

멘토링 2주차 내용 정리 pt2

CSS Layout 구현 관련

제가 팀채널에 올린 미션 5코드 참고하시면 레이아웃 짜시는데 많이 도움 될 것 같습니다. 코드 보다 궁금한거 있으면 질문 남겨주시면 시간될 때 답변해드리겠습니다.

생산성 향상 팁

코드리뷰 받은 내용을 수정하고 싶은데 수정할 내용이 너무 많아서 힘들어 하는 것 같아, 코드 리팩토링을 쉽게 만들어주는 확장 프로그램과 vscode(cursor) 기능을 소개해드리겠습니다.

eslint & prettier

eslint는 코드 규칙을 강제시키고, prettier는 코드 구조를 자동으로 일치시켜주는 라이브러리입니다. 일관된 코드를 위해 사용하시는걸 권장드립니다. eslint를 엄격하게 규칙을 설정하면 개발 시 불편할 수 있으니 이걸 적절하게 조절하시면 됩니다.

css sorter

css property를 일관되게 정렬해주는 확장프로그램이 있습니다. 사용하시면 좀더 css 가독성이 올라갈 수 있을거예요 (css 파일을 사용한다는 가정하에)

<https://marketplace.visualstudio.com/items?itemName=mrmlnc.vscode-postcss-sorting>

정렬되게하는 방법은 **cmp + p**로 명령창을 띄운 후 **>PostCss Sorting: Run** 명령 적어주시면 됩니다.

세팅은 settings.json에 다음 코드블럭 추가해주시는걸 추천드립니다. NHN 컨벤션을 따르는 설정입니다.

```
"postcssSorting.config": {
  "order": [
    "custom-properties",
    "dollar-variables",
    "declarations",
    "at-rules",
    "rules"
  ],
  "properties-order": [
    /* Layout */
    "display",
    "grid",
    "grid-column-gap",
    "grid-row-gap",
    "grid-auto-flow",
    "grid-auto-rows",
    "grid-auto-columns",
    "justify-items",
    "align-content",
    "place-items",
    "gap",
    "align-items",
    "justify-content",
    "flex-wrap",
    "flex-basis",
    "flex-grow",
    "flex-shrink",
    "flex",
    "align-self",
    "flex-direction",

    /* Box */
    "margin",
    "margin-top",
    "margin-right",
    "margin-bottom",
    "margin-left",
    "padding",
    "padding-top",
    "padding-right",
    "padding-bottom",
    "padding-left",
```

```
"border",
"border-top",
"border-bottom",
"border-right",
"border-left",
"border-style",
"border-color",
"border-top-width",
"border-right-width",
"border-bottom-width",
"border-left-width",
"border-top-style",
"border-right-style",
"border-bottom-style",
"border-left-style",
"border-top-color",
"border-right-color",
"border-bottom-color",
"border-left-color",
"border-top-left-radius",
"border-top-right-radius",
"border-bottom-right-radius",
"border-bottom-left-radius",
"outline",
"outline-width",
"outline-style",
"outline-color",
"outline-offset",
"box-shadow",
"overflow",
"overflow-x",
"overflow-y",
"clip",
"position",
"top",
"right",
"bottom",
"left",
"z-index",
"width",
"min-width",
"max-width",
"height",
```

```
"min-height",
"max-height",
"float",
"clear",
"visibility",
"vertical-align",

/* Background */
"background",
"background-color",
"background-image",
"background-repeat",
"background-attachment",
"background-position",
"background-clip",
"background-origin",
"background-size",

/* Font */
"font-family",
"font-size",
"font-style",
"font-weight",
"line-height",
"color",
"text-align",
"text-decoration",
"text-transform",
"letter-spacing",
"text-shadow",
"white-space",
"word-spacing",
"word-break",
"word-wrap",
"text-indent",
"direction",
"unicode-bidi",
"hyphens",

/* Animation and Transition */
"animation",
"animation-name",
"animation-duration",
```

```
"animation-timing-function",
"animation-delay",
"animation-iteration-count",
"animation-direction",
"animation-fill-mode",
"animation-play-state",
"transition",
"transition-property",
"transition-duration",
"transition-timing-function",
"transition-delay",

/* Other */
"content",
"counter-reset",
"counter-increment",
"quotes",
"list-style",
"list-style-position",
"list-style-type",
"caption-side",
"empty-cells",
"table-layout",
"pointer-events",
"cursor",
"resize",
"overflow-wrap",
"scroll-snap-type",
"scroll-padding",
"scroll-padding-top",
"scroll-padding-right",
"scroll-padding-bottom",
"scroll-padding-left",
"scroll-behavior",
"scroll-snap-align",
"scroll-snap-margin",
"scroll-snap-stop",
"scrollbar-width",
"scrollbar-color"
]
},
```

