1) HTML and the Web

Agenda

- Introduction of the course instructor
- Introduction of participants (shortly: name, why did you apply, do you have any prior knowledge?)
- The basics of the internet and World Wide Web
- Intro to HTML

Note for students:

- Make sure to **attend** course sessions. Attend even if you skipped a few sessions before or you didn't do your homework.
- If you'd like to become a programmer, **do homework regularly**.
- If you joined the course the understand coders and coding better (and you don't intend to become a programmer), then you don't have to do homework (unless you'd like to). But still make sure to attend the course sessions regularly and do coding examples at the sessions.
- If you want, **you can read each lesson** in the curriculum even **before you have a**

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

session with instructor. This way you come to the session **more prepared** and you can ask better questions.

- Students can also gather at **informal meetings** in order to do a homework together. This meetings can take place in a library, a coffee place etc. - it's up to you. But you have to **organize this yourselves**.
- Also search Meetup.com (https://www.meetup.com) for **local coding meetups** and **attend them** (either alone or - even better - as a group!).

Introduction to the World Wide Web

Let's start with a **question** for the students first:

What is the internet?

Short answer: a network of computer networks.

Okay. But what is then the World Wide Web (WWW)?

Short answer: WWW is one of the things that run on the internet. The term applies to websites, browsers, HTTP protocol and HTML language.

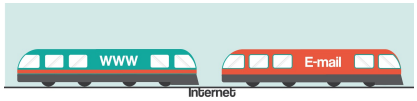
One of the other things that also run on the internet is email. Email is not part of the WWW - it has its own protocols and it is actually older than World Wide Web!

Internet and World Wide Web **are not the same thing**. In short: The term **internet** means a net of **computer networks**, while **World Wide Web** applies to a **browser**, such as Chrome, Firefox, Safari etc.

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

> *You can imagine a relationship between internet and WWW as a railway. Internet is the rails and WWW is a train that runs on that rails. WWW is totally useless without the rails.*
>
> 
>

A brief history of the internet

Internet is a network of computer networks. It started coming into existence in **late 60s** when 4 U.S. universities connected into the **first (long distance) computer network called ARPANET**. This project was financed by the U.S. Department of Defense.

Later some other US universities and research facilities joined the project, but there were also some that weren't invited. Some of them decided to create their own computer network. The same thing started happening in Europe, especially in the UK and France. This resulted in many different computer networks all across the world that initially were not connected to each other. Later they decided to also **connect to each other** (via the common protocol called TCP/IP) and thus the **Internet was born**. The network of networks.

How the computer networks were being used?

You could connect to any computer in your network and access their public folders. Computer did not have a graphical user interface (GUI) back then, so you had to do everything

through the **terminal** by writing text commands.

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

> ***Example in the terminal***
>
> *You could connect to another computer via special command and access it's public folder. Then you'd type* `ls` *which would show you a list of all the files and subfolders within that public folder. If you wanted to open certain document, you'd use a program like vim and type* `vim document_name` *to open the document and read it.*

The problem with terminals was that reading documents through it kind of... sucked. :) You could only have **one font type** in the terminal. You could not make the text bold or color it. And most importantly - **you could not add a link to some other document**.

If you wanted to let the reader of your document know that there's another good article to read, you had to write the IP address of that computer and write down in which folder and by what name

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

can that article be found. So then the reader had to disconnect from your computer, connect to another computer and try to find the recommended article... if it was still there at all.

This was quite a slow and frustrating process. Just think how upset you can get nowadays if a website takes too long to load. :)

Solution: World Wide Web

As we said, the first steps into building the internet were made in the 60s. But the World Wide Web came much later, in the **year 1990**. It was invented by Tim Berners-Lee, a CERN scientist, who was also frustrated by the limitation of the internet that we discussed before. That's why he built the **first browser**, the **HTTP protocol** and the **HTML language**.

