

기술

오타를 막으려고 고른 방법이 그림의 생김새까지 정해버렸습니다

방동걸 (주)웨이스 대표이사

2026년 9월 8일

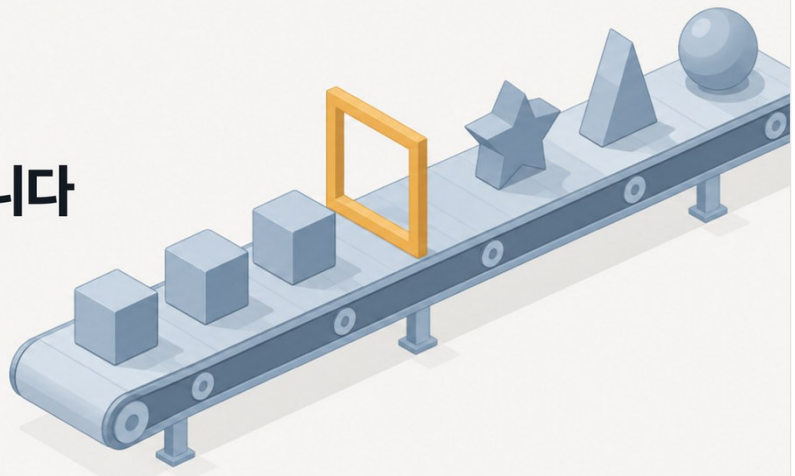
글에 넣을 이미지 세 장을 만들었는데, 다 내놓고 보니 셋이 한 형제처럼 생겼습니다. 스타일을 그렇게 고른 적이 없습니다. 오히려 "한 글 안에서 계열을 섞는다"는 규칙을 두고 있었습니다. VEXPLOR를 개발하는 웨이스가 기술블로그 이미지를 만들다 겪은 일입니다.

원인을 되짚으니 스타일을 고른 사람이 없었습니다. 오타를 막으려고 고른 생성 경로가 룩까지 같이 정한 것이었습니다.

AI 이미지 만들기

오타를 막으려다 그림이 같아졌습니다

경로를 먼저 고르면
그 경로가 모양을 정합니다



그림을 만드는 경로를 먼저 고르면, 그 경로가 낼 수 있는 모양만 나옵니다

AI로 이미지를 만들어 글에 넣다 보면 한 번은 밟는 자리라 적어 둡니다. 결론부터 적으면 순서를 뒤집어야 합니다.

그리고 이 자리는 이미지에만 있는 것이 아닙니다. 뒤에서 그 이야기를 합니다.

01

한글은 생성 모델이 자주 깨뜨립니다

이미지 생성 모델에 한글 문구를 넣으면 글자가 자주 무너집니다. 받침이 빠지거나, 없는 글자가 만들어지거나, 획이 뭉개집니다. 영문보다 눈에 띄게 나쁘고, 문장이 길수록 심해집니다.

발행물에서 이건 그냥 넘길 문제가 아닙니다. 그림 안의 오타는 본문 오타와 달리 **고치려면 이미지를 다시 만들어야** 하고, 이미 배포됐으면 회수할 방법이 없습니다. 그래서 저희는 "이미지 속 오타 0"을 타협하지 않는 조건으로 두고 있습니다.

문제는 그 조건을 어떻게 보증하느냐입니다.

02

그래서 글자를 모델에서 떼어냅니다

방법은 단순합니다. 그림과 글자를 다른 도구가 만들게 합니다.

1. 생성 모델에는 **글자 없는 그림**만 시킵니다. 프롬프트 끝에 글자 금지 구를 붙입니다.
2. 그 위에 얹을 글자는 **HTML**을 렌더해서 그림입니다. 폰트로 그리니 변형될 여지가 없습니다.
3. 둘을 합쳐 최종 이미지를 만듭니다.

저희는 문안을 JSON으로 적고 렌더러가 그것을 판으로 그리게 합니다. 대충 이런 모양입니다.

```
{ "type": "image", "src": "plate.png", "labels": [ { "x": 11, "y": 26, "text": "실체가 있다" } ] }
```

여기서 오타가 날 자리는 **JSON에 적은 문안 자체**뿐입니다. 그건 본문에서 복사해 오면 되고, 사람이 눈으로 대조할 수 있습니다. 모델이 글자를 임의로 바꿀 통로가 아예 없어집니다.

