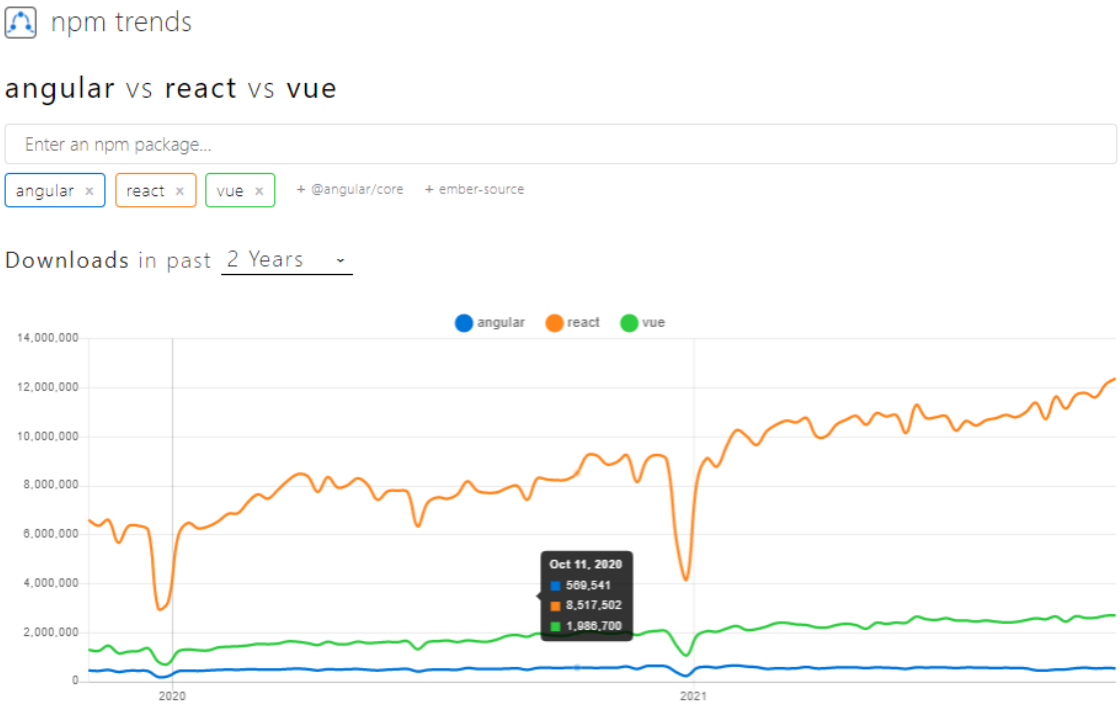
01장 리엑트 시작하기

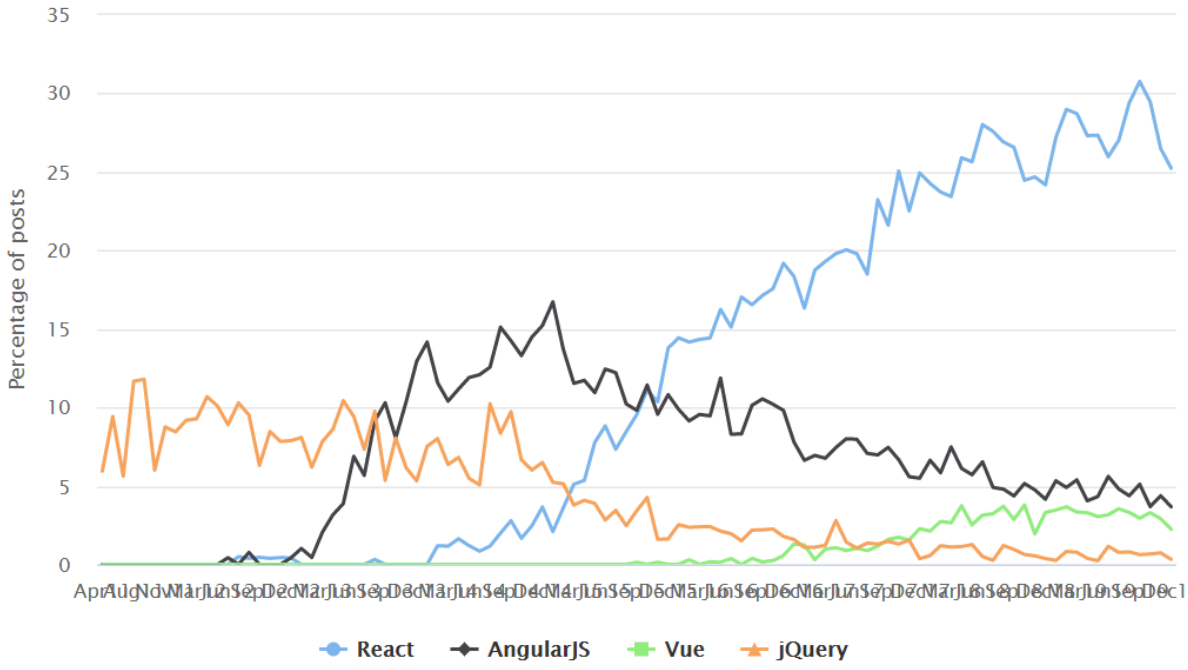
1.1 리액트의 정체를 알아보자!

