**1쇄**

**1.3.2 (pg. 46)**

macOS에서 yarn 설치 방법이 바뀌었습니다. 명령어에 --without-node를 없애야 합니다.

$ brew update

- $ brew install yarn --without-node

+ $ brew instal yarn

$ yarn config set prefix ~/.yarn

$ echo 'export PATH="$(yarn global bin):$PATH"' >> ~/.bash\_profile

**3.4.1 (pg. 105)**

컴포넌트가 상속받고 있는데, "상속하는" 이라고 잘못 설명하여 이를 수정합니다.

이는 컴포넌트 생성자의 메서드인데요. 클래스형 컴포넌트에서 ...

+ 이 함수가 호출되면 현재 클래스형 컴포넌트가 상속하고 있는 리액트의...

- 이 함수가 호출되면 현재 클래스형 컴포넌트가 상속받고 있는 리액트의...

**4.2.2.1 (pg. 127)**

페이지 상단의 설명에 오탈자가 있습니다.

- SyntheticEvent 및 네이티브 이벤트와 달리 이벤트가 끝나고 나면...

+ SyntheticEvent는 네이티브 이벤트와 달리 이벤트가 끝나고 나면...

**8.2.3 (pg. 199) - 개선사항**

useEffect를 사용하는 부분의 deps 배열이 비어있어서 ESLint 경고가 나타납니다. 이 경고를 없애기 위하여 deps 배열에 name 을 넣어주세요.

useEffect(() => {

console.log('effect');

console.log(name);

return () => {

console.log('cleanup');

console.log(name);

};

- }, []);

+ }, [name]);

**9.2.2 (pg. 230, 231)**

sass-loader가 업데이트 됨에 따라 설정 방법이 바뀌었습니다. includePaths를 설정 할 때 sassOptions 객체 안에 넣어서 설정하셔야 합니다.

{

test: sassRegex,

exclude: sassModuleRegex,

use: getStylesLoaders({

importLoaders: 2,

sourceMap: isEnvProduction && shouldUseSourceMap

- }).concat({

- loader: require.resolve('sass-loader'),

- options: {

- includePaths: [paths.appSrc + '/styles'],

- sourceMap: isEnvProduction && shouldUseSourceMap }

- }),

+ }).concat({

+ loader: require.resolve('sass-loader'),

+ options: {

+ sassOptions: {

+ includePaths: [paths.appSrc + '/styles']

+ },

+ sourceMap: isEnvProduction && shouldUseSourceMap,

+ }

+ }),

sideEffects: true

}

추가적으로, data 가 prependData 로 변경됐습니다.

{

test: sassRegex,

exclude: sassModuleRegex,

use: getStylesLoaders({

importLoaders: 2,

sourceMap: isEnvProduction && shouldUseSourceMap

}).concat({

loader: require.resolve('sass-loader'),

- options: {

- includePaths: [paths.appSrc + '/styles'],

- sourceMap: isEnvProduction && shouldUseSourceMap,

- data: `@import 'utils';`

- }

+ options: {

+ sassOptions: {

+ includePaths: [paths.appSrc + '/styles']

+ },

+ sourceMap: isEnvProduction && shouldUseSourceMap,

+ prependData: `@import 'utils';`

+ }

}),

sideEffects: true

}

**11.6 (pg. 302)**

헷갈릴 수 있는 설명이 있어서 이를 수정합니다.

- 앞에서 useState를 사용해 만든 onToggle 함수를 다시 확인해볼까요?

+ 앞에서 useState를 사용해 만든 todos 배열과 setTodos 함수를 사용하는 onToggle 함수를 다시 확인해볼까요?

**12.1.5 (pg. 320)**

immer의 예시 코드에 오탈자가 있어 이를 수정합니다.

**수정 전:**

const update = draft => {

draft.value = 2;

};

**수정 후:**

const update = produce(draft => {

draft.value = 2;

});

**14.1.3 (pg. 359)**

예시 코드에서 increase가 increment로 잘못 들어가서 이를 모두 increase로 변경합니다.

async function runTasks() {

try {

// try/catch 구문을 사용하여 에러를 처리합니다.

let result = await increase(0);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

} catch (e) {

console.log(e);

}

}

**16.2.10 (pg. 426)**

페이지 하단 설명에 오탈자가 있습니다.