HTTP means HyperText Transfer Protocol and HTML means HyperText Markup Language. They both include a word **HyperText** which is the main element of the Web. HyperText means an **enriched text**. So instead of having only plain text in your document (as it is in the terminal), you could have a text enriched with images, you could make some words bold, italic, you could make it bigger or smaller - and most importantly, you could **add some links to another documents**! That's how Tim Berners-Lee solved a problem that was frustrating many users of the internet.

The first website ever created

Here's the first website ever, created by Tim Berners-Lee: http://info.cern.ch/hypertext/WWW/TheProject.html (http://info.cern.ch/hypertext/WWW/TheProject.html)

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

See this video at home:

Let's hack BBC!

Before we start coding with HTML, let's do something crazy - let's hack the BBC website!

Instructor: Go to BBC.co.uk (http://www.bbc.com/), click on some article and open developer tools (right mouse click: Inspect element). Then change the title of the article to whatever you want.

Now ask the students to find this website on their computers. Has the title really changed?

No.

What happens if instructor reloads website on his computer?

It shows the **original article title** again (all the changes are lost).

Why does this happen?

Because of the way how the World Wide Web works. When you open a browser and enter a URL to some website (like www.smartninja.org), your browser **sends a request to the server** that hosts that website.

Server is just another name for a **computer that's connected to the internet 24/7** and has a task to **host websites**. When server receives a request from a browser, it **returns a copy of the website** that was requested.

This copy is in the form of an HTML document

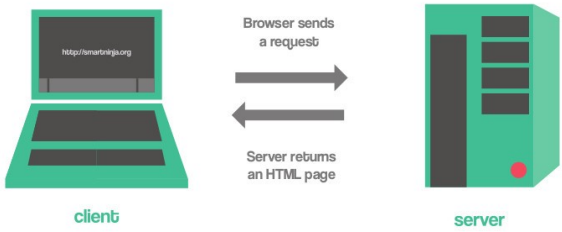
Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

and it **saves itself on our computer** (in some temporary folder). The original is still on the server and you get a COPY of it on your computer. That's the copy you can change using Developer Tools - but it **won't change the original** stored on the server.

When you reload the website, your browser sends another request to the server and receives another copy (and your old one gets removed).



Short exercise (5 min)

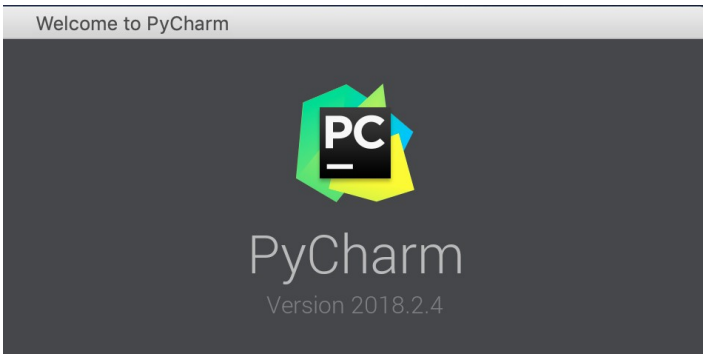
Use Developer tools to change the title and some text in the article of your choice.

Working environment

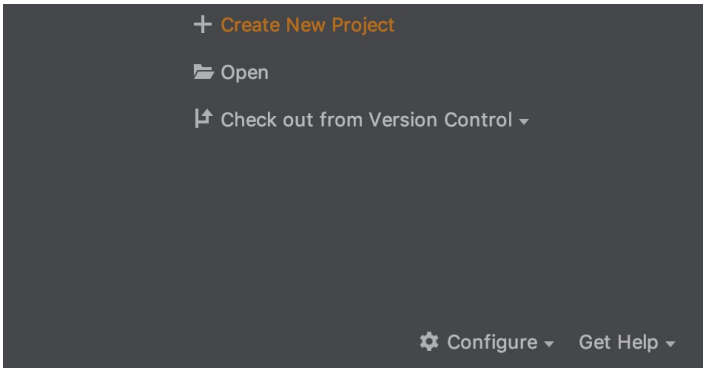
Before we start writing HTML, we have to prepare our working environment.

Let's start by opening PyCharm (you have to have it installed by now). PyCharm is a code editor, this means it's a program where we can write code in.