Vim

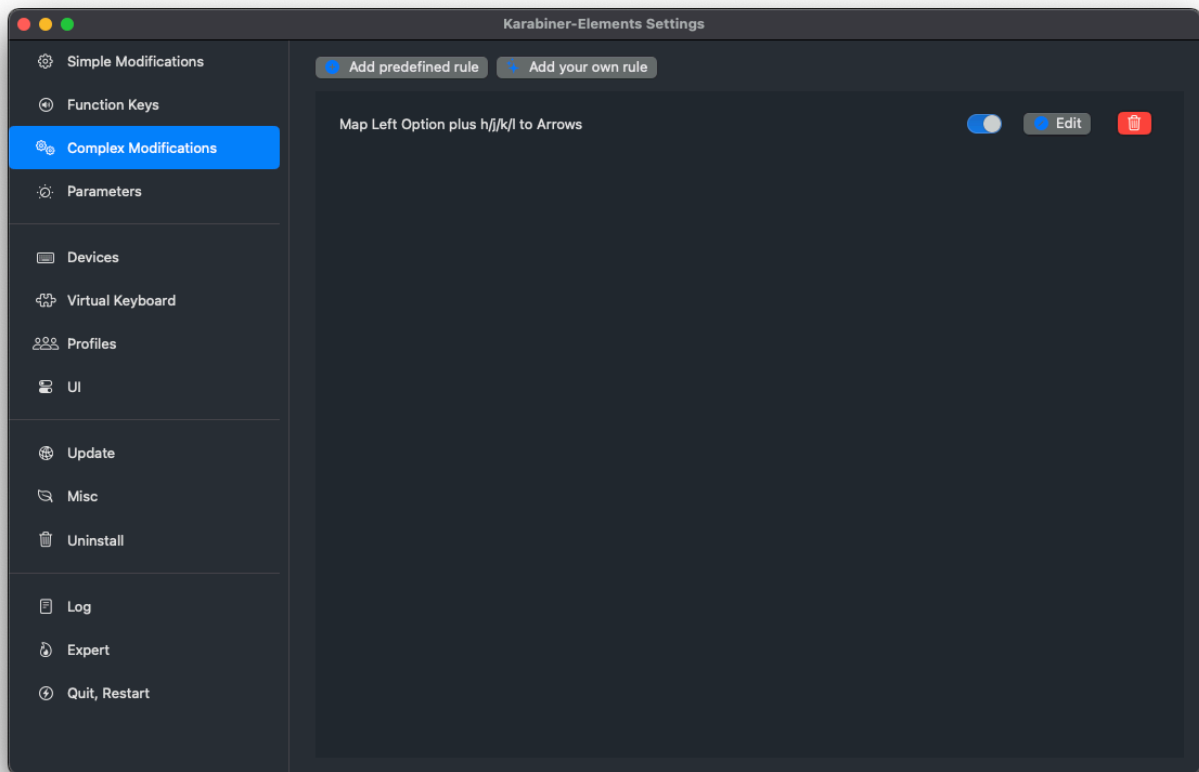
저는 vscode에서 vim 플러그인을 깔아서 vim으로 코드를 작성합니다. 익숙해지면 다른 단축키와 조합하기 편하고 마우스에 손댈일이 거의 없어지니 한번 공부해보시는걸 추천드립니다.

