uploding with the next structure:

```
<form action="feltolt.php" method="post"   enctype="multipart/form-data">
        <label>First: <input type="file" name="first" required></label>
        <label>Second: <input type="file" name="second"></label>
        <label>Third: <input type="file" name="third"></label>
        <input type="submit" name="send">
</form>
```

The system saves the uploaded files in a temporary folder with a temporary name. After uploading, we can check if there is a file with this name in the **images** folder. If there is a file with this name, do not overwrite it.
The system stores the data of the uploaded files in the **$_FILES two-dimensional** array, from where we can read them.

The structure of the **$_FILES** array:

- $_FILES["field-name"]["name"] file name
- $_FILES["field-name"]["type"] Size of the file
- $_FILES["field-name"]["size"] Type of the file (like .pdf, .zip, .jpeg…..etc)
- $_FILES["field-name"]["tmp_name"] A temporary address where the file is located before processing the upload request
- $_FILES["field-name"]["error"] Types of error occurred when the file is uploading

field-name: the name value of INPUT at FORM

```
<input type="file" name="second">
```

"error" codes:
https://www.php.net/manual/en/features.file-upload.errors.php

UPLOAD_ERR_OK, Value: 0; There is no error, the file uploaded with success.
UPLOAD_ERR_INI_SIZE, Value: 1; The uploaded file exceeds the upload_max_filesize directive in php.ini.
UPLOAD_ERR_FORM_SIZE, Value: 2; The uploaded file exceeds the MAX_FILE_SIZE directive that was specified in the HTML form.
…..
UPLOAD_ERR_NO_FILE, Value: 4; No file was uploaded.

We read the data form $_FILES array:

```
foreach($_FILES as $fajl) {
    if ($fajl['error'] == 4);
    ……..
}
```

Solution

We complete the previous gallery with **upload.php**.

Since the new images are uploaded with form, so the images can be in any folder on the computer.
We only do the uploading and copying. We do not display the images again.

In the exercise description: „…with maximum 500Kb sizes…”
=> $MAXMERET = 500*1024; 500 kB

upload.php

```
<?php
  // Application logic:
  include('config.inc.php');          // it's used again
// The messages after submitting are stored in this array. We'll print them after uploading.
//      "The type is not correct", "Too big file", "Already exist", "OK"
//                                                          => OK: check on the previous image
  $messages = array();
```

```
    // Form checking:
    if (isset($_POST['send'])) {
// At the first access of the upload.php script the $_POST array is empty, so this section is skipped
// The size of the $_FILES array is three after form submitting,
//         even if the user choose only one or two files.
// iterates over the $_FILES array (maximum 3 elements)
        foreach($_FILES as $file) {
            // If the user choose only one file, then for the other two files $file['error'] == 4
            // In this case nothing happens (empty statement)
            if ($file['error'] == 4);
            elseif (!in_array($file['type'], $MEDIATYPES)) // if the file-type is other than jpg and png
                $messages[] = " The type is not correct: " . $file['name'];
// 1: The file size exceeds the limit allowed in php.ini
//         c:\xampp\php\php.ini            upload_max_filesize
// 2: The file size exceeds the limit allowed in HTML Form.
//         It's not used in Form in this case, but you could use it (try it!):
//         e.g. <input name="MAX_FILE_SIZE" value="1048576" type="hidden"/>
// $MAXSIZE = 500*1024;   w eset in config.inc.php
            elseif ($file['error'] == 1
                    or $file['error'] == 2
                    or $file['size'] > $MAXSIZE)
                $messages[] = " Too big file: " . $file['name'];
            else {
// converts the filename to lowercase and copy after the foldername
                $target_dir = $FOLDER.strtolower($file['name']);
                if (file_exists($target_dir))           // if the file exists
                    $messages[] = " Already exist: " . $file['name'];
                else {
// uploads the file to the target directory


// $fajl['tmp_name']: file temporary name
// With the temporary file we can avoid deleting a file with the same name, if exists.
                    move_uploaded_file($file['tmp_name'], $target_dir);
                    $messages[] = ' Ok: ' . $file['name'];
                }
            }
        }
    }
    // Visualization logic:
?><!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Gallery</title>
    <style type="text/css">     // CSS is the topic of the first Exam
        label { display: block; }
    </style>
</head>
<body>
    <h1>Uploading to the gallery:</h1>
<?php
```

```
    if (!empty($messages))
    {     // Displays the messages in unordered list:
        echo '<ul>';
        foreach($messages as $m)
            echo "<li>$m</li>";
        echo '</ul>';
    }
?>
// Create the Form based on the sample provided in the description.
// multipart/form-data: This value is required when you are using forms that have a file upload control
// https://www.w3schools.com/tags/att_form_enctype.asp
    <form action="upload.php" method="post" enctype="multipart/form-data">
        <label>First:
            <input type="file" name="first" required>   // The first input element is required
        </label>
        <label>Second:
            <input type="file" name="second">
        </label>
        <label>Third:
            <input type="file" name="third">
        </label>
        <input type="submit" name="send">
      </form>
</body>
</html>
```