실제로 이 경로로 만든 이미지들은 글자 검사에서 오타 0이 나옵니다. 구조적으로 그럴 수밖에 없으니 당연합니다.

03

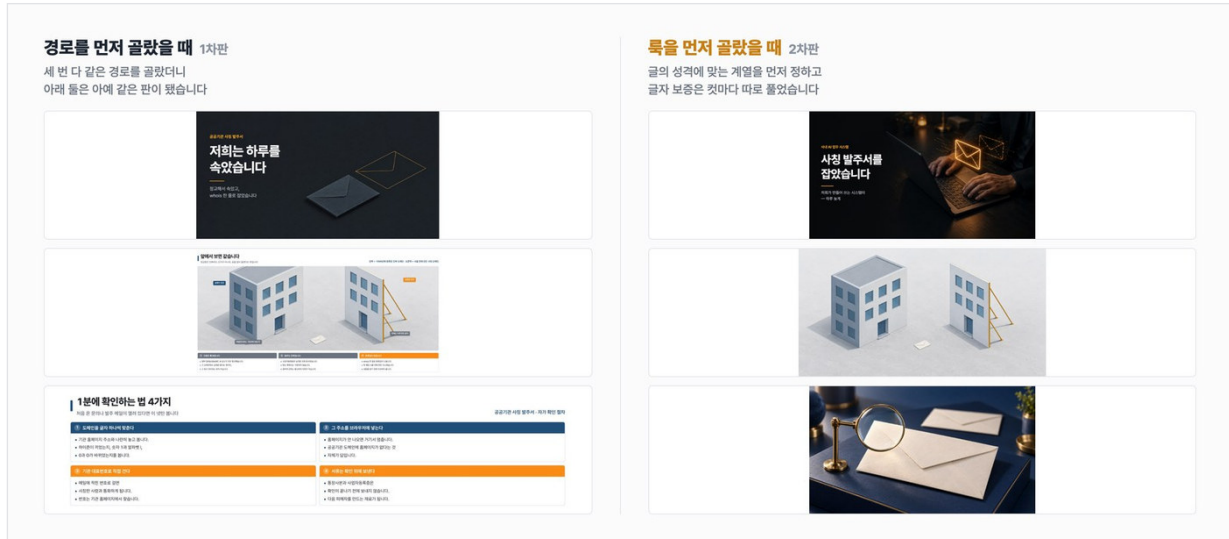
그런데 그 선택이 룩까지 정합니다

여기가 이번에 밝은 자리입니다.

글자를 렌더러가 그리게 하려면, **렌더러가 그릴 수 있는 판 위에 그림을 올려야** 합니다. 그 판은 정해진 모양을 갖고 있습니다. 저희 렌더러는 머리글 한 줄, 그림 자리 하나, 설명 카드 몇 장으로 구성된 문서형 판입니다. 보고서에 넣기 좋은 모양이고, 그래서 그렇게 만들어 두었습니다.

문제는 **글자를 넣고 싶으면 그 판을 쓸 수밖에 없다는** 것입니다. 그림이 실사든 3D든 일러스트든, 위에 판이 덮이는 순간 결과물은 "문서에 들어가는 도식"이 됩니다.

세 장을 만들면서 저는 세 번 다 이 경로를 골랐습니다. 오타 0을 보증하는 안전한 길이라서요. 스타일을 고민한 기억이 없는 이유가 여기 있습니다. **고를 필요가 없었습니다. 경로가 이미 정해 놓았으니까요.**



왼쪽이 경로를 먼저 고른 결과, 오른쪽이 룩을 먼저 고른 결과입니다. 왼쪽 아래 둘은 같은 판을 썼든 것입니다

왼쪽 세 장은 소재도 주제도 다른데 형제처럼 보입니다. 아래 둘은 아예 같은 판이고, 커버도 같은 계열의 평면도면입니다. 오른쪽은 같은 내용을 계열만 나눠 다시 만든 것입니다.

04

순서를 뒤집으면 풀립니다

고친 방법은 규칙을 더한 게 아니라 순서를 바꾼 것입니다.