<https://www.youtube.com/watch?v=cY0JxzENBJg>

이 영상에 필요한 핵심부분만 잘 요약해놔서 보시는걸 추천드립니다. 여기 나오는것 정도만 익히고 다 숙지되면 더 고급 기능들로 넘어가면 좋을 것 같아요.

vim을 쓰다보면 hjkl로 커서이동하는게 익숙해져서 vscode가 아닌 다른곳에서도 이 방법을 사용하고 싶어질텐데요.

Karabiner 라는 툴을 사용하면 키 바인딩을 커스텀할 수 있습니다. 저는 **cmd + hjkl**로 vim처럼 커서 이동만 가능하게 세팅해놨습니다.



Cursor editor

<https://www.cursor.com/>

vscode 기반으로 만들어진 AI 기능 탑재되어있는 에디터입니다. 코파일럿 기능을 켜놓으면 코드 짤 때 의도를 파악해서 자동으로 완성본을 추천해줍니다. 코드 자체는 아직 AI가 잘 못짜기 때문에 symbol rename안되는 변수들 rename하는 것 같은 노가다 작업 위주로 하게하면 좋습니다.

Css Peek

함수, 변수와 같이 html tag에 설정된 class/id가 어떤 css파일과 연결되어있는지 한번에 파악할 있습니다.

(react에서도 됩니다)

<https://marketplace.visualstudio.com/items?itemName=pranaygp.vscode-css-peek>

Preview

HTML, Markdown, SVG 파일 등을 vscode에서 미리보기를 할 수 있는 확장 프로그램이 있습니다.

이외에도 유용한 확장 프로그램들이 많으니 한번 찾아서 활용해보세요.

참고로 다운로드 수가 너무 적고 인증되지 않은 확장 프로그램은 설치를 지양하는게 좋습니다.

단축키

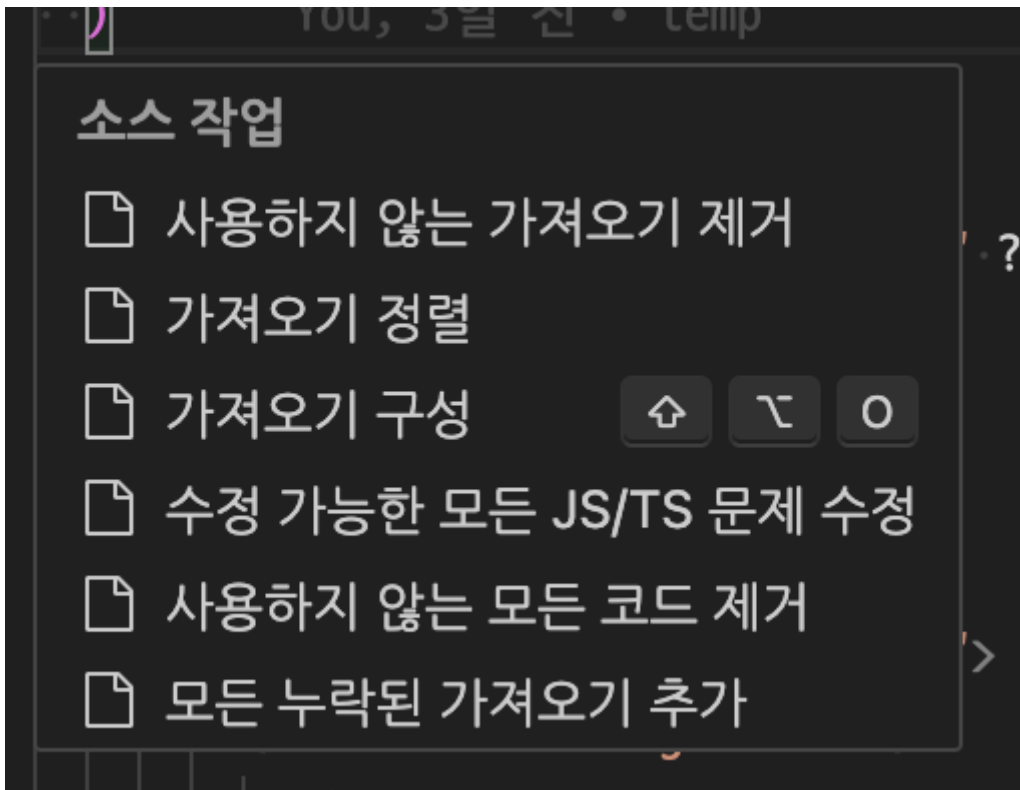
vscode에 유용한 단축키가 많은데 잘 활용하시면 생산성 향상에 매우 도움이 됩니다.

