1. 발표를 시작하겠습니다.

2. 안녕하세요 개발자 박원정입니다.

제 이름 앞에 개발자라는 단어가 붙는 것이 어색하네요.

항상 그 단어의 무게를 느끼고 제대로 된 개발자가 되려면 항상 공부하고 새로운 기술이나 지식을 받아 드리자는 마음이 들게 하는 단어인 것 같습니다.

2. 저는 17년 2월 경영학부를 졸업하고 19년 10월부터 Workplace Learning이라고 하여 기업에서 프로젝트를 진행하면서 쌓이는 자료들을 정보화해 기업교육에 활용하는 웹 기반 프로젝트 관리 툴을 개발하는 3개월차 개발자 박원정입니다.

저는 인문계에서 경상계열 학교를 졸업한 전형적인 문과생입니다. 졸업을 하고 입사까지 저에게 무슨 일이 일어났을까요?

3. 많은 이유와 계기가 있었지만 제가 개발자가 되겠다 생각하게 된 한마디가 있습니다.

졸업 후 미국대학원 입시 관련 스타트업의 초기멤버로 일을 했습니다. 그 때 당시만 해도 개발을

모르던 제가 사이트 런칭을 앞두고 당시 재직 중이던 개발자 분과 회의 중에

저의 아이디어를 사이트에 반영할 수 있을까요? 이런 기능이 실현 가능할까요? 라고 질문했을 때

그때 그 개발자 분께서 이렇게 말씀하셨습니다. ‘상상하는 무엇이든 다 만들 수 있어요.’

한동안 그 말이 머리에 남았습니다. 한번도 어느 사이트에 들어가서 어떤 버튼을 누르면

왜 페이지가 넘어가는지, 나의 정보가 나오는지 생각해본 적 없던 저에게

상상하는 모든 것을 만들 수 있다고? 그럼 해봐야 겠다라는 생각이 들었고

이 자리까지 오게 되었습니다.

그러나 처음부터 쉽지는 않았습니다.

여기있는 분들이 겪었고 저 또한 참 힘들고 막막했습니다.

주입식교육시대에 태어나 자란 만큼 개발 입문 강의로 유명한 인강도 다 들어봤고,

정부지원 오프라인 교육도 들어봤습니다.

사실 멍하니 흰색은 바탕이고 검은색은 글씨구나..하고 보낸 시간이었습니다.

가장 답답한 것이 내가 무엇을 모르는지 모르는 것. 어떤 검색어로 검색을 해야

내가 원하는 답을 찾을 수 있는지 모르는 것이었습니다. 공감하시나요?

4. 그 때 좋은 기회로 멘토님을 만나게 되었고 눈으로 보기만 하는 개발이 아니라

손가락을 움직이는 개발을 하게 되었습니다.

5. 바로 종필교의 시작이죠. 제가 어려운 에러에 빠져 있을 때 멘토링시간 멘토님이 해결방향을 제시해주시는 순간 프로필사진을 보고 마치 ‘내가 너희들을 개발 길로 인도하리라’하는 교주나 신 같다고 말씀드린 적이 있습니다. 그 뒤로 종교라고 생각하고 믿고 있습니다.

6. 처음 저의 개인 프로젝트는 제가 헬스장을 다니면서 느낀 니즈를 반영해서

Personal Trainer와 Trainee의 중개플랫폼 HowTo입니다.

7. 트레이너가 PT 가능 날짜를 여러 개 등록할 수 있도록 Multipledatepicker을 사용했고

8. 트레이니 입장에서보면 트레이너의 인적사항과 PT가능 날짜, 그리고 트레이너가 올린 운동 영상을 구분된 리스트로 살펴보고 컨택할 수 있게 구성하였습니다.

9.언어는 Javascript로 만들었고 Express기반 Rest API를 적용하여 node로 back단을 구성하였습니다.

10. 그리고 Front 단은 바로 React를 활용했는데요.

11. 혹시 리액트에 대해 아시는 분 계시나요? 🡪

12. 리액트는 많은 장점을 가진 아주 매력적인 라이브러리입니다.

간단 설명) 화면에 보이는 View에 집중하여 Frontend에서 UI만드는데 특화된 라이브러리입니다.

마치 포스트잇이 붙어있는 게시판과 같이 UI가 컴포넌트라는 요소로 구성되어 있어 재사용이 쉽고 코드파악이 쉬워 유지보수가 용이합니다.

