

Browser 모니터링 활용하기

이 문서는 Browser 모니터링 서비스를 이용해 여러분의 웹 애플리케이션의 문제점을 빠르게 파악하고 대응하는 방법에 대한 안내합니다. Browser 모니터링에 대한 기본 안내 문서는 [다음 문서](#)를 참조하세요.

페이지 로드 분석하기

웹 페이지 로드 속도는 사용자 경험에 큰 영향을 미칩니다. 페이지 로드가 느리면 사용자는 페이지를 떠나고 다른 사이트로 이동할 가능성이 높습니다. 이로 인해 사용자는 페이지를 볼 기회를 놓치게 되고 서비스나 제품을 이용하지 않거나 구매하지 않는 경우가 발생할 수 있습니다.

페이지 로드 속도를 최적화하고 사용자 경험을 개선하는 것은 매우 중요합니다. 페이지 로드 분석을 통해 어떤 요소가 페이지 로드를 느리게 만드는지 식별하고, 문제점을 개선하여 최적화된 사용자 경험을 제공할 수 있습니다. 이러한 노력은 사용자가 서비스나 제품을 더 많이 이용하고 더 긴 시간 동안 이용하도록 만들어 줍니다.

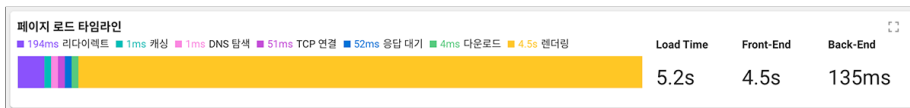
페이지 로드 분석은 모든 웹사이트와 서비스를 운영에 필수적입니다. 사용자 경험을 개선하고 더 나은 비즈니스 성과를 달성할 수 있습니다.

브라우저 애플리케이션 상태 파악하기

Browser 모니터링 서비스의 [브라우저 모니터링 대시보드](#) 메뉴를 이용해 먼저 브라우저 애플리케이션의 페이지 로드 속도를 확인해야 합니다.

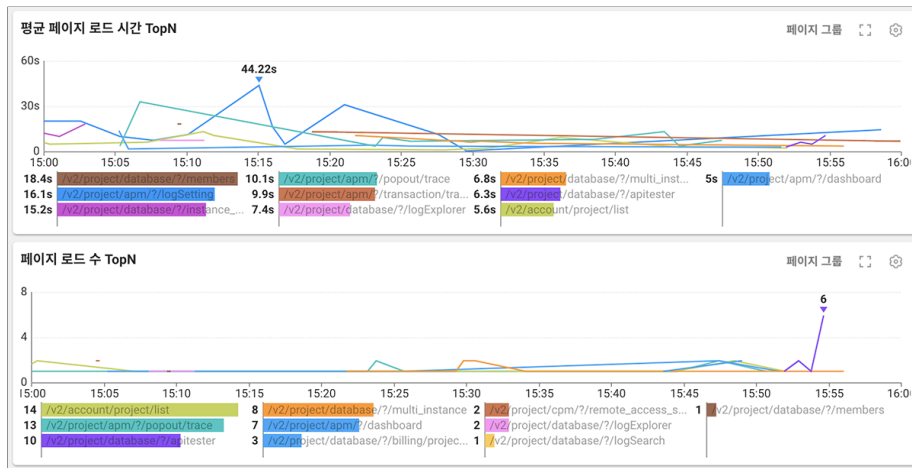
❗ 브라우저 모니터링 대시보드 메뉴에 대한 자세한 내용은 [다음 문서](#)를 참조하세요.