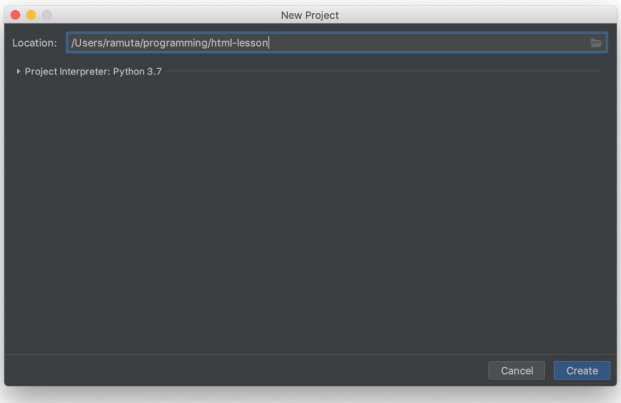
When the program opens, click on **Create new project**:



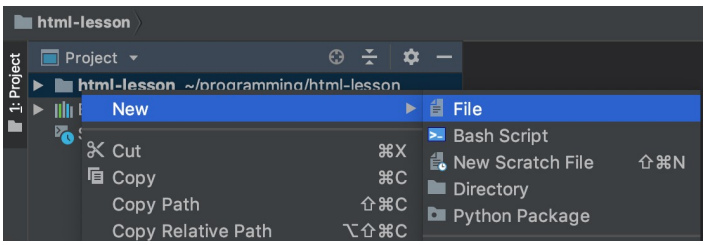
Courses (/student)
Forum (/student/forum)
Profile (/student/profile)



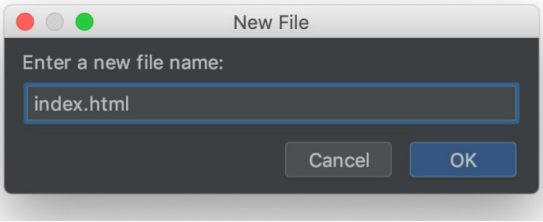
Then choose the correct folder (where you want your coding projects to be). We suggest to have a special folder named "programming". Then write the name of your project after it. It can be something like `html-lesson`. Then click create:



Next create a new file in the project. Right-click on the project name and select New --> File (don't select HTML file!):



Then enter `index.html` in the input box:



Courses (/student)

Forum (/student/forum)

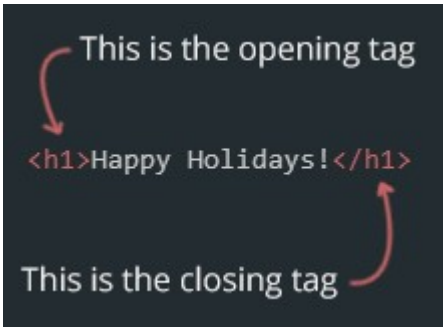
Profile (/student/profile)

Hit OK and that's it! We can start with the HTML lesson now.

What is HTML?

HTML is a HyperText Markup Language. We already know what a HyperText is - but why do we call it a **markup language**? Because, just like in Word (or similar program) we can **mark** some text and make it for example: bold, italic, bigger etc.

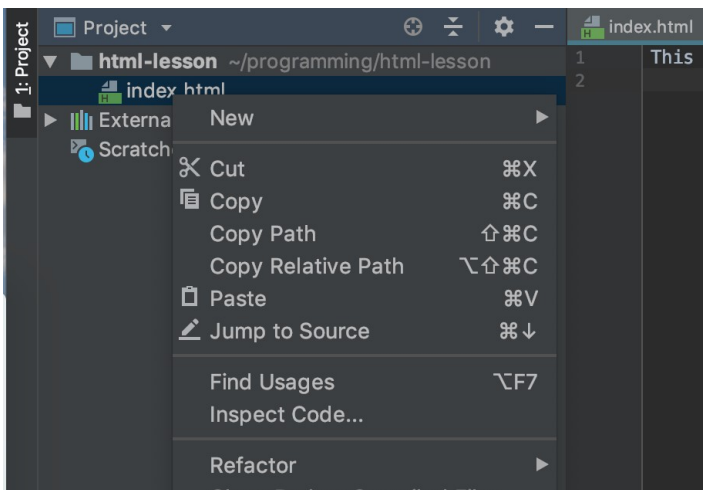
We mark the text with **tags**. We put the **start tag** (a.k.a. opening tag) at the beginning of some text and an **end tag** (a.k.a. closing tag) at the end of the text.



