

17. LCP

Largest Contentful Paint



Core Web Vitals

홈 > 검색 센터 > 문서 > 고급 검색엔진 최적화

평가 및 리뷰



Google 검색결과와 페이지 환경 이해

의견 보내기

페이지 환경은 순수한 정보 값을 넘어 사용자가 웹페이지에서 상호작용 경험을 어떻게 인식하는지를 측정하는 일련의 신호입니다. 환경 신호에는 로드 성능, 상호작용, 페이지의 시각적 안정성에 관한 실제 사용자 환경을 측정한 결과를 확인할 수 있는 [핵심 성능 보고서](#)가 포함됩니다. 또한, 기존의 Google 검색 신호인 [모바일 친화성](#), [세이프 브라우징](#), [HTTPS](#), [방해가 되는 전면 광고 가이드라인](#)도 포함됩니다.

페이지 환경이 순위에 영향을 미치는 방식 이해

[로드맵에서](#) 언급된 변경사항이 출시되고 나면 **Google 검색결과 생성 시 Google에서 고려하는 수많은 신호가 페이지 경험에 포함됩니다.**

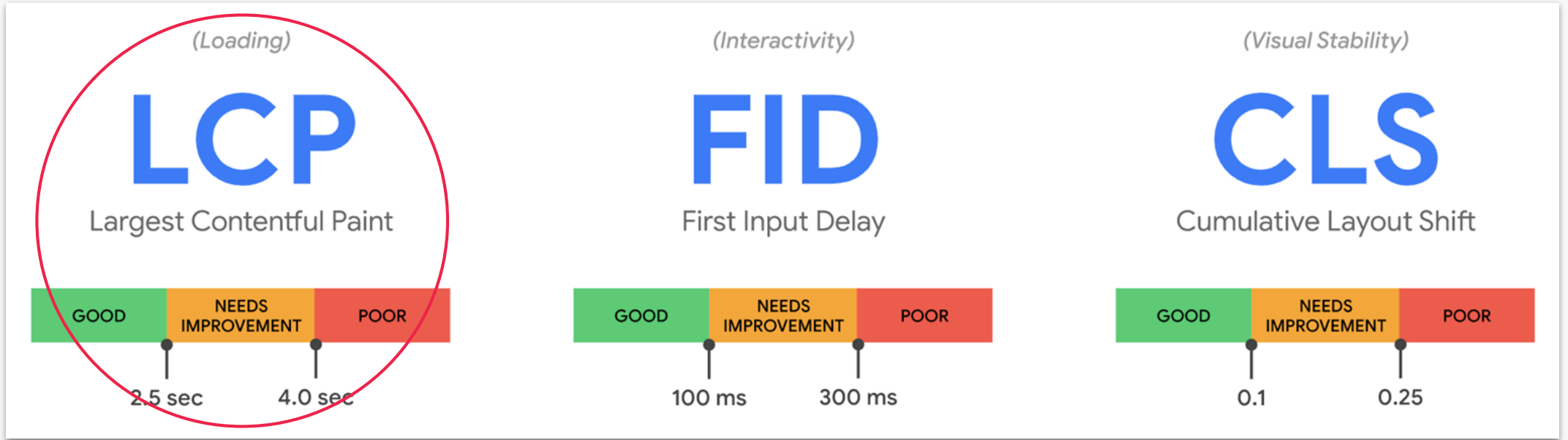
페이지 환경도 중요하지만, Google에서는 페이지 환경이 기대치에 미치지 못하더라도 전반적으로 최고의 정보를 제공하는 페이지를 기준으로 계속 순위를 매길 예정입니다. 훌륭한 페이지 환경보다는 뛰어난 페이지 콘텐츠가 더 중요합니다. 하지만, 다루는 콘텐츠가 비슷한 페이지가 여러 개인 경우 Google 검색에 표시되는 데 페이지 환경이 훨씬 더 중요하게 작용할 수 있습니다.



언제부터 적용되나요? 첫 번째 페이지 경험 업데이트는 **2021년 6월 중순부터 단계적으로 출시**되며 2021년 8월 말에 완료될 예정입니다. 이 페이지에 설명된 도구와 문서를 사용하여 사이트 평가를 시작하는 것이 좋습니다.