1. 타임 라인 위젯 확인하기



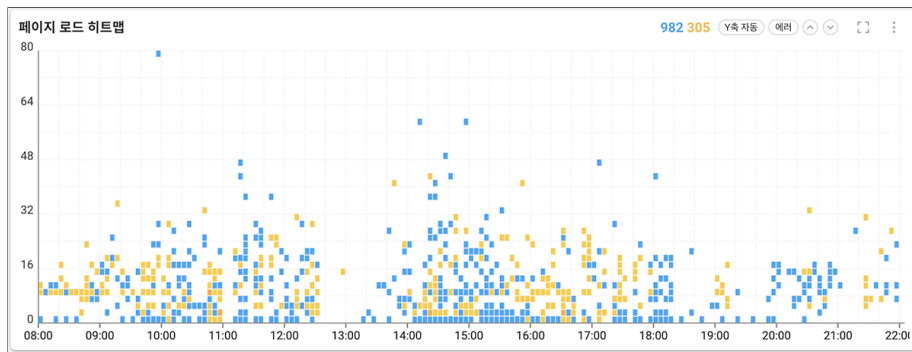
[페이지 로드 타임라인](#) 위젯을 통해서 전체 페이지 로드 성능을 모니터링하세요. 실시간으로 브라우저 애플리케이션의 전반적인 성능을 파악할 수 있습니다. 어느 구간에서 페이지 로드가 느린지 확인할 수 있습니다.

2. 페이지 로드 시간 및 로드 수 확인하기



많은 사용자가 접속하는 페이지의 로딩 시간이 길어진다면 해당 페이지의 성능을 개선하는 데 집중할 수 있습니다. 반대로 로딩 속도가 느린 페이지의 접속량이 크지 않다면, 해당 페이지의 개선보다는 다른 페이지에 더 많은 개발 리소스를 투자하는 것이 효율적일 수 있습니다.