또한 변경사항이 있는 경우 DOM에 직접 업데이트하지 않고 가상DOM에 그려본 뒤 변경된 부분만 반영하는 것으로 잦은 상태 변화나 업데이트가 있는 경우 특히 더 효율적인 구조입니다.

페이스북에서 개발한 라이브러리 답게 복잡한 앱에서도 데이터 흐름에서 일어나는 변화가 쉽게 예측 가능하도록 상위 컴포넌트에서 하위 컴포넌트로만 데이터를 전달하는 단방향 데이터 흐름을 가지고 있습니다. 그 것을 부모 컴포넌트에서 자식 컴포넌트로 데이터를 전달한다고도 하는데

하나 또는 여러 개의 컴포넌트가 하나의 컴포넌트에 속해 있다면 이것을 부모, 자식이라고~

제가 React를 선택한 이유도 여기에 있습니다.

리액트만의 특징은 아니지만 손쉽게 무료로 Virtual Dom을 활용해 효율을 높인다고 하니 속도개선에 좋다고 생각하였고,

또한 JSX라고 하여 자바스크립트와 HTML의 문법을 활용하여 코드를 작성 가능하기에 부담이 적었습니다.

단방향 데이터 흐름 역시 양방향보다 더 컨트롤하기 쉬울 것이라고 판단해서 리액트를 선택하였지만 React의 진입장벽은 상태관리가 상당부분 차지 한다는 사실을 잠시 간과했었네요.

13. 그래서 그 부분에 대해서 이야기 해보려고 합니다.

14. 리액트의 상태관리에 관하여 간단하게 알아보겠습니다.

사람들이 리액트를 어려워하는 이유 중에 하나가 바로 상태관리라고 하죠.

그래서 많은 사람들이 상태관리 라이브러리를 사용합니다.

각각의 원을 컴포넌트로 생각해서 간단한 구조의 프로젝트가 있다고 생각해보겠습니다.

15. 그리고 예를 들어 보편적인 상태관리 라이브러리 Redux의 개념을 잠깐 살펴보겠습니다.

16. Redux는 외부에 Store라 불리는 중간 매개체를 만들어 컴포넌트로 이루어진 리액트 프로젝트에서

17. 컴포넌트를 통한 데이터흐름이 아닌 스토어를 통해서 데이터 주고받는 것을 적용하는 것

아주 간단해보이지만 매우 복잡하고 까다로울 수 있습니다.

액션, 리듀서, 디스패치 등의 새로운 개념을 이해하고 적용하는 것이 어렵기 때문입니다.

Redux의 응용 강의가 아니기에 아주 정확한 개념이 아니더라도 흐름만 아주 빠르고 듣기 쉽게 말씀드리겠습니다. 부디 한 귀로 듣고 한 귀로 흘리시길 바랍니다. Redux를 배워야하는 분이라면... 그 분도 흘려 듣고 집에가서 다시 공부하시길 바랍니다.

여기서 느껴야하는 것은 리덕스와 같은 상태관리 도구를 왜 써야하는가? 왜 저렇게 써야하는가? 입니다.

18. 상태변화를 감지하기 위해 값이 필요한 컴포넌트는 스토어를 구독합니다.

19. 그 값을 변화시키는 컴포넌트는 값이 변화하면 어떻게 변화해야하는지 그 내용을 action이라 하고 그것을 store에 보내는 행위를 dispatcher가 dispatch한다고 합니다.

20. 그럼 스토어에 존재하는 리듀서가 이전 상태와 action을 합하여 새로운 상태를 만들죠.

21. 그렇다면 그 상태를 구독하고 있는 컴포넌트에 상태변화에 대한 알람을 보냅니다. 마치 여러분이 구독한 유튜버가 새로운 동영상을 올리면 알람이 가는 것 처럼요.

22. 그럼 알람을 받은 컴포넌트는 rerender라고 하는 과정을 실행하죠. 마치 새로고침과 같은 것입니다. 새로운 값을 반영하는 것이죠.

간단하죠? 하지만 이 과정들을 코드로 잘 짜셔야합니다. 상태의 수와 그 상태를 구독하는 컴포넌트의 수 만큼요.

제가 react를 처음 시작하면서 redux를 공부할 때 잠시 포기할까도 생각했습니다. 제게 더 어렵게 다가왔던 이유는 리액트의 데이터흐름과 상태관리과정을 쉽게, 편리하게 해주는 도구를 쓰려고하는데 복잡하고, 왜 이것을 써야하고 왜 액션을 디스패치하고 그것을 알려주고 새로고침을 해야하는지를 몰랐기 때문입니다.

