

Week 02: React, Node, HTML

...semantics, semantics, semantics.

Agenda

- Prerequisites
- Creating a React project from a vite template
- React project structure and practices
- HTML5 revision & document structure
- Inline, Block elements
- Tables, Forms
- Meta tags
- Lighthouse, Inspect
- Documentation for WEB development
- Demo - dummy form for investing

Let's start!

Prerequisites

You should have `node` and `npm` commands available. **These commands are available on school Linux computers (nymfe, musa), so you can skip this step.**

When you work on your own computers:

- If you have not installed a Node.js manager yet, do so via [nvm](#) or [fnm](#) install pages. We advise you to install the manager of your choice under the Unix environment (Linux, MacOS, WSL). Windows has its own version of `nvm` - `nvm-windows` which is not maintained by the creators of `nvm` for Unix computers. We prefer you to work under WSL so both `nvm/fnm` will work fine there.
- If you have installed your Node.js manager (`nvm/fnm`) and the commands are not available, it might mean you have not installed Node yet, or your path has not been updated. Check the links above to see how to install the latest version depending on your Node.js manager. There are FAQ & troubleshooting pages for both.

Let's create a React app via the vite template

```
npm create vite@latest seminar-02 --template react-ts
```

or just:

```
npm create vite@latest
```

Where you can specify the underlying frontend technology & TypeScript support.

Inspect your Node project

- Check out `package.json` & included scripts
- Install dependencies included in your `package.json` via:

```
npm i  
# or  
npm install
```

- Check out `package-lock.json`
- Check out the project structure:
 - `index.html` file
 - `src` & `public` folders
 - Check out where React is added injected onto the page

Run your React app

```
npm run dev
```

Possible discussion:

- How does it work? How do the files get in the browser?
- Now, try to change something on the page. What does HMR mean?
- Create a folder for pages/views and a folder for components? What's the difference between pages and components, when we represent them both as components?

JSX/TSX

JavaScript(/TypeScript) XML (formally JavaScript/TypeScript syntax eXtension) - allows writing logic & code along with the underlying markup

```
import { useState } from 'react'
import './App.css'

function App() {
  const [count, setCount] = useState(0);

  return (
    <>
      <h1>Vite + React</h1>
      <div className="card">
        <button onClick={() => setCount((count) => count + 1)}>
          count is {count}
        </button>
      </div>
    </>
  );
}

export default App;
```

For the demo, we will need to quickly revise HTML. Ready?

HTML5 revision

- HTML = HyperText Markup Language
- Used for web documents, gives meaning to the content
- **Not** a programming language
- Latest specification of HTML - **HTML5**
 - Current standard
 - New tags for semantic only (`<article>` , `<aside>` , `<footer>`)
 - Elements for multimedia (`<audio>` , `<video>`) - uses shadow dom
 - Since 2014, previous version from 1997

Document structure

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>A simple HTML document</title>
  </head>
  <body>
    <p>Hello World!<p>
  </body>
</html>
```

Document structure

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Page title</title>
    <link rel="stylesheet" href="style.css">
  </head>

  <body>
    <header>
      <h1>Header</h1>
    </header>

    <nav>
      <ul>
        <li><a href="#home">Home</a></li>
        <li><a href="#projects">Projects</a></li>
        <li><a href="#contact">Contact</a></li>
      </ul>
    </nav>

    <main>
      <article>
        <h2>Article heading</h2>
        <p>Lorem ipsum</p>
        <p>Lorem ipsum</p>
      </article>
    </main>

    <footer>
      <p>All rights reversed.</p>
    </footer>
  </body>
</html>
```

Common inline elements

```
<span>span</span>  
<a href="http://foo.com">link</a>  
<b>bold</b>  
<i>italic</i>  
<u>underline</u>  
<code>code</code>  
<input placeholder="Insert text here" type="text">  
<button type="button">button</button>  

```

span [link](#) **bold** *italic* underline code



Block elements

```
<div>
  block
</div>
<p>paragraph <i>holds</i> text</p>
<h1>heading 1</h1>
<h6>heading 6</h6>
<ul>
  <li>unordered</li>
  <li>list</li>
</ul>
<ol>
  <li>ordered</li>
  <li>list</li>
</ol>
```

block

paragraph *holds* text

heading 1

- unordered
- list

heading 6

1. ordered
2. list

Block vs inline elements

	Block	Inline
width	parent block	content only
break line	yes	no
children	any	inline or text

Tables

```
<table border="1">
  <thead>
    <tr>
      <th colspan="2" scope="col">Name</th>
      <th scope="col">Age</th>
    </tr>
  </thead>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>43</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>57</td>
  </tr>
</table>
```