3. 페이지 로드 히트맵 확인하기

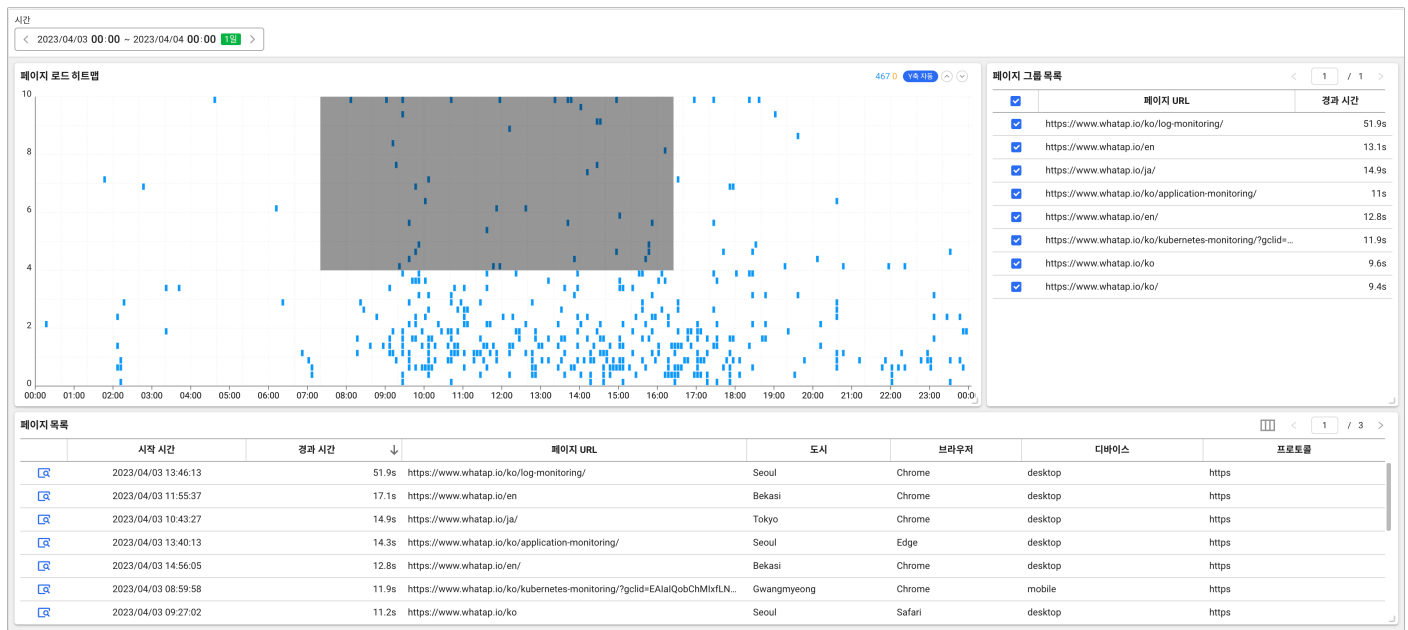


페이지 로드 이벤트를 히트맵 차트 형태로 제공합니다. 실시간으로 발생하는 페이지 로드 이벤트를 표시합니다. 페이지 로드가 느릴수록 차트 상단에 표시됩니다.

성능 저하 원인 파악 및 해결하기

페이지 로딩이 느린 페이지를 확인했다면 원인을 찾아 해결해야 합니다. [분석 > 페이지 로드 히트맵](#) 메뉴로 이동하세요. [페이지 로드 히트맵](#)은 시간에 따른 페이지 로드의 응답 시간을 분포도 차트로 표현합니다. 히트맵을 확인하면 개별 페이지 로드 이벤트에 대한 상세한 정보를 얻을 수 있습니다.

페이지 로드 히트맵 확인하기



1. 페이지 로드 히트맵으로 브라우저 애플리케이션 상태를 파악하세요.

특정 시간에 부하가 걸린 현상을 파악하거나 특정 페이지 로드 이벤트가 느린 현상을 파악할 수 있습니다. 차트 상단에 분포한 이벤트일수록 로딩 시간이 오래 걸린 경우이며 브라우저 애플리케이션의 작동 상태가 느리다는 것을 의미합니다.


2. 페이지 로드가 느린 영역을 드래그하면 페이지 목록에서 각 페이지 로드 이벤트를 확인하세요.

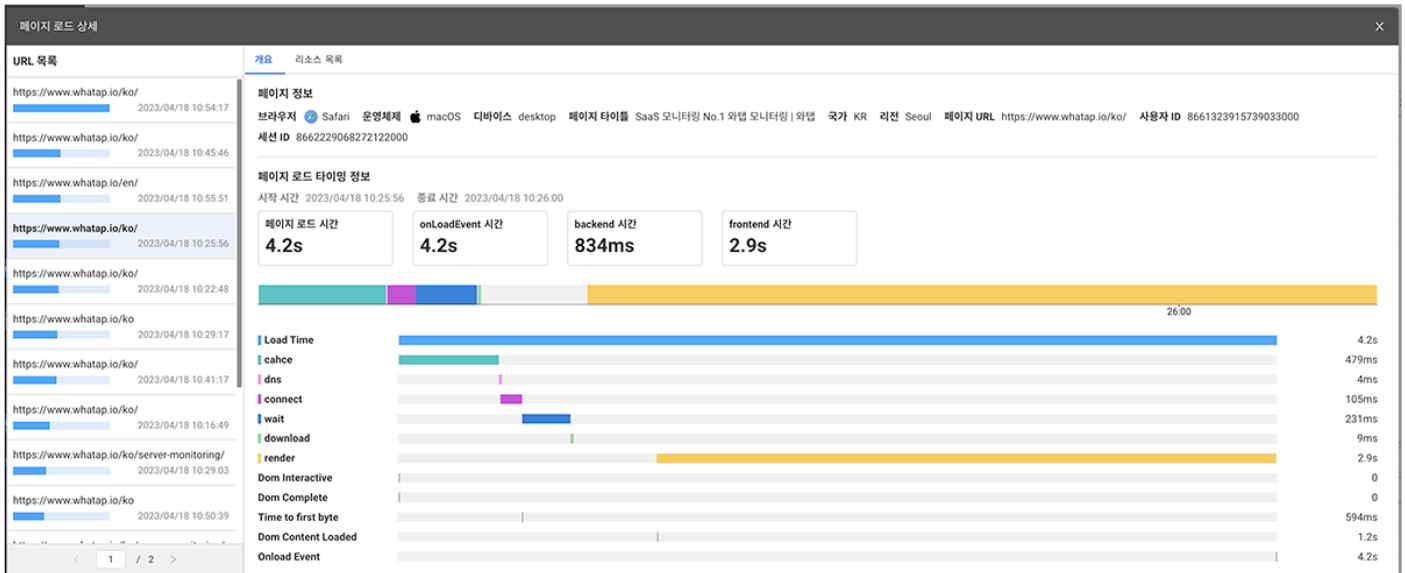
페이지 목록에서는 기본적으로 경과시간, 페이지 URL, 지역 정보, 브라우저, 디바이스 정보 등을 확인합니다. 특정 환경 또는 페이지에서 발생한 문제를 파악하는데 활용할 수 있습니다.