Article - <https://t.ly/OLWM>







Core Web Vitals



Core Web Vitals

Core Web Vitals

Now in your favorite developer tools

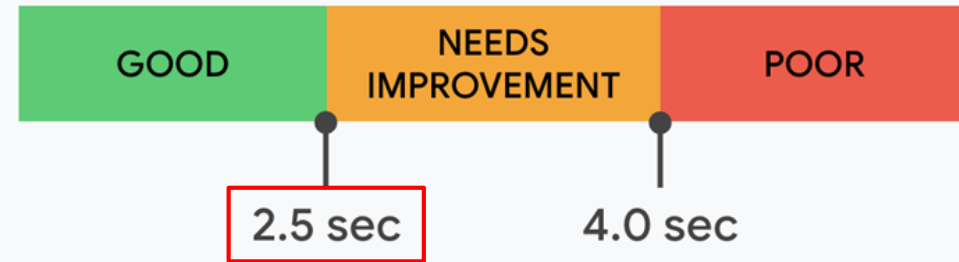
	LCP	FID	CLS
 PageSpeed Insights	✓	✓	✓
 Chrome UX Report Brand new API, BigQuery and Dashboard	✓	✓	✓
 Search Console	✓	✓	✓
 Chrome DevTools	✓	TBT	✓
 Lighthouse	✓	TBT	✓
 Web Vitals Extension	✓	✓	✓

LCP = Largest Contentful Paint, FID = First Input Delay, CLS = Cumulative Layout Shift, TBT = Total Blocking Time