전에는 이랬습니다. 글자가 필요하다 → 오타를 막아야 한다 → 렌더러 경로로 간다 → (룩이 따라 정해진다)

지금은 이렇게 합니다. 글의 성격을 본다 → 어울리는 룩을 고른다 → 그 룩에서 글자를 어떻게 보증할지를 그다음에 푼다

마지막 단계에서 선택지는 셋입니다.

- **킷에 글자가 꼭 필요한가**를 먼저 묻습니다. 대개는 아닙니다. 설명은 본문 캡션으로 내리면 되고, 그러면 그림은 생성 결과를 그대로 사용합니다.
- **표제 한 줄만 필요하다면** 그림 위에 텍스트 레이어만 얹습니다. 판 전체를 쓰지 않고 HTML로 글자만 그려 합성하면 룩은 그대로고 오타 위험도 없습니다.
- **표나 수치가 여럿이면** 그때 판을 사용합니다. 행과 열이 맞아야 하는 문안은 판이 제일 낫습니다.

세 장을 다시 만들 때는 커버만 표제를 얹고, 나머지 둘은 생성 결과를 그대로 쓰고 캡션은 본문에 뒀습니다. 글자 오타는 여전히 0이고, 세 장은 서로 다르게 생겼습니다.

05

라벨 좌표는 결국 눈으로 맞춥니다

판 경로를 쓰기로 했다면 알아 둘 것이 하나 더 있습니다. 그림 위 라벨의 자리는 퍼센트 좌표로 직접 적어야 합니다.

```
{"x": 11, "y": 26, "text": "실체가 있다"}
```

렌더러는 그림 안에 무엇이 어디 있는지 모릅니다. 그러니 좌표를 지능적으로 피해 주지 않습니다. 이번에도 첫 렌더에서 라벨 하나가 그림 한가운데 놓인 봉투를 통째로 덮었습니다. 문법 오류도, 경고도 없습니다. 렌더는 성공하고 그림만 못 쓰게 됩니다.

그래서 이 경로에는 렌더 결과를 사람이 보는 단계가 반드시 붙습니다. 좌표를 옮기고 다시 렌더하는 왕복이 컷당 한두 번은 생깁니다. "글자가 안 깨진다"를 얻는 대신 "자리를 손으로 맞춘다"를 지불하는 거래입니다.

컷이 늘수록 이 비용이 커집니다. 그래서 모든 컷에 판을 쓰지 않는 편이 낫습니다.

06

판이 못 내는 룩은 따로 짭니다

이번에 커버는 어두운 배경이 필요했습니다. 흰 표제를 얹어야 했으니까요. 그런데 저희 렌더러의 판은 흰 문서 시트만 그립니다. 다크 판은 지원하지 않습니다.

방법은 둘입니다. 렌더러에 다크 테마를 더하거나, 이 컷만 따로 짜거나. 컷 하나 때문에 렌더러를 고치는 건 과했습니다. 그래서 커버용 HTML을 30줄쯤 짜고 헤드리스 브라우저로 찍었습니다.

```
#sheet { position: relative; width: 1600px; height: 900px; } #sheet img { position: absolute; inset: 0; object-fit: cover; } #scrim { background: linear-gradient(90deg, rgba(8,9,11,.94), transparent 62%); }
```

그림을 깔고, 글자 쪽에만 어둠을 덧씌우는 층을 하나 얹고, 그 위에 텍스트를 놓습니다. 글자는 여전히 폰트로 그려지니 오타 위험은 판 경로와 같습니다. 판이 주는 보증만 가져오고 판의 생김새는 안 가져오는 방법입니다.

이게 앞서 말한 세 번째 선택지의 실물입니다. 표제 한 줄이면 판 전체가 필요 없습니다. 텍스트 레이어 하나면 됩니다.

07

그래도 마지막에는 기계로 검사합니다

경로를 어떻게 고르든 검사는 남습니다. 두 가지를 봅니다.

생성 컷에 글자가 섞였는지. 프롬프트에 글자 금지 구를 붙여도 모델이 간판이나 계기판에 문자를 그려 넣는 일이 있습니다. 그래서 생성 결과를 글자 검출로 한 번 훑습니다. 확정 검출이 0이 아니면 그 컷은 다시 만듭니다. 사람 눈은 배경의 작은 글자를 잘 놓칩니다.