다음과 같은 기능들이 제가 많이 사용하는 단축키들입니다. 저는 저한테 맞게 키바인딩을 커스텀해놔서 여러분들도 키바인딩을 본인이 기억하기 쉽게 만들어놓으면 좋을겁니다. (설정에 키바인딩 탭으로 이동하는 메뉴가 있습니다.)

명령	키 바인딩	언제	소스
<Tab> 키로 포커스 이동 설정/해제	M	-	시스템
가리키기 또는 포커스 표시	R I	editorTextFocus	시스템
가져오기 구성	O	textInputFocus && !editorReadonly && s...	시스템
개발자: 개발자 도구 설정/해제	I	isDevelopment	시스템
개발자: 창 다시 로드	R	isDevelopment	시스템
검색 편집기: 검색 편집기 입력 포커스	Escape	inSearchEditor	시스템
검색 편집기: 다시 검색	R	inSearchEditor	시스템
검색 편집기: 단어 단위로 토글	W	inSearchEditor && searchInputBoxFocus	시스템
검색 편집기: 대/소문자 구분 토글	C	inSearchEditor && searchInputBoxFocus	시스템
검색 편집기: 모든 일치 항목 선택	L	inSearchEditor	시스템
검색 편집기: 정규식 사용 토글	R	inSearchEditor && searchInputBoxFocus	시스템
검색 편집기: 컨텍스트 줄 늘이기	=	inSearchEditor	시스템
검색 편집기: 컨텍스트 줄 줄이기	-	inSearchEditor	시스템
검색 편집기: 파일 결과 삭제	Backspace	inSearchEditor	시스템
검색 편집기: 편집기에서 결과 열기	Enter	hasSearchResult && searchViewletFocus	시스템
검색: 검색 취소	Escape	listFocus && searchViewletVisible && !...	시스템
검색: 다음 검색 결과에 포커스	F4	hasSearchResult inSearchEditor	시스템
검색: 이전 검색 결과에 포커스	F4	hasSearchResult inSearchEditor	시스템
검색: 파일에서 바꾸기	H	-	시스템
검색: 파일에서 찾기	F	-	시스템
구현으로 이동	F12	editorHasImplementationProvider && edi...	시스템
기본 설정: 바로 가기 키 열기	R S	-	시스템
기본 설정: 색 테마	R T	-	시스템
기본 설정: 설정 검색 결과 지우기	Escape	inSettingsEditor && inSettingsSearch	시스템
기본 설정: 설정 검색에 포커스	F	inSettingsEditor	시스템
기본 설정: 설정 목록에 포커스		inSettingsEditor && settingRowFocus	시스템

Source Action 단축키

저는 **cmd + shift + a**로 바인딩 해놨습니다. 명령 이름은 "소스 작업"입니다.