LCP

LCP

Largest Contentful Paint



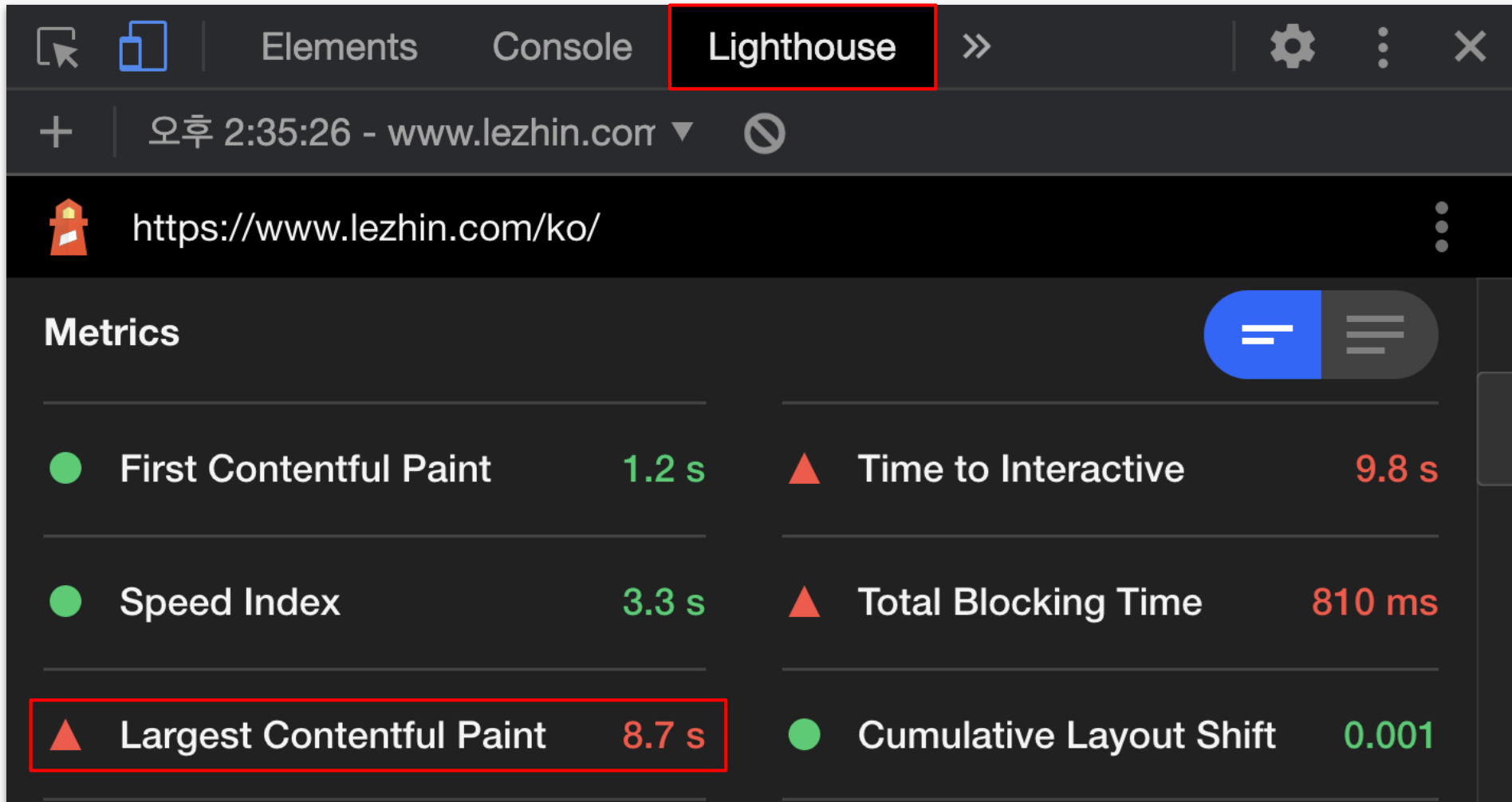
“

LCP

가장 큰 덩어리 콘텐츠



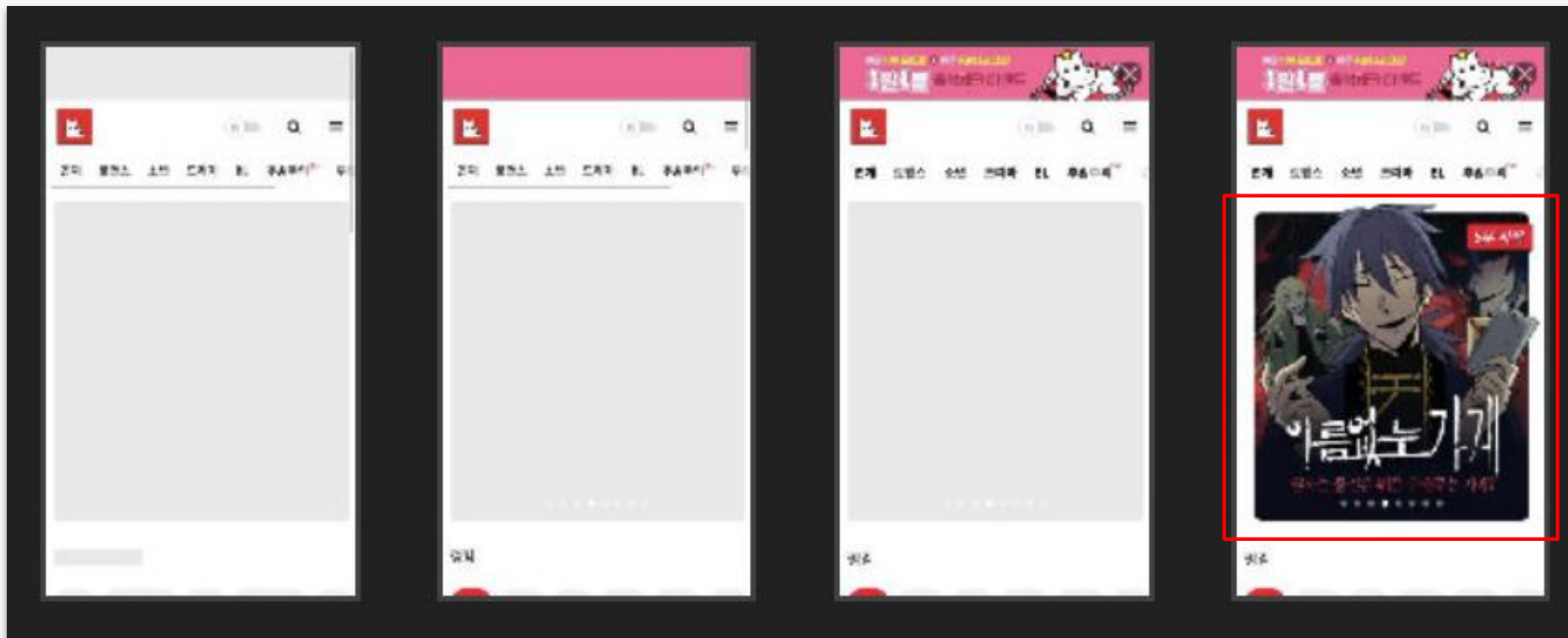
가장 큰 콘텐츠 그리기



The screenshot shows the Chrome DevTools interface with the Lighthouse panel open. The browser address bar shows the URL `https://www.lezhin.com/ko/`. The Lighthouse panel displays a list of performance metrics. The 'Lighthouse' tab is highlighted with a red box. The 'Largest Contentful Paint' metric is also highlighted with a red box.