3 – Web apps security – 30 min

A key issue for the reliability of web-based applications is the secure identification of the user. Identification is required not only at the time of login, but also continuously during the stay in the protected environment, until the user logs out and leaves the site.

Secure password

Most users usually choose some easy-to-remember, but not secure password. These passwords can be easily guessed using the "brute force attack" method.

Storing passwords in the database

Passwords may only be stored in encrypted form in the database!!!!!

Unencrypted storage of passwords should not occur in a final thesis (Szakdolgozat)!

1 - Sql injection - Why is the Prepared statement important, which we have used so far?

SQL injection is a common attack method that allows an attacker to read information from the database, change or delete data.

In XAMPP import **data.sql** file from **Solutions.zip**.

1st exercise - Logging in without knowing password!!!

login.php

- simple login form
- no protection against SQL injection attacks

- data submitted on the form is not validated
  it is inserted into an SQL statement without validation.
- On successful login, it prints the user's first name and last name from the table. On incorrect login, it prints an error message.

login.php

```
<?php
  if(isset($_POST['login']) && isset($_POST['password']))
  {
    try
    {
      $dbh = new PDO('mysql:host=localhost;dbname=datab', 'root', '',
array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
      $dbh->query('SET NAMES utf8 COLLATE utf8_general_ci');
      $username = $_POST['login'];
      $password = sha1($_POST['password']);
      $sql = "select * from users where login = '$username' and password = '$password'";
      $sth = $dbh->query($sql);
      $rows = $sth->fetch(PDO::FETCH_ASSOC);
      if(!$rows)
        echo "Incorrect login name - password pair!<br>";
      else
        echo $rows['first_name'] . " " . $rows['last_name'];
    }
    catch (PDOException $e)
    {
      echo "Error: ".$e->getMessage();
    }
  }
?>
<!DOCTYPE html>
<html>
  <head>
  <meta charset="utf-8">
  <title>PHP - SQL injection</title>
  </head>
  <body>
  <h1>Login</h1>
  <div>
    <div class="left">
      <form method="post">
      <label for="login">Login:<input type = "text" name="login"></label><br>
      <label for="jelszo">Password: <input type = "password" name="password"></label><br>
      <input type="submit" value="Login"><br>
      </form>
    </div>
  </div>
  </body>
</html>
```

Is it possible to log in to the system without knowing the user's password?

Enter the next **to the username textbox** (**a space is required at the end!!!**):
admin' --
You can enter anything in the password textbox or leave it blank.

You are logged in!
It displays the admin user's first name and last name. By logging in as admin, everything can be modified in a system!
Tom Smith

Ez az alap SQL utasítás:
"select * from users where login = '$username' and password = '$password'";

Entering the username admin' -- **this will be the final SQL statement:**
select * from felhasznalok where bejelentkezes = 'admin' -- and jelszo ……..

