React

- 든든한 facebook
- 큰 커뮤니티

→ so good!

뭐지? react

- 반응형 프로그래밍 (reactive programming)
- 의존된 것들이 같이 변한다.
  → like 엑셀
- MVC
- 컴포넌트 (메세지 주고받기 ok, 계층적, 조합 ok)
- 가상 DOM → 코드로 DOM트리를 조현한 것.

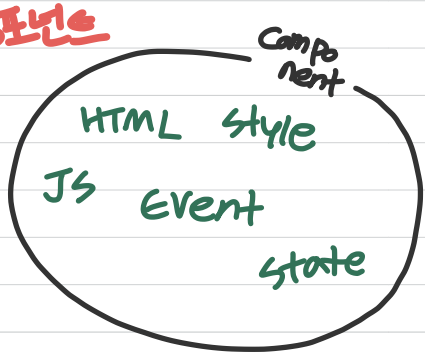
어떻게 공부?

- 깊게 쓰는것? → nope!
  (fork 딥고 빠름!!!!!)
- 최소노력, 최대효율
- 재사용성? 확장? → 나중에!

CRA . JSX

- Create React app
  → 빠르게 시작할 수 있다.
  - npx
  - npm
  - yarn
- /public/index.html
  이 root 역할 그리고 src 애들이 컴포넌트로 붙는다.
  index.js가 시작점. App.js가 root component. JSX는 가상 DOM을 의미한다.
  ↳ 안쓰는것도 가능!!
  컴포넌트의 최상위 컴포넌트는 1개
  → 변경을 더 잘보기 위해!!
  (감싸기 아 <> 쓰기)
  JSX 내 보간 ({ }) 가능
  반복 & 조건도 ok.
  ↳ map ↳ and / 삼항연산자

컴포넌트



그리고 상위 → 하위의 계층 &
위→아래로 흐른다.
재사용성? UI를 추상적으로! (???)
분류? ① 처음엔 걍 해! 갓갓!
② 엉 아 기존이 생길때
함수로 만들어보기

Props

Props 사용 → function(props)

→ tag 에서 <Logo size={180}> → string이 아닌경우

default value는 Logo.defaultProps = { size: 200 }

↳ 타입제한도 가능하다.

(prop-types import 사)

Logo.propTypes = { size: propTypes.number }

↳ 다른 Type은 타이핑 { size = ... } 도 가능

children props

하위컴포넌트의 내용을 넣는다?

children의 type은 Node

& isRequired도 가능

상태관리 w. hook

1) how to 지역상태관리?
2) 이벤트 바인딩
3) 부모에게 메세지 보내기

useEffect

변화가 있을 때 감지하여 반응하는 hook!

useEffect(func, var);

↳ 변화시 반응 ↳ 감시

만약 var => [] 라면 컴포넌트 로드 시 func 호출 (이벤트사이클처럼 사용가능)

↳ 초기이벤트나 fetch등 good!

또는 전역이벤트등록!

↳ 제거될 때 (return) 이벤트 해제해주어한다:)

useRefs

1) DOM 직접접근
2) 지역변수 사용할때

→ 값이 변경되어도 렌더링하지않는다. (useState는 렌더링한다.)

직접 말고도 컴포넌트 대고 DOM에 접근할 수 있다. ref...? 연결?

forwarding Ref

style

1) style 시트 쓰기

→ Tmp아 " "

2) Inline style
3) emotion Lib 쓰기

↳ CSS함수? styled?

useMemo for 최적화

렌더링 한다? → 함수가 호출된다.
즉, 변수, 함수가 다시 선언되거나 호출된다는 것 → re-rendering
(부모 props 변경, 나 변경, 부모 변경의 경우 다시 렌더링된다.)
→ use memo로 리렌더링을 막는다.

ex) const result = useMemo(() => sumCn(n), [n]);
→ n변경 시에만 sumCn() 호출

React.memo

부모변경시 리렌더링을 막는다.
→ 컴포넌트의 화살표 함수? 일반함수? 차이점이 무엇?????

useCallback

리렌더링? → 함수 재호출 → 즉 함수 재정의를 의미한다.
→ 왜 렌더링이 다시 되는가…?

Custom Hook

Story book

컴포넌트 관리를 쉽게 하게 해준다

form

Storybook, Styled