Kiro

📌 Kiro의 핵심 원칙

1. **마크다운(Markdown)의 모든 문법을 수용**한다. → 기존 문서를 쉽게 Kiro로 전환 가능해야 함
2. **스타일은 명시적으로 선언하고, 적용하며, 종료한다.**
3. **중첩 스타일은 허용하지 않는다.**
4. **리소스 삽입은 통일된 문법으로 표현하며, 자동 추론되지 않는다.**
5. **문서 원문을 읽어도 가독성과 구조가 명확해야 한다.**
6. **이미지, 링크는 반드시 Kiro 전용 문법을 사용해야 하며, 기존 마크다운 방식은 사용하지 않는다.**
7. **인용은 `|` 기호를 사용하며, 기존 마크다운의 `>`는 사용하지 않는다.**
8. **정의되지 않은 스타일 형식이나 기호 등은 모두 그대로 출력한다.** (스타일이 시작되지 않았는데 '<>' 가 있어도 그대로 출력.)
9. 공백 또한 의미의 구분이 아닌 레이아웃의 일부로 받아들인다. 예를 들어, 스타일 테그 앞, 리스트 마크다운 앞 공백, 연속된 개행 등을 모두 유지하여 출력한다.

🖌️ 기본 스타일 문법

🔷 스타일 적용

- `[스타일명] ... <>`
- 예시: `[tip] 이건 팁입니다 <>`
- 중첩 불가 → 새 스타일이 열리면 이전 스타일 자동 종료

🔷 스타일 선언

```
<style>
[스타일명] = {마크다운구조} [스타일속성1] [스타일속성2]
:하위 = [속성 덮어쓰기]
::하위하위 = [속성 덮어쓰기]
<>
```

🔷 스타일명 규칙

- 한글, 영문, 숫자, `_` 사용 가능
- 대소문자 구분 없음
- 공백 불가, `:`로 계층 구조 허용
- 최대 계층 깊이: **2단계까지 허용** (`[기본:하위::하위하위]`)

> **계층이란?**
> 최상위에서 최하위로 이어지는 상속 스타일.
> 적용 순서는 부모 + 상속 1 + 상속 2로 이어지며, 중복되는 스타일은 덮어써짐.

🔷 스타일 속성 키워드

문법	의미	예시
`[#색상]`	텍스트 색상 지정	`[#red]`, `[#004FFF]`
`[=폰트명]`	글꼴 지정	`[=Pretendard]`
`[+아이콘]`	접두사 추가	`[+💡]`, `[+🚨]`
`[$tailwind]`	Tailwind 클래스 지정	`[$text-lg]`, `[$bg-gray-100]`
`{마크다운구조}`	구조 삽입	`{#}`, `{##}`, `{> ###}`, `{-}`

🔪 폰트 적용 문법

🔷기본 폰트 종류

1. 구글폰트
    a. Noto Sans
    b. Roboto

c. JetBrains Mono

2. 고딕
    a. Pretendard
    b. MonoplexKR
3. 스타일
    a. RIDIBatang
    b. GowunDodum

🌐 외부 리소스 삽입 문법

🔷 문법: `@[타입]: URL ! 설명`

- 항상 `@` 기호로 시작하며, **타입을 명시**해야 함
- `:` 기호 뒤에는 리소스의 **URL**
- `!` 기호 뒤에는 해당 리소스의 **설명**
- **닫는 기호 없음** (한 줄이 하나의 블록)
- 설명은 자동으로 `title`, `alt`, `figcaption` 등에 활용됨

🔷 지원 타입

타입	설명	렌더링 요소
`img`	이미지	`<img> + <figcaption>`
`video`	동영상	`<video controls>`
`audio`	음성	`<audio controls>`
`link`	일반 링크	`<a href>`

🔷 문법적 특징

- **기존 마크다운 문법 (`![]()`, `[]()`)은 사용 불가**
    - Kiro는 스타일과 구조의 **명시성**을 중시하며

      `@img:`와 같은 **타입 선언 방식**으로

      문서의 역할과 구성 요소를 직관적으로 파악할 수 있도록 설계됨.
- **파일 확장자에 의존하지 않음**

- `.mp4`, `.jpg` 등으로 판단하지 않고,
  작성자가 직접 타입을 명시해야 함.

🔷 예시

```
@img: https://example.com/cat.jpg ! 고양이입니다
@video: https://example.com/intro.mp4 ! 소개 영상
@audio: https://example.com/bgm.mp3 ! 배경음악
@link: https://kiro.dev ! Kiro 공식 사이트
```

💬 인용 문법

- 마크다운과 달리 `>` 대신 `|` 사용
- 한 줄 또는 여러 줄 인용 가능
  - 여러 줄 렌더링 시에는, 하나의 테그와 개행문자로 렌더링

```
| 이건 인용이에요
| 한 줄 더
```

📚 리스트 문법

🧩 기본 규칙

1. **모든 리스트는 로 시작한다.**
2. 계층 구조는 `.`**(점)** 으로 구분하며, **리스트 키는 반드시** `.`**으로 끝나야 한다.**
3. 리스트 키는 **숫자, 알파벳 대문자/소문자** 조합이 가능하다. (예: `1.`, `1.1.`, `A.`, `2.B.1.`)
4. **숫자 리스트 이후에 일반 리스트는 불가능하다.** (→ `1.` → ❌)
5. 반대로, **숫자 리스트는 중간에 자유롭게 등장 가능**하다. (`1.1.` 다음에 `1.3.`이나 `1.1.`도 가능)
6. 들여쓰기는 절대 사용하지 않는다.

