UI 빌더를 지탱하는 레고 블록 같은 아키텍처 만들기

김훈민

NAVER ETECH / HomeBuilder

NAVER DEVIEW 2023

NAVER DEVIEW 2023

NAVER ETECH 를 소개합니다.

NAVER DEVIEW 2023

NAVER ETECH 를 소개합니다.

CONTENTS

01. Intro - HomeBuilder 소개

02. 왜 레고일까?
03. 우리의 레고를 찾아서

04. 문제 해결하기
- 너무 깐간한 코어 로직
- 너무 무거운 프로덕트 뷰
- 너무 긴 커스텀 동선

05. 플러그인의 발견
06. 다르게 풀어야 하는 문제

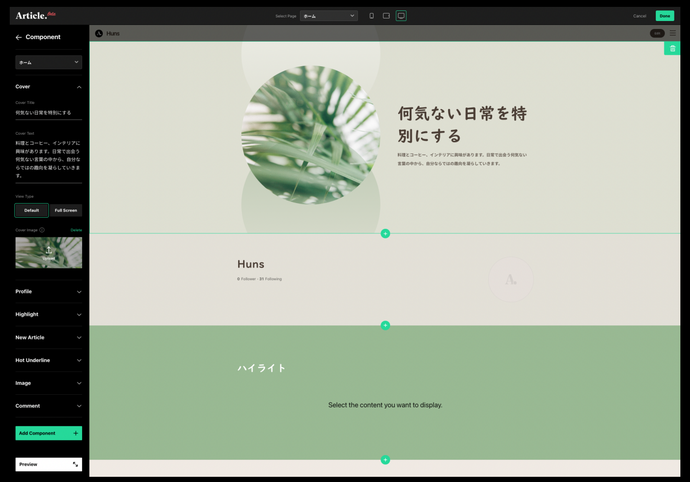
07. 정리하기

01. Intro

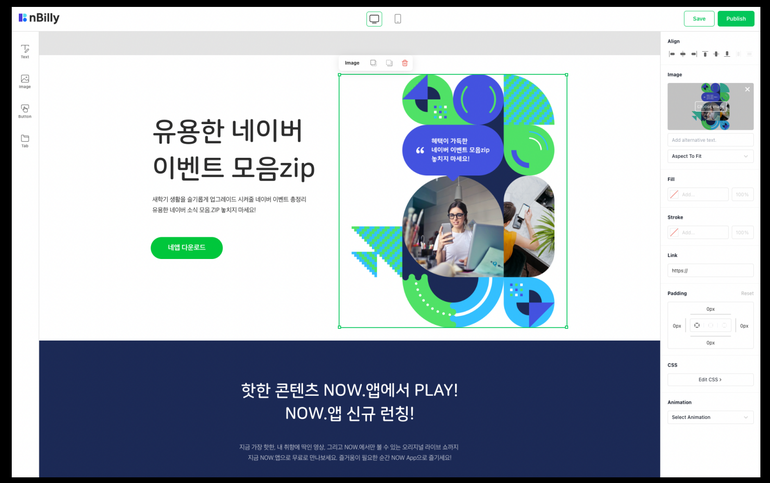
HomeBuilder 히스토리

NAVER
DEVIEW
2023

SmartEditor의 웹 콘텐츠 제작 기술을 UI 빌더로 확장



SDK



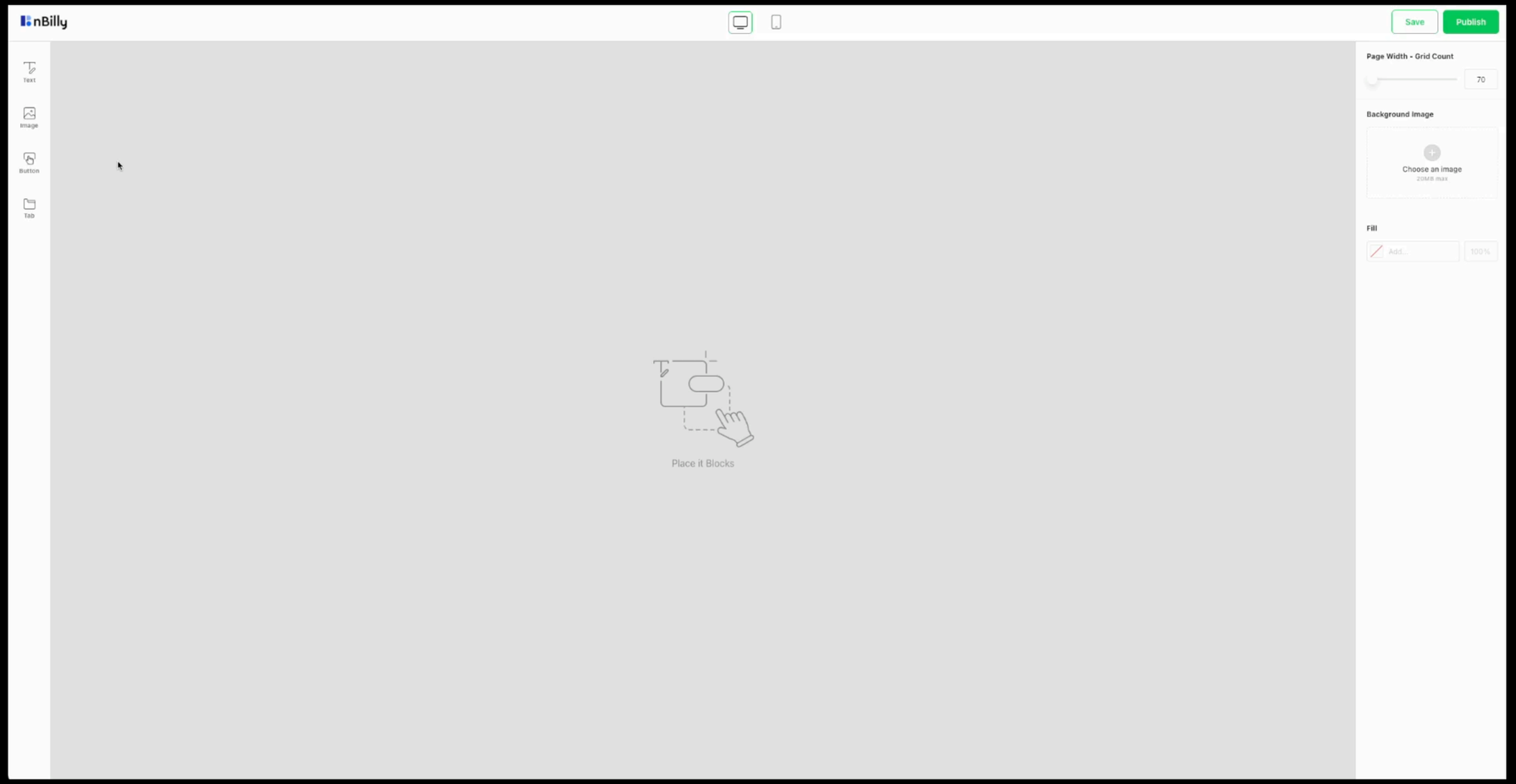
HomeBuilder 1.0
Article 서비스 탑재

HomeBuilder 2.0
자체 개발 프로덕트

nBilly

HomeBuilder 2.0의 새로운 이름

키노트를 만들 듯 웹 페이지를 만들어요



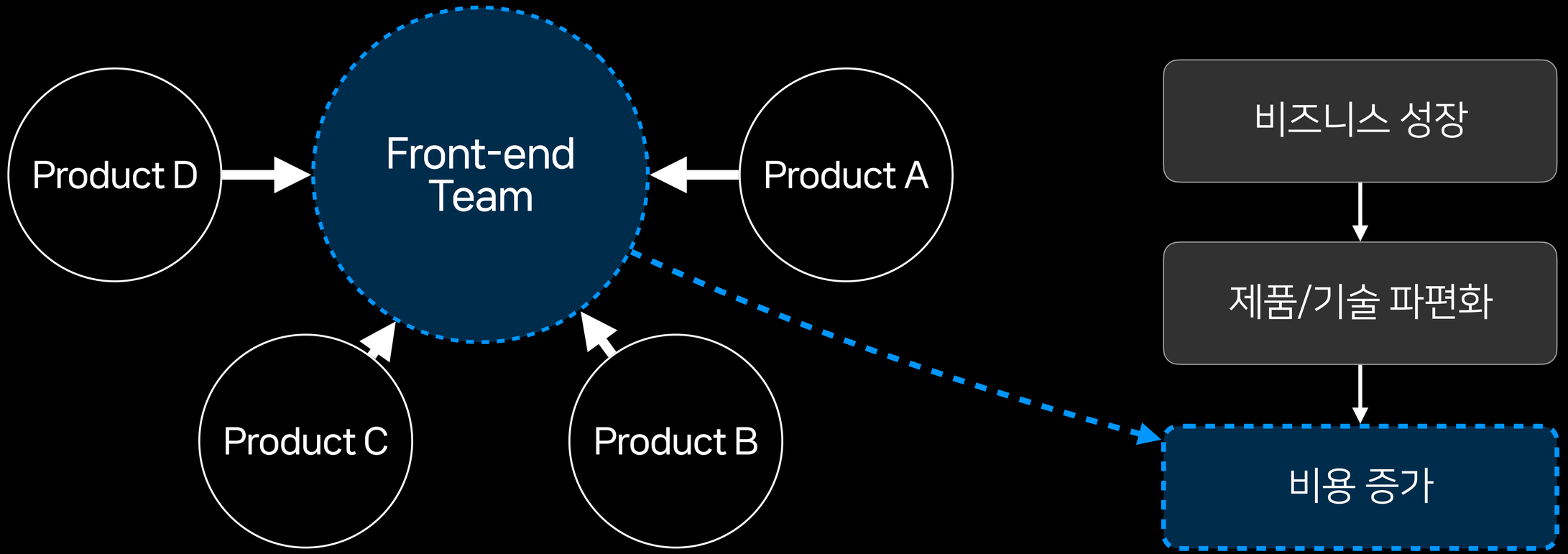
상상하기

내 마음대로
조립할 수 있는
레고 블록이 있다면?

02. 왜 레고일까?

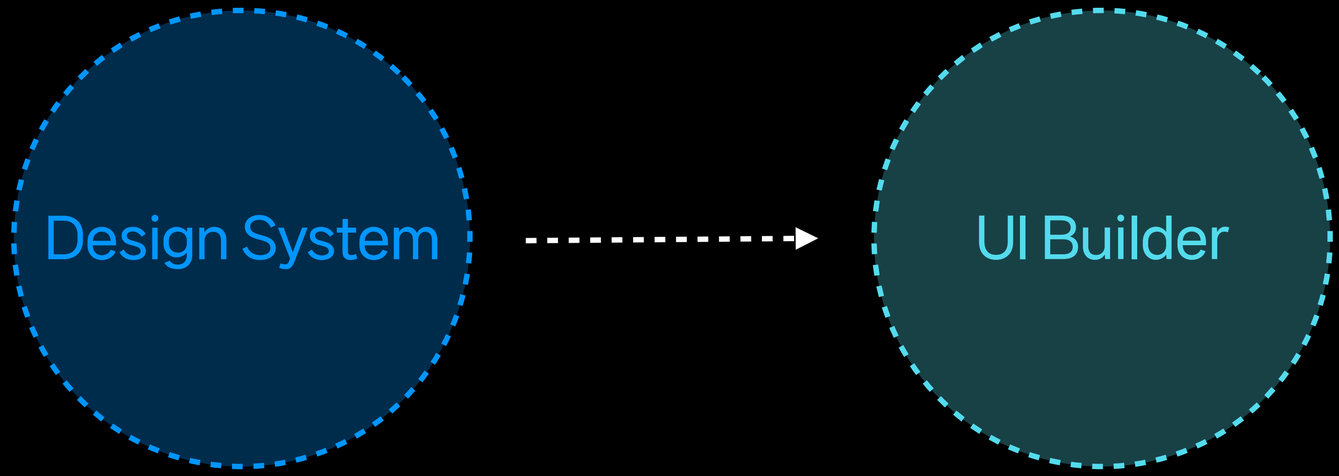
비즈니스 성장과 기술 파편화

체계화 속도 보다 성장 속도가 빠르면 흔하게 겪을 수 있는 일



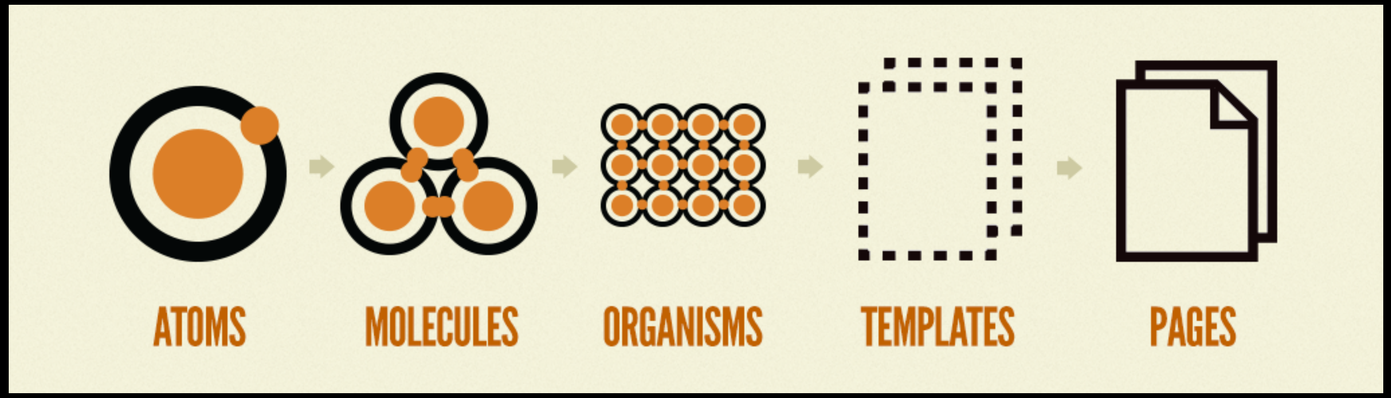
효율 높이기

디자인 시스템을 이용한
기술/커뮤니케이션 표준화는
UI 빌더로 진화 중



계층 결합 = 레고

잘 정의된 빌딩 블록과 계층이 있다면
작은 것을 결합해 더 큰 것을 만들 수 있다는 믿음



https://atomicdesign.bradfrost.com/chapter-2/

UI 빌더를 지탱하는 레고 같은 아키텍처

그래서 레고를 만들었군요?