Write the following code in the HTML file:

```
This is <strong>a bold part</strong> of the text
```

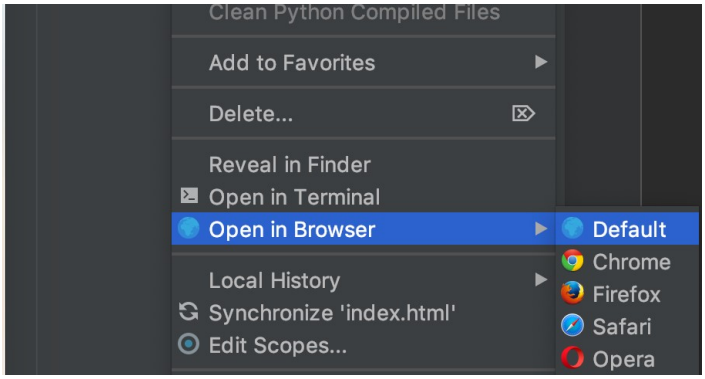
Then right click on the file name and select Open in browser --> Default:



Courses (/student)

Forum (/student/forum)

Profile (/student/profile)



A browser tab will open and show you the result of your HTML code.

Congrats! You've wrote your first piece of the HTML code!

Other HTML tags

There's a lot of HTML tags. You can see a list of **all the tags** here: http://www.w3schools.com /tags/ (http://www.w3schools.com/tags/).

Also try examples for h1 (and h2, h3), a, table, li:

```
<h1>Big title</h1>
<h2>A bit smaller title</h2>
<h3>Even smaller title</h3>

<a href="http://smartninja.org">I am a web link<

<table>
    <tr>
        <td>First field</td>
        <td>Second field</td>
        <td>Third field</td>
    </tr>

    <tr>
        <td>Field in the second row</td>
    </tr>
</table>

<ul>
    <li>first in the unordered list</li>
    <li>second in the unordered list</li>
    <li>third in the unordered list</li>
</ul>
```

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

> *Important: do not copy/paste the above examples! Type them by hand yourself ;)*

Not all tags are the same

We also have tags, that **don't have a closing tag**. Why not? Because they don't surround any text, but instead have other uses.

The most popular example is the `<img>` tag.

> *Try the `<img>` example. Other examples are `<br>`, `<hr>`, `<input>`.*

`<img>` introduced us to a new thing: **atributes**. Attributes lay inside a tag and are basically settings for tags..

The most important attribute inside the `<img>` tag is `src`. It tells us what image (or path to image) should `<img>` show:

```
<img src="https://storage.googleapis.com/smartni
```

In this case, the value of the `src` attribute is `https://storage.googleapis.com/smartninja/smartninja-logo-2-png1507892784.png` (this is a path to an image).

> *Question: what was an attribute within the `<a>` tag that we used in one of the previous examples?*

HTML structure

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

Every HTML document needs a **structure**. Having a right HTML structure is important, although even if you don't use it (correctly), the browser might still show you a desired result. But ignoring the HTML structure is a sin that will take you to a Coding Hell (and your website won't be on the first page on Google). :)

A basic HTML structure:

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

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

Every HTML document starts with a DOCTYPE tag that tells a browser what is the type of the document (in our case: html). This DOCTYPE tag does not have a closing tag.

Next we have an `<html>` tag that includes everything else within the document. It's closing tag (`</html>`) is the last thing in the document.

`<html>` has only two children: `<head>` and `<body>` . HTML elements are in a **family relationship** - they can be either children, parents or siblings.