🧾 예시

```
- 소개
-1. 시작하기
-2. 주요 개념
-2.1. 스타일 시스템
-2.1.A. 하위 스타일
-2.2. 리소스 삽입
-3. 마무리
```

🎨 렌더링 방식

- `<ul class="list-none">` 또는 `<div>` 구조로 출력
- 각 항목에 **계층 키를 `<span>`으로 명시적으로 출력**
- 들여쓰기 없이 **같은 수준의 리스트처럼 나열**

```
<ul class="list-none space-y-1">
  <li><span class="text-gray-400 font-mono">1.</span> 시작하기</li>
  <li><span class="text-gray-400 font-mono">2.1.A.</span> 하위 스타일</li
>
</ul>
```

🔽 토글 문법

🧩 기본 규칙

1. 토글은 `>` 기호로 시작한다.
2. 토글의 깊이에 따라 `>`, `>>`, `>>>` 등으로 계층 표현이 가능하다.
3. `### > 제목`과 같이 **헤딩과 결합하면 summary가 된다.**
4. 토글 본문은 다음 빈 줄이나 새로운 블록 요소(헤딩, 리스트 등)가 나오기 전까지 자동 포함된다.
5. 토글 내에 다른 토글을 중첩 가능 (`>>`, `>>>` 등), 최대 3단계 권장
6. 들여쓰기 대신 **기호(`>`) 반복으로 계층 표현**

🧾 예시

> 자주 묻는 질문

> Kiro는 뭔가요?
스타일 중심의 문서 포맷입니다.

>> 더 자세히 말해주세요
스타일 선언과 명시적 구조를 갖춘 문서 포맷입니다.

✅ 마크다운 완전 호환 요소

기능	예시	Kiro에서의 상태
헤딩	`# 제목`	✅ 유지
리스트	`- 항목` / `1. 항목`	✅ 유지
굵게	`**텍스트**`	✅ 유지
기울임	`_텍스트_`	✅ 유지
인라인 코드	`` `code` ``	✅ 유지
코드 블록	```` ```lang\ncode\n``` ````	✅ 유지
링크	`[텍스트](URL)`	❌ 사용하지 않음 → `^url ! 설명^` 사용
이미지	`![alt](URL)`	❌ 사용하지 않음 → `^img.jpg ! 설명^` 사용
구분선	`---`	✅ 유지
주석	`# 내부 메모`	✅ 렌더러 무시

🧩 스타일 시스템 확장 규칙

🔷 `[!global]` — 문서 전체에 적용되는 기본 스타일

- `<style>` 블록 내에서 `[!global]`을 선언하면, 문서 전체의 기본 스타일로 사용된다.
- 해당 스타일은 별도로 `[스타일명]`이 지정되지 않은 텍스트 블록에만 적용된다.
- `+아이콘`, `{마크다운 구조}` 속성은 사용할 수 없다.

```
<style>
[!global] = [#333] [=Pretendard] [$text-base]
<>
```

- 적용 가능한 속성:
  - 텍스트 색상: `[#색상]`
  - 글꼴: `[=폰트명]`
  - Tailwind 클래스: `[$...]`
- 렌더링 시, 해당 스타일은 `<body>` 또는 전체 wrapper에 Tailwind 클래스 형태로 삽입된다.
  - 예: `class="text-base text-[#333] font-pretendard"`

🎨 렌더링 시 적용되는 기본 스타일 원칙

🔶 Kiro는 기능적으로 필요한 시각 요소만 최소한으로 적용한다.

- Kiro는 **문서의 표현을 최대한 사용자 정의에 맡긴다.**
- 기본 디자인은 *문서 구조나 의미 전달에 꼭 필요한 범위*로 한정된다.

🔶 포함되는 Tailwind 기본 스타일

🔶 포함되지 않는 스타일

요소	이유
문서 배경색, 그림자, 테마	사용자 정의에 맡김
스타일 이름 없는 `+아이콘`, `{구조}`	의미 없는 장식 방지
마우스 hover, animation 등	문서의 기능성과 무관함

✨ 기본 HTML 템플릿 구조 (렌더링 기준)

```
<body class="font-sans text-base leading-relaxed text-gray-800">
  <main class="max-w-3xl mx-auto px-4 py-8">
    <!-- Kiro 렌더링 결과 삽입 -->
  </main>
</body>
```

- 위 구조는 *최소한의 레이아웃 + 가독성 확보*를 위한 것으로,
  사용자 테마가 존재할 경우 해당 클래스로 덮어쓴다.
- 커스터마이징이 필요한 경우, `[!global]` 문법을 통해 제어한다.

🧠 렌더러 기본 처리 로직 요약

- `[스타일]` → HTML span/div로 감싸며 선언된 속성 적용
- `@link: URL ! 설명` → 확장자에 따라 자동 렌더링 분기
- `|` 시작 줄 → blockquote 처리
- 마크다운 구조 → 원본대로 파싱
- 주석 `#` 시작 줄은 렌더 무시

🎯 Kiro의 철학

> Kiro는 의미 있는 기록을 위한 도구입니다.
>
> 사용자가 의식적으로 스타일을 설계하고, 문서의 구조를 명확하게 설계하며,
>
> 시각적으로 읽기 좋은 문서 그 자체를 목표로 합니다.