Building Web and Mobile Apps WS2025/26 (Prof. Dr. Cathrin Möller)

Exercise 1 - 27./28.10.2025

Practical Exercises

Ensure you did setup the private GitLab repo „bwma-25-26“ in the correct GitLab (https://git-ce.rwth-aachen.de/) and invited @*cathrin.moeller* as a “Reporter“.

Please also invite my tutors as a „Reporter“: **@pinky-jitendra.tinani** and **@rahul.patil**

You will need this in the following weeks for the submissions proving active participation. It also helps as we will continue working on the same application for multiple weeks.

Please remember to submit to your GitLab repo regularly. You will need the solutions from today for the upcoming exercise classes and for „Active Participation“ we will check that regularly commits happened.

1) As soon as a frontend gets more complex than pure HTML and CSS, we need pre-compilers like SASS to allow writing more maintainable styling code. Ensure Node.JS and NPM are installed and running by checking the installed versions

node -v

npm -v

Now navigate to your project folder (`cd vanilla`).

Run `npm init` to create a NPM project and follow the command line instructions (you can skip all answers, taking the default by pressing enter multiple times in a row).

How does the created package.json file look like?

Run `npm install sass –save–dev` to install sass just for the local project.

Add the package.json file to version control commit and push to your repo.

Do not add package-lock.json or node_modules folder to GIT. If you want to avoid them being added, please maintain a .gitignore file and add this to version control.

2) Create a main.scss file with the following content:

```
$primary-color: #24a0ed;
h1 {
  color: $primary-color;
}
button {
  color: #fff;
  background-color: $primary-color;
  border: none;
  padding: 10px 20px;
  border-radius: 15px;
  &:hover{
    color: $primary-color;
    background-color: #ffff;
    border: 1px solid $primary-color;
  }
}
```

Translate it to a main.css file. You can do this from the command line calling „sass main.scss css/main.css“

Inspect what gets created, add the files to git, commit and push.

3) Create a main.html file with a <h1> headline „Hello World“ and a <button> with some text and use the translated scss setting a reference to the external css file. Visit your styled website in a browser of your choice.

Hint: You can either use IDEA to see your HTML file within a browser or you can use the http-serve. NPX allows using the NPM package without installation. You can serve a complete folder with your HTML and CSS locally under http://localhost:8000

Check the path to project typing „pwd“ in the command line.

You can also add „.“ to render the current folder content.

```
npx http-server /path/to/project -o -p 8000
```

Now you should be able to access the index.html via http://localhost:8000 and your main.html file via http://localhost:8000/main.html.

Also the css should be retrievable from your webserver. Check the browser developer console to debug the download of your external css file.

Commit and push all created files.

4) Create `„scripts“: {}` within your package.json file. Here you can add command line commands. If you add `"serve": "npx http-server . -o -p 8000“` you can run ***`npm run serve`*** as a shortcut.

Create another script `„css"` for your scss to css translation.

And another one that combines both commands to ensure you have the latest css served assuming as a developer you only change the code in the HTML and the SCSS file. You can add multiple commands like ***„npm run css && npm run serve“*** here.

Modify the colors in the scss and re-compile to css using the ***npm run start*** command.

Commit and push the changes in your package.json file.

5) Imagine we have two HTML files with one separate CSS file that applies to each

=> dashboard.html & dashboard.scss (converted to dashboard.css) and home.html & home.scss (converted to home.css)

Add 4 boxes in both of the HTML files.

Each box shall get a different background-color. The colors should be defined by **variables** at the top of your **SCSS** and in the two SCSS files, the colors should be different. Each box should have the same width and height.

You need to ensure your SCSS gets compile to CSS after changes. Run the compilation commands or add a convenience script to your package.json file.

Run a webserver to visit localhost:8000/dashboard.html and localhost:8000/home.html and see the referenced externalized css being applied.

Add, commit and push all files.

6) You can round the borders of the boxes using border-radius in CSS.

Round the edges of the boxes a bit by trying different values.

What **border-radius** is needed in relation to the px of width and height of a box to make it a **circle**? Change the SCSS to display 4 circles in home.html.

Recompile the SCSS to CSS, restart the webserver and ensure no caching shows you outdated content.

Add, commit and push all files.

7) Write a **_shared.scss** file which provides a **mixin** for the box base layout. This way we want to avoid duplicate code in home.scss and dashboard.scss. And we avoid that the scss compile created a shared.css file which we do not need if we just import from that file to others.