23. 그럼 리덕스가 없으면 어떨까요?

리액트는 각각의 컴포넌트가 가질 수 있는 변경가능한 값과 자식컴포넌트에게 보내는 변경 불가능한 값인 props 위에서 아래로 내려주는 데이터 흐름만 가지고 있습니다.

24.만약 b에서 Root컴포넌트가 가진 값이 필요하면 어떻게할까요?

25. 방금 말한 것 처럼 값을 받아오면 됩니다.

반대로 Root에서 b의 값이 필요하다면 어떻게 될까요?

26. 값은 위로 보낼 수 없으니 자식컴포넌트가 부모가 가진 변경가능한 값을 변경하는 callback함수를 받아 부모의 변경가능 값을 자신의 값으로 변경합니다.

Props에 부모의 state를 변경하는 callback함수를 담아 내려보내면 됩니다.

B에서 R의 state값 변화시키는 함수를 실행하면 R의 값이 당연히 변하기 때문입니다.

27. 그렇다면 이런 경우는 어떨까요?

Root의 자식의 자식인 E가 root의 다른 자식인 b의 값이 필요하다면요?

28. 동공지진이 일어나시나요?

29. 생각보다 간단합니다. B의 state를 R의 state로 변경시키고 그 값을 e에게 내려주면 됩니다.

30. 하지만 프로젝트의 규모가 커진다면 매우 복잡한 상황이 펼쳐지겠죠.

값을 가진, 그 값을 원하는, 또 그 값을 변경하는 각각의 관계가 프로젝트 전반에 얽혀있다면.. 리액트는 대규모 프로젝트에는 사용하지 않게 될 것입니다.

기본 개념을 알고나니 상태관리 도구가 왜 필요한지 이해가 되셨나요?

31. 사실 Redux에서도 Redux, Redux-thunks, Redux-saga라고 해서 단순한 상태를 반환하는 redux

32.함수 실행 성과에 따라 다른 결과를 반환하거나 즉 action에 응답하는 redux-thunk

33. 더 나아가 yield를 기준으로 함수가 실행되거나 멈춰있거나 건너뛰는 제네레이터함수가 적용되어 action을 제어하는 redux-saga까지 설명드리려고 준비할까 했었는데

심리적으로 리액트와 리덕스에게 거리감을 느끼실까봐 넘어가도록하겠습니다.

사실 전 react를 처음 배우기 시작하면서 상태관리가 어렵다는 말을 익히 들었고 당연히 사람들이 많이 쓰는 redux도 공부해야겠다 싶어서 redux를 공부하다가 심리적 거리감을 많이 느꼈습니다.

사실 redux와 같은 상태관리 도구이지만 redux비해 상대적으로 복잡하지 않고 최적화 되어있는 Mobx라는 도구도 있습니다.

또한 더 이상 여러 컴포넌트를 거쳐서 값을 전달해주는 것이 아니라, Context 를 통해서 원하는 값이나 함수를 바로 쏴 줄 수 있는 Context API도 react 16.3버전부터 생겼습니다..

여러분이 만약 react를 배우는 입장이고 프로젝트를 처음 시작한다면 어떤 상태관리 라이브러리를 쓰시겠어요?

34.그 보다 상태관리 라이브러리를 사용하는 것은 필수적이라고 생각하시나요?

🡪 대답

저는 이렇게 생각합니다.

35. 잘 모르겠습니다. 진행하는 프로젝트의 규모와 데이터 흐름에 따라 또 개발자의 판단에 따라 달라진다고 생각합니다.

36. 제가 말씀드릴 수 있는 것은 결론적으로 멘토님의 조언으로 redux없이 howto를 만들었고  
redux를 공부하고 사용하는 지금 howto 개발 당시 redux를 사용하지 않은 것이 더 많은 도움이 되었다고 요새 많이 느낀다는 것입니다.  
redux를 사용하지 않고 복잡하고 힘들더라도 순수 react로 상태관리를 해봤기 때문에 지금 redux를 잘 이해하고 redux가 주는 편리함을 제대로 느낄 수 있는 것이라 생각합니다.  
꼭 redux뿐 아니라 여러분들이 개발을 하면서 저와 비슷한 상황이 된다면 꼭 쉬운 도구를 활용하기 전에 복잡하고 불편한 기본기를 겪어 보는 시간을 추천드리면서 발표를 마치겠습니다. 감사합니다.