React로 Callendar App 만들기 – 2.

## 헤더 만들기



헤더에 나타나는 표시사항으로는 현재 표시하고 있는 달, 오늘 날짜, 월 이동 입니다. 위의 모든 내용들은 App component에서 관리되고 Header component로 props로 전달할 겁니다.

앞서 말했듯이 이 프로젝트는 Date객체가 아닌 외부 라이브러리인 moment.js를 사용해서 더욱 빠르게 project를 향상 시키는 방법을 사용하도록 할겁니다. Moment.js의 설명은 git의 moment.js 페이지를 참조 해주시기 바랍니다.

Yarn add moment 명령어를 통하여 해당 프로젝트 폴더에 moment 라이브러리를 설치하도록 합니다.

이후 app.js에서 moment를 import하고 header component로 전달한 내용들을 구성하도록 합니다.

**/src/app.js**

import moment from 'moment'

…

export default class App extends Component {

state = {

calendarYM : moment(),

today : moment()

}

render() {

return (

<div className="test-layout">

<div className="RCA-app-container">

<Header calendarYM={this.state.calendarYM.format("YYYY년 MM월")}

today={this.state.today.format("현재 YYYY - MM - DD")}

/>

<Calendar />

</div>

</div>

)

}

}

State내부에는 현재 오늘을 표시해주기 위한 today와 calendarYM을 moment객체를 이용하여 저장했습니다. Header로 두개의 시간 정보 string을 props로 전달 해줍니다. 이후 Header.js에서 console.log(this.prop)로 props의 전달 내용을 확인 합니다.

/src/Header.js

export default class Header extends Component {

render() {

console.log(this.props)

return (

<div className="RCA-header-container">

</div>

)

}

}



이로써 props에 우리가 원하던 내용이 들어 있는 것을 확인 할 수 있습니다.

**Issue 발생** ! props를 확인하는 과정에서 render가 두번 일어나는 것을 확인 했습니다. /src/index.js를 다음과 같이 React.strict 를 제거하고 App 컴포넌트만 마운트 시켜주시기 바랍니다.

**/src/index.js**

ReactDOM.render(

<App />

,document.getElementById('root')

);

계속 해당 props의 내용을 화면에 표기하도록 하겠습니다.

**/src/header.js**

render() {

return (

<div className="RCA-header-container">

<h2>

{this.props.calendarYM}

</h2>

<h3>

{this.props.today}

</h3>

</div>

)

}



결과가 잘 보이시나요 ? 이제 달을 이동 할 수 있도록 이동 버튼을 만들어 주도록 하겠습니다. 괜찮은 곳에서 < > 형태의 png 파일을 다운로드 할 수 있도록 합시다. font awesome같은 css파일을 들고 올 수도 있지만, 우리가 사용하고자 하는 font하나 이외에 모든 style sheet가 우리 프로젝트에 같이 딸려 올 수 있으니, icon에 관련된 css를 직접 만들어 주었습니다.

**/src/Header.js**

(class 적용)

render() {

return (

<div className="RCA-header-container">

<h2 className="RCA-header-calendarYM RCA-header-middle">

{this.props.calendarYM}

</h2>

<h3 className="RCA-header-today RCA-header-middle">

{this.props.today}

</h3>

<ul className="RCA-header-buttons RCA-header-middle">

<li>

<i className="move-button left-img icon">

</i>

</li>

<li>

이동

</li>

<li>

<i className="move-button right-img icon">

</i>

</li>

</ul>

</div>

)

}

**/src/style/RCA.css**

/\* header \*/

.RCA-header-container{

flex-basis: 50px;

background-color: rgb(224, 215, 202);

display: flex;

}

.move-button{

width: 40px;

height: 40px;

display: inline-block;

}

.icon{

background-size: contain;

background-repeat: no-repeat;

cursor: pointer;

}

.left-img{

background-image : url('./left-icon.png');

}

.right-img{

background-image: url('./right-icon.png');

}

.RCA-header-middle{

line-height: 50px;

}

.RCA-header-calendarYM{

box-sizing: border-box;

padding-left: 5px;

flex-grow: 1;

}

.RCA-header-buttons{

flex-basis: 120px;

padding-left: 0;

padding-top: 4px;

display: block;

list-style-type: none;

margin: 0;

}

.RCA-header-buttons li{

float: left;

height: 40px;

line-height: 40px;

font-size: 20px;

}

Css를 적용이 제대로 되었다면 브라우저로 돌아와 header의 상태를 확인해볼까요 ?



이제 이동 버튼을 활성화 해주도록 하겠습니다.

**/src/app.js**

State아래에 moveMonth 함수를 작성해줍니다.

moveMonth = (month) => {

this.setState({

calendarYM : this.state.calendarYM.add(month,'M')

})

}

Add 메서드는 1번째 인자로 숫자를 받고 2번째 인자로 주 , 일 , 월 , 연도에 해당하는 char를 받아 현재 moment객체에 더해서 새로운 moment객체를 반환합니다.

(더 이상 참조하지 않는 객체는 알아서 가비지 컬렉터에 의해 폐기되니 걱정 마세요.)

<Header calendarYM={this.state.calendarYM.format("YYYY년 MM월")}

today={this.state.today.format("현재 YYYY - MM - DD")}

moveMonth={this.moveMonth}

/>

헤더 컴포넌트에 moveMonth props에 moveMonth 함수를 전달합니다.

이제 이동 옆 두개의 아이콘에 onClick이벤트를 달아줍니다.

<li>

<i className="move-button left-img icon" onClick={()=>{this.props.moveMonth(-1)}}>

</i>

</li>

<li>

이동

</li>

<li>

<i className="move-button right-img icon" onClick={()=>{this.props.moveMonth(1)}}>

</i>

</li>

이제 이동 옆 아이콘을 눌러보면 한 달씩 줄어들거나 한달씩 증가하여 화면에 표시되는 것을 볼 수 있습니다.

