2) Intro to CSS

Agenda

- Fakebook HTML homework
- Basics of CSS

HTML homework

Your homework at the first lesson was to **build a Fakebook profile** using your HTML skills.

At the end, your Fakebook should look something like this:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Matej Ramuta | Fakebook</title>
    </head>

    <body>
        <h1><i>FAKEBOOK</i></h1>
        Search: <input type="text" placeholder="Search by name">

        <br>

        <h2>Matej Ramuta</h2>
        <img src="https://storage.googleapis.com/smartninja-org-assets/curriculums/cw/de/sleepy
        <br>
        <br>

        <button>Add a friend</button>
        <br>

        <p><strong>City:</strong> Vienna</p>
        <p><strong>Age:</strong> 20+</p>
        <p><strong>Education:</strong> Ninja University degree</p>
        <p><strong>Hobby:</strong> sleeping</p>
        <p><strong>FB friends:</strong> 2341234235234523454235</p>
        <p><strong>FB enemies:</strong> -15</p>
    </body>
</html>
```

What is CSS?

HTML is the main language of the Web and CSS is a sort of a sidekick to HTML. They are like Batman and Robin. **HTML is Batman and CSS is Robin.** Batman can fight and save lives on his own, which Robin can't (at the beginning Batman actually was alone). But Robin can help him and make the world more colorful. :)

CSS means Cascade Style Sheets. In short: **CSS gives style to HTML**. It gives HTML colors, it's able to position HTML elements on the site etc. Let's see some examples!

First **create a new PyCharm project**. You can name it `css-lessons`.

Create index.html

Create index.html with the following HTML code:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>SmartNinja CSS lesson</title>
    </head>

    <body>
        <p>Red colored text</p>
    </body>
</html>
```

Run the code in the browser.

Create style.css

Right click on the project, select New --> File. Enter `style.css` in the input box. Enter the following code in:

```
p {
    color: red;
}
```

Connect style.css to the HTML file

Create a `<link>` element and place it within the `<head>` tags:

```
<head>
    <!-- other code -->
    <link rel="stylesheet" href="style.css">
</head>
```

Reload your browser tab. The text should be colored red now.

Using CSS

Let's learn using CSS by examples. Create another paragraph in your HTML body:

HTML:

```
 <p>Red colored text</p>

<p>Another red colored text</p>
```

These two paragraphs are now both red, because they use the same CSS code. This code colors the text based on the **selector**, which in this case is `p`. So every paragraph in the website will be colored red.

What if we want to have a different color for each paragraph?

HTML:

```
<p>Red colored text</p>

<p>Blue colored text</p>
```

CSS:

```
p {
    color: red;
}
```

The above example doesn't work as we wanted. The second paragraph should be blue, not red!

In this case we shouldn't give a color to the p tag (in our CSS code), but instead do something completely different. Every HTML tag can have an `id`, so we can give color to a specific paragraph based on it's id:

HTML:

```
<p id="red">Red colored text</p>

<p id="blue">Blue colored text</p>
```

CSS:

```
#red {
    color: red;
}

#blue {
    color: blue;
}
```

This looks good :)

But what if we want to have another blue paragraph?

HTML:

```
<p id="red">Red colored text</p>

<p id="blue1">Blue colored text</p>

<p id="blue2">Another blue colored text</p>
```

CSS:

```
#red {
    color: red;
}

#blue1 {
    color: blue;
}

#blue2 {
    color: blue;
}
```

This works, but is not the best solution, because we violate the DRY principle (the `color: blue;` code repeats unnecessarily).

Besides `id`, there's also a similar attribute, called `class`. Let's use this one!

HTML:

```
<p id="red">Red colored text</p>

<p class="blue">Blue colored text</p>

<p class="blue">Another blue colored text</p>
```

CSS:

```
#red {
    color: red;
}

.blue {
    color: blue;
}
```

So why can we use `class` and not `id`? Because **one** `id` can be used in **only one** HTML element. Two or more elements cannot share the same `id`. But they can share the same `class`.

Recap

As you can see, we can select HTML elements in three main ways in your CSS code:

- **element type selector**: when you use the html tag name as a selector (`p {}`)
- **id selector**: you assign an HTML element an id and then use it in the CSS code (`#red {}`)
- **class selector**: you assign one or many HTML elements a class and use it in the CSS code (`.blue {}`)

Where to find CSS properties examples?

- CSS properties: http://www.w3schools.com/cssref/default.asp (http://www.w3schools.com/cssref/default.asp)
- CSS selectors: http://www.w3schools.com/cssref/css_selectors.asp (http://www.w3schools.com/cssref/css_selectors.asp)
- StackOverflow: http://stackoverflow.com/ (http://stackoverflow.com/)
- CSS examples: http://www.w3schools.com/css/default.asp (http://www.w3schools.com/css/default.asp)

Remember, Google is your friend ;)

Example of using CSS documentation

Let's make an example where we add a dotted border to some image. When we go through the image with a mouse, the border should change to solid red.

Let's take a look at the CSS docs (links above) and find out how to do it:

HTML:

```
<img src="http://design.ubuntu.com/wp-content/uploads/ubuntu-logo32.png" class="image">
```

CSS:

```
.image {
  border: 6px black dotted;
  padding: 5px;
}
```

We added some `padding` to move the border a bit away from the image (we'll talk about this more in the next lesson).

Now let's add some code that will find out if we went over the picture with our mouse. In this case our image will get solid red border. Add the following code below `.image`:

```
.image:hover {
  border: 6px red solid;
}
```

We used a selector called `:hover` that can find out if we put a mouse cursor over some HTML element.

Searching for help on web & StackOverflow

> *If there's not enough time, cover this in one of the next lessons.*

No programmer can learn all the code by heart, less there's one that would know **everything about IT**. That's why coders often use **web searching** when they face challenges with programming.

Perhaps the most **important website** in this case is StackOverflow.com (http://stackoverflow.com/). StackOverflow is a Q&A (Questions & Answers) site which means users can post questions (related to programming) and help each other answer them.

Let's take a look, for example, at this question, where some user asks how to center a div within another div (http://stackoverflow.com/questions/114543/horizontally-center-a-div-in-a-div).

As you can see, the first post is a question and below are the answers. The order of answers is based on **how many votes they received** from other users. User, that posted the question, can also **choose some answer as the best** (that answer will then receive a green checkmark ✓).

A lot of questions, especially beginner ones, **have already been posted**. So often you don't really need to post a question, but rather to **search** if the same question already exists and see what was the best answer to it.

The best way to search on StackOverflow is **through Google**. Here you can use a **searching trick** that will only show you results from StackOverflow. Let's say, for example, that you'd like to know how to color some text red using CSS. You'd write something like this in Google:

```
how to make text color red in css site:stackoverflow.com
```

With `site:stackoverflow.com` you let Google know that you **only want results from the StackOverflow website**.

Q&A

Any question?

If there's enough time after Q&A, students can start working on their homework.

Homework 2.1: Make your Fakebook prettier

Make your Fakebook prettier using CSS. When you finish the project, paste the code to GitHub Gist: https://gist.github.com/ (https://gist.github.com/) and share it on the forum.

> *You can see one possible solution for the exercise here (https://github.com/smartninja/wd1-py3-exercises/tree/master/lesson-02/fakebook).*
>
> *Or even better structured (with subfolders) here: link (https://github.com/smartninja/wd1-py3-exercises/tree/master/lesson-02/fakebook-better-structure).*