

React — komponenty i stan

Czas: 2 × 45 min

praca indywidualna

Cel: Zbudowanie pierwszej aplikacji w React: podział interfejsu na komponenty, przekazywanie danych przez props, stan `useState`, renderowanie list z kluczem i formularz kontrolowany.

Po zajęciach potrafisz:

- tworzy projekt React w Vite i uruchamia serwer deweloperski
- dzieli interfejs na komponenty funkcyjne i przekazuje im dane oraz funkcje przez props
- przechowuje dane w stanie `useState` i zmienia go niemutowalnie (nowa tablica / nowy obiekt)
- renderuje listy przez `map()` z unikalnym `key` i stosuje renderowanie warunkowe
- obsługuje formularz kontrolowany z walidacją

Sprzęt i materiały

- komputer z Windows 11, Node.js LTS, dostęp do internetu (npm)
- plik `r2_22_index.css` z Pliki_do_zadan

Oprogramowanie

- **Node.js LTS** — Vite 8 wymaga Node.js 20.19+ lub 22.12+; sprawdź `node -v`
nodejs.org
- **Vite — dokumentacja**
vite.dev/guide
- **React — dokumentacja** — rozdziały Describing the UI i Adding Interactivity
react.dev/learn
- **Visual Studio Code + React Developer Tools** — rozszerzenie przeglądarki do podglądu komponentów i stanu
react.dev/learn/react-developer-tools

Przygotowanie przed zajęciami

- Przejrzyj wykład 2.15 (JSX, props, `useState`) i powtórz z wykładu 2.01 funkcje strzałkowe, destrukuryzację, spread, `map` i `filter`.

Bezpieczeństwo

- Instaluj pakiety tylko poleceniami z instrukcji; po instalacji sprawdź komunikat `npm audit` wyświetlany przez npm.
- Folder `node_modules` nie trafia do archiwum ani repozytorium (jest w `.gitignore`).

Przebieg zadania

1 Projekt Vite + React

```
npm create vite@latest planer -- --template react
cd planer
npm install
npm run dev
```

1. Utwórz projekt poleceniami z ramki (jeśli pojawią się pytania, wybierz React i wariant JavaScript). Otwórz adres podany przez Vite (domyślnie `http://localhost:5173`). Zanotuj wersje React i Vite z `package.json`.
2. Opisz rolę plików: `index.html`, `src/main.jsx`, `src/App.jsx`, `package.json`, `vite.config.js`. Zmień tekst w `App.jsx` i zaobserwuj odświeżenie bez przeładowania strony (HMR).
3. Wyczyść szablon: usuń zawartość `App.jsx` i plik `App.css`, zastąp `src/index.css` plikiem `r2_22_index.css`, ustaw w `index.html` `lang="pl"` i tytuł „Planer ucznia”. Komponenty twórz w folderze `src/components`.

2 Komponenty i props

Każdy komponent to funkcja z wielkiej litery w osobnym pliku `.jsx` z `export default`.

1. Komponent `Naglowek` przyjmuje props `tytuł` i `podtytuł` (wartość domyślna „React + Vite” w destrukuryzacji) i renderuje `<header className="naglowek">` z `h1` i akapitem. Użyj go w `App`.
2. Komponent `Licznik` przyjmuje `etykieta`, `start` (0), `krok` (1), `max` (10) i ma stan `wartosc`. Przyciski: `-krok`, `+krok` (wartość nie spada poniżej 0 ani nie przekracza `max`; przycisk wyłączony `disabled` na granicy), „Zeruj”. Po osiągnięciu maksimum wyświetl komunikat (renderowanie warunkowe `&&`).
3. Umieść w `App` dwa liczniki z różnymi props: „Szkłanki wody” (`max` 8) i „Minuty nauki” (`krok` 15, `max` 120). Sprawdź, że każdy ma własny stan. W React Developer Tools obejrzyj props i stan liczników.