파일 관련

- 파일 검색해서 창으로 열기 : **cmd + p** 로 명령어 창 열어서 파일이름 검색
- 최근 닫힌 파일(창)을 다시 열기 : 명령 이름 -> "보기: 닫힌 편집기 다시 열기"
- 현재 창 닫기 : 명령 이름 -> "창 닫기" + 언제 -> "!editorIsOpen && !multipleEditorGroups"
- 현재 창만 남기고 나머지 창 닫기(창 정리 필요할 때 유용) : 명령 이름 -> "보기: 그룹의 다른 편집기 닫기"
- 현재 커서 위치를 이전 커서 위치로 이동 : 명령 이름 -> "돌아가기"
- 현재 커서 위치를 다시 앞으로 이동 : 명령 이름 -> "앞으로 이동"

에디터 관련

- 왼쪽 사이드 바 토글 : **cmd + b**
- 내장 터미널 토글 : **cmd + j**
- 사이드 바 코드 탭으로 : **cmd + shift + e**
- 사이드 바 확장프로그램 탭으로 : **cmd + shift + x**
- 사이드 바 전역검색 탭으로 : **cmd + shift + f**
- 사이드 바 전역검색 탭 -> 치환 모드 : **cmd + shift + h**
- 사이드 바 디버거 탭 : **cmd + shift + d**

코드 관련

같은 문자 다중 선택

다중 선택하고싶은 문자를 드래그 후 **cmd + d**를 누르면 하나씩 위에서 아래로 선택할 수 있고 **cmd + shift + d**를 누르면 한번에 선택할 수 있습니다.

다중 커서를 활성화해 특정 반복적인 줄을 동시에 수정할 수 있습니다.


```

4  ..return (
5  ...<header className={style["gnb"]} > You, 5분 전 • Uncommitted changes
6  ...  <div className={style["gnb-container"]} > You, 5분 전 • Uncommitted changes
7  ...    <div className={style["gnb-left"]} > You, 5분 전 • Uncommitted changes
8  ...      <img className={style["logo"]} src={logoImg} alt="pandaLogo" /> You, 5분 전 • Uncommitted changes
9  ...      <div className={style["menu-container"]} > You, 5분 전 • Uncommitted changes
10 ...        <div className={style["menu"]} >자유게시판</div> You, 5분 전 • Uncommitted changes
11 ...        <div className={style["menu"]} >중고마켓</div> You, 5분 전 • Uncommitted changes
12 ...      </div>
13 ...    </div>
14 ...    <LoginButton>로그인</LoginButton>
15 ...  </div>

```

```

24 ..return (
25 ...<header (property) React.HTMLAttributes<T>.className?: string > You, 7분 전 • Uncommitted changes
26 ...  <div className={style["hello-gnb-container"]} > You, 7분 전 • Uncommitted changes
27 ...    <div className={style["hello-gnb-left"]} > You, 7분 전 • Uncommitted changes
28 ...      <img className={style["hello-logo"]} src={logoImg} alt="pandaLogo" /> You, 7분 전 • Uncommitted changes
29 ...      <div className={style["hello-menu-container"]} > You, 7분 전 • Uncommitted changes
30 ...        <div className={style["hello-menu"]} >자유게시판</div> You, 7분 전 • Uncommitted changes
31 ...        <div className={style["hello-menu"]} >중고마켓</div> You, 7분 전 • Uncommitted changes
32 ...      </div>
33 ...    </div>
34 ...    <LoginButton>로그인</LoginButton>
35 ...  </div>
36 ...</header>
37 ..)

```

문자열 치환

로컬 문자열 찾기 : **cmd + f**

로컬 문자열 치환 모드 전환 : **cmd + h**

문자열 찾기와 치환을 사용하면 한번에 원하는 문자의 형식을 바꿀수 있습니다.