Name		Age
Jill	Smith	43
Eve	Jackson	57

Forms

```
<form action="/path/to/handler" method="post">
  <label for="email">Email:</label>
  <input type="email" name="email" id="email" placeholder="john@example.com" required>

  <label for="password">Password:</label>
  <input type="password" name="password" id="password" pattern="[0-9a-fA-F]{4,50}"
  required>

  <button type="submit">Submit</button>
</form>

<!-- css showing HTML5 validation states
input:invalid { border: 2px dashed red; }
input:valid { border: 1px solid green; }
-->
```

Email: Password:

Meta tags

- Not rendered, information for search engines (SEO)
- Page title and description in search results
- `og:` properties for social media previews
- `title`, `favicon` in tab, color

```
<head>
  <title>My page</title>
  <meta charset="UTF-8">

  <!-- SEO -->
  <meta name="description" content="Free Web tutorials">
  <meta name="keywords" content="HTML, CSS, JavaScript">
  <meta name="author" content="John Doe">

  <!-- Open Graph protocol / social media previews -->
  <meta property="og:title" content="The Rock">
  <meta property="og:type" content="video.movie">
  <meta property="og:url" content="https://www.imdb.com/title/tt0117500/">
  <meta property="og:image" content="https://ia.media-imdb.com/images/rock.jpg">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="icon" type="image/svg" href="http://example.com/image.svg">
</head>
```

Reserved characters in HTML

If you use the less than (<) or greater than (>) signs in your text, the browser might mix them with tags.

Character entities are used to display reserved characters in HTML.

Result	Description	Entity Name
	non-breaking space	
<	less than	<
>	greater than	>
&	ampersand	&
"	double quotation mark	"
'	single quotation mark (apostrophe)	'

Lighthouse demo

<https://pagespeed.web.dev>

Documentation you should use

[Mozilla developer docs](#) - gold mine for HTML and JavaScript related information

[React.js docs](#) - Docs for React - we will talk more about it throughout the semester

[Google's web development page](#) - Another great resource. Use in conjunction with Lighthouse to see whether your page is up to standard.

Inspect demo - optional

CSS seminar will use this technique more

The image shows a browser's developer tools interface. The top toolbar includes icons for Inspector, Console, Debugger, Network, Style Editor, Performance, Memory, Storage, Accessibility, and Application. The 'Inspector' tab is active, showing a search bar and a 'Filter Styles' dropdown. The HTML tree on the left shows a document structure starting with `<!DOCTYPE html>` and `<html class="taiwrwljcy idc0_337" itemscope="" itemtype="http://schema.org/WebPage" lang="en">`. The `<body>` contains a `<div id="root">` which includes a `<div id="wrapper">` with a `<header id="header">`. The selected element is `<header id="header">`, which contains `<h1 class="index-link">`, `<nav class="links">`, `<div class="language-select">`, and `<div class="hamburger-container">`. The `<div id="main">` contains an `<article id="index" class="post">` with a `<header>` and three `<p>` elements, and a `<section id="sidebar">`. The CSS styles on the right show the default `display: inline` for the element, and a custom style from `_post.scss:9` that sets `display: flex` and `width: calc(100% + 6em);`. Other styles from `_skel.scss:538` are also visible, including `display: block` for various elements.

Demo - create markup for this simple form

Don't forget to use correct **semantic elements**! We advise you to open the slides on your PC to not have to rely on the projected image from your tutor. **No styling is required! (yet)**

The form is titled "Demo - create markup for this simple form" and contains several input fields and buttons. At the top, there are two tabs: "Koľko chcem vložiť" (selected) and "Koľko chcem získať". Below the tabs are three sliders and two button groups. The first slider is for "Pravidelný mesačný vklad" (Regular monthly deposit) with a value of 100 €, ranging from 0 € to 5 000 €. The second slider is for "Jednorazový počiatočný vklad" (One-time initial deposit) with a value of 1 000 €, ranging from 0 € to 250 000 €. The third slider is for "Doba trvania investície" (Investment duration) with a value of 8 rokov (years), ranging from 3 roky to 30 rokov. Below the sliders are two button groups. The first group is for "Zvoľte investičnú stratégiu" (Choose investment strategy) with options: Konzervatívna (selected), Vyvážená, Rastová, Dynamická, and Vlastná. The second group is for "Zvoľte scenár vývoja trhov" (Choose market development scenario) with options: Pesimistický, Neutrálny (selected), and Optimistický.

- Identify the elements
- Work your way from the whole form to individual elements making it up
- Split it into components -> think about the difference between a component and an HTML element

Demo - create markup for this simple form

Bonus:

- Add another field where the user can select multiple elements.
- Add a field where user can write their email into.
- [Advanced]: **Use flexbox/grid** to get the **layout** close to the one shown on this picture*

**Your tutor might show this to you as a sneak peek into layouting, which will be one of the two major parts of the next week's seminar (the other being styling).*

Questions?

Thank you for your attention!