판 경로로 얹은 글자가 문안과 같은지. 폰트로 그리니 변형은 없지만, 문안 자체에 오타가 있으면 그대로 그려집니다. 그래서 문안은 본문에서 복사해 오고, 렌더 뒤에 이미지 속 글자를 전수로 다시 읽습니다.

이 두 검사는 서로를 대신하지 못합니다. 앞은 "안 넣기로 한 글자가 들어왔나"를 보고, 뒤는 "넣기로 한 글자가 맞나"를 봅니다. 경로를 섞어 쓰면 컷마다 어느 검사를 걸어야 하는지도 달라지니, 그것도 함께 적어 둡니다.

08

일반화하면

이번 일은 이미지 이야기지만, 형태는 훨씬 흔합니다.

어떤 품질을 보증하려고 수단을 먼저 고르면, 그 수단이 결과물의 성격을 함께 정합니다.

타입 안정성을 보증하려고 코드 생성기를 도입하면 코드 모양이 생성기를 닮습니다. 재현성을 보증하려고 컨테이너를 사용하면 배포 구조가 컨테이너를 닮습니다. 일관성을 보증하려고 템플릿을 사용하면 문서가 전부 템플릿을 닮습니다.

이게 나쁘다는 말이 아닙니다. 보증은 실제로 얻어집니다. 다만 **따라오는 것을 안 보고 지나가기 쉽습니다.** 보증은 눈에 보이고, 성격은 안 보이니까요.

그래서 물어야 할 것은 이겁니다. **이 수단을 고르면 결과물의 무엇이 함께 정해지는가.** 그게 받아들일 만한 것인가.

09

덤으로 밝은 것 둘

같은 작업에서 걸린 자리를 적어 둡니다. 둘 다 오류를 안 냅니다.

프롬프트 묶음을 읽는 정규식이 형식을 하나만 받습니다. 생성 스크립트가 `## 1. \파일명.png`·비율·트랙`

A` 형태의 제목만 인식하는데, 저는 트랙 자리에 스타일 이름을 적었습니다. 결과는 "생성할 항목이 없다"에 종료 코드 0입니다. 실패로 안 보여서 한 번 더 돌릴 뻔했습니다.

이미지 검사 도구가 파이썬 경로 때문에 죽습니다. PYTHONPATH가 설정된 셀에서 돌리면 numpy를 소스 디렉터리에서 불러오려다 실패합니다. `env -u PYTHONPATH`를 앞에 붙이면 됩니다. 도구가 나쁜 게 아니라 셀이 오염돼 있던 것입니다.

둘 다 사소하지만, 사소한 것이 조용히 실패하면 결과물이 없는 채로 다음 단계로 넘어갑니다.

10

정리

- 이미지에 한글을 넣으려면 글자를 생성 모델에서 떼어내는 편이 안전합니다.
- 다만 그 방법이 룩을 함께 정합니다. 판을 사용하면 결과물은 판을 닮습니다.
- 순서를 뒤집습니다. 룩을 먼저 고르고, 글자 보장은 그 안에서 푸는 순서입니다.
- 컷마다 다르게 가도 됩니다. 표제 한 줄이면 텍스트 레이어만, 표와 수치가 많으면 판으로.

AI로 무언가를 만들 때 도구를 먼저 고르는 일이 맞습니다. 되는 방법을 아니까요. 그런데 그 도구가 만들어 낼 수 있는 모양이 곧 만들 수 있는 전부가 됩니다.

여러분은 어떤 품질을 보증하려다 결과물의 성격까지 넘겨준 적이 있으신가요?

WACE(웨이스)는 공장이 스스로 판단하고 멈추지 않게 하는 제조 AI 운영체제 VEXPLOR를 만드는 회사다.



원문 blog.vexplor.com/post/the-path-you-pick-decides-the-look

발행 2026년 9월 8일 · 방동걸 (주)웨이스 대표이사

회사 주식회사 웨이스 (WACE Inc.) · 대표이사 방동걸 · 사업자등록번호 439-87-02852

주소 (18462) 경기 화성시 동탄구 동탄영천로 150, 에이동(A동) 1702·1706호 (영천동, 실리콘밸리동탄)

연락처 전화 070-4730-0629 · contact@wace.me

홈페이지 wace.me · linkedin.com/company/wace-inc



원문



홈페이지



링크드인