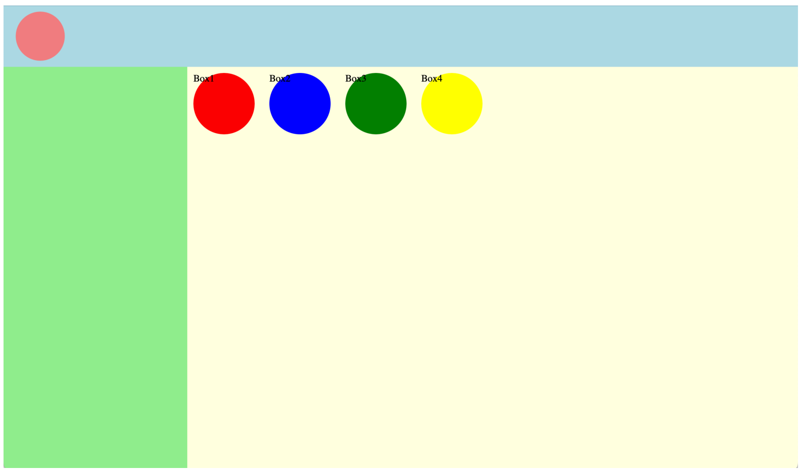
Write a mixin using **@mixin** and reuse it with **@use "shared"** in the two SCSS files which want to inherit the code. You will also need using **@include shared.<mixin name>** to apply the same rules in both home.scss and dashboard.scss.

8) A basic **Dashboard Layout** is something commonly used for all web applications. Revisit your dashboard.html. Please add a <**header>** at the top of the <body> content in the HTML file, height is 100px, width is full width. Inside that header at the top left, there should be a circle made with CSS in a different color which is aligned 20px from the left and 10px from top and bottom of header. Now add below the header a **sidebar** using <aside> (300px width fixed) and a variable **content** area using <main> tag that takes the remaining space. In the content area you can keep the old content with the 4 boxes from previous exercises.

Set different background colors for the header, sidebar and content area to make them visible. Ensure the sidebar and content stretch over the whole screen down to the bottom. CSS should be loaded from the external dashboard.css file as a reference in the <head> section of HTML. If something is not working properly, maybe you might need a reset CSS as well.

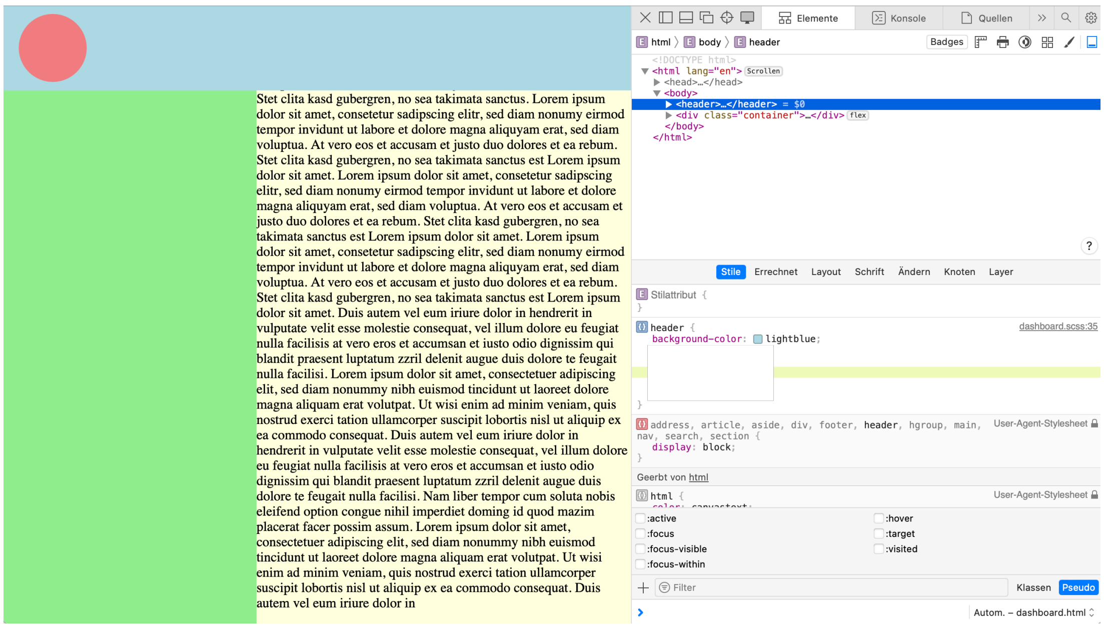
Be aware CSS caching can be hard to overcome, open Web developer console and disable cache and reload page. You can inspect what CSS rules are applied to your HTML by right clicking on rendered elements and selecting sth like „Element information / Inspect element“.

Result might look similar like this:



Commit and push

9) Make the header „**sticky**“ using position „sticky“. Add a longer placeholder content (e.g. lots of text) in the content area until the content does not fit on a screen anymore and scrollbars appear. Generating such text can be done from „ https://www.loremipsum.de/ “. If you scroll down, the header should always stay visible on top of the screen.


Stet clita kasd gubergren, no sea takimata sanctus. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in

```
<!DOCTYPE html>
<html lang="en"> Scrollen
  <head>…</head>
  <body>
    <header>…</header> = $0
    <div class="container">…</div> flex
  </body>
</html>
```

```
Stilattribut {
}

header {                                             dashboard.scss:35
  background-color: lightblue;
}

address, article, aside, div, footer, header, hgroup, main,   User-Agent-Stylesheet
nav, search, section {
  display: block;
}

Geerbt von html

html {                                               User-Agent-Stylesheet
```

:active
:focus
:focus-visible
:focus-within
:hover
:target
:visited

Commit and push