The **parent-child** relationship means that one HTML element wraps (or encompasses) another element. In our case `<html>` element wraps `<head>` and `<body>` elements, so `<html>` is a parent to `<head>` and `<body>` (they are children).

This hierarchy is very important to understand and to follow.

All the HTML elements that we tried in the previous examples (such as h1, h2, table etc.) have to go inside the `<body>` element. They

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

can't go inside the `<head>`. Try to add couple of them inside the body and show them in the browser.

So what do we put inside `<head>`?

All the HTML elements visible in the browser are in the body. But in the **head** we put elements that **are not visible in the browser**. Why not? Because this are usually some meta data that help search engines (like Google) to easily find and categorize our website.

Here are some examples of these meta tags:

```
<meta charset="UTF-8">
<meta name="description" content="My first websi
<meta name="keywords" content="HTML, CSS, SmartN
<meta name="author" content="Matej Ramuta">
```

We also put a path to our CSS file into `<head>`, but more about that later. As we said, none of the tags within the head is not visible in the browser... except one - it's called `<title>`. Where do we see it? In the tab.

Try it out yourself:

```
<title>Hi, you can see me in the tab</title>
```

Q&A

Any question?

If there's enough time after Q&A, students can start working on their homework. The instructor will briefly present the task for the homework and the "Read-Search-Ask" approach.

Homework 1.1: Fakebook

Now that you know the basics of HTML, you're ready to build your first project - your very own

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

social network!

Don't worry, it's not as hard as it sounds. You will only build an example of your profile on this imaginary social network called Fakebook. And you'll use HTML for that.

But before you start, set up the new project environment. Open PyCharm, click on Create New Project and create the Fakebook project (the same way as earlier in this lesson).

Now you can start building you Fakebook profile with the HTML language you just learned. It will probably look something like this: https://storage.googleapis.com/smartninja-org-assets/curriculums/cw/de/fb-de.html (https://storage.googleapis.com/smartninja-org-assets/curriculums/cw/de/fb-de.html).

When you finish the project, paste the code you build into GitHub Gist: https://gist.github.com/ (https://gist.github.com/) and share it on the forum. It will serve you there as a proof of you first coding project. :)

> *You can see one possible solution for the exercise here: https://github.com/smartninja/wd1-py3-exercises/tree/master/lesson-01/fakebook (https://github.com/smartninja/wd1-py3-exercises/tree/master/lesson-01/fakebook). You should know that in coding, there are many different ways of how to do a task.*

Read-Search-Ask

When you do your homework, follow the so

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

called “read-search-ask” approach when you face some challenge. It consists of 3 steps:

1) Read

For example, if you get an error with your code, check the console/terminal (if possible) to determine what kind of error did you get. If there’s no error message to read and the things still don’t work the way you want, check the documentation for the coding language or framework that you use.

2) Search

If the documentation can’t help you or you don’t know where to find it, use (Google) web search to find it. If you got an error message in the console/terminal, copy that error in the search and find the solution. One of the best websites that help you find solution to your coding problem is StackOverflow (http://stackoverflow.com/).

3) Ask

If none of the above steps work, ask your question on the SmartNinja forum or ask instructor at the lesson. You can also post your question on the previously mentioned StackOverflow.

Useful resources

- HTML validator to validate your HTML code: http://validator.w3.org/ (http://validator.w3.org/)
- Basics of HTML (Mozilla) (https://developer.mozilla.org/en-US/Learn/Getting_started_with_the_web/HTML_basics)
- Microsoft Virtual Academy: http://www.microsoftvirtualacademy.com/training-courses/html5-css3-fundamentals-development-for-absolute-beginners (http://www.microsoftvirtualacademy.com

Courses (/student)

Forum (/student/forum)

Profile (/student/profile)

/training-courses/html5-css3-fundamentals-development-for-absolute-beginners)

- Check the HTML part: HTML&CSS for the entrepreneurs (https://www.udemy.com/programming-for-entrepreneurs/#/)
- How browser communicates with a server: https://www.youtube.com/watch?v=kBXQZMmiA4s (https://www.youtube.com/watch?v=kBXQZMmiA4s)