아니요, 그런 건 세상에 '아직' 없습니다

하지만 비슷해 질 수는 있습니다

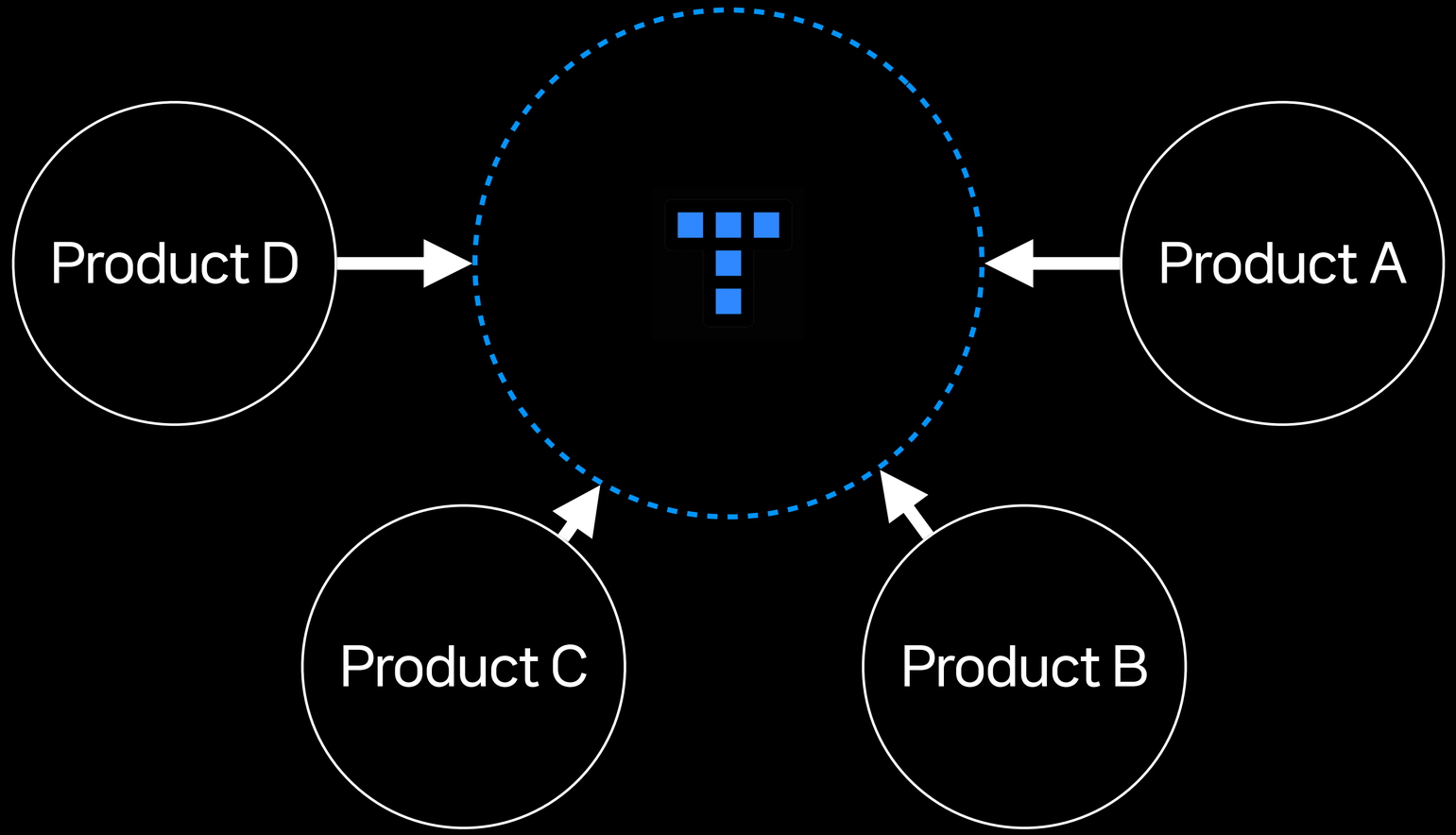
고민하고 시도하며 꿈을 좇는 이야기입니다

03. 우리의 레고를 찾아서

개발 대리인 → 기술 제공자

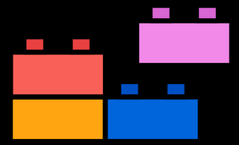
NAVER
DEVIEW
2023

팀과 기술이 지속 성장할 수 있는 기반 갖추기



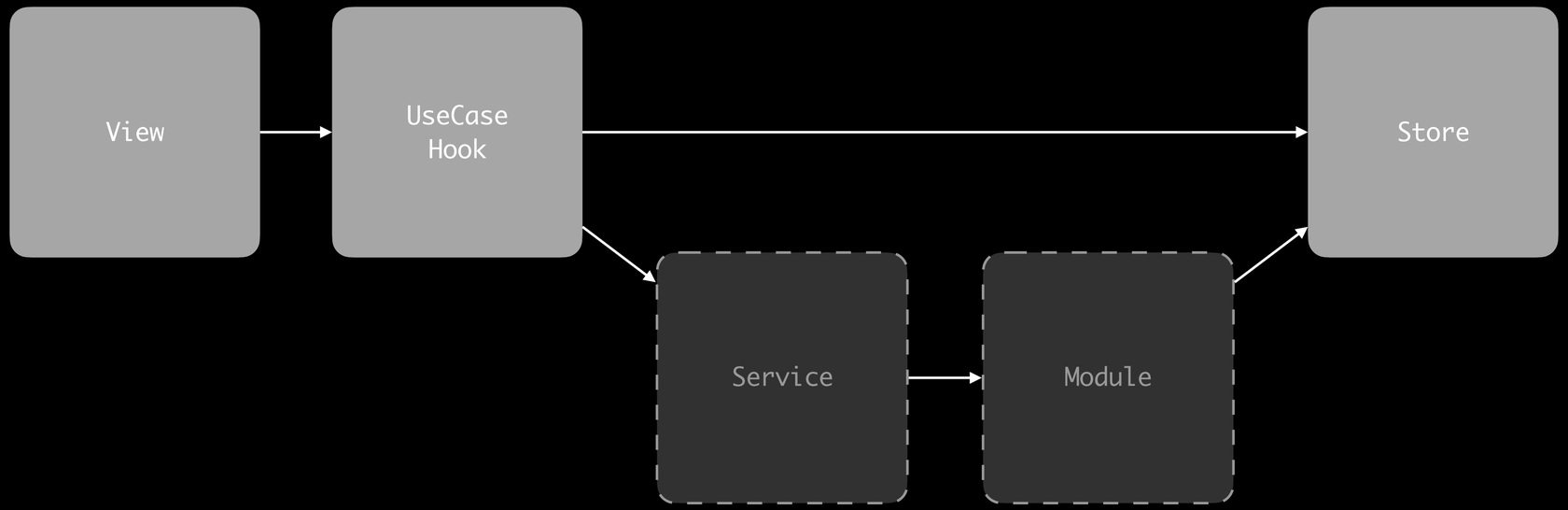
우리가 가진 UI 빌딩 기술을

레고 같은 SDK로 만들자



데이터 플로우

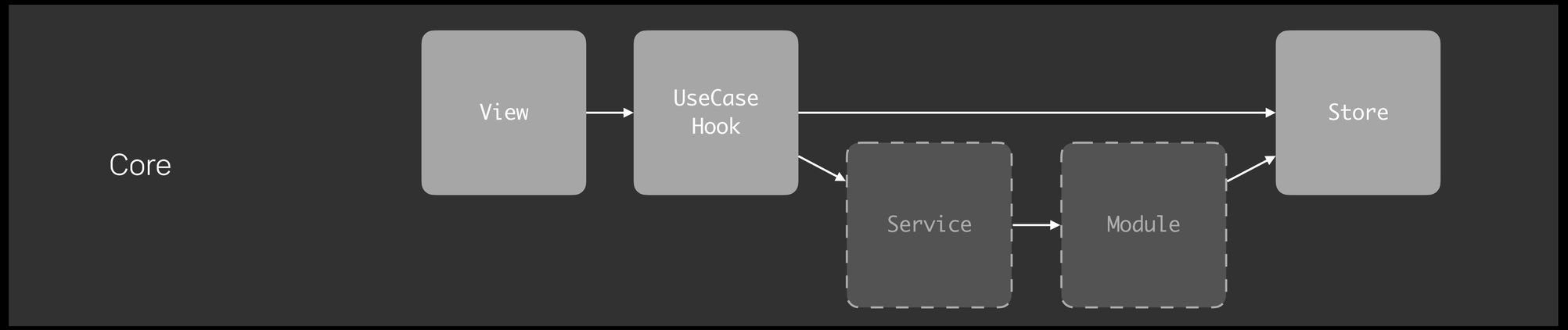
Redux 아키텍처 일부 커스텀
Undo/Redo 구현 유리



2-계층 아키텍처

NAVER
DEVIEW
2023

하나, 기본 재료를 Core에 배치한다

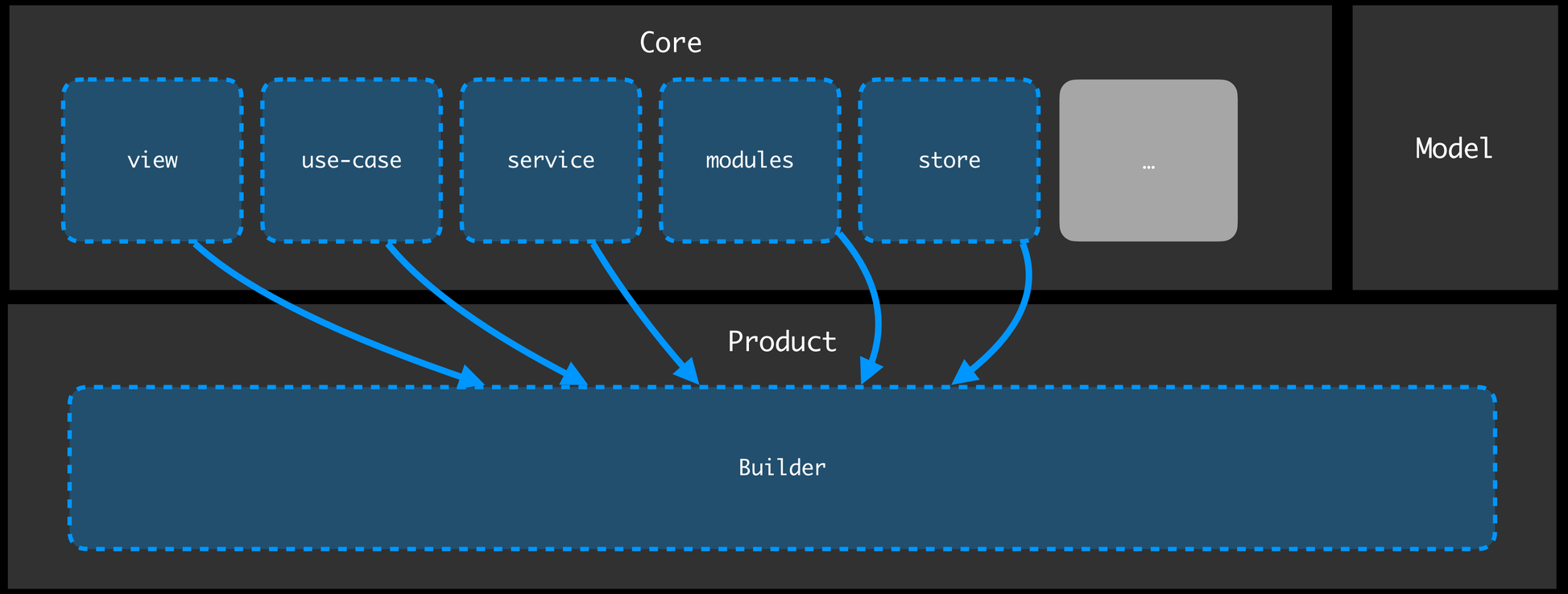


2-계층 아키텍처

NAVER
DEVIEW
2023

하나, 기본 재료를 Core에 배치한다

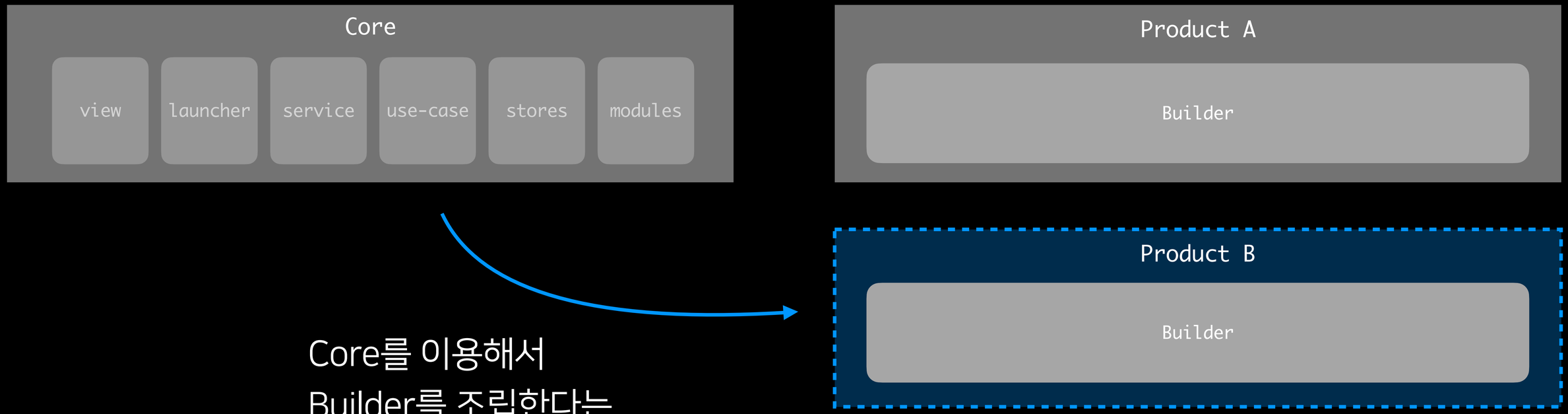
둘, Core와 Model의 빌딩 블록을 이용해 Product에서 Builder 조립한다



2-계층 아키텍처

NAVER
DEVIEW
2023

셋, 새로운 Product 요구는 Core를 이용해서 개발한다



Core를 이용해서
Builder를 조립한다는
평범한 아이디어

누구에게나 계획이 있다

NAVER
DEVIEW
2023

우리에게도 멋진 계획이 있었다
이 문제를 만나기 0.17345초 전까지 😐

04. 문제 해결하기

증상 → 진단 → 처방

문제 #1

너무 깐깐한 코어 로직

NAVER
DEVIEW
2023

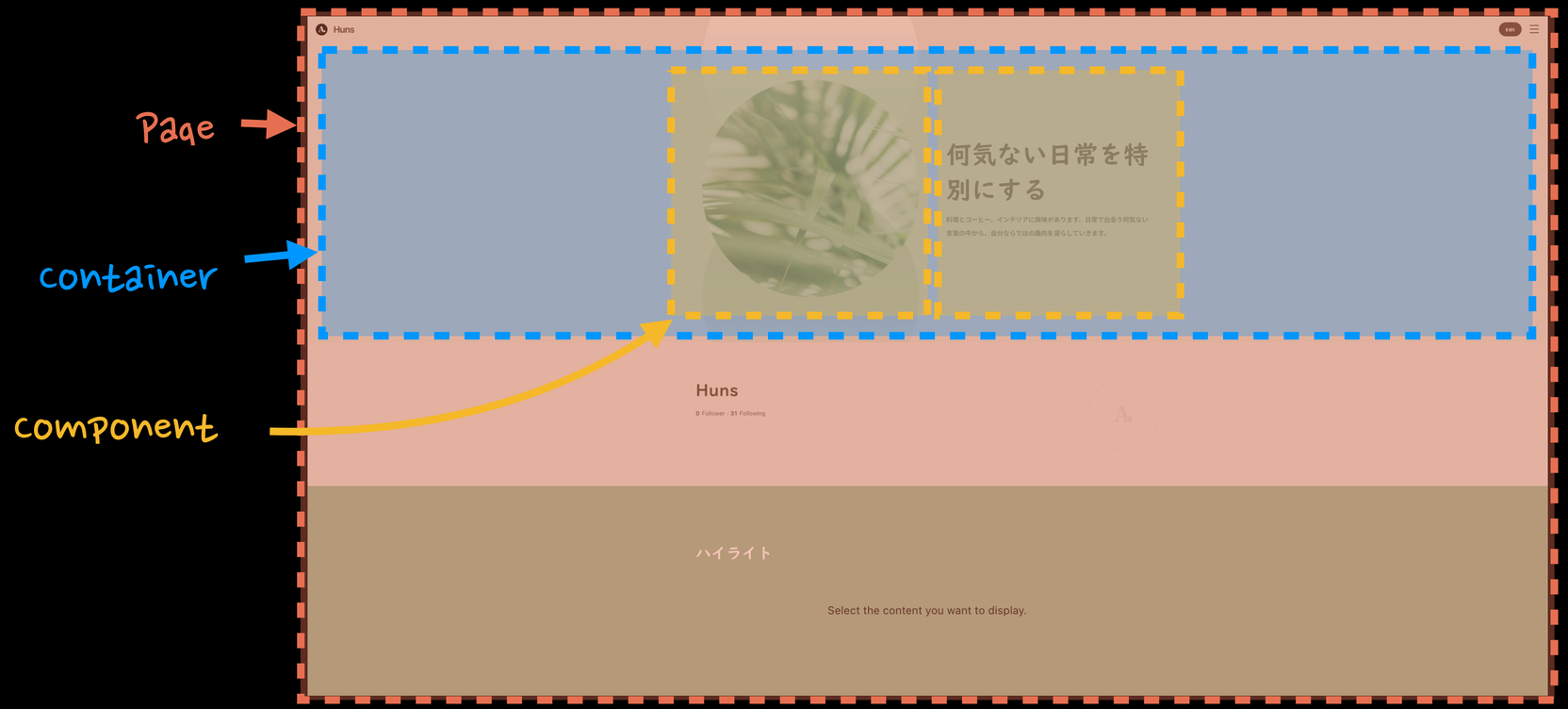
😷 에디터의 스펙을 결정하는 Hook이 코어에 있다!?

문제 #1

너무 깐깐한 코어 로직

NAVER
DEVIEW
2023

😷 에디터의 스펙을 결정하는 Hook이 코어에 있다!?

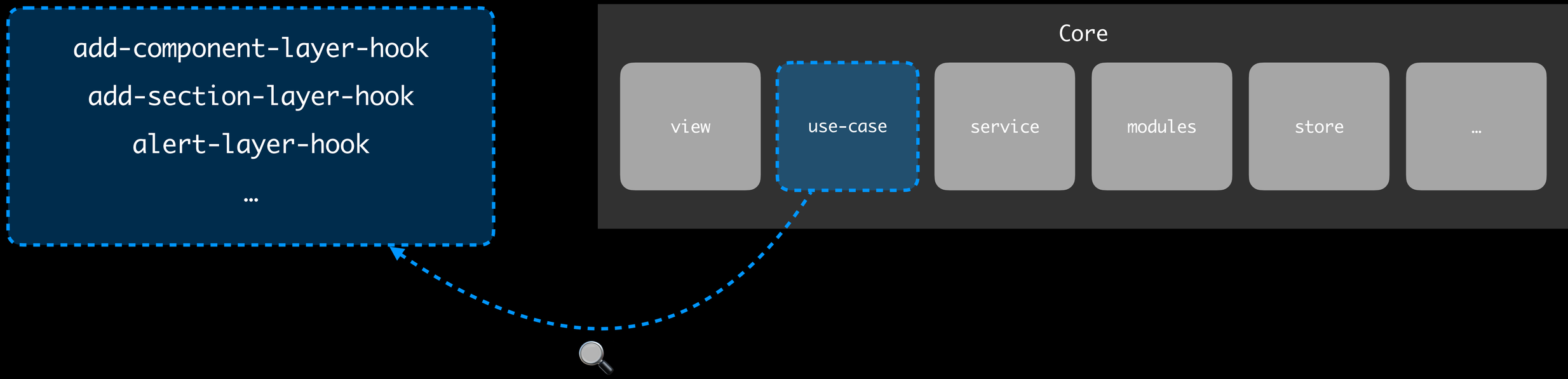


문제 #1

너무 깐깐한 코어 로직

NAVER
DEVIEW
2023

😷 에디터의 스펙을 결정하는 Hook이 코어에 있다!?



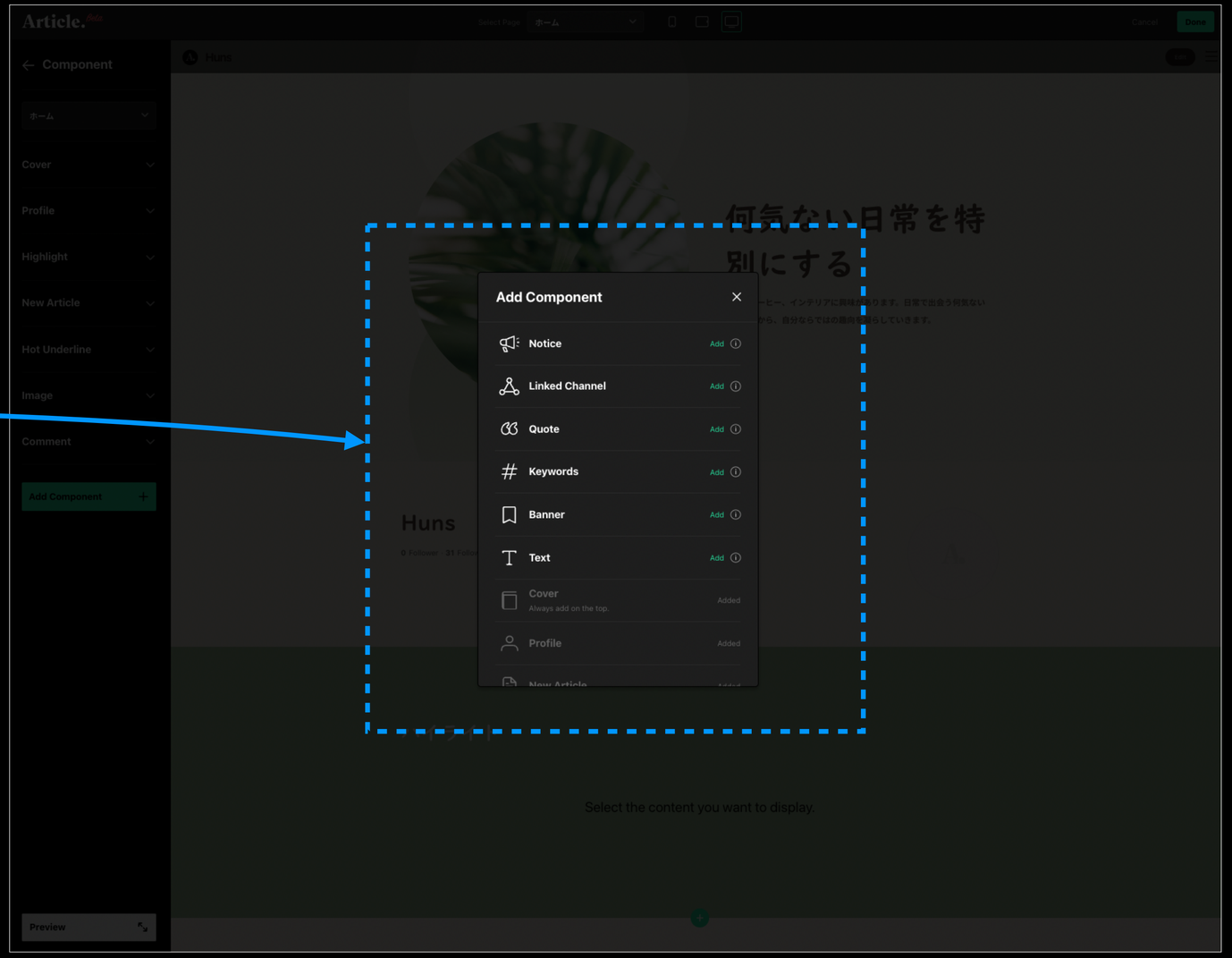
그때 우리는 이렇게 생각했다

Headless UI 같은 고급스러운 느낌!?

