


# [사용자] Next.js 15에서 metadata 설정하기

☰ 구현 어려움 정도	☆☆
☑ 구현 완료	☑
📅 기간	@December 27, 2024 → December 28, 2024
☑ 정리 완료	☑

## Optimizing: Metadata

Use the Metadata API to define metadata in any layout or page.

 <https://nextjs.org/docs/app/building-your-application/optimizing/metadata>

Optimizing: Metadata

메타데이터는 페이지의 HTML `<head>` 태그 내에 삽입되는 정보로, 검색엔진 최적화(SEO), 소셜 미디어 공유, 페이지 정보 표시를 위해 사용됩니다.

Next.js에서는 페이지의 메타데이터를 설정할 수 있는 방법에는 정적 메타데이터(Static Metadata)와 동적 메타데이터(Dynamic Metadata)가 있습니다.

## Static Metadata

정적 메타데이터는 변경되지 않는 정보를 기준으로 설정하는 메타데이터입니다. 주로 사이트의 기본적인 정보나 변경되지 않는 데이터를 사용해 생성됩니다.

- 예) 사이트의 기본 제목, 설명, 이미지

```
// layout.tsx나 page.tsx에 설정
import type { Metadata } from "next";

export const metadata: Metadata = {
  title: "가상자산 플랫폼 | Coin Market",
  description: "Coin Market에서 코인 시세를 확인해보세요.",
};
```

## Dynamic Metadata

동적 메타데이터는 변경되는 정보를 기준으로 설정하는 메타데이터입니다. 페이지의 내용이나 url 파라미터에 따라 메타데이터가 달라지게 됩니다.

- 예) 특정 코인 상세페이지에 맞는 제목, 설명 등
- 실시간으로 변경되는 데이터에 따라 메타데이터를 설정 가능
- 주로 동적 콘텐츠(사용자의 입력, API data, url 파라미터 등)에 기반한 메타데이터를 생성하는데 사용
- SEO 최적화와 소셜 미디어 공유 시, 각 페이지에 맞는 정보 제공 가능

```
// app/[id]/layout.tsx
import { ReactNode } from 'react';
import { Metadata } from 'next';
import axios from 'axios';

type LayoutProps = {
  children: ReactNode;
  params: Promise<{ id: string }>;
};

export async function generateMetadata({ params }: LayoutProps): Promise<Metadata> {
  const { id } = await params; // 페이지 경로가 [id]이기 때문에 동적 경로 사용 가능
```

```

const coinListAll = await axios.get(`API 주소`);
const product = coinListAll.data.find((item: any) => item.symbol === id); // 사용할 값 필터링

if (!product) {
  return {
    // API 불러오지 못했을 때 처리
  };
}


return {
  title: `${product?.name} | ₩ ${product?.total_volume.toLocaleString('ko-KR')}` || 'Coin Market',
  description: `${product?.name}은 현재 ₩ ${product?.total_volume.toLocaleString('ko-KR')}입니다. Coin Mark
  openGraph: {
    images: [product?.image || 'https://raw.githubusercontent.com/wwowww/crypto/9630be17f62e7ebf89d447f09
  },
};
}

```

하지만 과도하게 사용 시 아래 문제가 발생할 수 있다고 합니다.

- 성능 저하
  - 동적으로 메타데이터를 생성하려면 서버에서 데이터를 가져오거나 클라이언트에서 javascript로 처리해야 하기 때문에 성능에 영향을 줄 수 있음
- 캐싱 문제
  - 동적 메타데이터는 캐시가 불가능한 경우가 대부분 → 서버에 불필요한 요청을 발생시킬 수 있음
- 복잡성 증가
  - 설정이 복잡하고 코드가 길어짐

SEO 및 소셜 미디어 공유 시에 더 많은 이점(최적화)이 있으니 적절하게 사용하는 것이 좋겠습니다!

 다음 글: [\[사용자\] Next.js SEO\(검색 엔진 최적화\) 적용하기, 색인 차단 오류](#)