In SQL, -- means a comment. Anything after that is treated as a comment and ignored.

2nd exercise - Delete the user table!

You need to know the table name. In an open-source application, for example, Wordpress, all table names are known. If, for example, we insert a form into Wordpress and use an unprepared statement to process the data, the system can be easily attacked.

Enter to the first textbox:
something'; Drop table users;
anything can be written in place of something

Deletes the users table!!!
It gives an error message when executed: Fatal error: Uncaught TypeError: count(): …….

There are many similar tricks against systems that do not have proper protection.

1/B - Protection against Sql injection - Check incoming data!

In **login.php**:

Instead of it:

```
$sql = "select * from users where login = '$username' and password = '$password'";
$sth = $dbh->query($sql);
```

this:

```
$sql = "select * from users where login = :login and password = :passw";
$sth = $dbh->prepare($sql);
$sth->execute(Array(':login' => $username, ':passw' => $password));
```

Try it:

- login works well
- protects against SQL injection

B, with regular expression – we have already used it – Only informative text

For example, check the first input field:
Only letters, numbers, period(.) and underscore(_) are allowed for the login name.
Here we also require the following: The first character must be a letter, and the length must be at least 5 characters.

```
$username = $_POST['login'];
// http://php.net/manual/en/function.preg-match.php
if (preg_match('/^[A-Za-z]{1}[A-Za-z0-9._]{4,20}$/', $username)) {
        ……….
}
else {
        echo "Rossz bejelentkezési név formátum!";
}
```

2nd - Can we get the original password for a password code? – in many cases yes

If you can get someone's encrypted password from a database, you can often get the original password.
In the previous example, in the **data.sql** file, we saw that the login3 password has the following sha1 code:

df4d8ad070f0d1585e172a2150038df5cc6c891a

Can we get the original password for the code?

Copy the code to https://crackstation.net/ page. You can see the original password!:

Hash	Type	Result
df4d8ad070f0d1585e172a2150038df5cc6c891a	sha1	login3

Crackstation's lookup tables were created by extracting every word from the Wikipedia databases and adding with every password list we could find.
For MD5 and SHA1 hashes, we have a 190GB, **15-billion-entry lookup table**, and for other hashes, we have a 19GB **1.5-billion-entry lookup** table.

It knows several types of encodings: *LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults*

MD5 and sha-1 encryption are not completely secure, it is recommended to use the stronger sha-256, sha-512 instead.

I generated several types of encoding for the password **password12345**:

```
<?php
        $pass = "password12345";
        echo "md5: " . hash("md5",$pass)."<br>";
        echo "sha1: " . hash("sha1",$pass)."<br>";
        echo "sha256: " . hash("sha256",$pass)."<br>";
        echo "sha512: " . hash("sha512",$pass)."<br>";
?>
```

I got the next codes:
md5: 365d38c60c4e98ca5ca6dbc02d396e53
sha1: ae9030c665364eb2651d450e8321ae62dd51a726
sha256: 3700adf1f25fab8202c1343c4b0b4e3fec706d57cad574086467b8b3ddf273ec
sha512:
fb997d5c01ebcf962d820b3b0e7f8bfeeb7f4bd337cc83682f2af90d252c20c5d85744b7c6bb94f48139f690a61e4ad317d6107e4310efc016d9287266b5172b

Copy them to https://crackstation.net/ page. **It will find them all. sha512 too!**

It won't find this!: e.g. proba12345
Try your own code too. ☺

3 - Password salting – Improvements to previous encoding methods

https://crackstation.net/ writes: *This only works for "**unsalted**" hashes.*

With **password salting**, a random piece of data is added to the password before it runs through the hashing algorithm, making it unique and harder to crack. When using both hashing and salting, even if two users choose the same password, salting adds random characters to each password when the users enter them.

For example: proba1237SL2cRcLAP2FE7H, where proba123 is th original password.