17.1 절이 아니라 16.1 절로 돌아가야합니다.

- 모두 제대로 작동한다면 17.1절로 돌아가서 다시 한번 개념을 훑어 보세요. 그리고 다음 내용을 계속 진행해주세요.

+ 모두 제대로 작동한다면 16.1절로 돌아가서 다시 한번 개념을 훑어 보세요. 그리고 다음 내용을 계속 진행해주세요.

**17.6.1.2 (pg. 460)**

REMOVE 액션 리듀서에서 action.payload를 사용해야 하는데 action.id를 사용하고 있습니다.

[REMOVE]: (state, action) => ({

...state,

+ todos: state.todo.filter(todo => todo.id !== action.payload)

- todos: state.todo.filter(todo => todo.id !== action.id)

})

**17.7.6 (pg. 470)**

파일명이 잘못되었습니다.

- containers/CounterContainer.js

+ containers/TodosContainer.js

**18.3.1.5 (pg.500)**

페이지 상단 connect 함수의 loadingPost 와 loadingUsers 부분에 오탈자가 있습니다.

export default connect(

({ sample, loading }) => ({

post: sample.post,

users: sample.users,

- loadingPost: loading.GET\_POST,

- loadingUsers: loading.GET\_USERS

+ loadingPost: loading['sample/GET\_POST'],

+ loadingUsers: loading['sample/GET\_USERS']

}),

{

getPost,

getUsers

}

)(SampleContainer);

**20.3.2 (pg. 549-550)**

css-loader가 업데이트 됨에 따라 기존 exportOnlyLocals 라는 옵션이 onlyLocals로 변경되었습니다.

CSS, CSS Module, Sass, Sass Module 총 4가지 부분에서 옵션 이름을 교체해주세요.

- exportOnlyLocals: true,

+ onlyLocals: true

**20.3.5 (pg. 560)**

manifest[key] 가 아닌 manifest.files[key] 로 바꿔주어야 합니다.

const chunks = Object.keys(manifest.files)

.filter(key => /chunk\.js$/.exec(key)) // chunks.js 로 끝나는 키를 찾아서

- .map(key => `<script src="${manifest[key]}"></script>) // 스크립트 태그로 변환하고

+ .map(key => `<script src="${manifest.files[key]}"></script>) // 스크립트 태그로 변환하고

.join(''); // 합침

그리고 그 하단의 main.css 와 main.js 도 마찬가지로 manifest[] 대신 manifest.files[] 를 사용하세요. runtime~main.js 의 경우도 마찬가지이며, 최근 CRA 업데이트로 인하여 기존 runtime~main.js 에서 runtime-main.js 로 파일 이름이 변경되었습니다.

- <link href="${manifest['main.css']}" rel="stylesheet" />

+ <link href="${manifest.files['main.css']}" rel="stylesheet" />

- <script src="${manifest['runtime~main.js']}"></script>

+ <script src="${manifest.files['runtime-main.js']}"></script>

- <script src="${manifest['main.js']}"></script>

+ <script src="${manifest.files['main.js']}"></script>

**20.5.2 (pg. 592)**

LoadablePlugin을 적용하는 부분에서 webpack.config.js에 plugins: 가 여러개 있어서 헷갈릴 수 있습니다. 따라서 webpack.config.js 의 내용을 다음과 같이 수정합니다.

const LoadablePlugin = require('@loadable/webpack-plugin');

(...)

plugins: [

new LoadablePlugin(),

// Generates an `index.html` file with the <script> injected.

new HTMLWebpackPlugin(

(...)

].filter(Boolean),

(...)

**22.2 (pg. 641) - 업데이트**

macOS에서 MongoDB 설치하는 명령어가 바뀌었습니다.

- $ brew update

- $ brew install mongodb

- $ brew services start mongodb

- ==> Successfully started `mongodb` (label: homebrew.mxcl.mongodb)\

+ $ brew tap mongodb/brew

+ $ brew install mongodb-community@4.2

+ $ brew services start mongodb-community@4.2

+ ==> Successfully started `mongodb-community` (label: homebrew.mxcl.mongodb-community)

**22.4 (pg. 646)**

ESModule 기능이 Node v12에서 정식 지원되지 않았는데, 잘못된 정보가 기재되었습니다. 따라서, pg 646의 **Node.js v12부터 ES Module 정식 지원** 참고 블록을 무시해주세요.