아래처럼 정규식을 사용하면 쉽게 치환할 수 있습니다. 이걸 전역 검색 탭에서 치환하면 모든 파일에 적용됩니다.

src > layouts > GNB.jsx > GNB

~/codeit-mentor/codeit-sprint-2-example/src · 강조 표시한 항목 포함

```
1 import { createButton } from "hocs/createbutton"
2 import React, { useEffect, useState } from "react"
3 import { useViewport } from "../contexts/viewportContext"
4 import "../css/GNB.css"
5 import pandaLogo from "../images/pandaLogoSm.svg"
6 import pandaTypo from "../images/pandaLogoTypo.svg"
7
8 const LoginButton = createButton({
9   buttonType: "small-button",
10   backgroundColor: "bg-primary-100",
11   color: "txt-gray-100"
12 })
13
14 export function GNB() {
15   const viewport = useViewport()
16   const [logoImg, setLogoImg] = useState(
17     viewport === "mobile" ? pandaTypo : pandaLogo
18   )
19
20   useEffect(() => {
21     setLogoImg(viewport === "mobile" ? pandaTypo : pandaLogo)
22   }, [viewport])
23
24   return (
25     <header className="gnb">
26       <div className="gnb-container">
27         <div className="gnb-left"> You, 3일 전 · temp
28         <img className="logo" src={logoImg} alt="pandaLogo" />
29         <div className="menu-container">
30           <div className="menu">자유게시판</div>
31           <div className="menu">중고마켓</div>
32         </div>
33       </div>
34       <LoginButton>로그인</LoginButton>
35     </div>
36   </header>
37 )
38 }
39
```

className="([^\"]+)"