1.1.1 어떤 프레임워크를 공부해야 할까?

- 리액트는 프런트앤드 프레임워크 중 하나이다. 현재 가장 주목받는 프레임워크는 react, angular, vue를 꼽을 수 있다.

- 프레임워크 다운로드와 개발자 수요 현황 그래프
  - https://www.npmtrends.com/angular-vs-react-vs-vue
  - https://bit.ly/2Rmx1Zn





1.1.2 리액트의 개념과 장점은 무엇일까?

- 리액트는 게이스북을 개발할 때 사용한 기술이며, 공개 소프트웨어이다. 리액트의 가장 큰 특징은 ‘화면 출력에 특화된 프레임워크’라는 것이다.
- 컴포넌트로 화면 구성을 효율적으로 할 수 있다.
- 게임 엔진 원리를 도입하여 화면 출력 속도가 빠르다.

1.1.3 노드 패키지 매니저란 무엇일까?

- npm은 https://www.npmjs.com에서 필요한 자바스크립트 라이브러리를 내려받아 설치하고 삭제하는 등의 관리를 해주는 프로그램이다.
- npm은 실제로 node_modules 폴더에 라이브러리를 내려받아 저장하고 package.json이라는 파일에 설치된 라이브러리의 정보를 적어 저장한다.

1.1.4 웹팩이란 무엇일까?

- 웹팩은 프로젝트에 사용된 파일을 분석하여 기존 웹 문서 파일로 변환하는 도구이다. 웹 브라우저가 해석할 수 없는 .hbs, .cjs, .sass 등의 파일을 웹팩이 분석하여 .js, .css 등의 파일로 변환해 준다.
- 웹팩은 간단한 노드 기반의 웹 서버를 구동하기도 한다.

1.2 리액트 개발 환경 설치하기

~~1.2.0 노드 버전 매니저로 노드제이에스 설치하기~~

- 노드제이에스는 구글에서 공개한 소프트웨어로 V8 엔진을 기반으로 만든 자바스크립트 런타임 도구이다. 쉽게 말해 웹 브라우저가 아닌 서버(컴퓨터)에서 자바 스크립트를 실행할 수 있게 해준다.

- NVM은 아래의 주소에 nvm-setup.zip을 내려받아 압축을 해제하고 설치 파일을 실행한다.
  - https://github.com/coreybutler/nvm-windows/releases

```
# NVM 버전 확인하기
d:\prod\nvm>nvm -v
Running version 1.1.8.

# NVM으로 노드제이에스 설치하기
d:\prod\nvm>nvm install 10.10.0
Downloading node.js version 10.10.0 (64-bit)...
Complete
```

1.2.1 노드제이에스 설치하기

- 노드제이에스(Node.js)는 서버 사이드 자바스크립트로 서버 측에서 실행되는 자바스크립트 실행 환경이다. 뷰 CLI(Command Line Interface)를 이용하여 쉽게 뷰 프로젝트를 구성하려면 노드제이에스가 설치되어야 한다.

- 설치
  - https://nodejs.org/dist/v16.13.0/ 공식 사이트에서 "node-v16.13.0-x64.msi" 파일을 다운로드하여 설치한다.

```
# node 버전 확인
D:\dev\workspace\react2>node -v
v16.13.0

# npm 버전 확인
D:\dev\workspace\react2>npm -v
8.3.0

# 추가 작업(옵션): 명령 프롬프트를 관리자 권한으로 실행하여 아래와 같이 수행한다.
# C:\Windows\system32>npm install --global --production windows-build-tools@4
# C:\Windows\system32>choco install python visualcpp-build-tools
# C:\Windows\system32>npm config set msvs_version 2017
```

