오픈소스에서 터득한 프론트엔드 실무기술들

React, Fluent UI, Vite 같은 오픈소스 코드를
실무에 적용한 경험을 공유합니다.

송현지 2025. 11. 12

발표자 소개

송현지

커리어
이것저것

2년 풀스택
2년 웹스크래핑
잠깐 창업
…

지금은 프론트 개발 2년차..!

귀엽게 봐주세요!

Roseline

Unlock portfolio의 치트키는
“**Show me the hj**”

Portfolio

Blog

오픈소스 기여 경험

summernote
shopify
typeorm
pnpm
...

- **Summernote** - WYSIWYG Editor 커스터마이징으로 오픈소스 활동 시작 (2019.08)
- **Shopify**/graphql-design-tutorial - 튜토리얼 번역 및 개선 (2019.09 ~ 2021.03)
- 오픈소스 스터디 진행 - PR/리뷰 기반 학습 주도 (2019.10 ~ 2019.12)
- Marmelab/React-Admin - 첫 PR 거절 경험, 코드 품질 개선 계기 (2019.11)
- **DefinitelyTyped** - TypeScript 타입 정의 커뮤니티 기여 (2020~)
- **TypeORM**/mongodb-bug-fix - MongoDB 관련 버그 수정 PR 머지 (2022.08)
- **pnpm** - patch edit-dir - 패치 시 디렉토리 편집 기능 추가 (2022.09)
- influxdb-light/write-options - Write 쿼리 옵션 추가 (2023.01)
- React-Native-Progress-Bar - NPM 배포용 프로그레스 바 컴포넌트 개발 (2024.03)
- CodeReview-Action - PR 템플릿 기반 코드리뷰 GitHub Action 공개 (2024.06)
- Open Storybook - VSCode용 Storybook 실행 확장 프로그램 출시 (2025.03)
- React Tree - 커스터마이징 및 json 트리에 특화된 Tree 컴포넌트 (2025. 09)

Index

오픈소스 코드 왜 봐야할까?

오픈소스 코드 어떻게 분석할까?

오픈소스 코드를 실무에 적용하기

Q&A

오픈소스 코드

왜 봐야할까?

CTO님 왈

“ 회사 코드에서 배우지 말고
오픈소스 코드에서 배우세요 ”

메인테이너들의 코드 리뷰

첫번째 이유

세계 최고들로부터
배울 수 있다.

packages/plugin-commands-patching/src/patch.ts Outdated Hide resolved

```
+      title: 'Options',
+      list: [
+        {
+          description: 'set a package directory path for patching',
```

zkochan on Sep 3, 2022 Member

I think this option name is confusing. Maybe something like `--extract-dir` would be better.

The description is also not clear to me. Also it should be capitalized.

roseline124 on Sep 4, 2022 • edited Contributor Author

I get it. How about 'user-dir' or 'patch-dir'?

fzaninotto reviewed on Jul 9, 2020 View reviewed changes

packages/ra-core/src/form/validate.ts Outdated

```
@@ -1,4 +1,5 @@
import lodashMemoize from 'lodash/memoize';
+ import format from 'date-fns/format';
```

fzaninotto on Jul 9, 2020 Member

Browsers have an API for formatting dates (Intl), no need for an external library

👍 1

roseline124 on Jul 10, 2020 • edited Author

> Hi, these validators look great 👍
>
> Thanks for contributing 😊.

두번째 이유

오픈소스를 자유자재로

Typeorm 기여 경험

```
/**
 * @Note this hook's caller must be wrapped with `React.Suspense`
 */
Qodo Gen: Test this function
export function loadModel(params: LoadModelParams): LoadableObject {
  if (modelsCache.has(params)) {
    return modelsCache.get(params)!
  }

  const model = suspend(() => loadModelPromise(params.src), [params.src])

  modelsCache.set(params, model)

  return model
}
```

suspend-react를 사용해 만든 3D 오브젝트 로드 모델 유틸

R3F, drei 등 pmndrs 조직의
오픈소스 코드를 둘러보다가
발견한 suspend-react.