Aa ab

7의 3

↑ ↓ ≡ ×

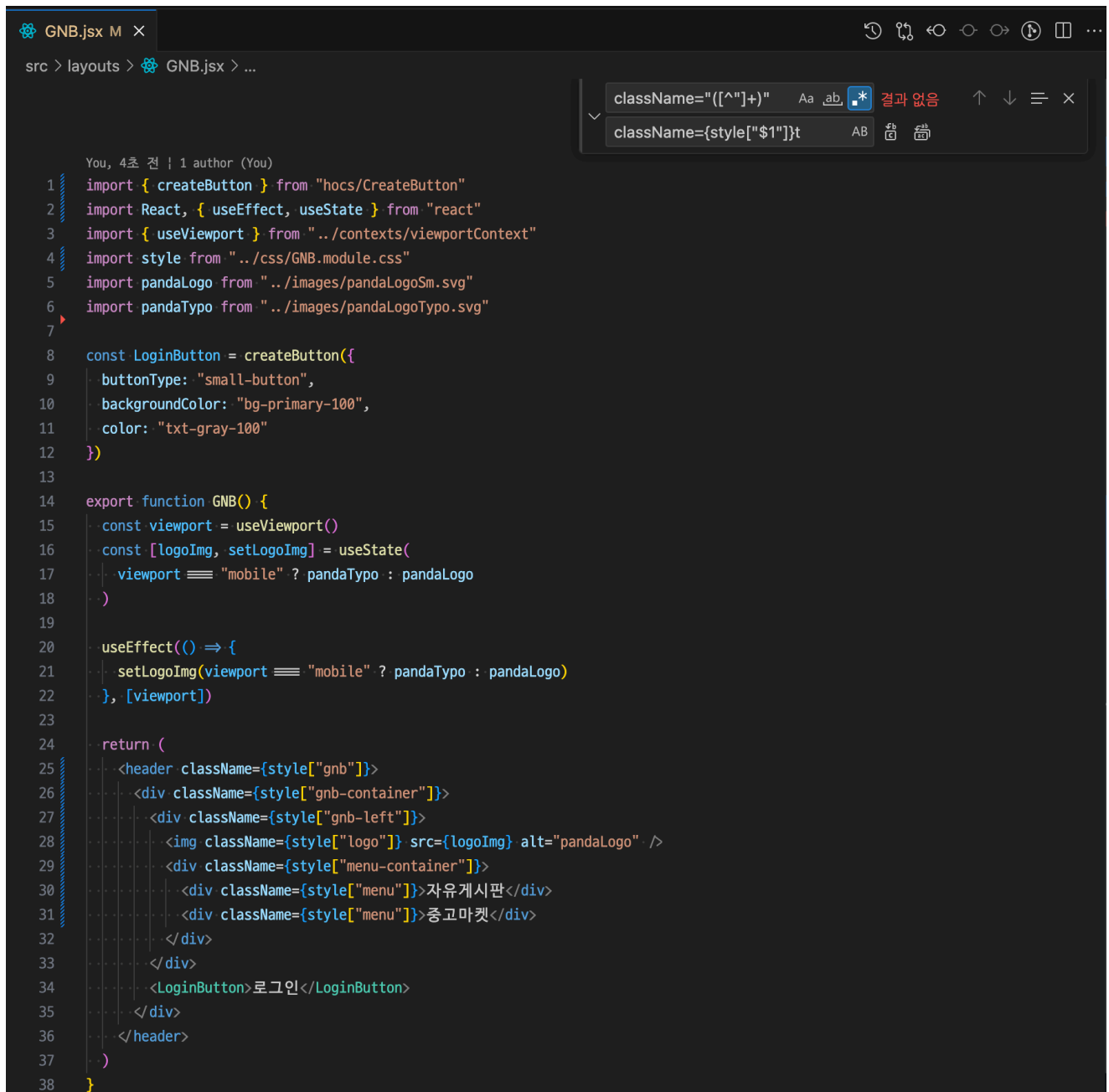
className={style["\$1"]}

AB

🔍 📄

Add to Chat 📄L

Edit 📄K



```
GNB.jsx M X
src > layouts > GNB.jsx > ...

You, 4초 전 | 1 author (You)
1 import { createButton } from "hocs/CreateButton"
2 import React, { useEffect, useState } from "react"
3 import { useViewport } from "../contexts/viewportContext"
4 import style from "../css/GNB.module.css"
5 import pandaLogo from "../images/pandaLogoSm.svg"
6 import pandaTypo from "../images/pandaLogoTypo.svg"
7
8 const LoginButton = createButton({
9   buttonType: "small-button",
10   backgroundColor: "bg-primary-100",
11   color: "txt-gray-100"
12 })
13
14 export function GNB() {
15   const viewport = useViewport()
16   const [logoImg, setLogoImg] = useState(
17     viewport === "mobile" ? pandaTypo : pandaLogo
18   )
19
20   useEffect(() => {
21     setLogoImg(viewport === "mobile" ? pandaTypo : pandaLogo)
22   }, [viewport])
23
24   return (
25     <header className={style["gnb"]}>
26       <div className={style["gnb-container"]}>
27         <div className={style["gnb-left"]}>
28           <img className={style["logo"]} src={logoImg} alt="pandaLogo" />
29           <div className={style["menu-container"]}>
30             <div className={style["menu"]}>자유게시판</div>
31             <div className={style["menu"]}>종고마켓</div>
32           </div>
33         </div>
34         <LoginButton>로그인</LoginButton>
35       </div>
36     </header>
37   )
38 }
```

에러로 이동

현재 커서 기준으로 다음 에러로 이동 : 저는 **cmd + e**로 바인딩을 해봤는데 명령 이름은 "파일의 다음 문제로 이동 (오류, 경고 정보)" 입니다.

```
26  
27 I'm Error line You, 3분 전 • Uncommitted changes  
⊗ GNB.jsx 문제 3개 중 1개  
예기치 않은 키워드 또는 식별자입니다. ts(1434)  
28  
29 return (  
30   <header className="gnb">  
31     <div className="gnb-container">  
32       <div className="gnb-left">  
33         <img className="logo" src={logoImg} alt="pandaLogo" />  
34         <div className="menu-container">  
35           <div className="menu">자유게시판</div>  
36           <div className="menu">중고마켓</div>  
37         </div>  
38       </div>  
39       <LoginButton>로그인</LoginButton>  
40     </div>  
41   </header>  
42 )  
43 }
```

