Spis treści

[**1.** **RWD** 4](#_Toc508041967)

[1.1. max-device-width ogarnąć 5](#_Toc508041968)

[**2.** **CSS** 5](#_Toc508041969)

[2.1. BEM 5](#_Toc508041970)

[2.2. Kaskadowość 5](#_Toc508041971)

[2.3. Dziedziczenie 5](#_Toc508041972)

[2.4. Specyficzność 5](#_Toc508041973)

[**3.** **Bootstrap** 5](#_Toc508041974)

[3.1. kiedy i dlaczego używać 5](#_Toc508041975)

[**4.** **SASS** 5](#_Toc508041976)

[4.1. po co używać 5](#_Toc508041977)

[4.2. loader do reacta 5](#_Toc508041978)

[4.3. co ma ponad css 5](#_Toc508041979)

[4.4. tworzenie plików css preprocesorem 5](#_Toc508041980)

[4.5. zobaczyć LESS – co to i podstawowe różnice do SASS 5](#_Toc508041981)

[4.6. SCSS 5](#_Toc508041982)

[**5.** **GIT i jego funkcje** 6](#_Toc508041983)

[5.1. Mergowanie 6](#_Toc508041984)

[5.2. Branchowanie 6](#_Toc508041985)

[5.3. pull requesty 6](#_Toc508041986)

[5.4. cofanie się do commitów 6](#_Toc508041987)

[5.5. Stash 6](#_Toc508041988)

[5.6. Konflikty - skąd się biorą i jak je rozwiązywać 6](#_Toc508041989)

[5.7. poprosić Ozza 6](#_Toc508041990)

[5.8. git flow 6](#_Toc508041991)

[**6.** **Npm, a yarn** 6](#_Toc508041992)

[6.1. popytać Ozza 6](#_Toc508041993)

[**7.** **JS** 6](#_Toc508041994)

[7.1. Ify 6](#_Toc508041995)

[7.2. Pętle 6](#_Toc508041996)

[7.3. Funkcje 6](#_Toc508041997)

[7.4. Prototypy 7](#_Toc508041998)

[7.5. operatory logiczne 7](#_Toc508041999)

[7.6. undefined vs null 7](#_Toc508042000)

[7.7. Asynchroniczność 7](#_Toc508042001)

[**8.** **JQuery** 7](#_Toc508042002)

[8.1. Funkcjonalności 7](#_Toc508042003)

[8.2. dlaczego wypadł z łask 7](#_Toc508042004)

[8.3. zalety na swój czas 7](#_Toc508042005)

[**9.** **OOP** 7](#_Toc508042006)

[9.1. dlaczego obiektowość 7](#_Toc508042007)

[9.2. implementacja w react 7](#_Toc508042008)

[**10.** **Ajax** 7](#_Toc508042009)

[10.1. ajax a redux 7](#_Toc508042010)

[10.2. po co używać 7](#_Toc508042011)

[10.3. zady i walety 7](#_Toc508042012)

[**11.** **API** 7](#_Toc508042013)

[11.1. api na stronach 8](#_Toc508042014)

[11.2. api do mongo w mern 8](#_Toc508042015)

[11.3. hash użytkownika API 8](#_Toc508042016)

[**12.** **Node** 8](#_Toc508042017)

[12.1. przypomnieć sobie jak postawić serwer na czystym node 8](#_Toc508042018)

[12.2. różnice z php 8](#_Toc508042019)

[12.3. back-end – serwer i zapytania do niego 8](#_Toc508042020)

[**13.** **ReactJS** 8](#_Toc508042021)

[13.1. szablony (PuG EJS) 8](#_Toc508042022)

[13.2. struktura danych 8](#_Toc508042023)

[13.3. cykl życia komponentu 8](#_Toc508042024)

[13.4. ReactDOM 8](#_Toc508042025)

[13.5. state vs props 8](#_Toc508042026)

[13.6. dlaczego React 8](#_Toc508042027)

[13.7. podstawowe różnice z AgularJS 8](#_Toc508042028)

[**14.** **ES6** 8](#_Toc508042029)

[14.1. co wprowadza 9](#_Toc508042030)

[14.2. jaki sposób kompiluje się do es2015 (babel) 9](#_Toc508042031)

[**15.** **Webpack** 9](#_Toc508042032)

[15.1. Ozz 9](#_Toc508042033)

[15.2. Pluginy 9](#_Toc508042034)