오픈소스를 둘러보다보면
쓸만한 라이브러리들을
발견하게 된다.

```
export const Default: Story = {
  render: () => (
    <TreeWithJson
      treeProps={{ open: true }}
      data={sampleData}
      renderTreeItem={({
        level: _level,
        key,
        value,
        itemType,
        keyPath: _keyPath,
      }) => (
        <div>
          [{itemType}] {key} {typeof value === 'string' ? `: ${value}` : ''}
        </div>
      )}
    />
  ),
};
```

Fluent-ui 중 Tree 컴포넌트의 영감을 받아 만든 TreeWithJson

fluent-ui 전체를 쓰지 않고 tree
컴포넌트를 분석해
나만의 컴포넌트를 만들 수 있었다.

세번째 이유

바이브코딩에 좋은 소스

React에서 RenderLanes로
렌더링 우선순위를 결정하는데,
이때 비트마스크 연산을 통해 복합 상태를
관리해.

지금 내 코드에서도 상태가 복합적으로 적용
되는데, 이걸 비트마스크 연산을 통해
쉽게 관리할 수 있을까?

오픈소스 코드

어떻게 분석할까?

With AI

Entry Point를 따라 흐름 찾기

내가 뭐부터 봐야할까?

원하는 라이브러리의 코드를 git clone한 후…

Tiptap이 Prosemirror와 React를 어떻게 연결하는지 궁금해. 뭐부터 보면 될까?

코드를 타고타고 내려가기

ReactNodeViewRenderer에서
어떻게 ReactComponent를
프로스미러 editor뷰랑 연결하는 거지?
코드의 흐름 좀 보여줘

Tiptap은 Prosemirror와 React를 어떻게 연결하는가?

Toss slash 라이브러리 useOverlay 코드 분석

nestjs가 http module을 생성하는 방법

Tiptap Content Editable

코드가 복잡한 경우

코드에 주석으로 설명을 달아줘

```
/**
 * [단계 1] useEditor 훅: Editor 인스턴스 생성
 *
 * 사용 방법:
 *   const editor = useEditor({ extensions: [StarterKit] })
 *
 * 이 훅은:
 * 1. Editor 인스턴스를 생성합니다 (아직 DOM에 연결되지 않음)
 * 2. EditorInstanceManager를 통해 인스턴스 생명주기를 관리합니다
 * 3. 의존성이 변경되면 Editor를 재생성합니다
 *
 * 실제 DOM 연결은 EditorContent 컴포넌트에서 수행됩니다.
 */
Qodo Gen: Test this function
export function useEditor(options: UseEditorOptions = {}, deps: DependencyList = []): Editor | null {
  const mostRecentOptions = useRef(options)

  mostRecentOptions.current = options

  // Editor 인스턴스 관리자 생성 (한 번만 생성)
  const [instanceManager] = useState(() => new EditorInstanceManager(mostRecentOptions))

  // [단계 1] Editor 인스턴스를 외부 스토어처럼 사용
  // 인스턴스가 생성/제거될 때 컴포넌트가 자동으로 리렌더링됨
  const editor = useSyncExternalStore(
    instanceManager.subscribe,
    instanceManager.getEditor,
    instanceManager.getServerSnapshot,
  )
```

Tiptap은 Prosemirror와 React를 어떻게 연결하는가?

Toss slash 라이브러리 useOverlay 코드 분석

nestjs가 http module을 생성하는 방법

코드의 원리를 파악하고 싶은 경우

코드의 핵심만 간추려서 의사코드를 짜줘

```
export class Editor extends EventEmitter<EditorEvents> {

  /**
   * Creates a ProseMirror view.
   */
  Qodo Gen: Test this method
  private createView(element: NonNullable<EditorOptions
    this.editorView = new EditorView(element, {
      ...this.options.editorProps,
      attributes: {
        // add `role="textbox"` to the editor element
        role: 'textbox',
        ...this.options.editorProps?.attributes,
      },
      dispatchTransaction: this.dispatchTransaction.bin
      state: this.editorState,
    })

    // `editor.view` is not yet available at this time.
    // Therefore we will add all plugins and node views
    const newState = this.state.reconfigure({
      plugins: this.extensionManager.plugins,
    })

    this.view.updateState(newState)

    this.createNodeViews()
    this.prependClass()
    this.injectCSS()

    // Let's store the editor instance in the DOM eleme
    // So we'll have access to it for tests.
    // @ts-ignore
    const dom = this.view.dom as TiptapEditorHTMLElemen

    dom.editor = this
  }
```