Metrics					
●	First Contentful Paint	1.2 s	▲	Time to Interactive	9.8 s
●	Speed Index	3.3 s	▲	Total Blocking Time	810 ms
▲	Largest Contentful Paint	8.7 s	●	Cumulative Layout Shift	0.001

가장 큰 콘텐츠 그리기



가장 큰 콘텐츠 그리기

Opportunity

Estimated Savings

▲ Preload Largest Contentful Paint image


2.25 s ^

Preload the image used by the LCP element in order to improve your LCP time. [Learn more](#).

☐ Show 3rd-party resources (0)

URL

Potential Savings



...media/upperBannerMobile.webp?updated=162...
&width=688 (ccdn.lezhin.com)

2,250 ms

가장 큰 콘텐츠 그리기

LCP

1. 가장 큰 덩어리 콘텐츠를 화면에 표시하는데 걸리는 시간.
2. 웹 페이지에서 가장 큰 덩어리 콘텐츠는 90% 확률로 히어로 이미지.
3. 히어로 이미지를 표시하기 전 발생하는 모든 성능 이슈가 LCP 문제를 해결하는 열쇠.

“

LCP 개선 사례

LCP 3.6초 단축한 이야기



LCP 개선 사례

Before - <https://t.ly/tyMg>



<https://naradesign.github.io/lcp/>



Largest Contentful Paint

4.6 s

After - <https://t.ly/a7uy>



<https://naradesign.github.io/lcp/solution>



Largest Contentful Paint

1.0 s

“

라이브러리 의존 줄이기

jquery, lodash, normalize...

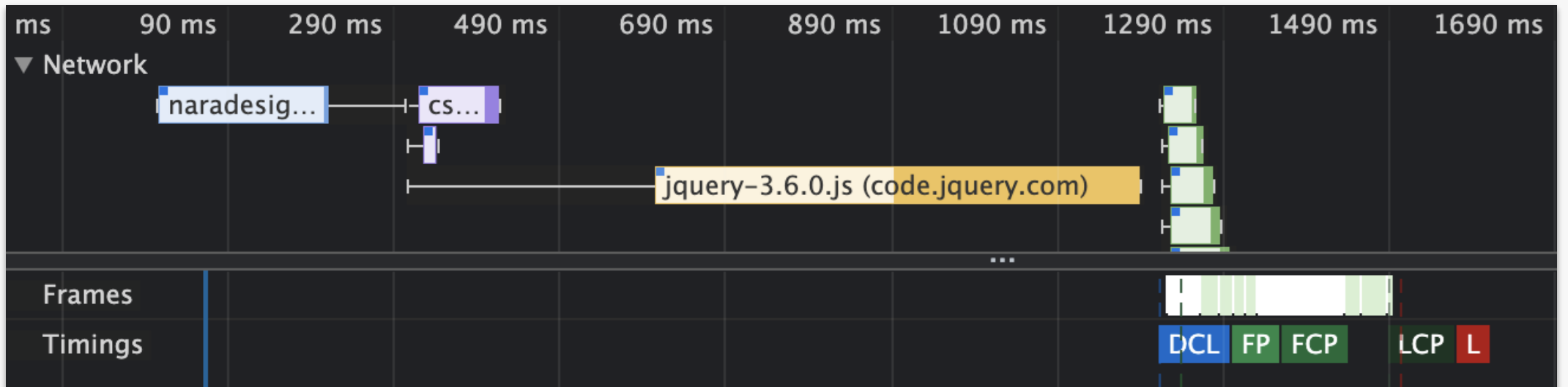


As is

```
<head>
```

```
  <script src="jquery-3.6.0.js"></script> 🤖
```

```
</head>
```



To be

```
<body>
```

```
...
```

```
<script src="jquery-3.6.0.js"></script> ☐
```

```
</body>
```

```
<body>
```

```
...
```

```
<script src="jquery-3.6.0.js"></script> 🙅
```

```
</body>
```

You might not need *.