작아지는 프로덕트 커스텀 범위

프로덕트 스펙을 팝업 UI로 가정
만약 팝업이 아니라면?

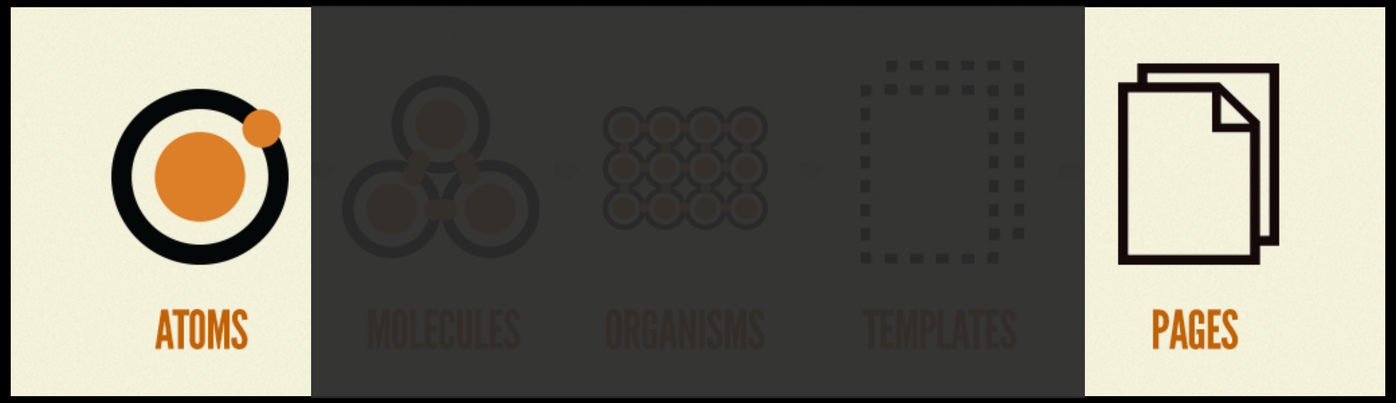
add-component-layer-hook



Core와 Product의 책임 혼선

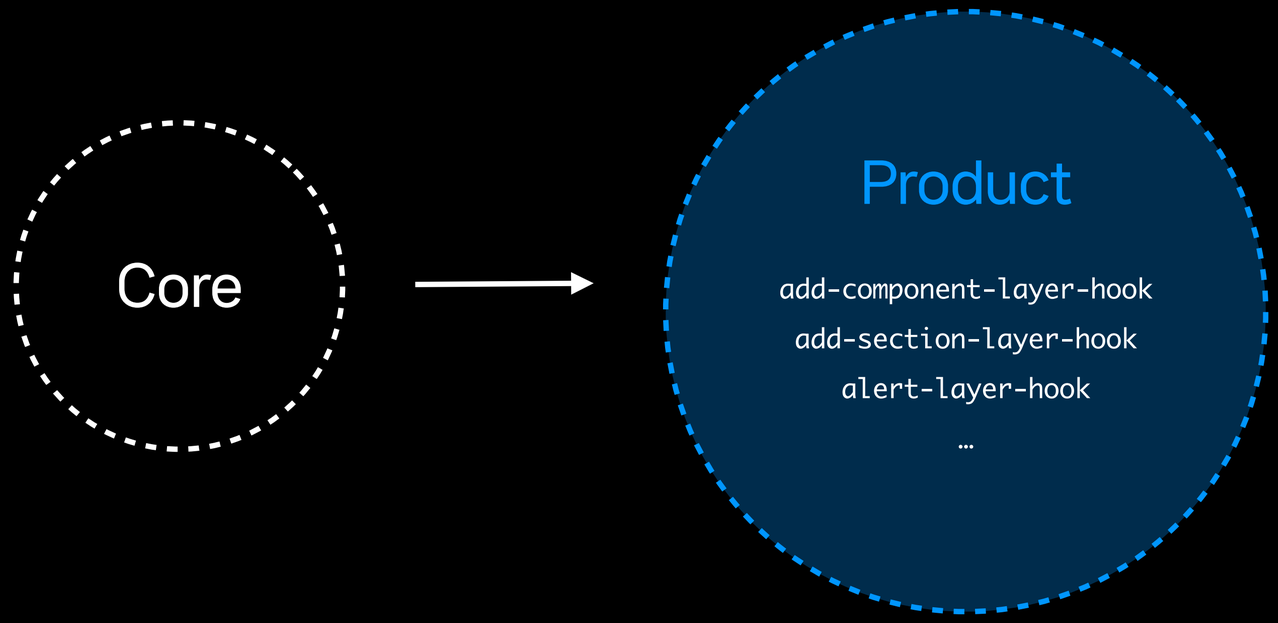
Core는 계층의 Foundation, Product는 제품 조립 현장
Foundation이 제품의 스펙을 결정한다면?

이런 상황과 비슷해요
1. 계층에 Atom과 Pages만 있다
2. Atom이 Page의 모습을 결정한다



Product로 다 넘기면 될까!?

2-계층 아키텍처는 '책임을 두 곳에 분산하겠다'는 의지
Core가 작아질수록 반대로 커지는 Product



진단 #1

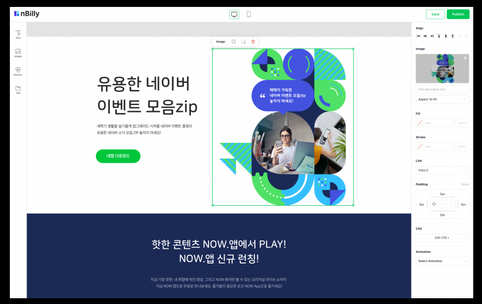
부족한 계층과 불명확한 책임

처방 #1

계층을 나누고 책임을 재설계 하라

파운데이션은 어떤 과정을 거쳐 서비스 애플리케이션이 되는가?

SDK



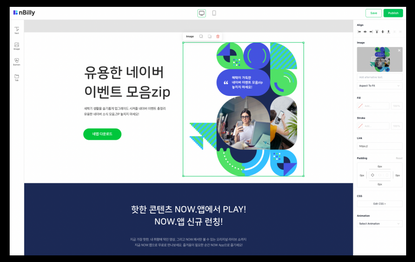
에디터/ 뷰어

서비스
애플리케이션

제품 공정을 토대로 초안을 그리기

계층과 공정 모두 위로 올라갈수록 추상화 수준이 높아진다

서비스
애플리케이션



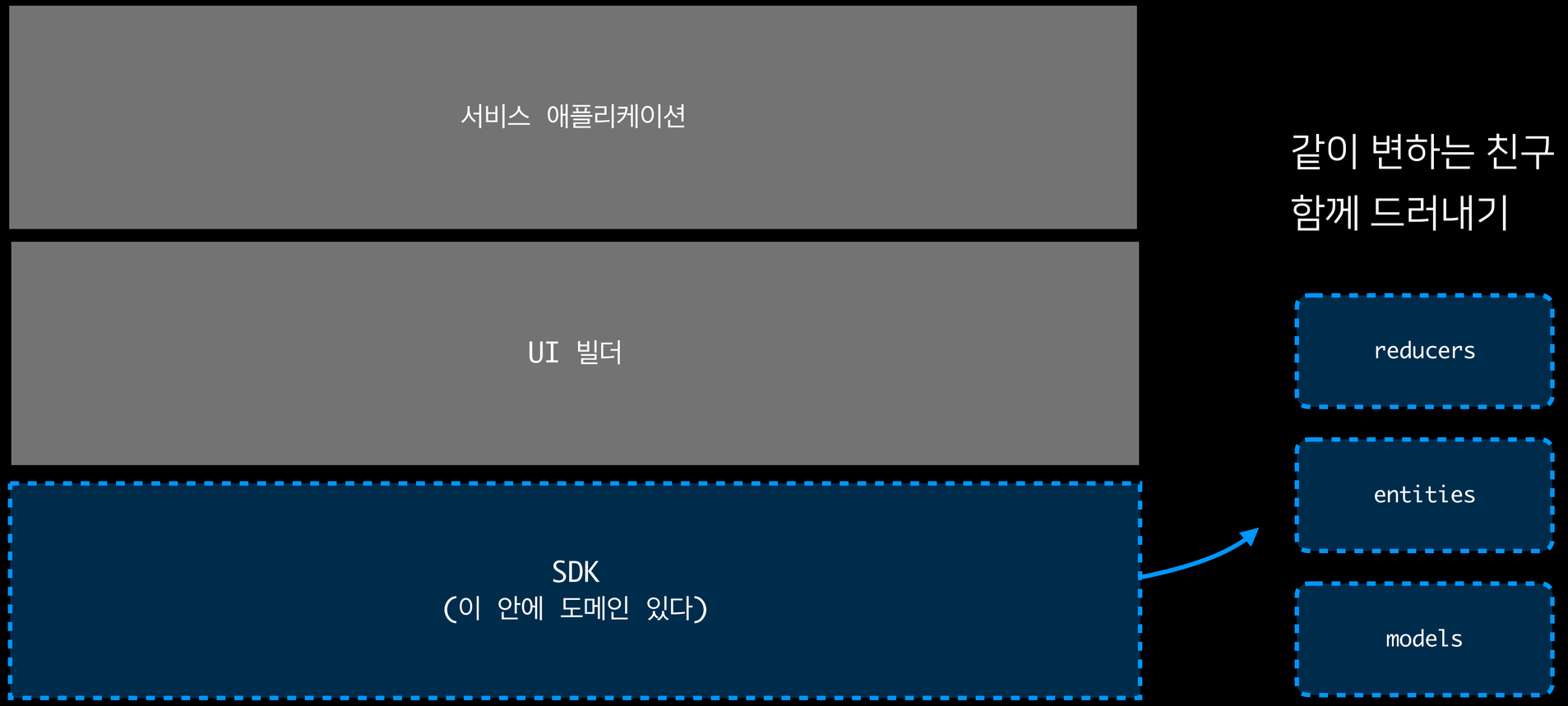
SDK

서비스 애플리케이션

UI 빌더

SDK(도메인, 유틸리티…)

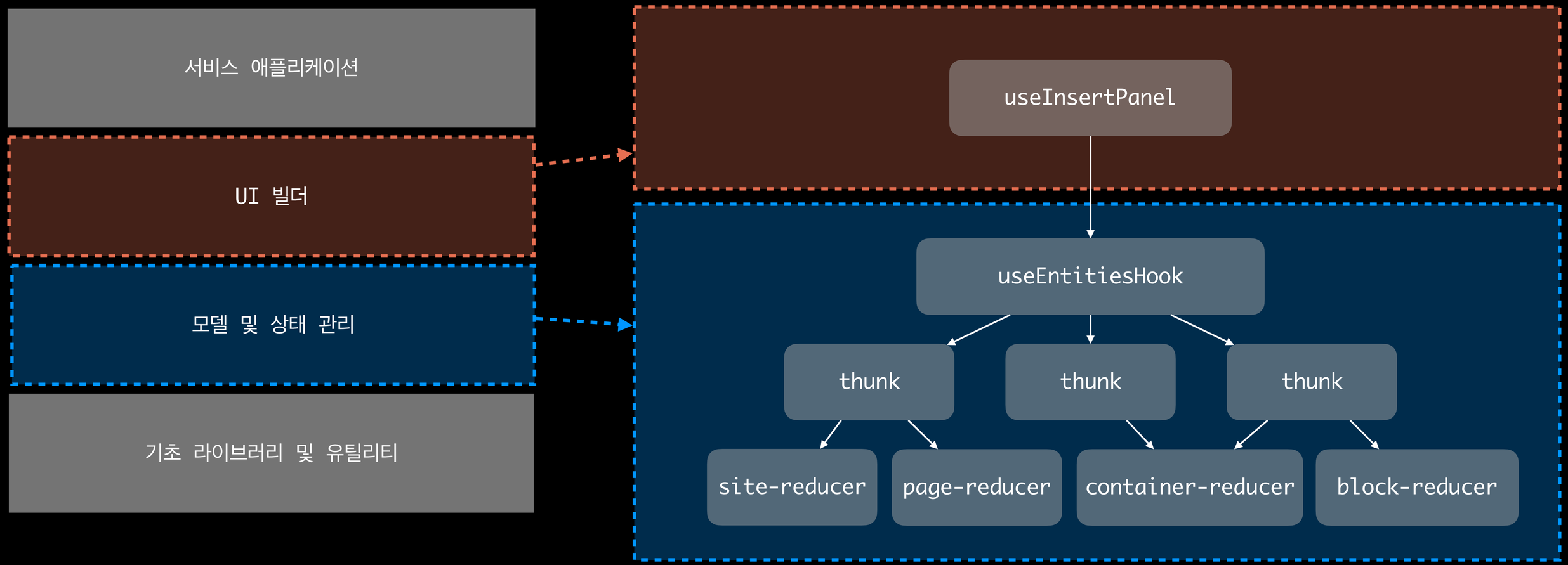
숨어 있는 도메인을 찾아서 계층 분리하기




서비스 애플리케이션
UI 빌더
모델 및 상태 관리
reducers
entities
models
기초 라이브러리 및 유틸리티

계층을 나누고 책임을 재설계 한 후

무엇이 달라졌나!?

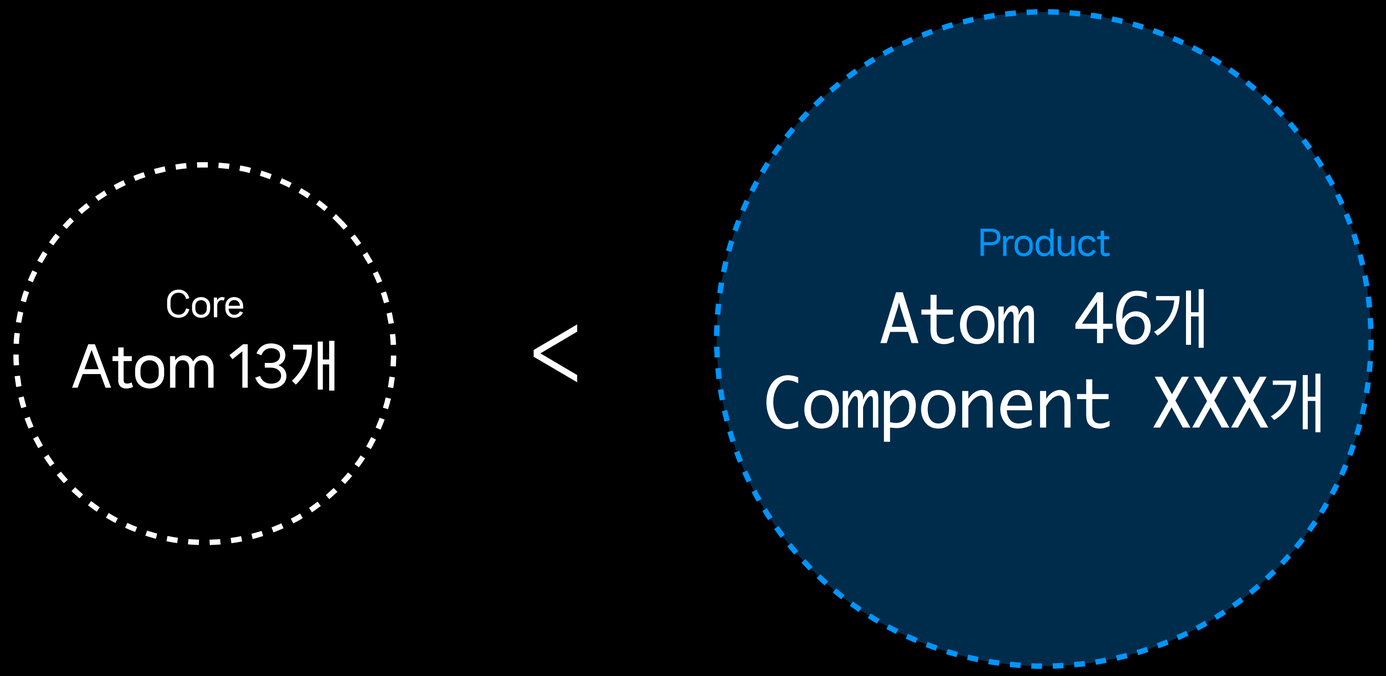


문제 #2

너무 무거운 프로덕트 뷰

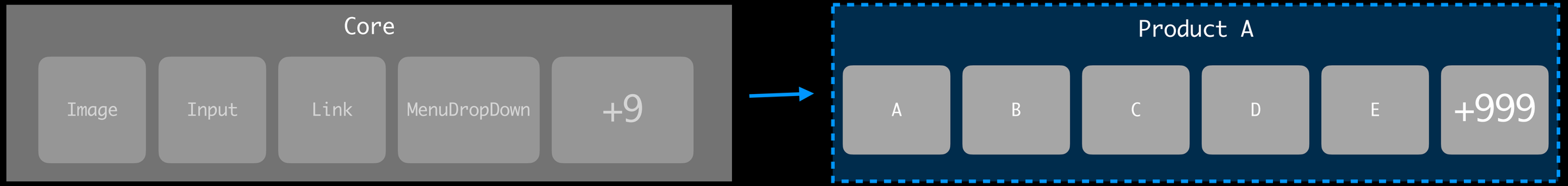
NAVER
DEVIEW
2023

😷 프로덕트 계층에서 만들어야 하는 UI 컴포넌트가 왜 이렇게 많죠?



부품으로 사용할 수 있는 범위가 좁다

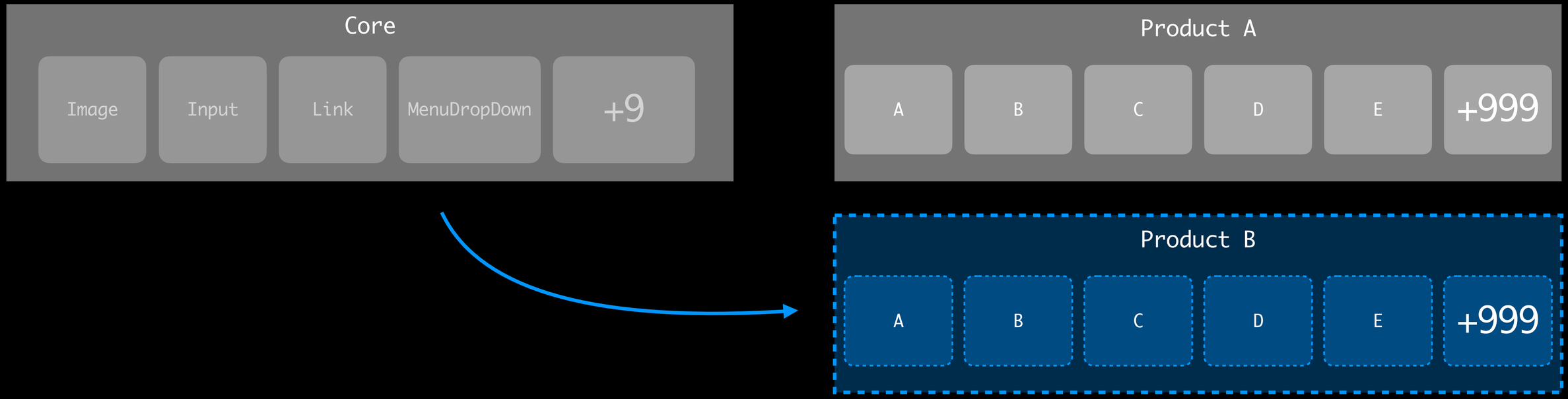
코어가 부품을 주지 않는다!?