원래 코드

```
class Editor {
  create() {
    // 1. ProseMirror Schema, State 생성
    this.schema = this.extensionManager.schema
    this.editorState = EditorState.create({ schema: this.sche

    // 2. ProseMirror EditorView 생성
    // options.element에 DOM을 생성하고 연결
    this.createView(this.options.element)
  }

  createView(element) {
    // 핵심: ProseMirror EditorView 생성
    // 이때 element 내부에 ProseMirror DOM이 생성됨
    this.editorView = new EditorView(element, {
      state: this.editorState,
      dispatchTransaction: this.dispatchTransaction,
      // NodeView는 나중에 추가됨
    })
  }

  createNodeViews() {
    // 모든 extension의 NodeViewRenderer를 실행
    // 각 NodeView는 React 컴포넌트를 렌더링함
    const nodeViews = {}

    this.extensions.forEach(ext => {
      if (ext.nodeViewRenderer) {
        // ReactNodeViewRenderer가 실행됨
        nodeViews[ext.name] = ext.nodeViewRenderer({
          editor: this,
          node,
          // ...
        })
      }
    })

    // EditorView에 NodeView 등록
    this.editorView.setProps({ nodeViews })
  }
}
```

Review next f

의사코드

Tiptap은 Prosemirror와 React를 어떻게 연결하는가?

Toss slash 라이브러리 useOverlay 코드 분석

nestjs가 http module을 생성하는 방법

오픈소스 코드에서 배우기

React

인라인 최적화

```
// This is a constructor function, rather than a POJO constructor, still
// please ensure we do the following:
// 1) Nobody should add any instance methods on this. Instance methods can be
//    more difficult to predict when they get optimized and they are almost
//    never inlined properly in static compilers.
// 2) Nobody should rely on `instanceof Fiber` for type testing. We should
//    always know when it is a fiber.
// 3) We might want to experiment with using numeric keys since they are easier
//    to optimize in a non-JIT environment.
// 4) We can easily go from a constructor to a createFiber object literal if that
//    is faster.
// 5) It should be easy to port this to a C struct and keep a C implementation
//    compatible.
Qodo Gen: Test this function
function createFiberImplClass(      Sebastian "Sebbie" Silbermann, 4 months ago • [F
  tag: WorkTag,
  pendingProps: mixed,
  key: null | string,
  mode: TypeOfMode,
): Fiber {
  // $FlowFixMe[invalid-constructor]: the shapes are exact here but Flow doesn't lik
  return new FiberNode(tag, pendingProps, key, mode);
}
```

https://roseline.oopy.io/dev/optimization-in-react-library

오픈소스 코드에서 배우기

React

인라인 최적화

1. this에 instance method 추가하지 않기
2. 상태 업데이트 로직은 외부 함수로 분리
3. 반복 호출되는 코드는, prototype 체인 접근 최소화

https://roseline.oopy.io/dev/optimization-in-react-library

오픈소스 코드에서 배우기

React

비트마스크 플래그로 복합 상태 처리하기

```
export const getState = ({
  timeSlot,
  todayStartRecord,
  todayRecords,
}: GetStateProps) => {
  // 시작 기록 자체가 없으면 전역적으로 시작 전 상태
  if (!todayStartRecord) return RecordFlags.NO_RECORD;

  // 특수 탭 처리
  if (timeSlot === null) return RecordFlags.PROGRESS;
  if (timeSlot === "goal") return RecordFlags.GOAL;

  // 아침/점심/저녁 공통 처리
  const slotFlag = TimeSlotFlagMap[timeSlot];
  const record = todayRecords?.find((r) => r.timeSlot === timeSlot);

  // 해당 슬롯 기록이 없으면 슬롯 + NO_RECORD
  if (!record) return setFlag(RecordFlags.NO_RECORD, slotFlag);

  // 성공 여부가 명확하면 슬롯 + (성공/실패)
  if (typeof record.isSuccess === "boolean") {
    return setFlag(
      slotFlag,
      record.isSuccess ? RecordFlags.NOT_SMOKED : RecordFlags.SMOKED
    );
  }

  // 그 외에는 진행 중으로 본다
  return RecordFlags.PROGRESS;
};
```