Another way to increase the security of encryption is to **use multiple algorithms in series**. For example:
md5(sha1(password))
md5(md5(salt) + sha512(password))
sha1(sha256(password))
sha1(str_rot13(password + salt))
md5(sha1(md5(md5(password) + sha1(password)) + md5(password)))

This way the method will be unpredictable, the attacker will not be able to decipher whatever dictionary he/she has.

Others – Only informative text

Brute force attack

Brute force attack, also known as the full-trial method, is an attack method used against encryption systems.
It determines the key used by trying all possible keys. Its effectiveness is determined only by the technical (IT) background and the available time. Fast and high-capacity hardware (target hardware) is required. The breaking time depends on the number of possible keys, i.e. the size (length) and complexity (number of characters to choose from) of the key.

Number of possible keys = (number of characters)key length

Number of possible keys = $(\text{number of characters})^{\text{key length}}$
For a 128-bit key, there are 2^{128} possible keys.
340 282 366 920 938 463 463 374 607 431 768 211 456.
If there were a device that could check 1 trillion (10^{18}) keys in 1 second, it would take that machine 10 trillion (10^{13}) years to try all the keys.

Defense.
By changing the key frequently, we can force the attacker to constantly search for the key.

Login captcha solutions
In order to protect against "brute force" attacks, the system should filter automated attack attempts with a captcha solution after the first three login attempts.

Waiting time
After 3-5 unsuccessful attempts with the same username, we should disable login for a specified time, and log the unsuccessful attempts.
To slow down the login process, we should insert a 3-5 second wait into the login script. This can reduce the number of attempts that can be made during a given time.

Include files
It is recommended to always put include files outside the web-root folder, in a directory with a non-traditional name.
The include files should never have the extension .inc, but e.g. .inc.php, so that they run as php files, not as txt files, because in the latter case the information stored in them will be displayed immediately!
It is recommended to put an empty index.html file in every include folder so that it will run if someone browses the given path.

Directory listing (in .htaccess or httpd.conf) should always be disabled on the server!

Secure SSL connection
From the perspective of the security of the web application and the user data stored in it, it is worth considering the use of the SSL protocol, which provides an encrypted communication channel between the user's computer and the server.

For databases, do not use the old, no longer supported mysql_query function.
http://php.net/manual/en/function.mysql-query.php
Use the PDO driver instead.
http://php.net/manual/en/ref.pdo-mysql.php

Filtering input fields
Not only in the login form but everywhere in the application, we should strive to limit the length and data type of input fields. Where possible, use drop-down lists, and in the case of text fields, limit the number of characters in HTML.

Data leakage prevention
Data Loss Prevention (DLP) literally means “data loss prevention”, which would mean preventing the physical destruction of data, but is actually used in the sense of “data leakage prevention”. Data leakage is a security-related information technology term used in cases where protected, confidential data somehow escapes, “leaks” from a protected environment.
DLP systems are designed to detect and prevent unauthorized use and transmission of confidential information. DLP software and solutions are IT protection systems that identify, monitor and protect confidential data, both at the endpoint, in the network and on data storage. This protection is achieved through comprehensive content-based scanning, security analysis of data streams (authority of the creator, subject of the data, timing, recipient/destination, etc.), and a centralized management system.

Google dorks and the Google hacking based on them

Googlebot – Google's web crawling robot – basically indexes and caches everything it sees, and there is no specific mark on the web server that it is not going to be cached by any search engine. This also happens with documents that the user has uploaded somewhere knowing that no one will find them anyway.
So, for example, you can search for files with Google that are likely to contain username-password pairs, e.g. in Excel format. With one such dork, you can find tens of thousands of documents on the web.

Google search: password filetype:xls
there are many such files!!!
Searching for such documents is not illegal – but using the data extracted from them is, of course.

The **Offensive Security** website has a collection of tricks available to everyone that can be used to extract similar sensitive data:
https://www.exploit-db.com

Others

Cross-Site Scripting (XSS)
SESSION hijacking
SESSION fixation
Cross-Site Request Forgeries (CSRF)
stb…..

12 – Exam-2 – with computer based exercises