14. 폰트와 이미지 최적화

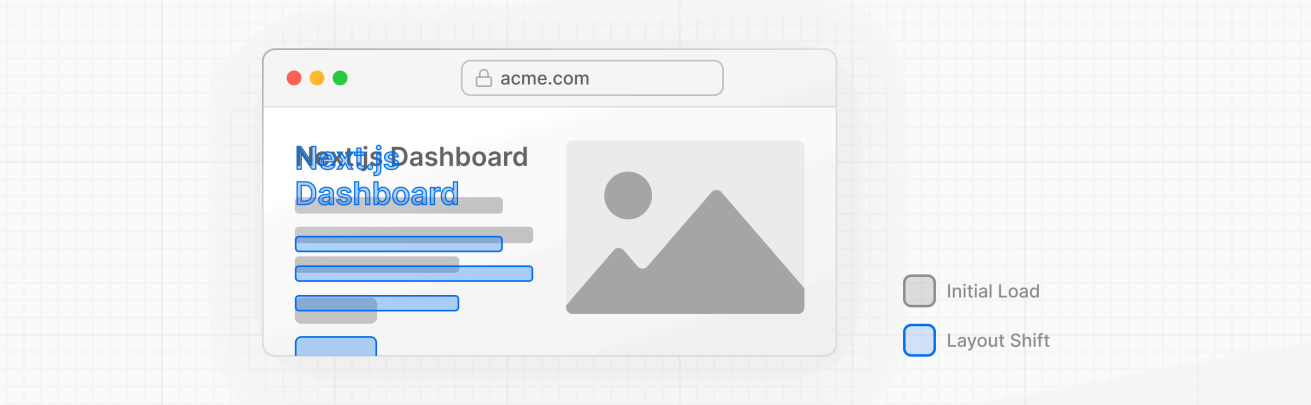
이번 챕터에서는

- 커스텀 폰트 추가하기
- 히어로 이미지 추가하기
- Next.js에서 폰트와 이미지를 최적화하는 방법 배우기

폰트를 최적화해야 하는 이유

웹사이트 디자인에서 폰트는 큰 역할을 합니다.
하지만 커스텀 폰트를 사용하면 별도의 폰트 파일을 다운로드해야 해서
성능에 영향을 줄 수 있습니다.

특히 **CLS(Cumulative Layout Shift)** 문제를 유발할 수 있습니다.
(초기에는 시스템 기본 폰트로 렌더링되었다가,
나중에 커스텀 폰트로 교체되면서 레이아웃이 밀리는 현상)



Next.js는 `next/font` 모듈을 통해 빌드 타임에 폰트를 다운로드하고 정적 자산으로 제공하기 때문에 이런 문제를 자동으로 해결합니다.

기본 폰트 추가하기

1. `/app/ui/fonts.ts` 파일을 새로 만듭니다.
2. `next/font/google` 모듈에서 `Inter` 폰트를 가져옵니다:

```
import { Inter } from 'next/font/google';
```

```
export const inter = Inter({ subsets: ['latin'] });
```

3. 그리고 `/app/layout.tsx` 파일의 `<body>` 태그에 적용합니다:

```
import '@/app/ui/global.css';
import { inter } from '@/app/ui/fonts';

export default function RootLayout({ children }: { children:
React.ReactNode }) {
  return (
    <html lang="en">
      <body className={`${inter.className} antialiased`}>{children}</body>
    </html>
  );
}
```

> `antialiased` 클래스는 글꼴 렌더링을 부드럽게 만듭니다. 선택사항이지만 가독성이 좋아집니다.

실습: 보조 폰트 추가하기

`fonts.ts` 파일에 보조 폰트로 `Lusitana`를 추가해봅시다.