https://roseline.oopy.io/dev/optimization-in-react-library

오픈소스 코드에서 배우기

React

비트마스크 플래그로 복합 상태 처리하기

1. 상태가 독립적이면서 동시에 켜질 수 있는 경우
2. 복잡한 if문을 비트 연산으로 단순화
3. enum보다 빠른 비트 단위 표현

https://roseline.oopy.io/dev/optimization-in-react-library

오픈소스 코드에서 배우기

Vite

Proxy로 하위호환성 지키며 기능 확장하기

```
// Proxy를 사용해서 router의 모든 메서드를 가로채서 beforeUnload 체크를 적용
const guardedRouter = new Proxy(router, {
  get(target, prop) {
    const originalMethod = target[prop as keyof typeof target];

    // 네비게이션 관련 메서드들만 가로채기
    if (
      typeof originalMethod === "function" &&
      ["push", "replace", "back", "forward", "refresh"].includes(
        prop as string
      )
    ) {
      return async (...args: any[]) => {
        if (isDirty) {
          const ok = await beforeUnloadCallback();
          if (!ok) return;
        }

        // 프로그래밍 방식 네비게이션임을 표시
        isProgrammaticNavigationRef.current = true;

        // back() 메서드인 경우 prevUrl 사용
        if (prop === "back" && prevUrl) {
          const result = (target.push as any).apply(target, [prevUrl]);

          // URL 업데이트
          setTimeout(() => {
            currentUrlRef.current = window.location.href;
          }, 0);
```

https://roseline.oopy.io/dev/what-i-learned-in-vite

오픈소스 코드에서 배우기

Vite

렉싱과 State를 통해
빠르게 문자열 분석하기

```
Hyunji Song, last week | 1 author (Hyunji Song) | Qodo Gen: Test this class
class StreamingJsonParser {
  private mode: "none" | "page" | "category" | "key" | "value" | "dataType" =
    "none";
  private buffer = "";
  private result = "";

  Qodo Gen: Test this method
  feed(chunk: string): string {
    if (!chunk) return "";      Hyunji Song, last week • fix: 버그 수정 2 (#60) …

    const normalized = chunk
      .replace(/\\n/g, "\n")
      .replace(/\\t/g, "  ")
      .replace(/\\"/g, '"')
      .replace(/\\\\/g, "\\");

    let newOutput = "";

    for (let i = 0; i < normalized.length; i++) {
      const char = normalized[i];
      this.buffer += char;
```

https://roseline.oopy.io/dev/what-i-learned-in-vite

오픈소스 코드에서 배우기

Fluent-ui

Tree 컴포넌트의 상태 관리 방법

https://roseline.oopy.io/dev/react-tree-open-source

```
Qodo Gen: Test this function
export const TreeWithJson = ({
  treeProps,
  ...renderTreeProps
}: TreeWithJson) => {
  return (
    <Tree {...treeProps}>{renderTree({ treeProps, ...renderTreeProps })}</Tree>
  );
};

You, 2 months ago | 2 authors (Hyunji Song and one other)
interface getTreeItemLayoutPropsParams {
  level: number;
  key: string;
  value: TreeData;
  itemType: "leaf" | "branch";
}

Hyunji Song, 2 months ago | 2 authors (You and one other)
interface RenderTreeItemProps {
  level: number;
  key: string;
  value: TreeData;
  itemType: "leaf" | "branch";
  keyPath: string[];
}
```

감사합니다

Q&A