부품으로 사용할 수 있는 범위가 좁다

코어가 부품을 주지 않는다!?

그럼 또 만들어야죠... 🤷‍♂️



진단 #2

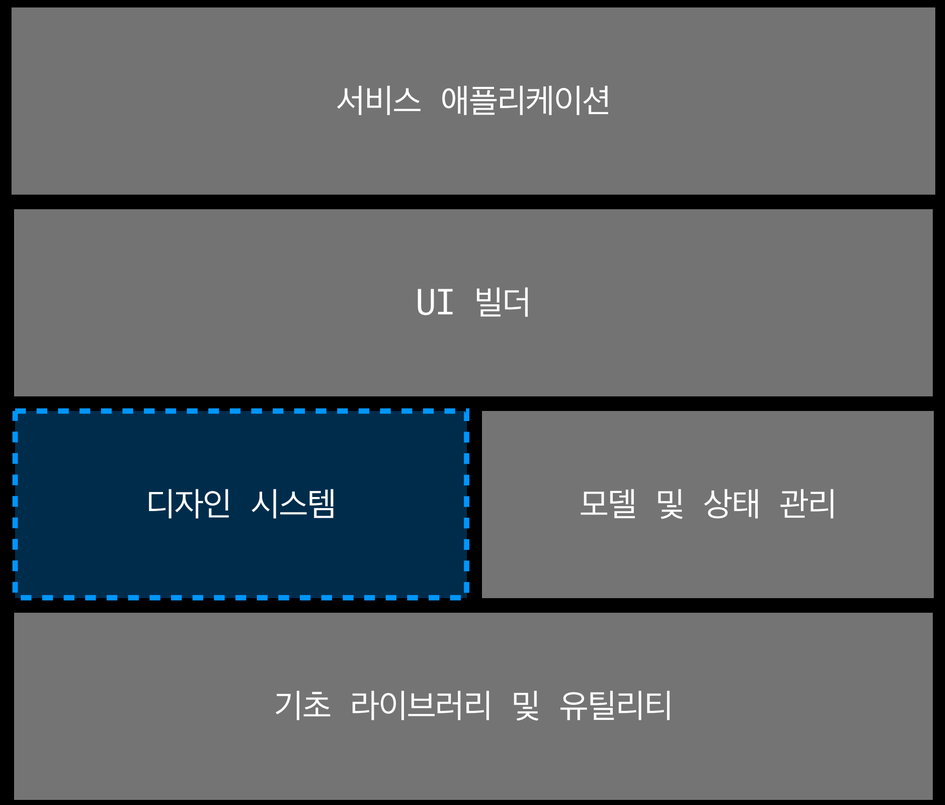
공용 UI 체계 부재

처방 #2

디자인 시스템을 갖자

HomeBuilder 1.0 → 서비스 지원 제품
HomeBuilder 2.0 → 자체 프로덕트

자체 프로덕트를 만들 때는
UI 체계를 갖추자

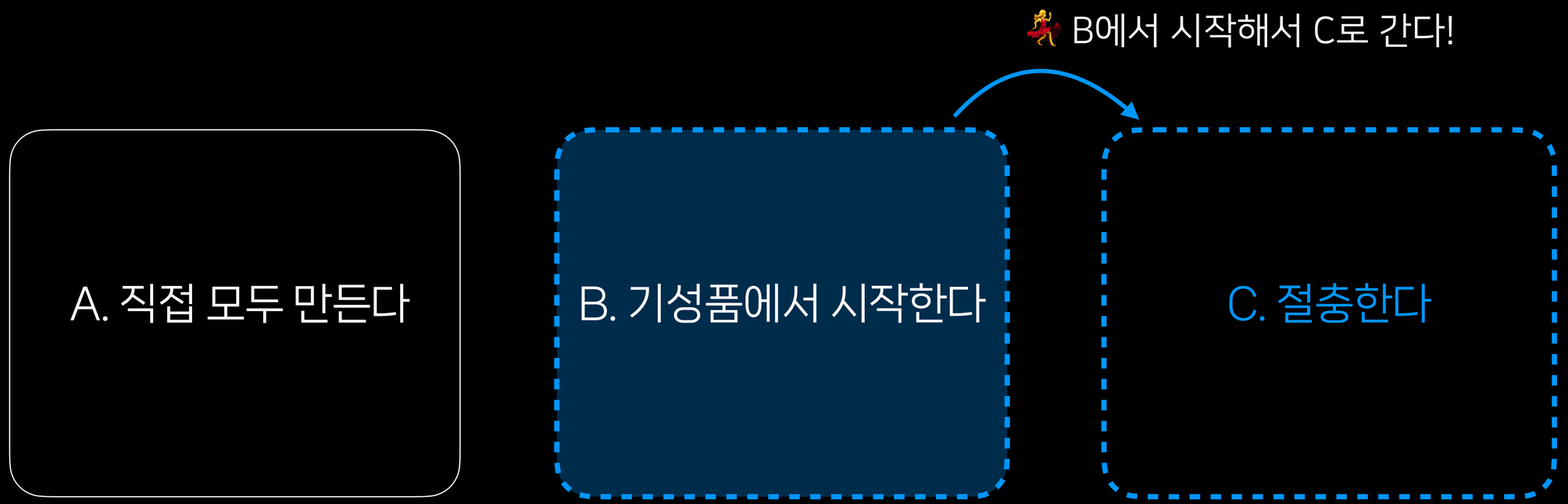


디자인 시스템을 갖고 싶다

그런데 우리는 디자이너가 없네? 🤔

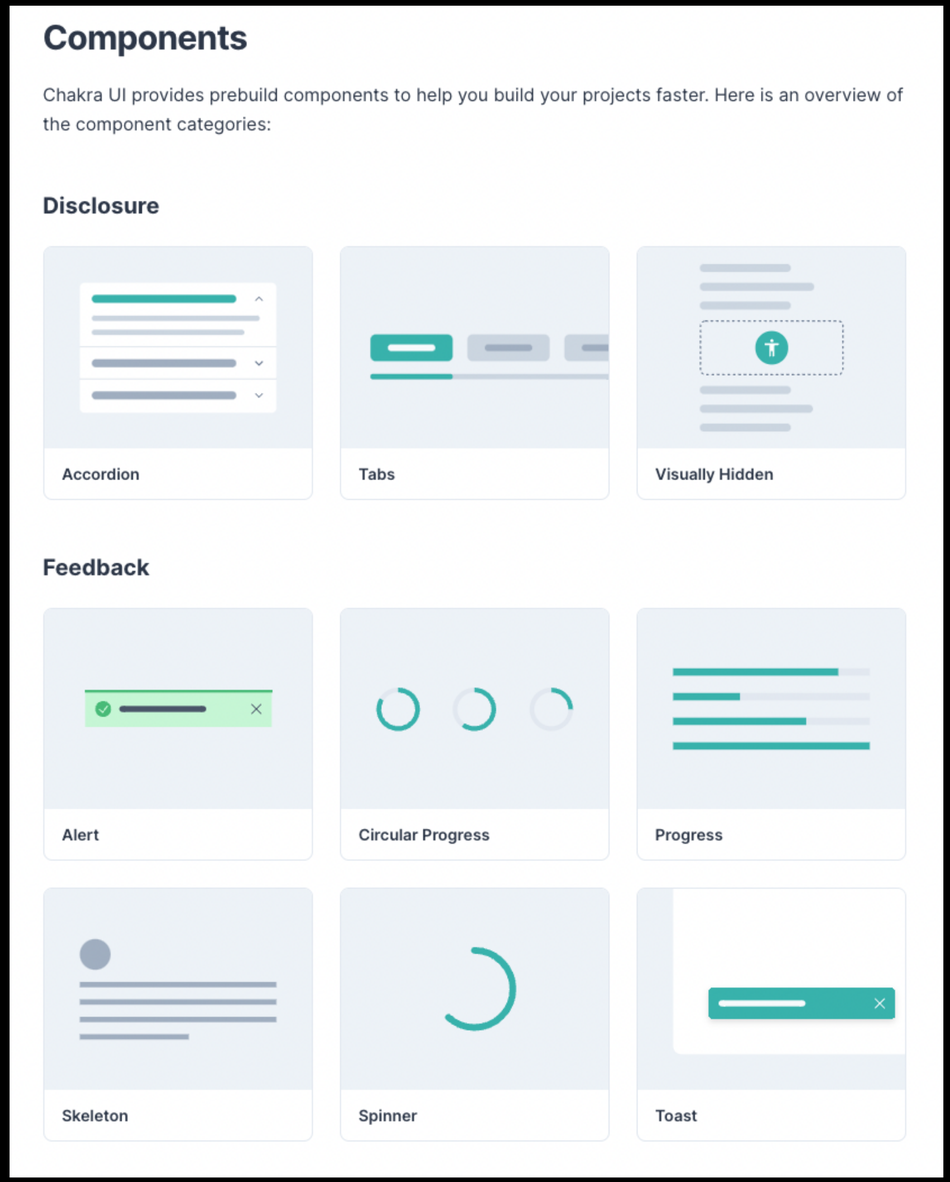
디자이너 있는 척 일하기

전담 디자이너 없던 HomeBuilder 2.0 초창기
디자이너 없지만 있는 척 할 순 있다



Chakra-ui, 너 나하고 일 하나 같이 하자

컴포넌트가 풍부할 것 → 61종



Chakra-ui, 너 나하고 일 하나 같이 하자

컴포넌트가 풍부할 것 → 61종

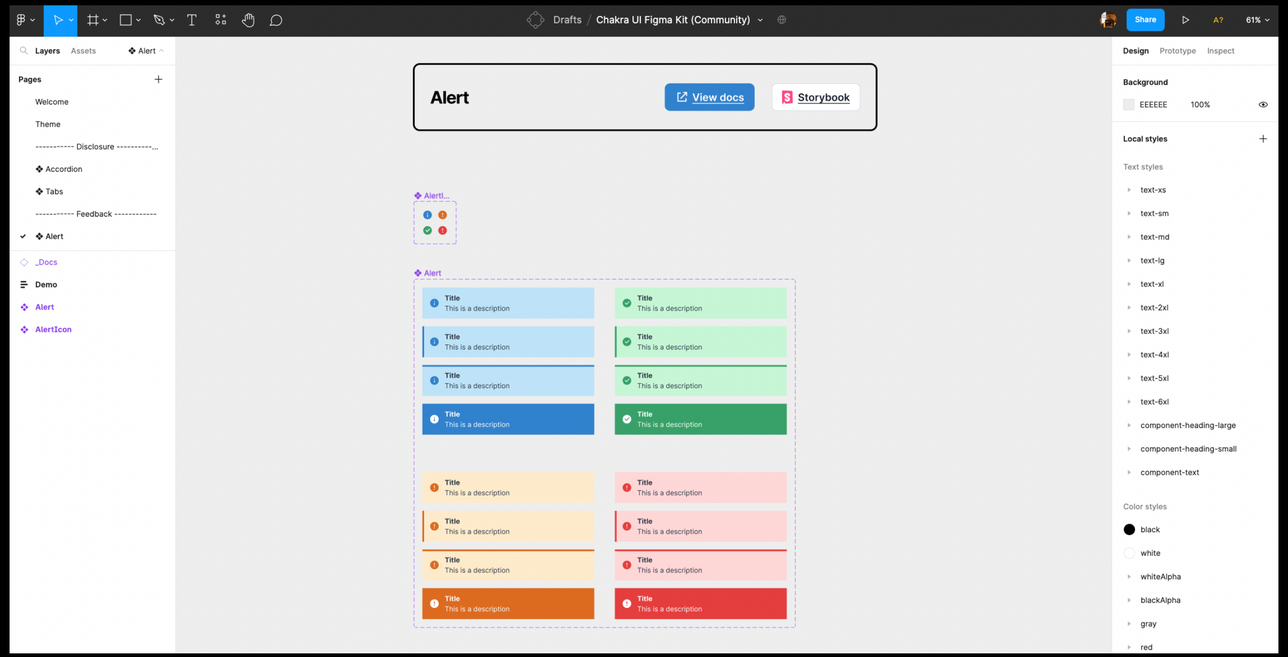
커뮤니티가 잘 형성되어 있을 것 → ⭐ 31.3k

Chakra-ui, 너 나하고 일 하나 같이 하자

컴포넌트가 풍부할 것 → 61종

커뮤니티가 잘 형성되어 있을 것 → ★ 31.3k

Figma 리소스 지원될 것 → Chakra Figma UI Kit



Chakra-ui, 너 나하고 일 하나 같이 하자

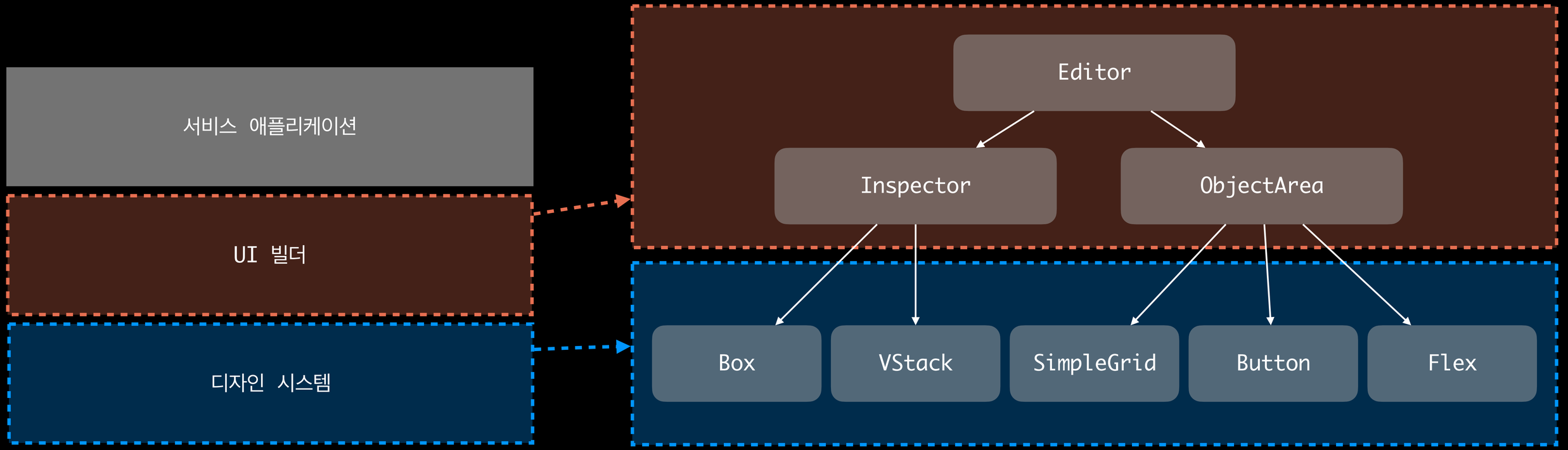
컴포넌트가 풍부할 것 → 61종
커뮤니티가 잘 형성되어 있을 것 → ★ 31.3k
Figma 리소스 지원될 것 → Chakra Figma UI Ki
커스텀 하기 좋을 것 → Theme, Style Props

```
<Box m={2}>Tomato</Box>
<Box maxW="960px" mx="auto" />
<Box color='#f00' />
<Box bg='tomato' />
<Box backgroundColor='tomato' />
<Text
  bgGradient="linear(to-l, #7928CA, #FF0080)"
  bgClip="text"
  fontSize="6xl"
  fontWeight="extrabold"
>
  Welcome to Chakra UI
</Text>
```

디자인 시스템을 가진 후

무엇이 달라졌나!?