힌트:

- `Lusitana` 폰트를 가져올 때 subsets는 `'latin'`
- weight는 `400`, `700` (보통/볼드)

```
// /app/ui/fonts.ts
import { Inter, Lusitana } from 'next/font/google';

export const inter = Inter({ subsets: ['latin'] });

export const lusitana = Lusitana({
  weight: ['400', '700'],
  subsets: ['latin'],
});
```

```
// /app/page.tsx

import AcmeLogo from '@/app/ui/acme-logo';
import { ArrowRightIcon } from '@heroicons/react/24/outline';
import Link from 'next/link';
import { lusitana } from '@/app/ui/fonts';
```

```
export default function Page() {
  return (
    // ...
    <p
      className={`${lusitana.className} text-xl text-gray-800 md:text-3xl
md:leading-normal`}
    >
      <strong>Welcome to Acme.</strong> This is the example for the{' '}
      <a href="https://nextjs.org/learn/" className="text-blue-500">
        Next.js Learn Course
      </a>
      , brought to you by Vercel.
    </p>
    // ...
  );
}
```

추가로, `/app/page.tsx` 파일에서
`<AcmeLogo />` 컴포넌트가 Lusitana를 사용하는데 주석처리 되어 있던 것을 이제 주석 해제할 수 있습니다.

```
// ...

export default function Page() {
  return (
    <main className="flex min-h-screen flex-col p-6">
      <div className="flex h-20 shrink-0 items-end rounded-lg bg-blue-500
p-4 md:h-52">
        <AcmeLogo />
        {/* ... */}
      </div>
    </main>
  );
}
```

이미지를 최적화해야 하는 이유

Next.js에서는 `/public` 폴더에 이미지를 저장하고 접근할 수 있습니다.

기존 HTML에서는 이렇게 이미지를 추가했겠죠:

```
<img src="/hero.png" alt="대시보드 프로젝트 스크린샷" />
```

하지만 이 방법은:

- 반응형(Responsive) 처리가 안되고
- 다양한 디바이스 크기에 맞춰 사이즈 조절이 필요하며
- 로딩 시 레이아웃 쉬프트가 발생할 수 있고
- 뷰포트 밖에 있는 이미지를 수동으로 Lazy Load 해야 합니다.

이 모든 문제를 `next/image` 컴포넌트가 자동으로 해결해줍니다!

`<Image>` 컴포넌트

`next/image` 컴포넌트는 HTML `<img>`의 확장판입니다.

장점:

- 자동으로 레이아웃 쉬프트 방지
- 디바이스에 맞게 크기 조절
- Lazy Loading 기본 적용
- WebP, AVIF 등 최신 이미지 포맷 지원

데스크탑용 히어로 이미지 추가하기

`/public` 폴더에는

- `hero-desktop.png`
- `hero-mobile.png`

두 가지 이미지가 준비되어 있습니다.

`/app/page.tsx` 파일에서
`next/image`를 import 한 다음, 데스크탑 이미지를 추가합니다:

```
import Image from 'next/image';

export default function Page() {
  return (
    <div className="flex items-center justify-center p-6 md:w-3/5 md:px-28
md:py-12">
      <Image
        src="/hero-desktop.png"
        width={1000}
        height={760}
        className="hidden md:block"
        alt="대시보드 프로젝트 데스크탑 버전 스크린샷"
```

```
      />
    </div>
  );
}
```

- 데스크탑에서는 `md:block`으로 보이게 하고
- 모바일에서는 `hidden`으로 숨깁니다.

> 좋은 습관: width/height를 지정해 레이아웃 쉬프트를 방지합니다.

실습: 모바일용 히어로 이미지 추가하기

방금 추가한 데스크탑 이미지 아래에, 모바일용 이미지를 추가합니다.