**22.9.2 (pg.669 - 672)**

기존 joi 모듈이 deprecated 되고, @hapi/joi 로 이름이 바뀌었습니다. 따라서 설치 방법도 바뀌고, 사용 방법도 조금 바뀌었습니다. 기존에 Joi.validate 가 schema.validate 로 바뀝니다.

설치 방법 변경:

- yarn add joi

+ yarn add @hapi/joi

**posts.ctrl.js** - import 하는 부분과 write 함수에서 validate 부분 코드 변경

- import Joi from 'joi';

+ import Joi from '@hapi/joi';

// 검증하고 나서 검증 실패인 경우 에러 처리

- const result = Joi.validate(ctx.request.body, schema);

+ const result = schema.validate(ctx.request.body);

**posts.ctrl.js** - update 부분도 변경

- const result = Joi.validate(ctx.request.body, schema);

+ const result = schema.validate(ctx.request.body);

**23.3.1 (pg.691)**

Joi 버전이 업데이트됨에 따라 코드도 업데이트 합니다.

+ import Joi from '@hapi/joi';

- import Joi from 'joi';

import User from '../../models/user';

+ const result = schema.validate(ctx.request.body);

- const result = Joi.validate(ctx.request.body, schema);

**23.4.2 (pg. 697)**

user.js 파일의 경로에 오탈자가 있습니다. modles 디렉터리가 아닌 models 디렉터리입니다.

- src/modles/user.js - generateToken

+ src/models/user.js - generateToken

**24.2.6 (pg. 784)**

비밀번호가 일치하지 않을 때 액션을 디스패치하는 부분에 오탈자가 있습니다. 액션 생성 함수를 호출 후 dispatch 해주어야합니다.

- changeField({ form: 'register', key: 'password', value: '' });

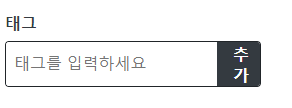
- changeField({ form: 'register', key: 'passwordConfirm', value: '' });

+ dispatch(changeField({ form: 'register', key: 'password', value: '' }));

+ dispatch(changeField({ form: 'register', key: 'passwordConfirm', value: '' }));

**25.2.1 (pg. 814)**

일부 환경에서 TagForm 이 다음과 같이 버튼이 찌그러지는 현상이 나타납니다.

[](https://user-images.githubusercontent.com/17894639/65041447-c039a780-d991-11e9-8bb7-b9a3419eed90.png)

해결 방안은 input 태그에 min-width: 0; 속성을 넣는 것 입니다.

input {

padding: 0.5rem;

flex: 1;

+ min-width: 0;

}

**26.2.2 (pg. 857 - 858)**

username을 match.params 에서 조회해야 하는데 query에서 조회를 하는 실수가 있었습니다. match.params 에서 불러오도록 수정합니다.

-const PostListContainer = ({ location }) => {

+const PostListContainer = ({ location, match }) => {

useEffect(() => {

+ const { username } = match.params;

+ const { tag, page } = qs.parse(location.search, {

- const { tag, username, page } = qs.parse(location.search, {

ignoreQueryPrefix: true,

});

dispatch(listPosts({ tag, username, page }));

}, [dispatch, location.search]);

**26.2.4 (pg. 869 - 870)**

username을 match.params 에서 조회해야 하는데 query에서 조회를 하는 실수가 있었습니다. match.params 에서 불러오도록 수정합니다.

+const PaginationContainer = ({ location, match }) => {

-const PaginationContainer = ({ location }) => {

+ const { username } = match.params;

// page가 없으면 1을 기본값으로 사용

+ const { tag, page = 1 } = qs.parse(location.search, {

- const { tag, username, page = 1 } = qs.parse(location.search, {

ignoreQueryPrefix: true,

});

**27.1.2 (pg. 880)**

ownPost 값을 설정하는 로직에 오탈자가 있습니다. 추가적으로 PostViewer 에서 ownPost 를 사용하는 코드가 빠졌습니다. 따라서, 페이지에 최소한의 수정을 하기 위하여 ownPost 라는 상수를 따로 선언하고 actionButtons props 를 설정하는 과정에서 조건부 렌더링을 하는 방식으로 수정됐습니다.