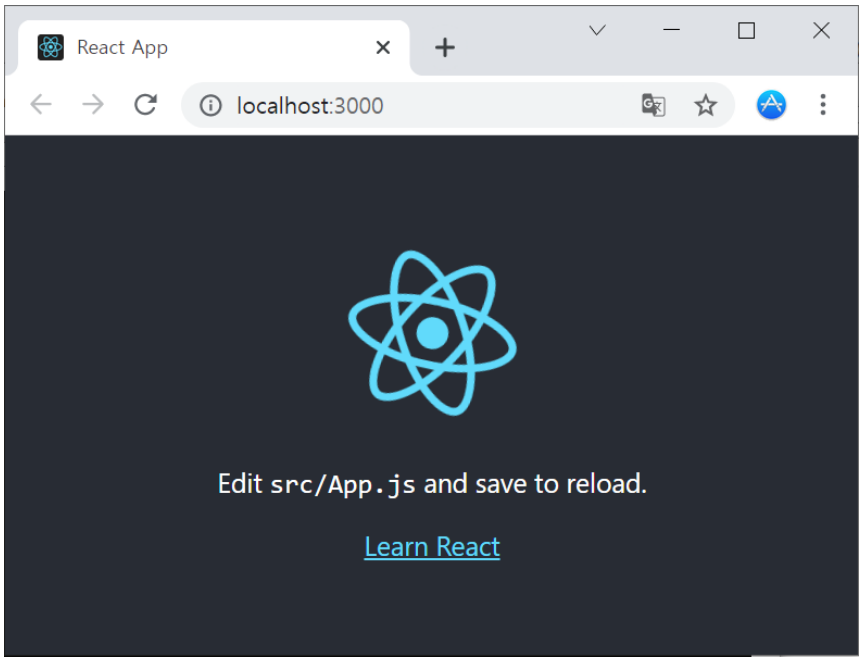
1.2.2 yarn과 create-react-app 설치하기

- npm으로 yarn을 설치한다. 앞으로 각종 설치는 npm 대신 yarm을 사용하도록 한다.

```
# 1. yam 설치하기
D:\dev\workspace\react2>npm install -g yarn
```

```
# 2. 리액트 앱 생성하기
# D:\dev\workspace\react2>yarn create react-app do-it-example-lab --scripts-version 2.1.7

# 3. 리액트 앱 구동하기
D:\dev\workspace\react2>cd do-it-example-lab
D:\dev\workspace\react2\do-it-example-lab>yarn start
yarn run v1.22.17
$ react-scripts start
```



1.2.3 예제에 필요한 라이브러리 미리 설치하기

■ package.json을 통해 프로젝트에 필요한 라이브러리를 미리 설치한다.

■ package.json 수정하기
- 프로젝트 루트 폴더(do-it-example-lab)에 있는 package.json 파일을 다음과 같이 수정한다.

[./package.json]

```
1  {
2    "name": "do-it-examples-lab",
3    "version": "0.1.0",
4    "private": true,
5    "dependencies": {
6      "axios": "0.18.1",
7      "enzyme": "^3.8.0",
8      "enzyme-adapter-react-16.3": "^1.4.1",
9      "moment": "^2.24.0",
10     "next": "^8.1.0",
11     "react": "^16.7.0",
12     "react-dom": "^16.7.0",
13     "react-redux": "^6.0.0",
14     "react-router-dom": "^5.0.0",
15     "react-scripts": "2.1.7",
```

```
    "react-test-renderer": "^16.7.0",
    "react-with-styles": "^3.2.1",
    "recompose": "^0.30.0",
    "redux": "^4.0.1",
    "redux-pack": "^0.1.5",
    "redux-thunk": "^2.3.0",
    "reselect": "^4.0.0",
    "selector-action": "^1.1.1"
  },
  "scripts": {
    "dev": "next",
    "predeploy": "yarn build-all",
    "deploy": "firebase deploy",
    "build-all": "yarn ssrbuild && yarn build-firebase",
    "build-firebase": "cd \"./functions\" && yarn --ignore-engines",
    "ssrbuild": "next build",
    "storybook": "start-storybook -p 9001 -c .storybook",
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "mockserver": "json-server --watch --delay 500 --port 4000 mock/create.js",
    "errorserver": "node mock/fake.js",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": [
    ">0.2%",
    "not dead",
    "not ie <= 11",
    "not op_mini all"
  ],
  "devDependencies": {
    "@babel/core": "7.5.5",
    "@babel/plugin-syntax-object-rest-spread": "^7.2.0",
    "@storybook/addon-actions": "^5.2.6",
    "@storybook/addons": "^5.2.6",
    "@storybook/react": "^5.2.6",
    "aphrodite": "^2.2.3",
    "babel-loader": "^8.0.5",
    "json-server": "^0.14.2",
    "node-sass": "^4.11.0",
    "react-with-styles-interface-aphrodite": "^5.0.1",
    "redux-devtools-extension": "^2.13.8",
    "sass-loader": "^7.1.0",
    "storybook-addon-jsx": "^7.1.13"
  }
}
```

■ package.json에 적힌 라이브러리 모두 설치하기

```
# 1. yarn 명령어를 실행한다.
D:\dev\workspace\react2\do-it-example-lab>yarn add node-sass
D:\dev\workspace\react2\do-it-example-lab>yarn
yarn install v1.22.17
warning ..\package.json: No license field
[1/4] Resolving packages...
success Already up-to-date.
Done in 0.66s.

# 2. 에러가 발생할 경우, Windows PowerShell 관리자 권한으로 아래와 같이 실행한다.
```

```
D:\dev\workspace\react2>npm install --global --production windows-build-tools@4
```

- babel-loader 호환성 오류 해결하기
  - babel-loader 버전 오류가 발생할 경우 루트 폴더에 npm 환경변수 파일을 만들어 아래와 같이 저장한다.