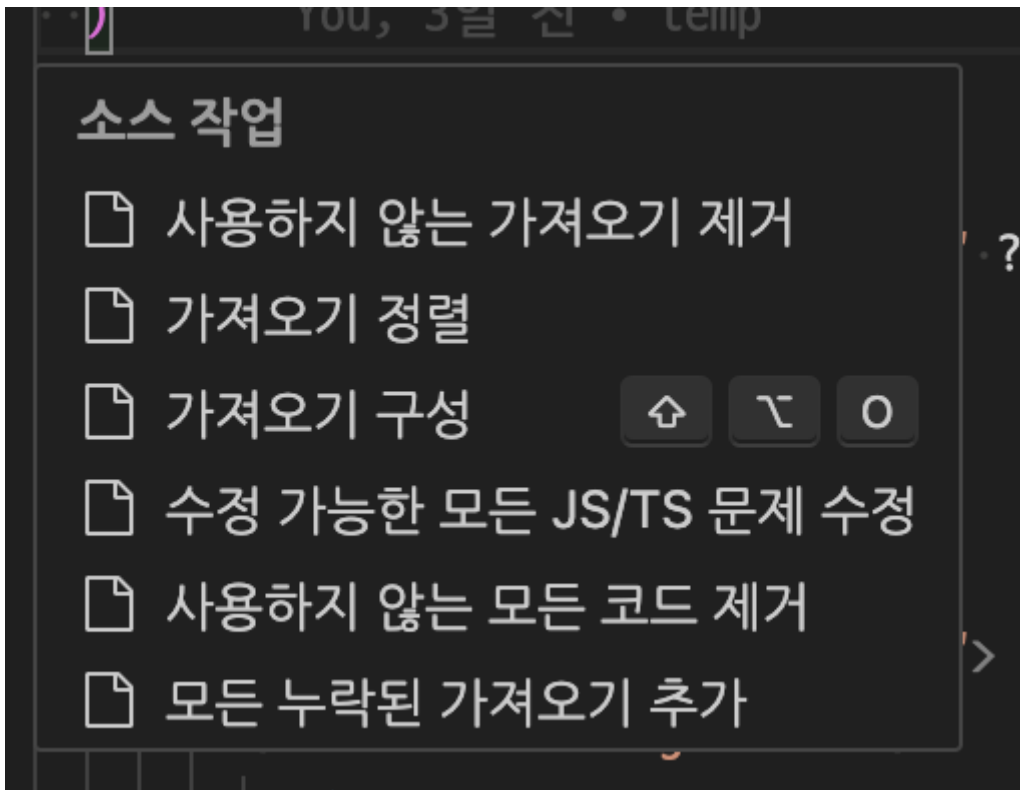
Source Action 기능들

symbol rename - 변수나 함수이름을 변경할 때 한번에 같은 이름의 변수/함수를 변경할 수 있습니다.
명령 이름 -> "기호 이름 바꾸기"

https://code.visualstudio.com/docs/editor/refactoring#_rename-symbol

js source action - import하지 않은 모듈들을 자동으로 import, 사용하지 않는 모듈 자동으로 제거,
모듈을 이름순으로 정렬 등 다양한 기본 기능이 source action 등에 탑재되어 있습니다.

저는 **cmd + shift + a**로 바인딩 해놨습니다. 명령 이름은 "소스 작업"입니다.



react에서는 해당 jsconfig.json 파일을 레파지토리 최상단(package.json 있는 곳)에 만들어서 옵션을 줘야 해당 source action을 원활하게 사용할 수 있습니다.

```
// jsconfig.json
{
  "compilerOptions": {
    "baseUrl": "./src",
    "checkJs": true,
    "jsx": "react"
  }
}
```

자세한건 아래 링크를 통해 어떤 것들을 할 수 있는지 확인해보세요! 잘나와있습니다.

https://code.visualstudio.com/docs/languages/javascript#_organize-imports