3 Lista zadań

Stan listy trzymaj w komponencie `ListaZadan`; zadanie to obiekt `{ id, tresc, priorytet, zrobione }`. Zaczynasz od 3 przykładowych zadań.

```
setZadania((z) => [...z, nowe]); // dodanie
setZadania((z) => z.map((x) => x.id === id ? { ...x, zrobione: !x.zrobione } : x));
setZadania((z) => z.filter((x) => x.id !== id)); // usunięcie
```

1. Komponent `Zadanie` (props: `zadanie`, `onPrzelacz`, `onUsun`) renderuje `<li>` z polem wyboru, treścią, priorytetem i przyciskiem usuwania. Klasy CSS: priorytet (`niski` / `normalny` / `wysoki`) oraz `zrobione`.
2. W `ListaZadan` wyrenderuj zadania `map()` z `key={z.id}`. Napisz funkcje `przelacz` i `usun` według ramki i przekaż je do `Zadanie`. Pod listą: „Do zrobienia: X z Y”.
3. Dodaj trzy przyciski filtra (wszystkie, do zrobienia, zrobione) — stan `filtr`, aktywny przycisk z klasą `aktywny`. Gdy lista po filtrze jest pusta, wyświetl „Brak zadań w tym widoku”. Sprawdź w konsoli F12, co się stanie, gdy na chwilę usuniesz `key` (zanotuj ostrzeżenie).

4 Formularz kontrolowany

1. Komponent `FormularzZadania` (prop `onDodaj`) ma stan `tresc`, `priorytet`, `bład`. Pola mają `value` i `onChange` (formularz kontrolowany); pod polem licznik znaków „N/80”, przycisk „Dodaj” wyłączony, gdy pole jest puste.
2. Po wysłaniu (`onSubmit`, `preventDefault`): treść krótsza niż 3 znaki → komunikat błędu; poprawna → `onDodaj(tresc, priorytet)`, wyczyszczenie pól i błędu. Nowe zadanie dostaje `id` z `crypto.randomUUID()`.
3. Wykonaj `npm run build` i obejrzyj folder `dist`. Zrób zrzuty: aplikacja z licznikiem na maksimum, błąd formularza, lista po filtrze, React Developer Tools. Spakuj projekt **bez** `node_modules` i `dist` jako `nazwisko_imie_zad22.zip` — użyjesz go w zadaniu 23.

W sprawozdaniu umieść

- ☐ tabela wykonanych poleceń (krok.punkt) z fragmentami kodu
- ☐ opis plików projektu Vite i wersje React/Vite
- ☐ drzewo komponentów aplikacji z zaznaczeniem, gdzie jest stan, a co przechodzi przez props
- ☐ zrzuty: aplikacja, błąd formularza, filtr, React Developer Tools, ostrzeżenie o braku `key`
- ☐ archiwum `nazwisko_imie_zad22.zip` i odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Czym różnią się props od stanu? Kto może je zmieniać?
2. Dlaczego nie wolno zmieniać stanu przez `zadania.push(...)` i po co forma `setZadania((z) => ...)`?
3. Do czego React używa atrybutu `key` i dlaczego indeks tablicy jest złym kluczem dla listy z usuwaniem?
4. Co oznacza „formularz kontrolowany” i jakie daje korzyści w porównaniu z odczytem pól w zdarzeniu submit?
5. Porównaj dodawanie elementu listy w React i w czystym JS z DOM (zadanie 3).

Kryteria oceny

Kryterium	Punkty
Projekt Vite uruchomiony, opis plików, wyczyszczony szablon	2
Komponenty <code>Naglowek</code> i <code>Licznik</code> z props i wartościami domyślnymi; dwa niezależne liczniki z limitami	5
Lista zadań: <code>map</code> z <code>key</code> , przełączanie, usuwanie bez mutacji, podsumowanie, filtr	6
Formularz kontrolowany z licznikiem znaków, walidacją i czyszczeniem	4
Build, drzewo komponentów, zrzuty, archiwum, pytania	3
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).