+ const ownPost = (user && user.\_id) === (post && post.user.\_id);

return (

<PostViewer

post={post}

loading={loading}

error={error}

- actionButtons={<PostActionButtons onEdit={onEdit} />}

+ actionButtons={ownPost && <PostActionButtons onEdit={onEdit} />}

- ownPost={user && user.id === post && post.id}

/>

);

**27.2 (pg. 892)**

pg.880과 같은 이유로 수정되었습니다.

+ const ownPost = (user && user.\_id) === (post && post.user.\_id);

return (

<PostViewer

post={post}

loading={loading}

error={error}

- actionButtons={<PostActionButtons onEdit={onEdit} />}

+ actionButtons={ownPost && <PostActionButtons onEdit={onEdit} onRemove={onRemove} />}

- ownPost={user && user.id === post && post.id}

/>

);

## 2쇄

## 3.4.1 (pg. 105) 개선

컴포넌트가 상속받고 있는데, "상속하는" 이라고 잘못 설명하여 이를 수정합니다.

이는 컴포넌트 생성자의 메서드인데요. 클래스형 컴포넌트에서 ...

+ 이 함수가 호출되면 현재 클래스형 컴포넌트가 상속하고 있는 리액트의...

- 이 함수가 호출되면 현재 클래스형 컴포넌트가 상속받고 있는 리액트의...

## 9.2.2 (pg. 230, 231) 업데이트

sass-loader가 업데이트 됨에 따라 설정 방법이 바뀌었습니다. includePaths를 설정 할 때 sassOptions 객체 안에 넣어서 설정하셔야 합니다.

{

test: sassRegex,

exclude: sassModuleRegex,

use: getStylesLoaders({

importLoaders: 2,

sourceMap: isEnvProduction && shouldUseSourceMap

- }).concat({

- loader: require.resolve('sass-loader'),

- options: {

- includePaths: [paths.appSrc + '/styles'],

- sourceMap: isEnvProduction && shouldUseSourceMap }

- }),

+ }).concat({

+ loader: require.resolve('sass-loader'),

+ options: {

+ sassOptions: {

+ includePaths: [paths.appSrc + '/styles']

+ },

+ sourceMap: isEnvProduction && shouldUseSourceMap,

+ }

+ }),

sideEffects: true

}

추가적으로, data 가 prependData 로 변경됐습니다.

{

test: sassRegex,

exclude: sassModuleRegex,

use: getStylesLoaders({

importLoaders: 2,

sourceMap: isEnvProduction && shouldUseSourceMap

}).concat({

loader: require.resolve('sass-loader'),

- options: {

- includePaths: [paths.appSrc + '/styles'],

- sourceMap: isEnvProduction && shouldUseSourceMap,

- data: `@import 'utils';`

- }

+ options: {

+ sassOptions: {

+ includePaths: [paths.appSrc + '/styles']

+ },

+ sourceMap: isEnvProduction && shouldUseSourceMap,

+ prependData: `@import 'utils';`

+ }

}),

sideEffects: true

}

## 11.6 (pg. 302) 개선

헷갈릴 수 있는 설명이 있어서 이를 수정합니다.

- 앞에서 useState를 사용해 만든 onToggle 함수를 다시 확인해볼까요?

+ 앞에서 useState를 사용해 만든 todos 배열과 setTodos 함수를 사용하는 onToggle 함수를 다시 확인해볼까요?

## 12.1.5 (pg. 320) 오탈자

immer의 예시 코드에 오탈자가 있어 이를 수정합니다.