[./.env]

```
1  SKIP_PREFLIGHT_CHECK=true
```

1.2.4 비주얼 스튜디오 코드와 플러그인 설치하기

- Reactjs code snippets 플러그인 설치하기
  - 리엑트에서 자주 사용하는 코드 뭉치를 자동으로 완성해 준다.
  - Extensions에서 ‘reactjs code snippets’를 검색한 다음 플러그인을 설치한다.

- RCC.jsx 파일 생성
  - 편집 화면에서 rcc라고 입력하여 목록이 나오면 rcc를 선택하여 자동으로 코드를 생성한다.
  - 파일 이름이 RCC.jsx이므로 리액트 컴포넌트의 클래스 이름을 RCC로 만든다.

[./src/RCC.jsx]

```
1  import React, { Component } from 'react';
2
3  class RCC extends Component {
4      render() {
5          return (
6              <div>
7
8              </div>
9          );
10     }
11 }
12
13 export default RCC;
```

- Prettier - Code formatter 플러그인 설치하기
  - 코드의 줄바꿈 등의 스타일을 자동으로 변환하여 프로젝트의 코드 입력 스타일을 동일하게 유지시켜준다.
  - 프로젝트 폴더(do-it-example-lab)에 Prettier 설정 파일을 생성한다. 이때 파일 이름에 반드시 점(.)을 포함해야 한다.
  - 2: 탭을 사용할 때 빈칸으로 채운다.
  - 3: 파일 최대 길이를 100칸으로 지정한다.
  - 4: 탭의 빈칸을 두칸으로 지정한다.
  - 5: 나열 항목의 마지막에 항상 쉼표(,)를 붙인다.
  - 6: 실행 줄 마지막에 항상 세미콜론(;)을 붙인다.
  - 7: 문자 따옴표를 작은따옴표('')로 통일한다.

[./.prettierrc]

```
1  {
```

```
    "useTabs": false,
    "printWidth": 100,
    "tabWidth": 2,
    "trailingComma": "all",
    "semi": true,
    "singleQuote": true
}
```

■ prettier로 코드 입력 스타일 적용하기

- RCC.jsx 파일을 열고 Ctrl+Shift+P를 누르면 명령어 팔레트가 나타난다. 'Format'을 검색하여 Format Document라는 명령어를 선택하여 Enter를 눌러 명령어를 실행한다(혹은 Ctrl+Shift+F). 코드가 다음과 같이 정리되는 것을 확인할 수 있다.
- 5: 문자열에 큰따옴표를 사용하였고 끝에 세미콜론(;)이 빠졌다.
- 6: HTML 코드의 문자에 작은따옴표를 사용하였다.

[./src/RCC.jsx]

```
import React, { Component } from 'react';

class RCC extends Component {
  render() {
    var text = "따옴표"
    return <div name="name">{text}</div>
  }
}

export default RCC;
```

[./src/RCC.jsx]

```
import React, { Component } from 'react';

class RCC extends Component {
  render() {
    var text = '따옴표';
    return <div name="name">{text}</div>;
  }
}

export default RCC;
```

1.3 리액트 앱 수정하기

■ App.css 수정하기

- App.css 맨 아래에 다음 코드를 추가한다.

[./src/App.css의 일부]

```
...(생략)...

.title {
  font-style: italic;
}
```

■ 스타일 반영하기

- 5: render() 함수는 HTML을 반환한다. HTML의 스타일 클래스 이름은 자바스크립트 클래스(class) 키워드와 같으므로 리액트에서는 class가 아니라 className으로 정의하여 사용한다.

[./src/App.js]

```
1  import React, { Component } from 'react';
2  import './App.css';
3
4  class App extends Component {
5    render() {
6      return (
7            <div className="App">
8              <div className="title">두잇! 리액트 시작하기</div>
9            </div>
10     );
11   }
12 }
13
14 export default App;
```

■ 리액트 핫 리로딩으로 변경된 화면 확인하기

- 리액트 앱을 구동한 상태라면 파일을 저장한 즉시 화면이 바뀐다.