- `hero-mobile.png` 사용
- width: 560px
- height: 620px
- 모바일에서만 보이고 데스크탑에서는 숨기기

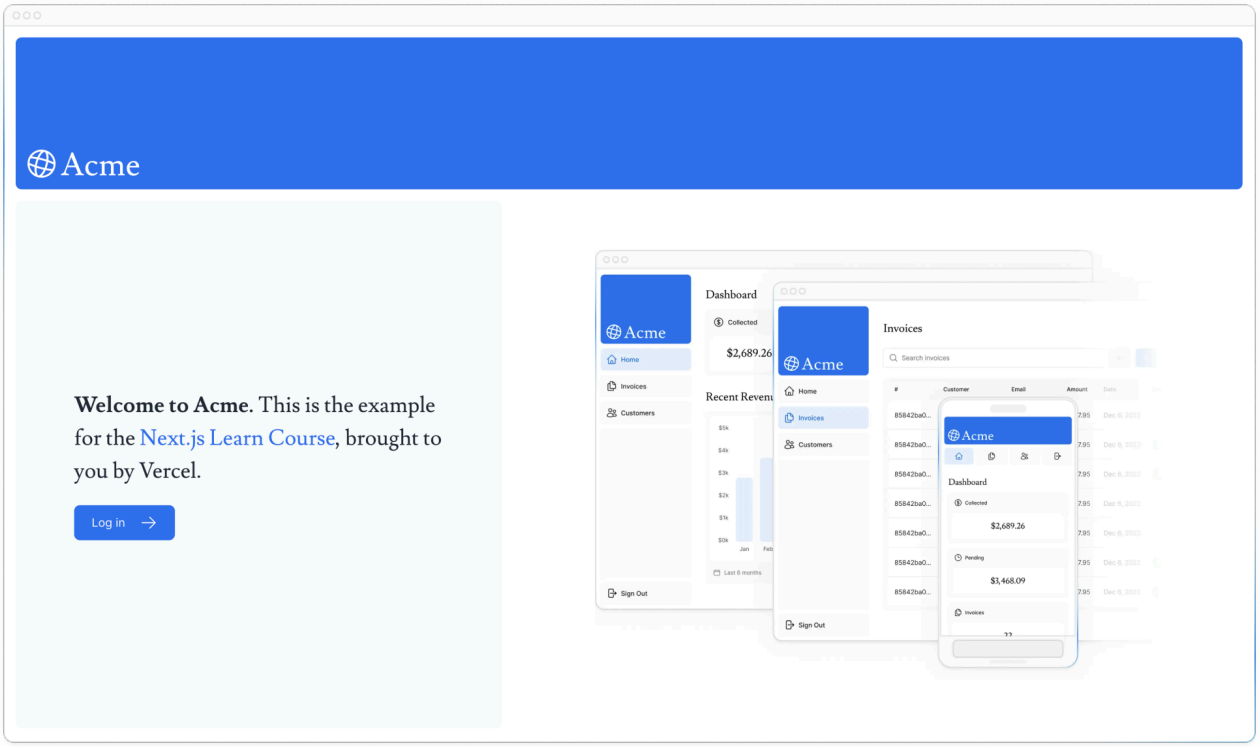
힌트: Tailwind 클래스로 `block md:hidden` 조합 사용.

```
// /app/page.tsx
import AcmeLogo from '@/app/ui/acme-logo';
import { ArrowRightIcon } from '@heroicons/react/24/outline';
import Link from 'next/link';
import { lusitana } from '@/app/ui/fonts';
import Image from 'next/image';

export default function Page() {
  return (
    // ...
    <div className="flex items-center justify-center p-6 md:w-3/5 md:px-28
md:py-12">
      {/* Add Hero Images Here */}
      <Image
        src="/hero-desktop.png"
        width={1000}
        height={760}
        className="hidden md:block"
        alt="Screenshots of the dashboard project showing desktop version"
      />
      <Image
        src="/hero-mobile.png"
        width={560}
```

```
          height={620}
          className="block md:hidden"
          alt="Screenshot of the dashboard project showing mobile version"
        />
      </div>
      //...
    );
}
```

결과 화면



(※ 복원된 이미지로 정상 표시)

추천 읽을거리

- Image Optimization 공식 문서
- Font Optimization 공식 문서

🎉 Chapter 3 완료!

홈페이지에:

- 커스텀 폰트 추가 완료
- 히어로 이미지 추가 완료
- 성능 최적화 적용 완료

다음은?

👉 Chapter 4: 레이아웃과 페이지 만들기

(중첩 레이아웃과 파일 기반 라우팅을 배우게 됩니다!)

요약

- next/font로 커스텀 폰트 최적화
- next/image로 이미지 최적화
- 레이아웃 쉬프트 방지
- 반응형 히어로 이미지 적용

다음 **Chapter 4 (Creating Layouts and Pages)** 도 바로 이어서 번역해드릴까요? 🚀
진행할까요? 🎯
(필요하면 여기까지 내용 zip 파일로도 묶어드릴 수 있어요!)