[15.3. Loadery 9](#_Toc508042035)

[15.4. HMR 9](#_Toc508042036)

[**16.** **ExpressJS** 9](#_Toc508042037)

[16.1. Middleware 9](#_Toc508042038)

[16.2. routing – endpointy 9](#_Toc508042039)

[16.3. CRUD 9](#_Toc508042040)

[16.4. JSON w JS 9](#_Toc508042041)

[16.5. autoryzacja 9](#_Toc508042042)

[16.6. serwowanie plików 9](#_Toc508042043)

[**17.** **Inne** 9](#_Toc508042044)

[17.1. Makiety 9](#_Toc508042045)

[17.2. SCRUM 10](#_Toc508042046)

[17.3. Polling 10](#_Toc508042047)

[17.4. Streaming 10](#_Toc508042048)

[17.5. Long-polling 10](#_Toc508042049)

[17.6. Socket.io 10](#_Toc508042050)

[**18.** **Redux** 10](#_Toc508042051)

[18.1. Flux 10](#_Toc508042052)

[18.2. Cqrs 10](#_Toc508042053)

[18.3. event sourcing 10](#_Toc508042054)

[18.4. zasady korzystania 10](#_Toc508042055)

[18.5. programowanie funkcyjne 10](#_Toc508042056)

[18.6. store w mern 10](#_Toc508042057)

[18.7. React-redux 10](#_Toc508042058)

[18.8. middleware redux 10](#_Toc508042059)

[18.9. data flow 10](#_Toc508042060)

[**19.** **React-router** 10](#_Toc508042061)

[19.1. co to jest i jak działa 11](#_Toc508042062)

[**20.** **MongoDB** 11](#_Toc508042063)

[20.1. nierelacyjne bazy danych 11](#_Toc508042064)

[20.2. SQL 11](#_Toc508042065)

[20.3. Mongoose 11](#_Toc508042066)

[20.4. operacje na bazie Mongo 11](#_Toc508042067)

[**21.** **MERN** 11](#_Toc508042068)

[21.1. inne wersje np. MEAN 11](#_Toc508042069)

[21.2. kompletnie inne technologie PERL 11](#_Toc508042070)

[21.3. aplikacje izomorficzne 11](#_Toc508042071)

[21.4. server side rendering 11](#_Toc508042072)

[21.5. wirtualizacja (VBox, vagrant, docker) – Ozz 11](#_Toc508042073)

[21.6. generator kodu 11](#_Toc508042074)

[21.7. Redux-thunk 11](#_Toc508042075)

[21.8. Fetch 11](#_Toc508042076)

[21.9. przejrzeć dokumentację 11](#_Toc508042077)

[**22.** **Pozostałe** 12](#_Toc508042078)

[22.1. REST API 12](#_Toc508042079)

[22.2. normalizacja danych 12](#_Toc508042080)

[22.3. ReactDnD 12](#_Toc508042081)

[22.4. compose i connect 12](#_Toc508042082)

[22.5. obadać punkty połączenia frontu z back-endem 12](#_Toc508042083)

[**23.** **Zasady** 12](#_Toc508042084)

[23.1. DRY 12](#_Toc508042085)

[23.2. KISS 12](#_Toc508042086)

[23.3. poziomy abstrakcji 12](#_Toc508042087)

[23.4. SOLID Principles 12](#_Toc508042088)

[**24.** **Wzorce projektowe** 12](#_Toc508042089)

[**25.** **Migracje** 12](#_Toc508042090)

[**26.** **Zagadnienia serwerowe** 12](#_Toc508042091)

[26.1. deployowanie, CI/CD 12](#_Toc508042092)

[26.2. Cache’owanie (memcache, redis, cloudflare) 12](#_Toc508042093)

[26.3. Domeny (DNS, Vhosty) 13](#_Toc508042094)

1. **RWD**

<https://www.w3schools.com/css/css_rwd_intro.asp>

* 1. max-device-width ogarnąć

DEPRECATED!!! Nie używać!

Rzeczywista ilość pikseli wyświetlacza (są np. wyświetlacze, w których na jeden piksel w CSS składają się 2 piksele).