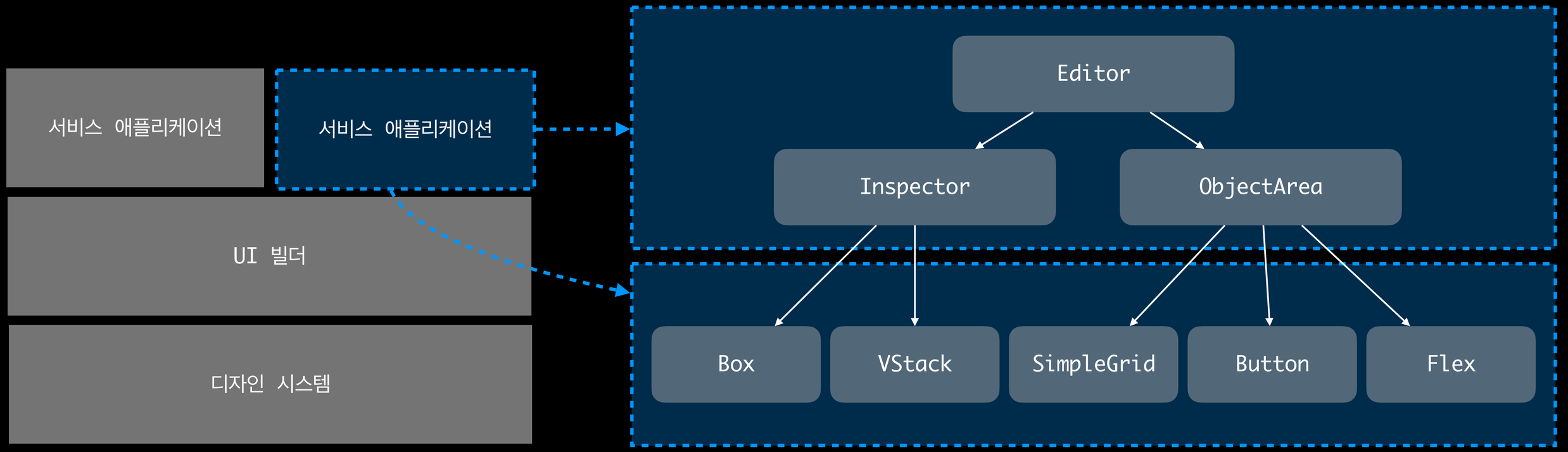
새로운 서비스 애플리케이션 개발시 기존 계층 활용



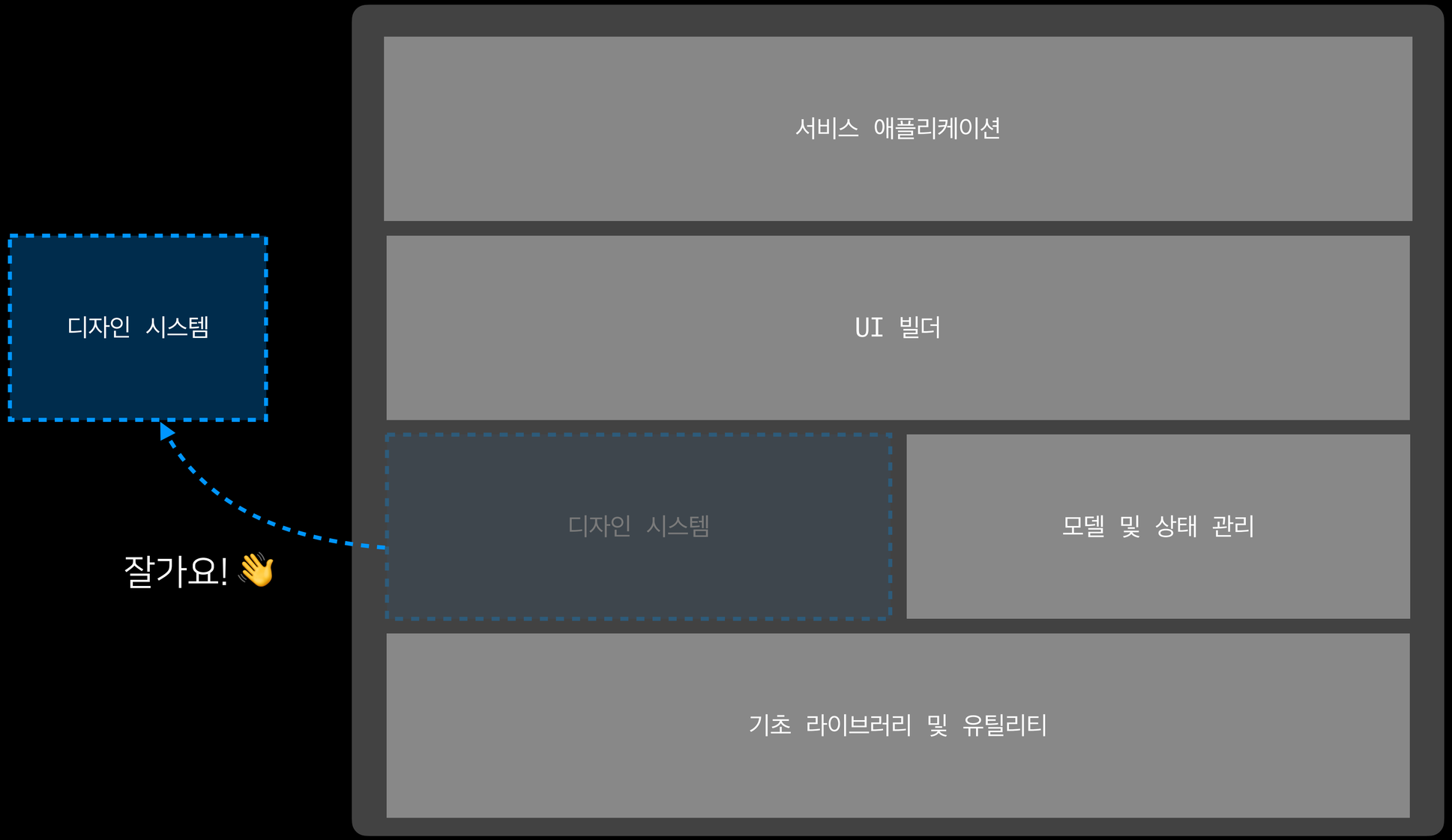
디자인 시스템을 가진 후

무엇이 달라졌나!?

새로운 서비스 애플리케이션 개발시 기존 계층 활용



디자인 시스템과 애플리케이션 계층 분리



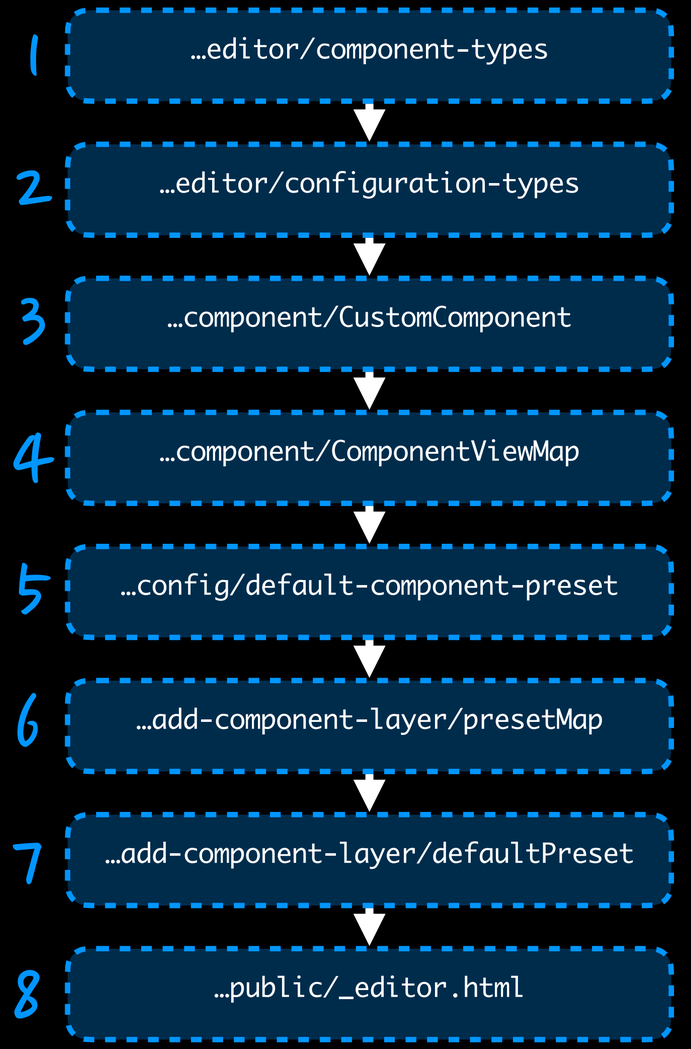
문제 #3

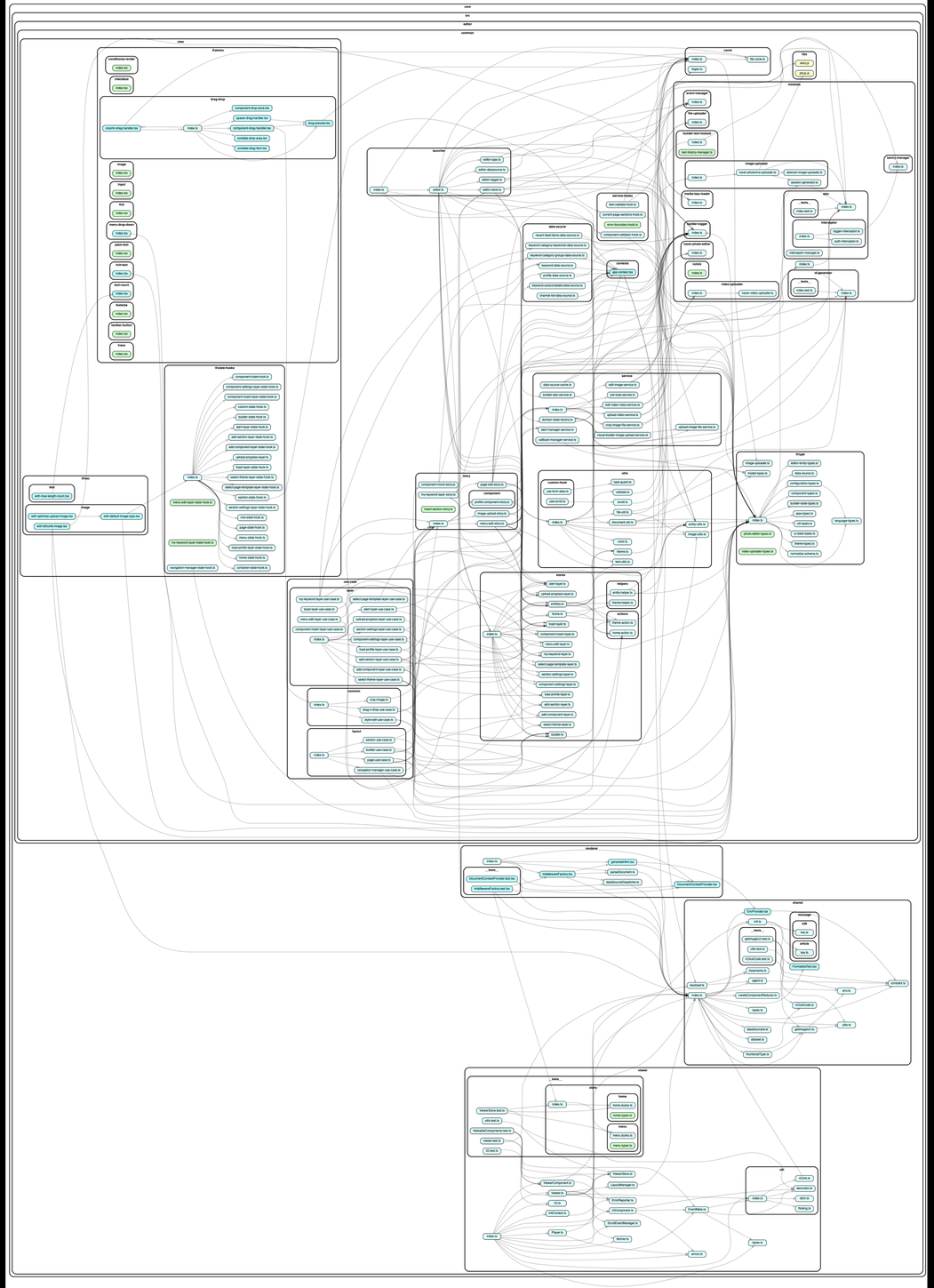
너무 긴 커스텀 동선

NAVER
DEVIEW
2023

커스텀 컴포넌트 추가할 때 8군데를 고쳤어요

다 했는데 왜 에러가 나죠!?

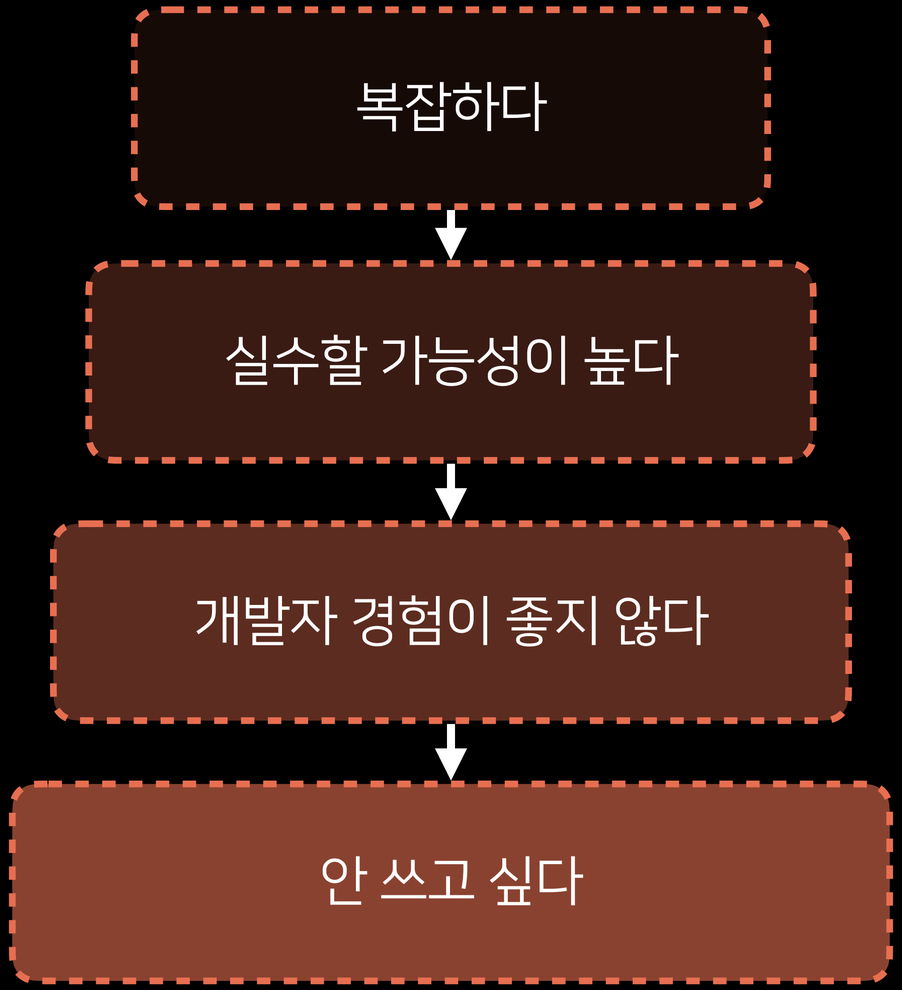




왼쪽에서 오른쪽 찾기

1. …editor/component-types
2. …editor/configuration-types
3. …component/CustomComponent
4. …component/ComponentViewMap
5. …config/default-component-preset
6. …add-component-layer/presetMap
7. …add-component-layer/defaultPreset
8. …public/_editor.html

어렵다



진단 #3

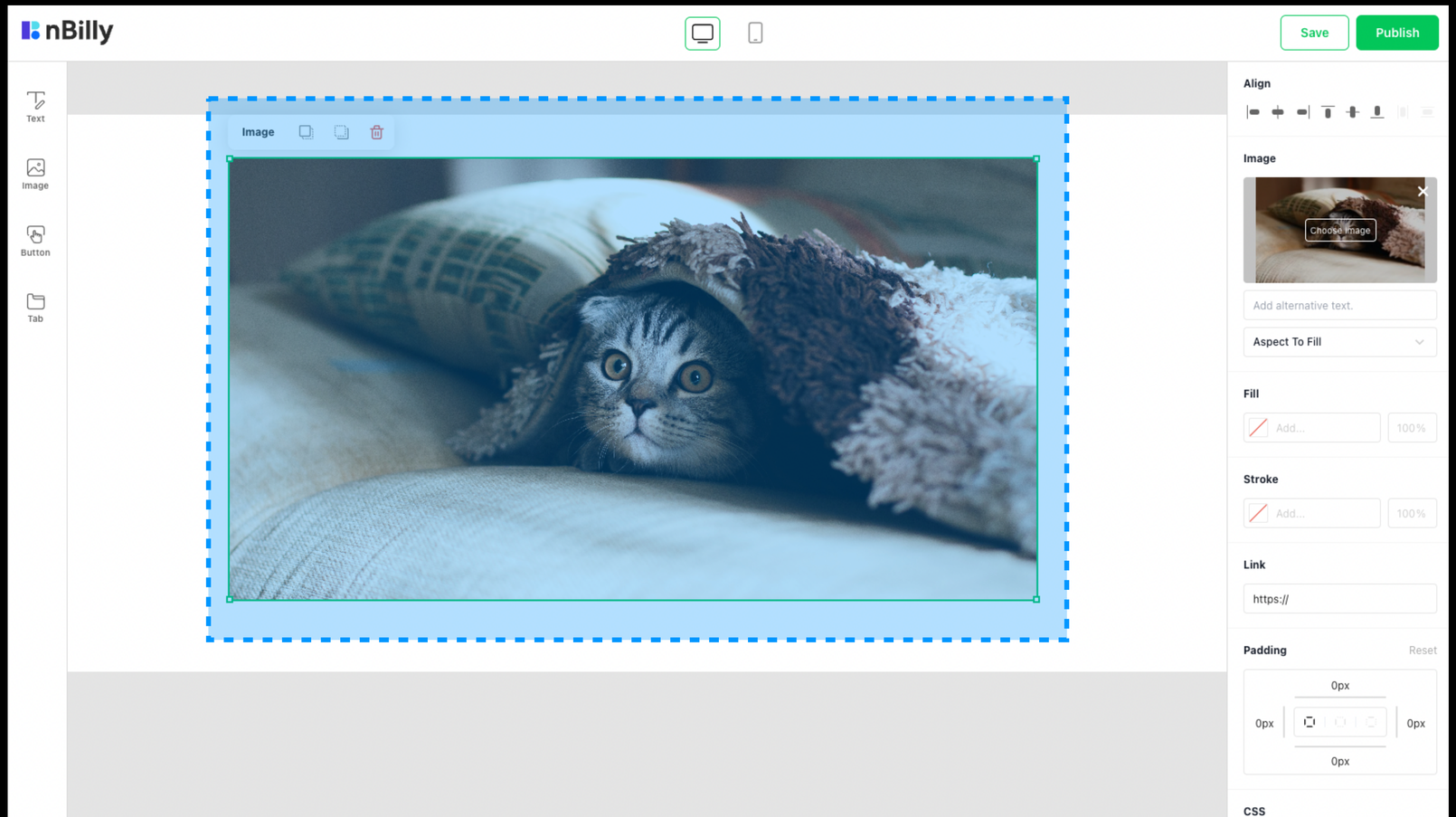
사용자 관점의 부재

처방 #3

변경을 모듈화 하라

블록의 자격 4가지

1. 모델(Model)
2. 뷰(View)
5. 로직(Logic)
4. 프로퍼티(Property)

