1. Komponenty

a) tworzony przez ciebie komponent musi zaczynać się z wielkiej litery i zwracać HTML container, domyślne kontenery (podstawowe) zaczynają się z małych liter

b) komponent to funkcja w jsx (js)

c) Shift + Alt + F – formatowanie kodu

d) używając komponentów wrapujemy je w <>, np. używamy funkcji Header

e) można szybko dać <Header />

f) rozszerzenie pliku to .jsx (mogą też być .js)

g) renderowanie componentu

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

ReactDOOM tworzy korzeń na podstawie pobranego elementu i wstrzykuje do niego nasz komponent

2. Zmienne dynamiczne

a) wstawiasz zmienne opatulając {zmienna}, może to być wynik funkcji. Nie może to być if, loop, tylko wartość

b) można importować w następujący sposób



Potem wstawiasz w klamry {}

3. Rekwizyty (props) (Wstawianie danych do komponentu)

a) tworzysz komponent, który przyjmie obiekt

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

b) wywołując komponent przekazujesz parametry

Obraz zawierający tekst, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

c) Możemy też zaimportować inny komponent (rozszerzenie .js)



Opatulamy w {} bo nie jest default export

d) operator rozprzestrzeniania

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

To samo co



e) można też

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

4. Dekompozycja komponentów

a) komponenty zazwyczaj tworzymy w folderze o nazwie components

b) nazwa pliku to zazwyczaj nazwa komponentu

c) musimy użyć: export default (lub export wtedy importując dokładamy {})

d) css także dekomponujemy. Dodajemy pliki css do folderu components

importujemy w potrzebnym komponencie



e) przy dekompozycji css, będzie on i tak stosowany do każdego elementu, wystarczy, że go raz zaimportujemy i będzie działał na każdym komponencie

f) przekazywanie danych. Tworząc komponent z parametrami props, zawsze otrzymasz parametr children



Chcąc wykorzystać Components musisz zrobić coś takiego (może to być i jsx, tzn. divy, headery itd.)

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Takie coś nazywamy kompozycją komponentów

5) Obsługa przycisków

a) Ctrl + Space – podpowiedzi

b) dodajemy Onclick parametr, przyjujący funkcję, zazwyczaj robimy to wewnątrz komponentu

c) nazywamy od handle lub kończymy (taka konwencja) i dodajemy nazwę wydarzenia

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

d) jeśli coś ma się stać po wykonaniu danego zdarzenia, to nazwę parametru zaczynamy od on (taka konwencja)

e) przekazywanie funkcji jako parametr do przycisku:

i) określasz funkcję w komponencie

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

ii) przekazujesz jako parametr do innego komponent 

iii) w niestandardowym komponencie przekazujesz zmienną do natywnego komponentu

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

f) chcąc przekazać parametr do funkcji onClick robimy sztuczkę (niestandardowy parametr):

i) Dodajemy parametr do funkcji obługującej wydarzenie

Obraz zawierający tekst, Czcionka, zrzut ekranu, Grafika

Opis wygenerowany automatycznie

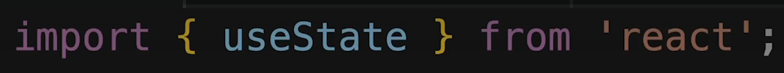
ii) tworzymy funkcję, wykonującą inną funkcję, przekazując parametr



6) Aktualizacja komponentów

a) Komponenty są renderowane tylko na samym początku, więc jeśli chcemy coś zmienić w komponencie, musimy zmienić stan

b) musimy zaimportować ReactHook (funkcje które zaczynają się od use



Mogą być one wywoływane tylko **wewnątrz komponentów reacta**, lub innych react hook, **na samym początku** (nie może być zagnieżdżone w innej funkcji)

c) używanie hook’a:

i) definiujemy

konwencja, useState przyjmuje wartość domyślną

selectedTopic – przechowuje zmienioną wartość

setSelectedTopic – funkcja aktualizująca zmienną, mówiąca reactowi o

aktualizacji komponentu

d) zastosowanie

i) dodajemy plik data.js

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

ii) importujemy



iii) odnosimy się do zmiennych ustawianych w event handler



Należy pamiętać o prawidłowej wartości początkowej

7) Zmienne warunkowe

a) ustawiamy useState() usuwając wartość początkową i dajemy {state ? :}

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Można połączyć w jedno 😊

b) użycie operatora and (&&)



c) użycie zmiennej trzymającą kod JSX

i) definicja zmiennej

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

ii) wyświetlanie



8) Dynamiczna stylizacja

a) podobnie jak z przekazaniem funkcji. Po prostu przekazujemy przy wywołaniu parametr odpowiadający za przekazanie klasy:

i) modyfikujemy klasę przyjmującąObraz zawierający zrzut ekranu, tekst, Oprogramowanie multimedialne, oprogramowanie

Opis wygenerowany automatycznie

ii) przekazujemy wartość dynamicznie na podstawie stanu

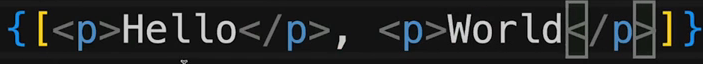


9) Dynamiczne wyświetlanie list

a) tablicę danych można wyświetlić dynamiczne

Obraz zawierający Czcionka, zrzut ekranu, linia, typografia

Opis wygenerowany automatycznie



b) ideą jest mapowanie obiektu na element typu HTML

i) mamy dane

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

ii) musimy je zmapować

Obraz zawierający tekst, Czcionka, zrzut ekranu, Grafika

Opis wygenerowany automatycznie

Jest to równoważne wszystkim powtórzeniom tego



iii) żeby pozbyć się warningu ustawiamy key