<https://youmightnotneed.com/> 

You might not need jQuery

YOU MIGHT NOT NEED LODASH

Remove unused CSS

```
<head>
```

```
  <link rel="stylesheet" href="normalize.css"> 🤖
```

```
</head>
```

```
<head>
```

```
  <link rel="stylesheet" href="normalize.css"> 🙅
```

```
</head>
```

“

Preconnect / Preload



웹 폰트 preconnect / CSS preload

```
<head>
```

```
  <link rel="stylesheet" href="*.css"> 🤖
```

```
</head>
```

```
<head> 🙌🙌🙌
```

```
  <link rel="preconnect" href="https://fonts.gstatic.com">
```

```
    <link rel="preload" as="style" href="*.css"  
onload="this.onload=null;this.rel='stylesheet'">
```

```
</head>
```

Preconnect / Preload

<link rel="preconnect">

- 도메인을 알지만 자원의 최종 경로를 모르는 경우 서버와의 연결을 미리 설정.
- DNS(Domain Name Server), TCP(Transmission Control Protocol), TLS(Transfer Layer Security) 왕복에 필요한 시간을 단축.
- 서드 파티 자원 연결에 적합.


Preconnect / Preload

<link rel="preload">

- 필요한 자원을 병렬 다운로드.
- 자원을 로딩하는 동안 렌더링을 차단하지 않음.
- as 속성을 함께 명시해 주어야 한다.

예) as="style", as="script", as="image".

히어로 이미지 preload

```
<head>   
  <link  
    rel="preload" as="image"  
    media="(max-width:640px)" href="small.avif">  
  <link  
    rel="preload" as="image"  
    media="(min-width:641px)" href="large.avif">  
</head>
```

“

Feature detection

Image type / Viewport width



As is

```
<!-- UNOPTIMIZED HERO IMAGE -->  
 🤖
```


To be

<!-- OPTIMIZED HERO IMAGE -->

```
<picture>
  <!-- AVIF && SMALL SCREEN -->
  <source srcset="small.avif" type="image/avif" media="(max-width:640px)">
  <!-- AVIF && LARGE SCREEN -->
  <source srcset="large.avif" type="image/avif">
  <!-- WEBP && SMALL SCREEN -->
  <source srcset="small.webp" type="image/webp" media="(max-width:640px)">
  <!-- WEBP && LARGE SCREEN -->
  <source srcset="large.webp" type="image/webp">
  <!-- FALLBACK -->
  
</picture>
```



“

Image Loading / Decoding



Image Loading / Decoding

```
<!-- UNOPTIMIZED IMAGE -->  
 🤖
```

Image Loading / Decoding

```
<!-- OPTIMIZED IMAGE -->  

```

Image Loading / Decoding

``

뷰포트 밖에 있는 이미지를 로딩하지 않는다.

대략 뷰포트 높이의 1~2배 지점까지 근접하면 로딩 됨.

Demo - <https://mathiasbynens.be/demo/img-loading-lazy>

Image Loading / Decoding

``

이미지 디코딩(복호화)을 병렬 처리.

디코딩을 지연시켜 다른 콘텐츠의 표시 속도가 빨라짐.

Article - <https://t.ly/o2cR>

MDN - <https://developer.mozilla.org/ko/docs/Web/HTML/Element/img>

“

SUMMARY



“ Summary

1. LCP는 뷰포트에 표시하는 가장 큰 콘텐츠 렌더링 성능.
2. 가장 큰 덩어리 콘텐츠를 2.5초 이내로 표시해야 한다.
3. JS/CSS 라이브러리 의존도를 낮추어야 한다.
4. preconnect/preload 속성으로 외부 자원 최적화.
5. photo 요소의 type, media 속성으로 이미지 전송량 최적화.
6. loading/decoding 속성으로 이미지 렌더링 최적화.

“

실습 과제



실습 과제

1. <https://github.com/naradesign/lcp> 저장소를 포크하세요.
2. index.html 파일의 LCP를 개선하세요.
3. <https://github.com/naradesign/lcp> 저장소에 Pull Request를 보내주세요.



참고: 보내주신 Pull Request는 병합하지 않습니다.

Thank you !