**수정 전:**

const update = draft => {

draft.value = 2;

};

**수정 후:**

const update = produce(draft => {

draft.value = 2;

});

## 14.1.3 (pg. 359) 오탈자

예시 코드에서 increase가 increment로 잘못 들어가서 이를 모두 increase로 변경합니다.

async function runTasks() {

try {

// try/catch 구문을 사용하여 에러를 처리합니다.

let result = await increase(0);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

result = await increase(result);

console.log(result);

} catch (e) {

console.log(e);

}

}

## 17.6.1.2 (pg. 460) 오탈자

REMOVE 액션 리듀서에서 action.payload를 사용해야 하는데 action.id를 사용하고 있습니다.

[REMOVE]: (state, action) => ({

...state,

+ todos: state.todo.filter(todo => todo.id !== action.payload)

- todos: state.todo.filter(todo => todo.id !== action.id)

})

## 20.3.2 (pg. 549-550) 업데이트

css-loader가 업데이트 됨에 따라 기존 exportOnlyLocals 라는 옵션이 onlyLocals로 변경되었습니다.

CSS, CSS Module, Sass, Sass Module 총 4가지 부분에서 옵션 이름을 교체해주세요.

- exportOnlyLocals: true,

+ onlyLocals: true

## 20.5.2 (pg. 592) 개선

LoadablePlugin을 적용하는 부분에서 webpack.config.js에 plugins: 가 여러개 있어서 헷갈릴 수 있습니다. 따라서 webpack.config.js 의 내용을 다음과 같이 수정합니다.

const LoadablePlugin = require('@loadable/webpack-plugin');

(...)

plugins: [

new LoadablePlugin(),

// Generates an `index.html` file with the <script> injected.

new HTMLWebpackPlugin(

(...)

].filter(Boolean),

(...)

## 22.9.2 (pg.669 - 672) 업데이트

기존 joi 모듈이 deprecated 되고, @hapi/joi 로 이름이 바뀌었습니다. 따라서 설치 방법도 바뀌고, 사용 방법도 조금 바뀌었습니다. 기존에 Joi.validate 가 schema.validate 로 바뀝니다.

설치 방법 변경:

- yarn add joi

+ yarn add @hapi/joi

**posts.ctrl.js** - import 하는 부분과 write 함수에서 validate 부분 코드 변경

- import Joi from 'joi';

+ import Joi from '@hapi/joi';

// 검증하고 나서 검증 실패인 경우 에러 처리

- const result = Joi.validate(ctx.request.body, schema);

+ const result = schema.validate(ctx.request.body);

**posts.ctrl.js** - update 부분도 변경

- const result = Joi.validate(ctx.request.body, schema);

+ const result = schema.validate(ctx.request.body);

## 22.4 (pg. 646) 정정

ESModule 기능이 Node v12에서 정식 지원되지 않았는데, 잘못된 정보가 기재되었습니다. 따라서, pg 646의 **Node.js v12부터 ES Module 정식 지원** 참고 블록을 무시해주세요.

## 23.3.1 (pg.691) 업데이트

Joi 버전이 업데이트됨에 따라 코드도 업데이트 합니다.

+ import Joi from '@hapi/joi';

- import Joi from 'joi';

import User from '../../models/user';

+ const result = schema.validate(ctx.request.body);

- const result = Joi.validate(ctx.request.body, schema);

## 26.2.2 (pg. 857 - 858) 실수

username을 match.params 에서 조회해야 하는데 query에서 조회를 하는 실수가 있었습니다. match.params 에서 불러오도록 수정합니다.

- const PostListContainer = ({ location }) => {

+ const PostListContainer = ({ location, match }) => {

useEffect(() => {

+ const { username } = match.params;

+ const { tag, page } = qs.parse(location.search, {

- const { tag, username, page } = qs.parse(location.search, {

ignoreQueryPrefix: true,

});

dispatch(listPosts({ tag, username, page }));

}, [dispatch, location.search]);

## 26.2.4 (pg. 869 - 870) 실수

username을 match.params 에서 조회해야 하는데 query에서 조회를 하는 실수가 있었습니다. match.params 에서 불러오도록 수정합니다.

+const PaginationContainer = ({ location, match }) => {

-const PaginationContainer = ({ location }) => {

+ const { username } = match.params;

// page가 없으면 1을 기본값으로 사용

+ const { tag, page = 1 } = qs.parse(location.search, {

- const { tag, username, page = 1 } = qs.parse(location.search, {

ignoreQueryPrefix: true,

});