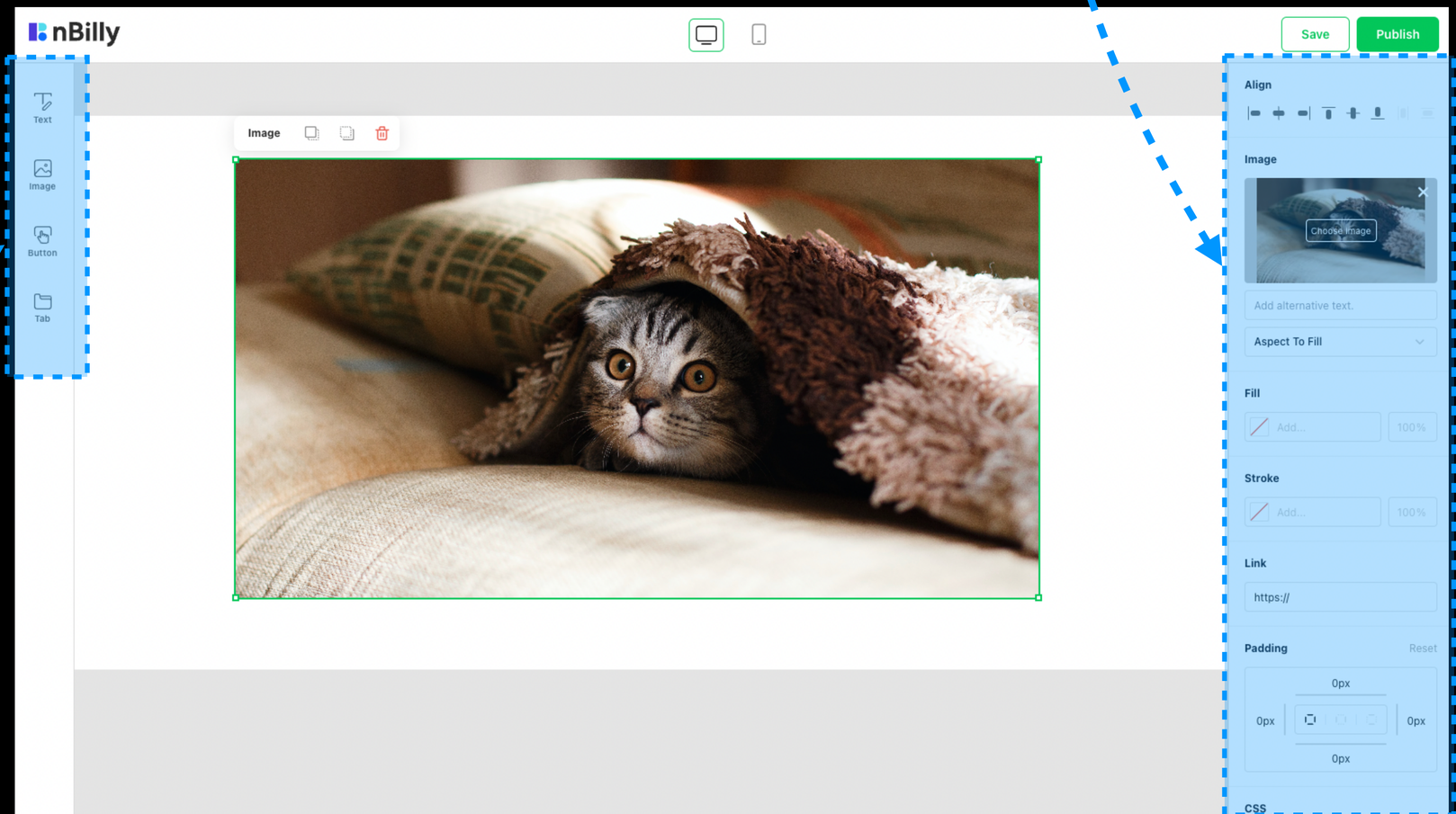


처방 #3

변경을 모듈화 하라

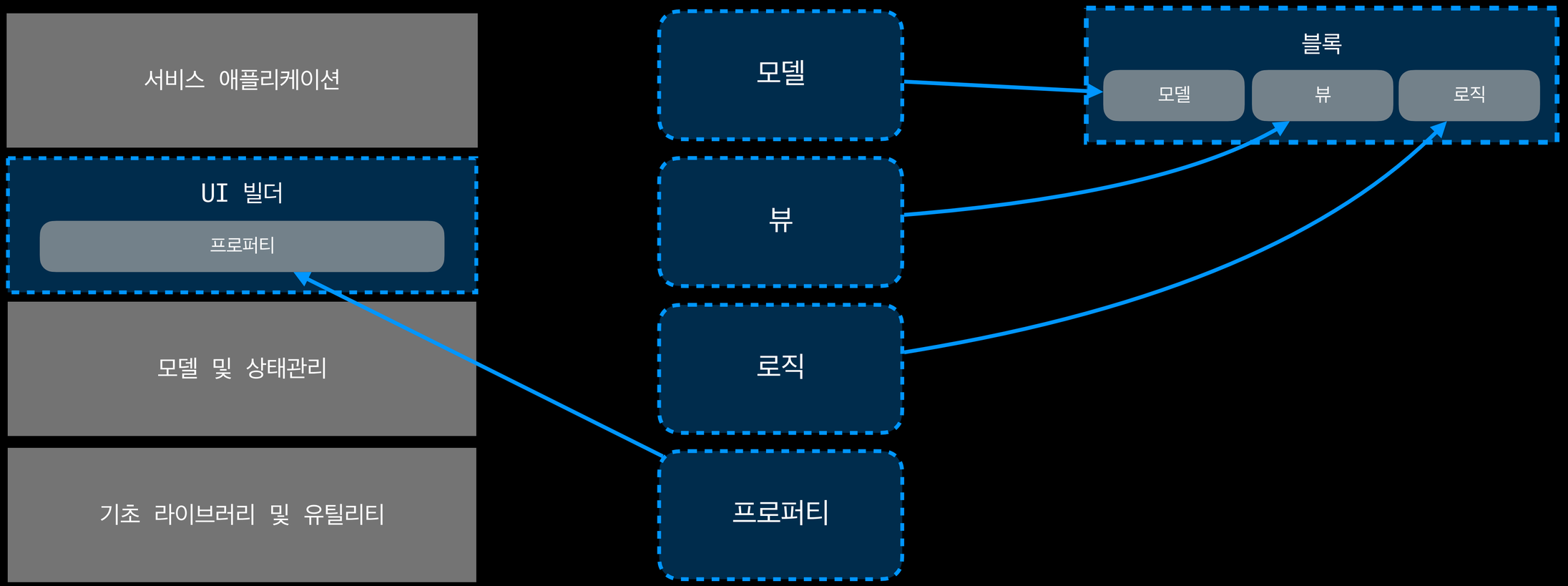
블록의 자격 4가지

1. 모델(Model)
2. 뷰(View)
5. 로직(Logic)
4. 프로퍼티(Property)



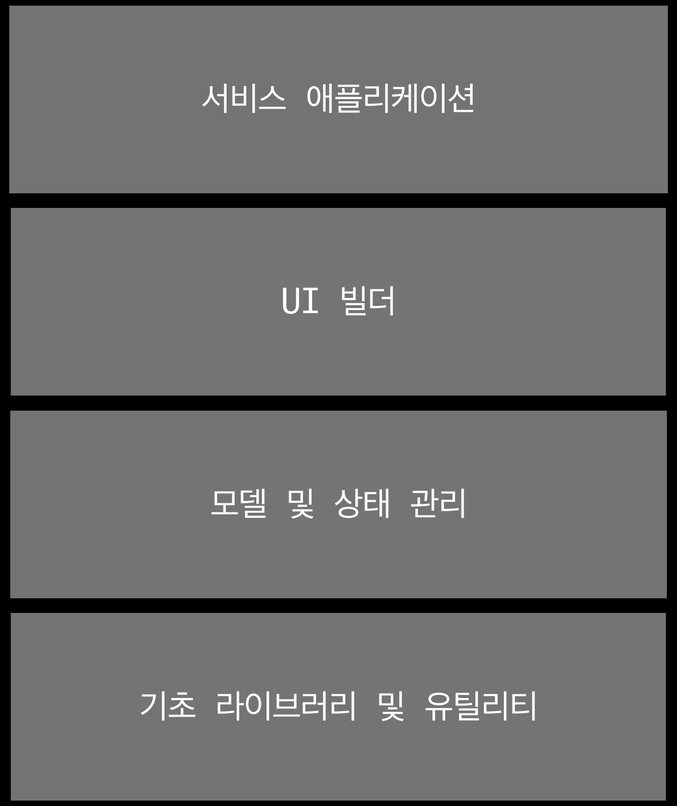
하나. 부품을 적절한 계층에 배치한다

사정을 가장 잘 아는 곳에 두기



둘. 동선을 생각하며 결합을 한 지점으로 모은다

manifest 파일에 블록의 스펙을 선언하기




```
NbillyPlugin.registerPlugin<IBlockSpec>({
  name: 'Image',
  category: PluginCategory.BLOCK,
  specs: [
    {
      name: 'Image',
      icon: ImageBlockIcon,
      type: IMAGE_BLOCK_TYPE,
      defaultModelValue: defaultImageBlockModel,
      properties: [
        BlockAlignProperty,
        BlockImageProperty,
        () => BlockColorProperty({ title: 'Fill', stylePropertyName: 'color' }),
        () => BlockColorProperty({ title: 'Stroke', stylePropertyName: 'strokeColor' }),
        LinkProperty,
        CustomCssProperty,
      ],
      view: ImageBlock,
    },
  ],
});
```

둘. 동선을 생각하며 결합을 한 지점으로 모은다

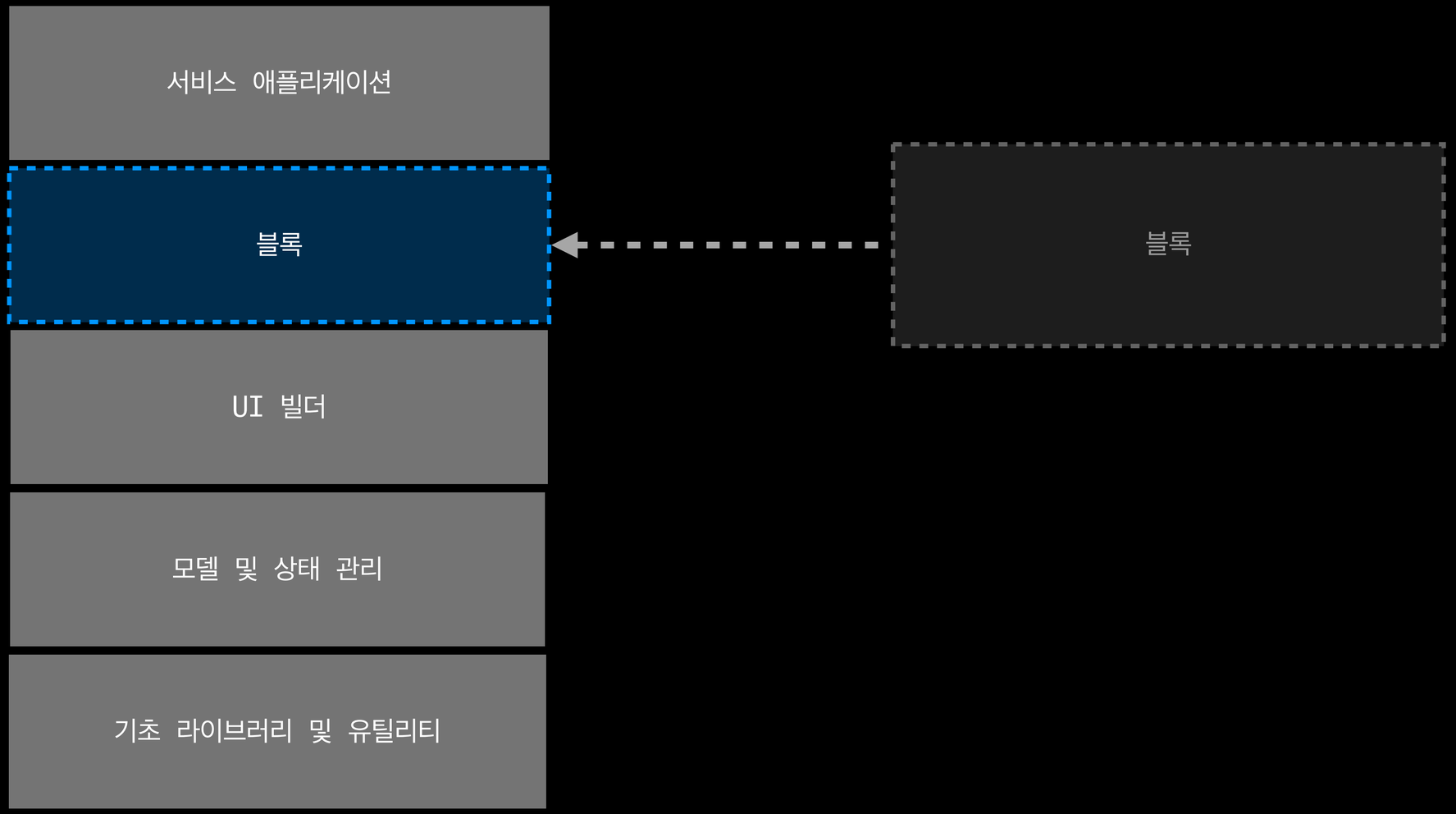
builderConfig에 등록하기


```
import { IBuilderConfig } from '@nbilly/builder';

function createBuilderConfig(): Partial<IBuilderConfig> {
  return {
    plugins: () => [
      require('../plugins/text-block'),
      require('../plugins/image-block'),
      require('../plugins/button-block'),
      require('../plugins/tab-block'),
    ],
  };
}

export default createBuilderConfig;
```

셋. 새로 생긴 구역을 계층에 통합한다

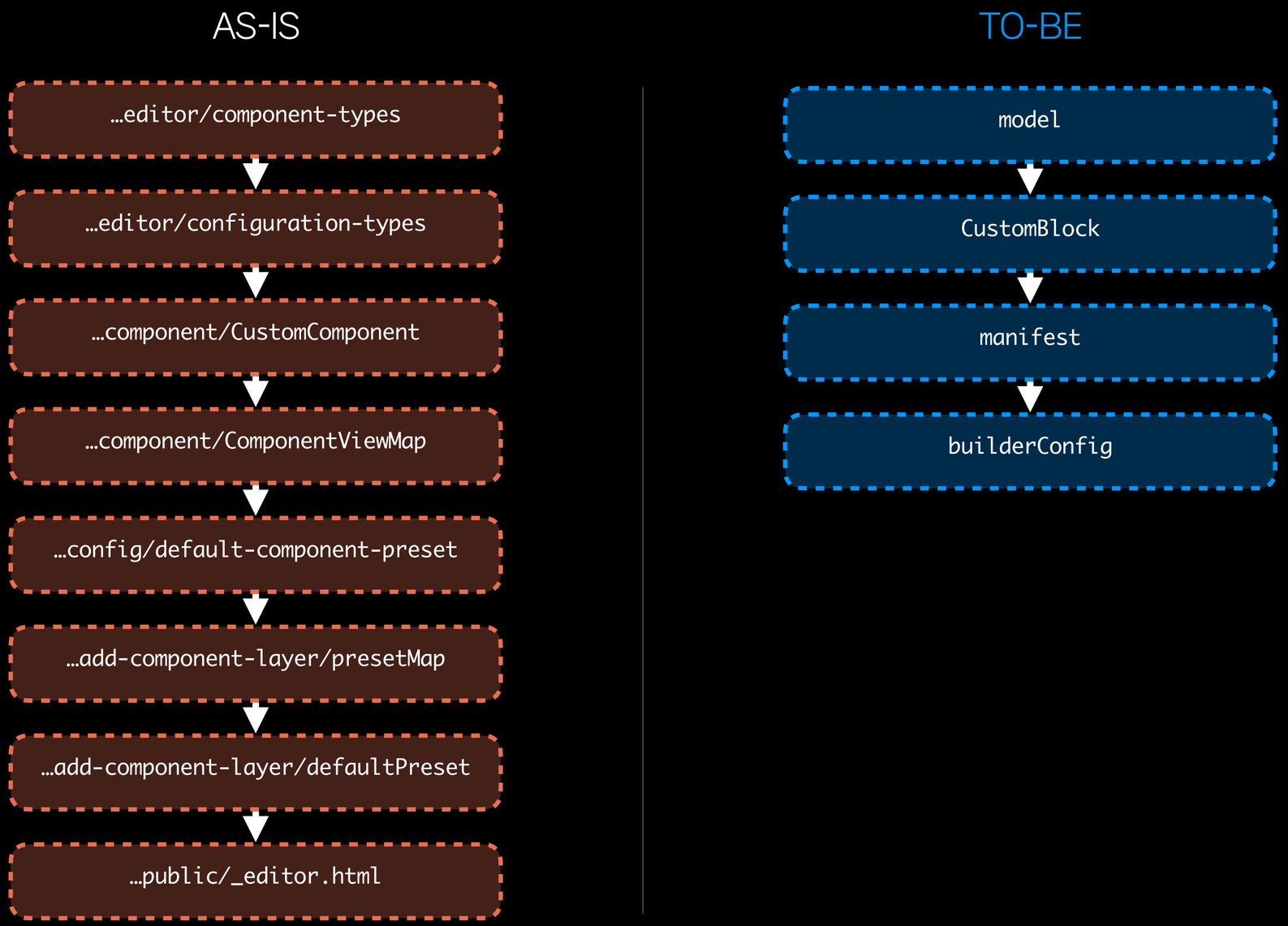


변경을 모듈화 한 후에

무엇이 달라졌나!?

반으로 줄어든 동선

더 높아진 예측 가능성



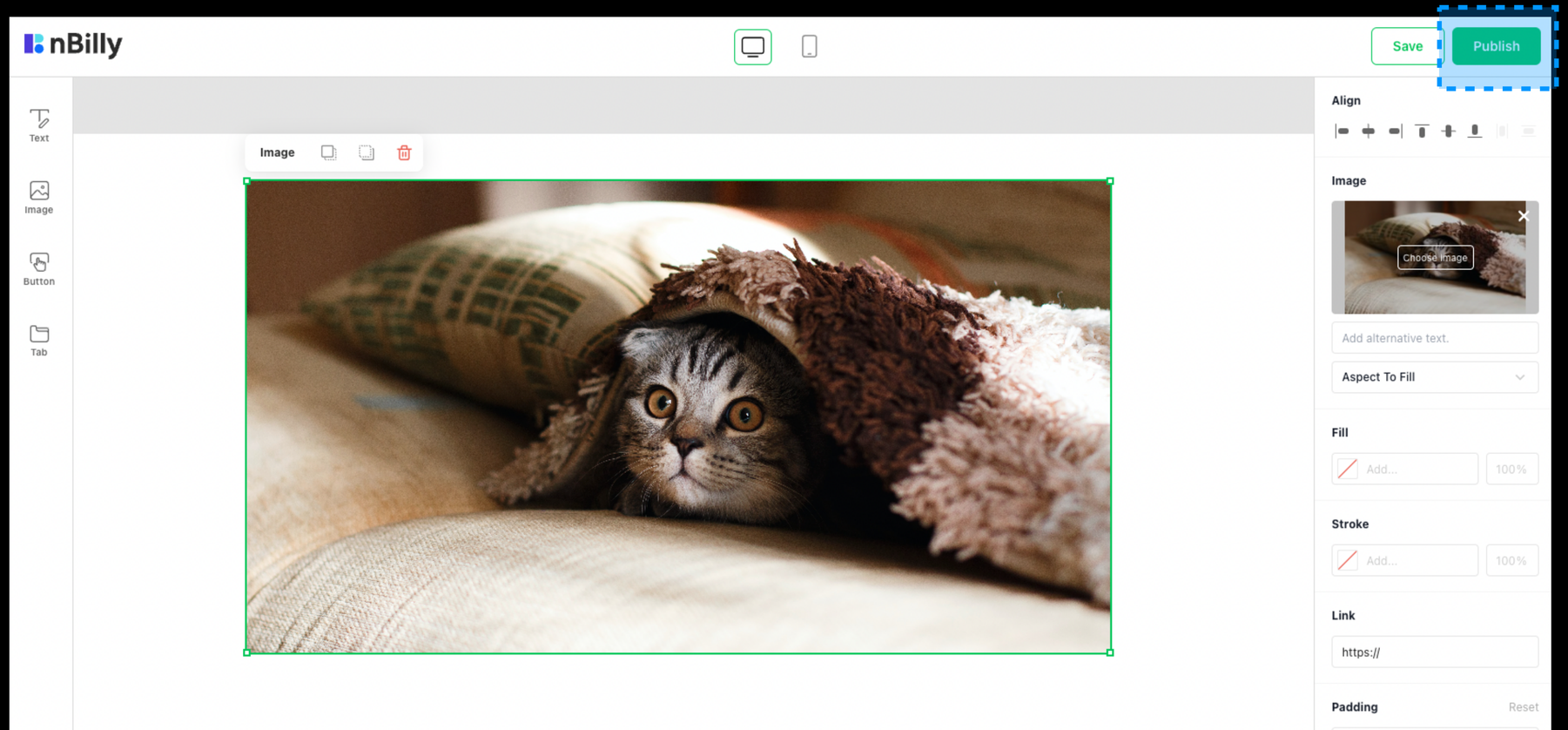
보너스 스테이지 #1

05. 플러그인의 발견

다른 기능도 블록 같을 순 없을까?