3. 평균 페이지 로드 시간이 길었던 페이지를 필터링하여 볼 수도 있습니다.

특정 페이지에 대한 페이지 로드 이벤트를 확인하려면 페이지 그룹 목록에서 필터링을 적용해 확인할 수 있습니다.

페이지 로드 상세 보기

페이지 목록에서 가장 왼쪽에 를 선택하면 페이지 로드 상세 창이 나타납니다. 페이지 로드 성능을 분석하기 위한 다양한 지표와 성능에 영향을 주는 요소를 파악할 수 있습니다.



페이지 로드 과정 중 구간별 소요된 시간을 그래프 차트로 제공합니다. 차트를 통해 페이지 로드 과정 중 느린 구간을 파악하고 최적화를 진행할 수 있습니다.

- **페이지 정보:** 최종 사용자의 접속 환경, 접속 페이지 정보, 사용자 ID, 세션 ID 정보를 제공합니다. 이 정보들로 특정 환경에서만 발생하는 이상 현상을 판단해 볼 수도 있습니다.
- **페이지 로드 타이밍 정보:** 페이지 로드 시 구간별 소요 시간을 확인할 수 있습니다. 각 구간에 대한 자세한 내용은 다음 문서를 참조하세요.
 - **리다이렉트:** 브라우저가 웹 페이지를 불러올 때 리다이렉트 과정에서 소요한 평균 시간입니다.
 - **캐싱:** 브라우저가 웹 페이지를 불러오면서 캐시된 리소스를 검색하는데 소요한 평균 시간입니다.
 - **DNS 탐색:** 브라우저가 웹 페이지를 불러오면서 웹사이트 도메인을 조회하는데 소요한 평균 시간입니다.
 - **TCP 연결:** 브라우저가 웹 페이지를 불러올 때 TCP 핸드셰이크 과정에서 소요한 평균 시간입니다.
 - **응답 대기:** 브라우저가 웹 페이지를 불러오면서 네트워크 요청을 보낸 후 서버로부터 첫 번째 바이트가 수신될 때까지 소요된 평균 시간입니다.
 - **다운로드:** 브라우저가 웹 페이지를 불러오면서 서버로부터 리소스를 다운로드하는데 소요한 평균 시간입니다.
 - **렌더링:** 서버로부터 다운로드한 리소스를 화면에 렌더링하고 페이지 로드 이벤트를 완료하는데 소요한 평균 시간입니다.
 - **Load Time:** 브라우저가 웹 페이지를 완전히 불러오는데 소요한 평균 시간입니다.
 - **Front-End:** 웹 페이지를 초기 렌더링하는데 소요한 평균 시간입니다.
 - **Back-End:** 페이지 로드 요청부터 리소스를 다운로드하는데 소요한 평균 시간입니다.