1. **CSS**

<https://www.w3schools.com/css/default.asp>

* 1. BEM

<http://getbem.com/>

Block - Standalone entity that is meaningful on its own. – naming: .block

Element - A part of a block that has no standalone meaning and is semantically tied to its block. – naming: .block\_\_elem

Modifier - A flag on a block or element. Use them to change appearance or behavior. – naming: .block--mod, .block\_\_elem--mod

* 1. Kaskadowość

<https://developer.mozilla.org/en-US/docs/Web/CSS/Cascade>

Najpierw autor, potem użytkownik, potem przeglądarka. CSS Animations przed nimi. Przed tym wszystkim te z flagą !important user-agent, user, author.

* 1. Dziedziczenie

Properties such as color, font-family are inherited, and border, margin are not. Check if property is inherited by default when used. Can use inherit, initial and unset to set inheritance.

* 1. Specyficzność

<https://the-awwwesomes.gitbooks.io/html-css-step-by-step/content/pl/css-basics/css-specificity/index.html>

Selektory: najpierw #identyfikator, potem .klasa/pseudoklasa, potem element/pseudoelement. Im dłuższy (bardziej dokładny selektor) tym wyżej w hierarchii.

1. **Bootstrap**
   1. kiedy i dlaczego używać
2. **SASS**
   1. po co używać
   2. loader do reacta
   3. co ma ponad css
   4. tworzenie plików css preprocesorem
   5. zobaczyć LESS – co to i podstawowe różnice do SASS
   6. SCSS
3. **GIT i jego funkcje**
   1. Mergowanie
   2. Branchowanie
   3. pull requesty
   4. cofanie się do commitów
   5. Stash
   6. Konflikty - skąd się biorą i jak je rozwiązywać
   7. poprosić Ozza
   8. git flow
4. **Npm, a yarn**
   1. popytać Ozza
5. **JS**
   1. Ify
   2. Pętle
   3. Funkcje
   4. Prototypy
   5. operatory logiczne
   6. undefined vs null
   7. Asynchroniczność
6. **JQuery**
   1. Funkcjonalności
   2. dlaczego wypadł z łask
   3. zalety na swój czas
7. **OOP**
   1. dlaczego obiektowość
   2. implementacja w react
8. **Ajax**
   1. ajax a redux
   2. po co używać
   3. zady i walety
9. **API**
   1. api na stronach
   2. api do mongo w mern
   3. hash użytkownika API
10. **Node**
    1. przypomnieć sobie jak postawić serwer na czystym node
    2. różnice z php
    3. back-end – serwer i zapytania do niego
11. **ReactJS**
    1. szablony (PuG EJS)
    2. struktura danych
    3. cykl życia komponentu
    4. ReactDOM
    5. state vs props
    6. dlaczego React
    7. podstawowe różnice z AgularJS
12. **ES6**
    1. co wprowadza
    2. jaki sposób kompiluje się do es2015 (babel)
13. **Webpack**
    1. Ozz
    2. Pluginy
    3. Loadery
    4. HMR
14. **ExpressJS**
    1. Middleware
    2. routing – endpointy
    3. CRUD
    4. JSON w JS
    5. autoryzacja
    6. serwowanie plików
15. **Inne**
    1. Makiety
    2. SCRUM
    3. Polling
    4. Streaming
    5. Long-polling
    6. Socket.io
16. **Redux**
    1. Flux
    2. Cqrs
    3. event sourcing
    4. zasady korzystania
    5. programowanie funkcyjne
    6. store w mern
    7. React-redux
    8. middleware redux
    9. data flow
17. **React-router**
    1. co to jest i jak działa
18. **MongoDB**
    1. nierelacyjne bazy danych
    2. SQL
    3. Mongoose
    4. operacje na bazie Mongo
19. **MERN**
    1. inne wersje np. MEAN
    2. kompletnie inne technologie PERL
    3. aplikacje izomorficzne
    4. server side rendering
    5. wirtualizacja (VBox, vagrant, docker) – Ozz
    6. generator kodu
    7. Redux-thunk
    8. Fetch
    9. przejrzeć dokumentację
20. **Pozostałe**
    1. REST API
    2. normalizacja danych
    3. ReactDnD
    4. compose i connect
    5. obadać punkty połączenia frontu z back-endem
21. **Zasady**
    1. DRY
    2. KISS
    3. poziomy abstrakcji
    4. SOLID Principles
22. **Wzorce projektowe**
23. **Migracje**
24. **Zagadnienia serwerowe**
    1. deployowanie, CI/CD
    2. Cache’owanie (memcache, redis, cloudflare)
    3. Domeny (DNS, Vhosty)