NAVER
DEVIEW
2023

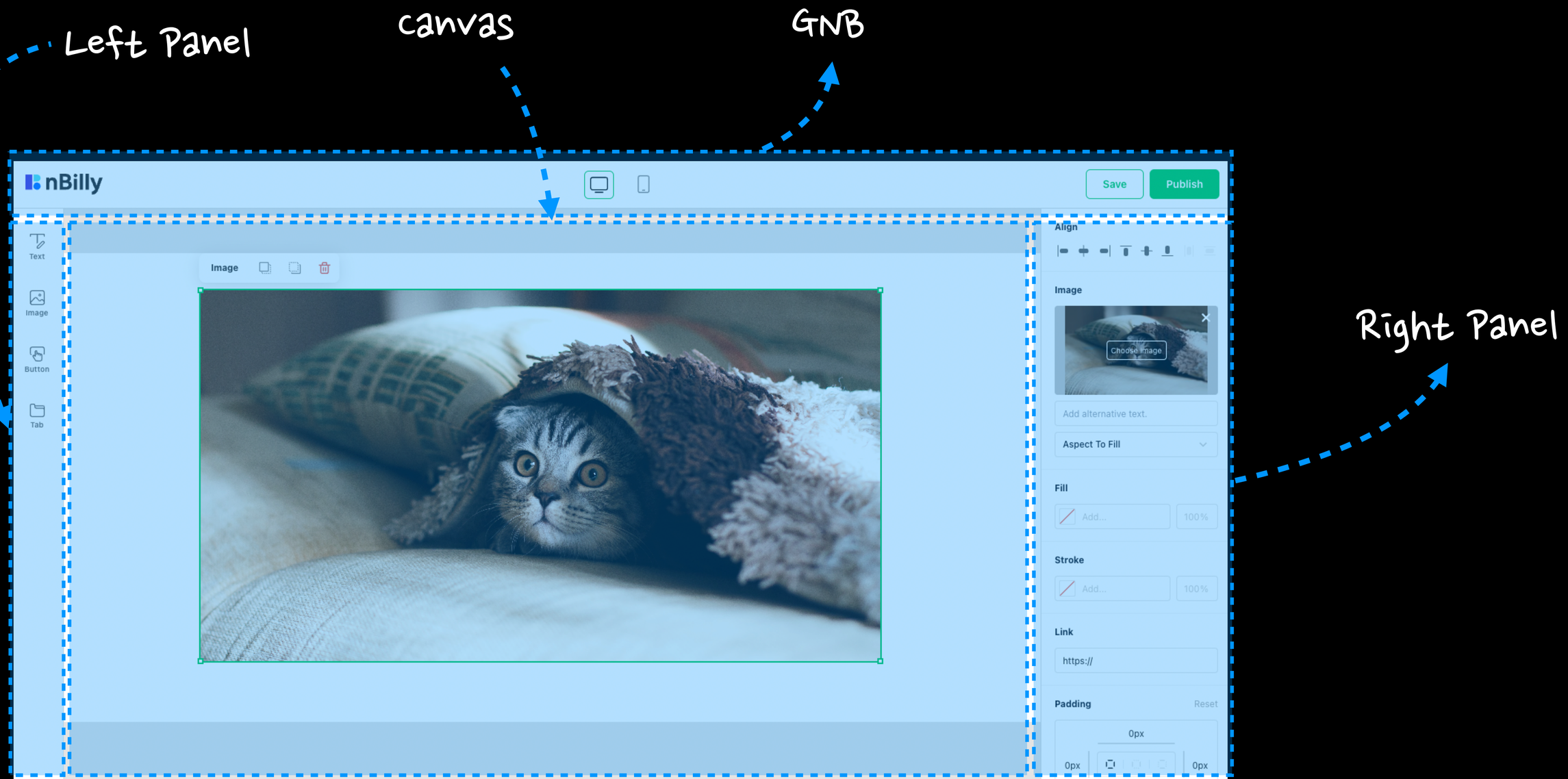
"발행한 사이트를 우리 서버에 저장할 수 있어요?"
"미리보기 기능을 넣을 순 없어요?"



변경을 바깥에서 만들어서 안으로 넣기

클라이언트가 무엇을 원할지 우리는 알 수 없다
알려고 하지도 말자

하나. UI 레이아웃을 몇 개의 구역으로 나누기



둘. 블록처럼 한 곳으로 모은다

manifest 파일에 UI 레이아웃의 사양을 선언하기

```
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'GlobalNavigationBar',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.GNB,
      view: () =>
        GlobalNavigationBar({
          components: [
            Logo,
            () => (
              <Centering>
                {ButtonGroup({ buttons: [DesktopPreviewButton, MobilePreviewButton], gap: '20px' })}
              </Centering>
            ),
            () => ButtonGroup({ buttons: [SaveButton, PublishButton] }),
          ],
        }),
    },
  ],
});
```

```
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'RightPanel',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.RIGHT_PANEL,
      view: () => RightPanel({ components: [Inspector] }),
    },
  ],
})
```

```
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'LeftPanel',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.LEFT_PANEL,
      view: () => LeftPanel({ components: [BlockArea] }),
    },
  ],
});
```

둘. 블록처럼 한 곳으로 모은다

manifest 파일에 UI 레이아웃의 사양을 선언하기

```
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'GlobalNavigationBar',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.GNB,
      view: () =>
        GlobalNavigationBar({
          components: [
            Logo,
            () => (
              <Centering>
                {ButtonGroup({ buttons: [DesktopPreviewButton, MobilePreviewButton], gap: '20px' })}
              </Centering>
            ),
            () => ButtonGroup({ buttons: [SaveButton, PublishButton] }),
          ],
        }),
    },
  ],
});
```

```
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'RightPanel',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.RIGHT_PANEL,
      view: () => RightPanel({ components: [Inspector] }),
    },
  ],
})
```

```
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'LeftPanel',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.LEFT_PANEL,
      view: () => LeftPanel({ components: [BlockArea] }),
    },
  ],
});
```

둘. 블록처럼 한 곳으로 모은다

builderConfig에 등록하기


```
import { IBuilderConfig } from '@nbilly/builder';

function createBuilderConfig(userId?: string): Partial<IBuilderConfig> {
  return {
    plugins: () => [
      require('../plugins/text-block'),
      require('../plugins/image-block'),
      require('../plugins/button-block'),
      require('../plugins/tab-block'),
      require('../plugins/gnb'),
      require('../plugins/left-panel'),
      require('../plugins/right-panel'),
    ],
  };
}

export default createBuilderConfig;
```

손님 마음에 안 드시면 수정해서 쓰세요

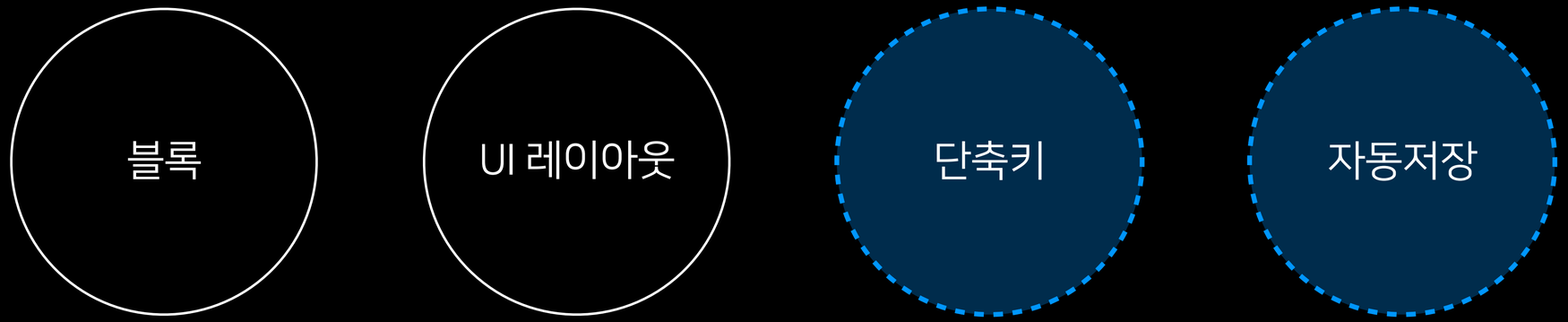
```
const apis = {
  saveSite: (site: ISite) => appSitesRemotes.saveSite(site),
};

NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'GlobalNavigationBar',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.GNB,
      view: () =>
        GlobalNavigationBar({
          components: [
            Logo,
            () => (
              <Centering>
                {ButtonGroup({ buttons: [DesktopPreviewButton, MobilePreviewButton], gap: '20px' })}
              </Centering>
            ),
            () => ButtonGroup({ buttons: [SaveButton, PublishButton({ apis })] }),
          ],
        }),
    },
  ],
});
```

다른 것도 할 수 있겠는데?

NAVER
DEVIEW
2023

블록과 UI 레이아웃을 해결했다면 단축키나 모달도 할 수 있어야지?



똑같이 만들어보자

유형으로 구분하고 인터페이스는 동일하게 맞추기

React Context

```
NbillyPlugin.registerPlugin<IGlobalFeatureSpec>({
  name: 'Shortcut',
  category: PluginCategory.COMPONENT_PROVIDER,
  specs: [
    {
      use: [PluginUse.EDITOR],
      view: CustomShortcut,
    },
  ],
});
```

UI가 없는 백그라운드 기능

```
NbillyPlugin.registerPlugin<IGlobalFeatureSpec>({
  name: 'AutoSave',
  category: PluginCategory.GLOBAL_FEATURE,
  specs: [
    {
      use: [PluginUse.EDITOR],
      view: () => AutoSave({ apis, router }),
    },
  ],
});
```

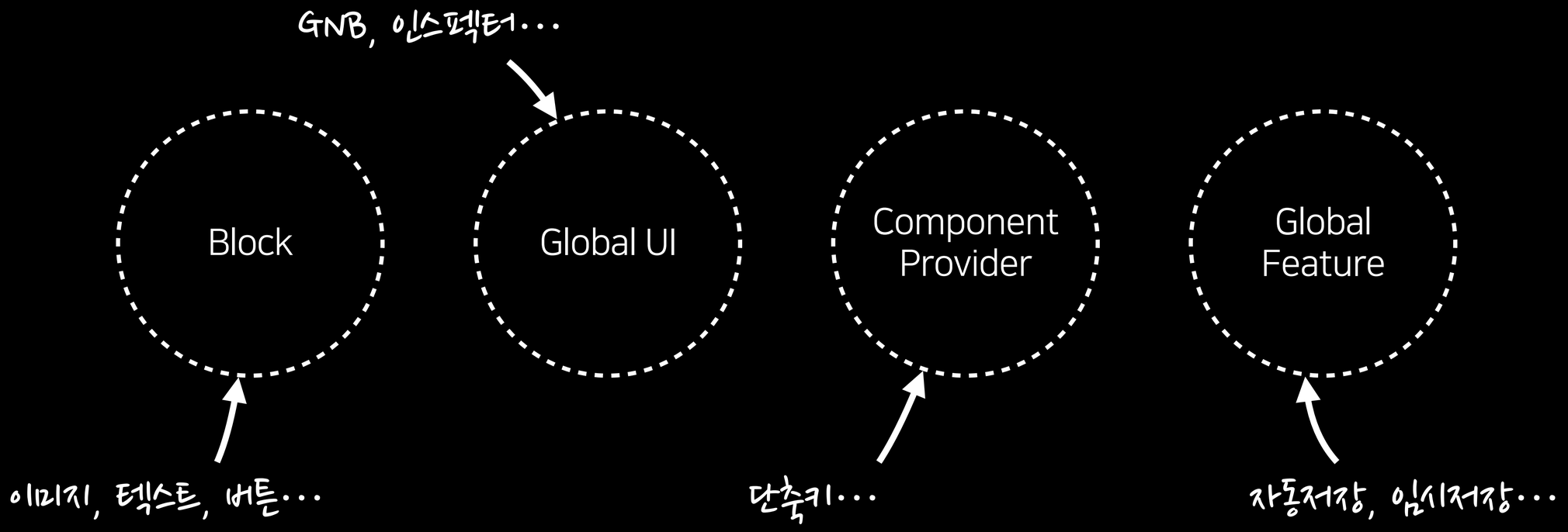
```ts
NbillyPlugin.registerPlugin<IBlockSpec>({
  name: 'Image',
  category: PluginCategory.BLOCK,
  specs: [
    {
      name: 'Image',
      icon: ImageBlockIcon,
      type: IMAGE_BLOCK_TYPE,
      defaultModelValue: defaultImageBlockModel,
      properties: [
        BlockAlignProperty,
        BlockImageProperty,
        () => BlockColorProperty({ title: 'Fill', stylePropertyName: 'color' }),
        () => BlockColorProperty({ title: 'Stroke', stylePropertyName: 'strokeColor' }),
        LinkProperty,
        CustomCssProperty,
      ],
      view: ImageBlock,
    },
  ],
});
```

```ts
NbillyPlugin.registerPlugin<IGlobalUISpec>({
  name: 'Left Panel',
  category: PluginCategory.GLOBAL_UI,
  specs: [
    {
      slot: GlobalUISlot.LEFT_PANEL,
      view: () => LeftPanel({ components: [BlockArea] }),
    },
  ],
});
```

```ts
NbillyPlugin.registerPlugin<IGlobalFeatureSpec>({
  name: 'AutoSave',
  category: PluginCategory.GLOBAL_FEATURE,
  specs: [
    {
      use: [PluginUse.EDITOR],
      view: () => AutoSave({ apis, router }),
    },
  ],
});
```

```ts
NbillyPlugin.registerPlugin<IGlobalFeatureSpec>({
  name: 'Shortcut',
  category: PluginCategory.COMPONENT_PROVIDER,
  specs: [
    {
      use: [PluginUse.EDITOR],
      view: CustomShortcut,
    },
  ],
});
```

We are all plugins...!



무엇이 더 좋아졌나?

NAVER
DEVIEW
2023

서로 다른 기능을 동일한 인터페이스로 모듈화

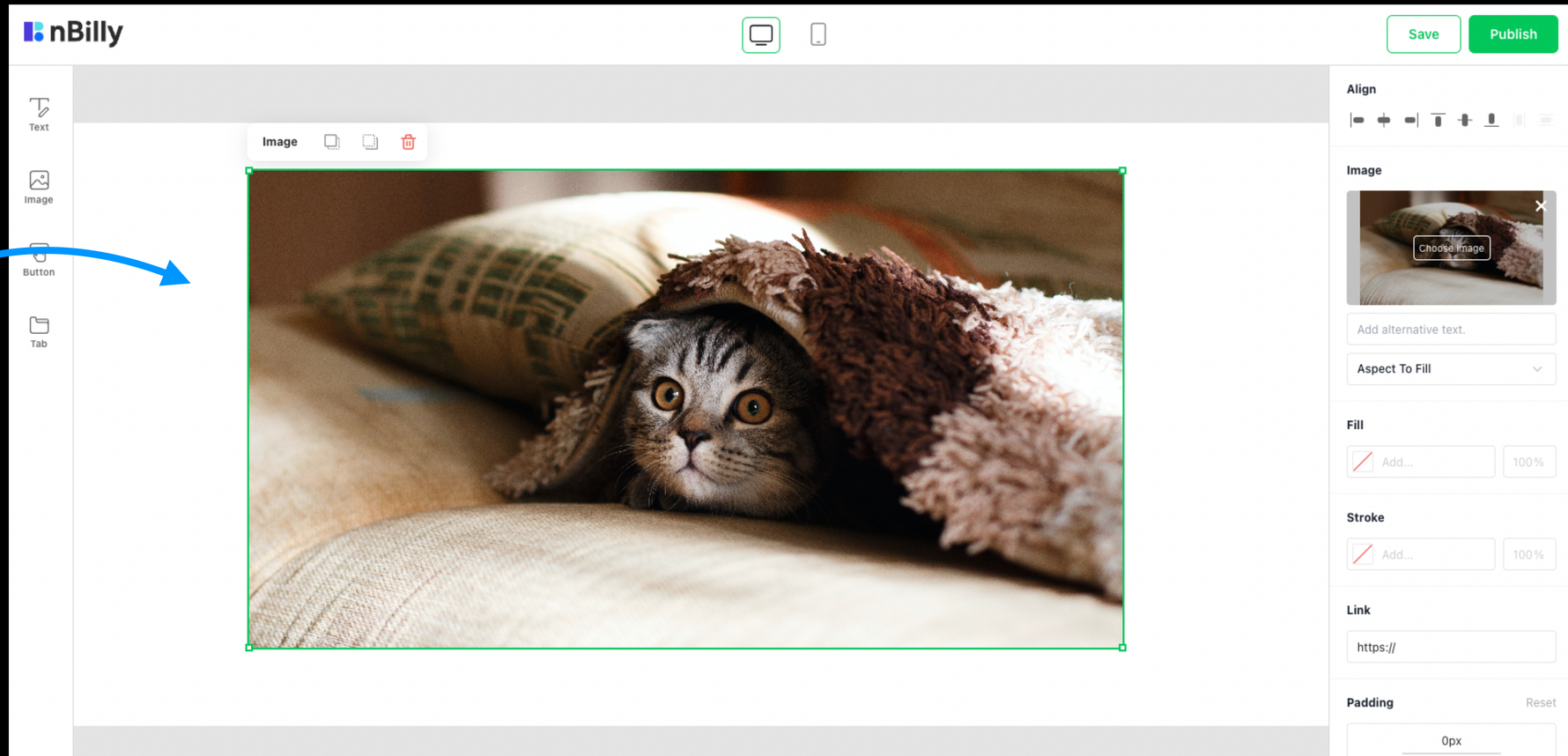
auto-save
shortcut
gnb
inspector
object-panel
text-block
image-block
button-block
tab-block

무엇이 더 좋아졌나?

NAVER
DEVIEW
2023

서로 다른 기능을 동일한 인터페이스로 모듈화
기존 체계를 훼손하지 않는 확장성 확보

auto-save
shortcut
gnb
inspector
object-panel
text-block
image-block
button-block
tab-block



잠깐만, 단점은 없을까?

NAVER
DEVIEW
2023

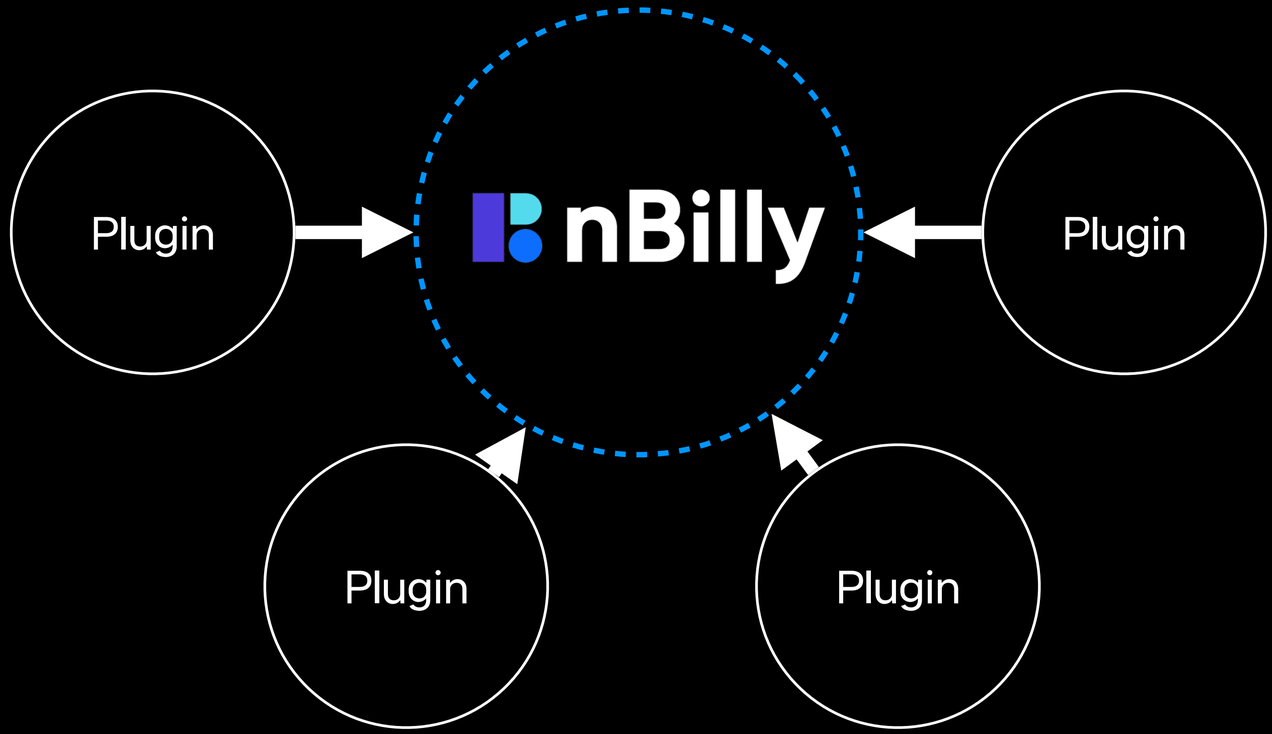
높은 확장성으로 얻을
비즈니스 가치가 없다면
오버엔지니어링

공짜 점심은 없다

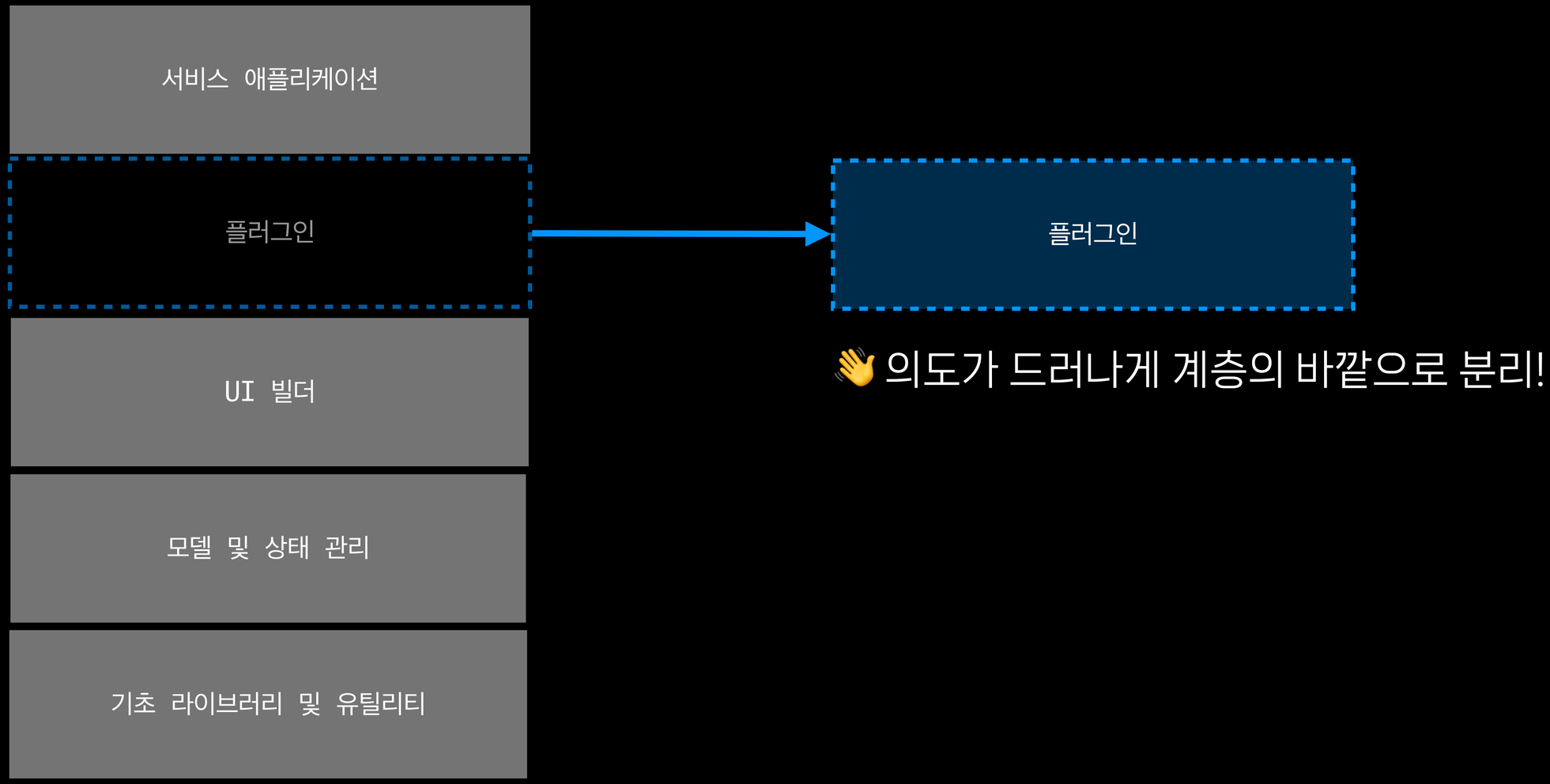
돌아가는 느낌
미지의 가능성 대비하기
어려워진 최적화
호환성 유지라는 산

우리에겐 중요한 확장성

프로덕트에 넣고 싶어요



변경을 밖에서 만들어 안으로 넣겠다는 의지



👋 의도가 드러나게 계층의 바깥으로 분리!

HomeBuilder SDK 2.0

nBilly

보너스 스테이지 #2

06. 다르게 풀어야 하는 문제

도메인의 행위 변경

NAVER
DEVIEW
2023

계층의 아래로 내려갈수록
재조립 비용 ⬆ 발생 가능성 ⬇

도메인의 행위 변경

NAVER
DEVIEW
2023

계층의 아래로 내려갈수록
재조립 비용 ⬆ 발생 가능성 ⬇

근간이 되는 라이브러리,
도메인의 동작이 바뀔 가능성은?

예) 컨테이너는 하나의 블록만 가져야 한다

한계를 설정하기

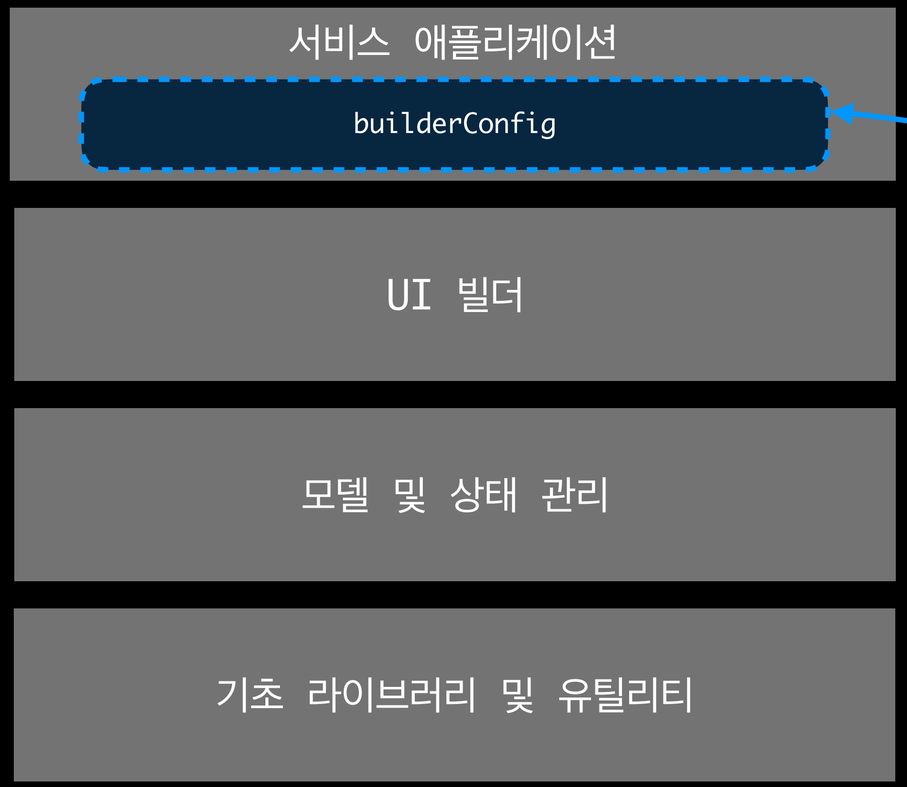
모든 걸 해줄 수는 없다
가능성이 낮은 일은 적당히 눈 감자

하지만 뭐든 해주고 싶은 우리는 개발자...

대신에 옵션을 열어 드릴게요

편집 규칙을 변경할 수 있는 옵션 열기

게으른 대응 → 요구사항이 들어오면 만든다




```
editorConfig: {
  textFormats: TEXT_FORMATS,
  maxHistoryCount: Infinity,
  radius: {
    min: 0,
    max: 999,
  },
  recentColorLimit: 23,
  gridTemplate: {
    row: 8,
    rowCount: 40,
    column: 8,
    gap: 8,
  },
  blockDndModifiers: restrictToVerticalAxis,
},
```

07. 정리하기

오늘 한 이야기

NAVER
DEVIEW
2023

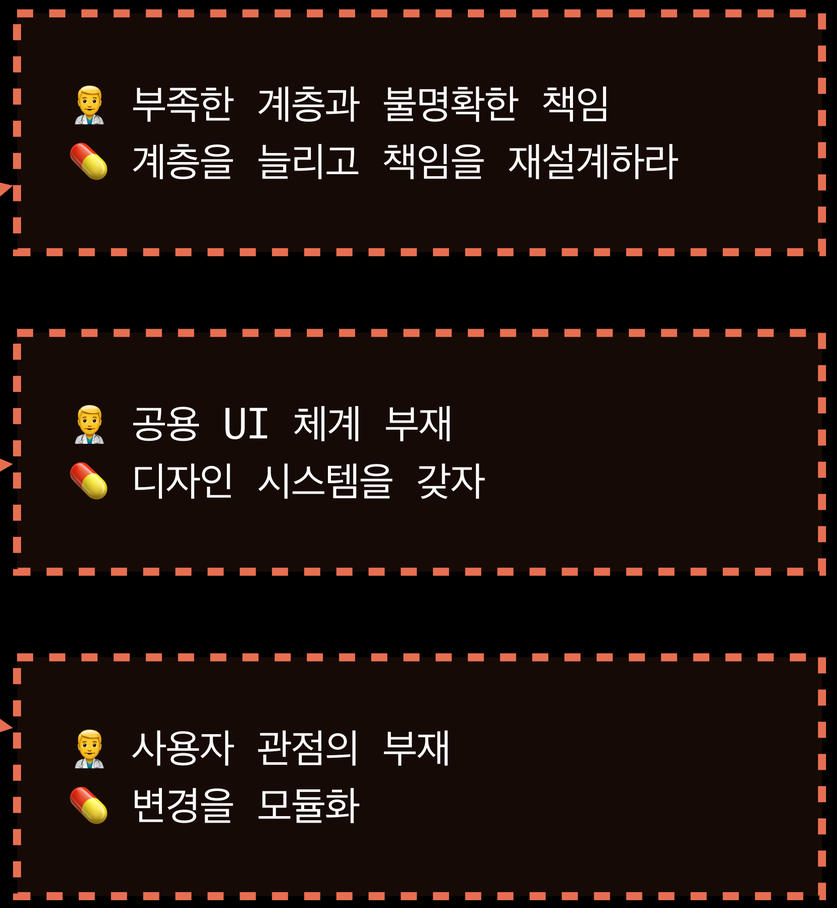
01. Intro - HomeBuilder 소개

02. 왜 레고일까?
03. 우리의 레고를 찾아서

04. 문제 해결하기
- 너무 깐간한 코어 로직
- 너무 무거운 프로덕트 뷰
- 너무 긴 커스텀 동선

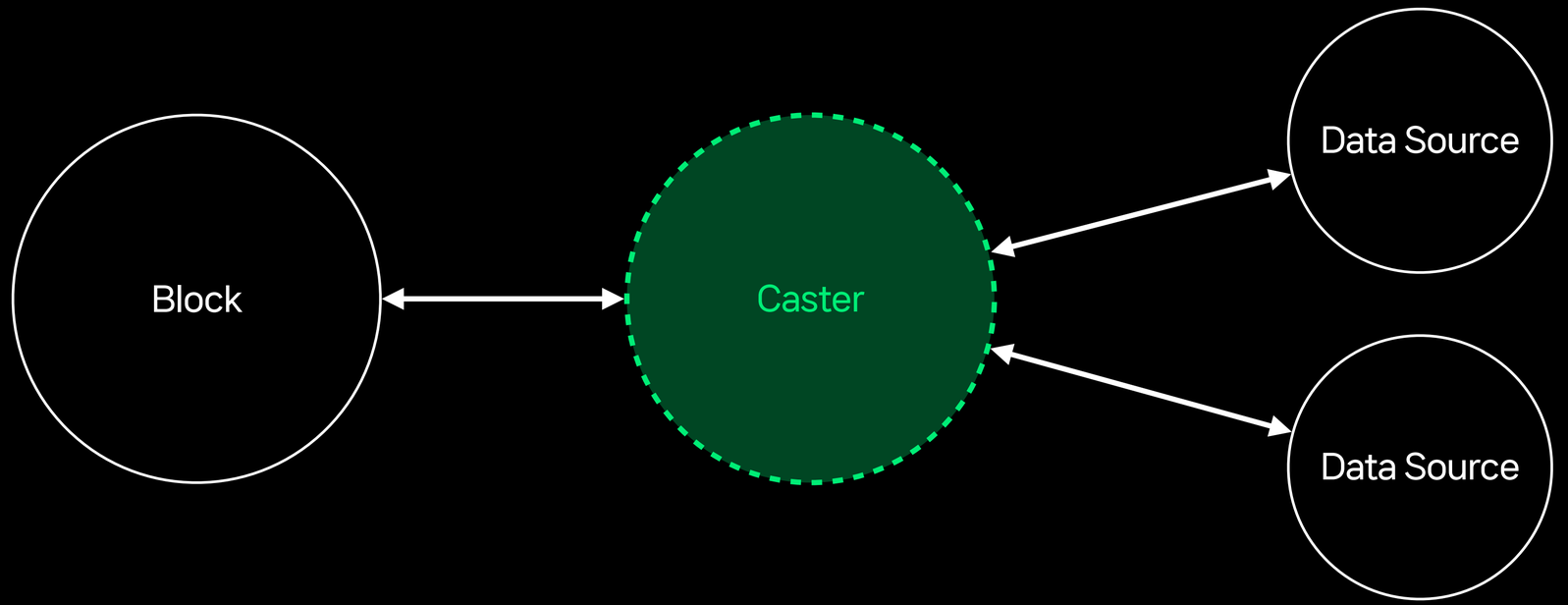
05. 플러그인의 발견
06. 다르게 풀어야 하는 문제들

07. 정리하기



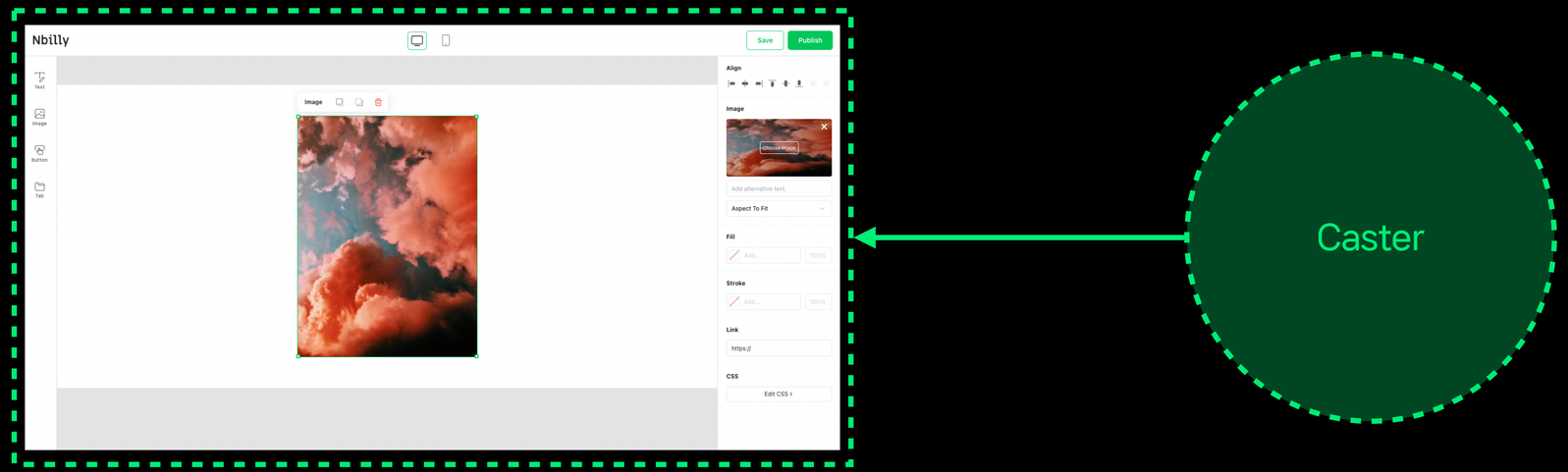
2023년

한 개의 블록에
N개의 데이터 소스 연결하기



2023년

한 개의 블록에
N개의 데이터 소스 연결하기



nBilly + Clova AI = ?

함께 할 수 있어 영광입니다

매일매일 고맙습니다

NAVER

ETECH 에서 함께 성장하실 분을 모십니다.

NAVER
DEVIEW
2023

Channels

- ETECH 직무 소개 : https://naver-career.gitbook.io/kr/service/etech

- ETECH 기술 문의 : etech@navercorp.com

- ETECH 채용 문의 : etech-recruit@navercorp.com

ETECH 직무 소개

Q & A

Thank You