백엔드 구간에서 최적화하기

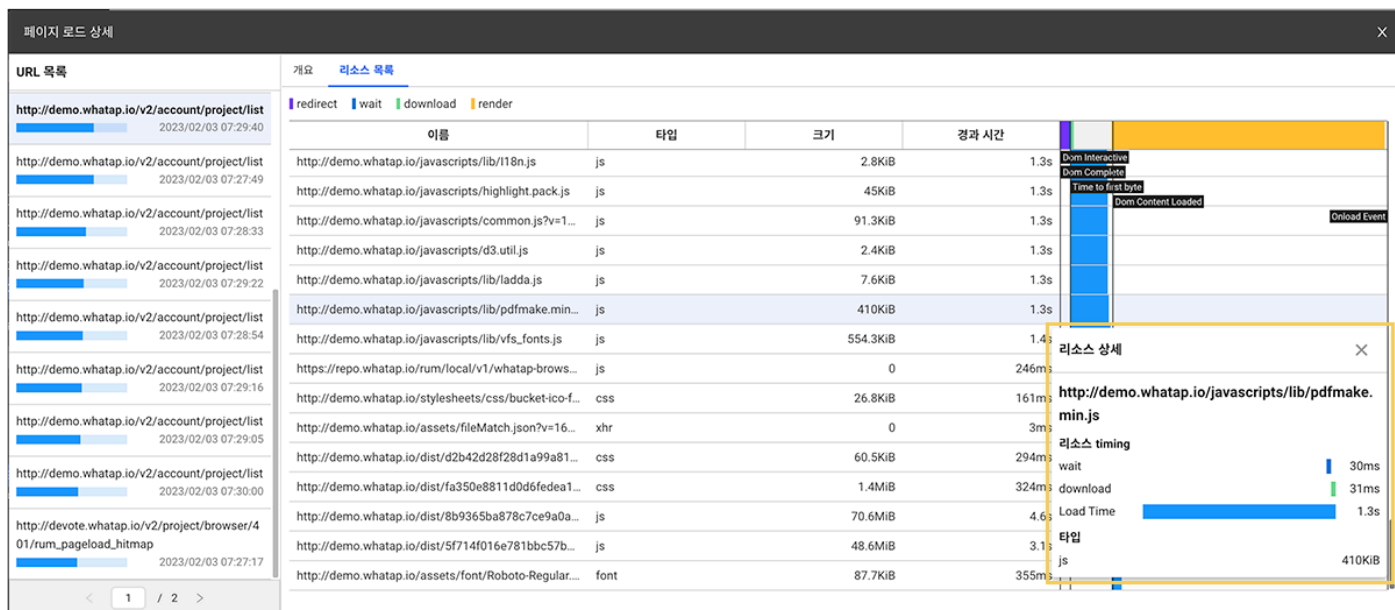
- **캐시 설정 최적화:** 웹 페이지의 캐시 설정이 최적화되지 않거나 캐시 만료 시간이 너무 짧으면 캐싱 속도가 느려질 수 있습니다. 콘텐츠 유형별로 적절한 캐시 만료 시간을 설정하는 것이 중요합니다.
- **캐시 미스:** 캐시에 저장되지 않은 콘텐츠가 많을 경우 캐싱이 제대로 작동하지 않을 수 있습니다.
- **DNS 서버 성능 문제:** 사용 중인 DNS 서버의 성능이 떨어지거나 과부하 상태인 경우 DNS 조회 시간이 길어질 수 있습니다. 이를 해결하기 위해 더 빠르고 안정적인 DNS 서버로 변경하거나 여러 DNS 서버를 사용하여 부하를 분산할 수 있습니다. 또는 가능한 한 적은 수의 도메인을 사용하고 외부 리소스를 최소화하여 DNS 탐색 시간을 줄이는 것이 좋습니다.
- **원격 DNS 서버 위치:** DNS 서버가 사용자와 너무 멀리 떨어져 있으면 조회 시간이 늘어날 수 있습니다. 지리적으로 가까운 DNS 서버를 사용하거나 글로벌 캐시를 제공하는 DNS 서비스를 사용하여 이 문제를 해결할 수 있습니다.
- **서버 응답 시간:** 서버의 처리 속도가 느리거나 서버가 과부하 상태인 경우 응답 시간이 느려질 수 있습니다. 서버 성능을 최적화하거나 서버 용량을 증가시키는 조치 등이 필요할 수 있습니다.
- **서버 위치:** 서버와 사용자 간의 거리가 멀 경우 지연 시간이 길어질 수 있습니다. 이를 해결하기 위해 Content Delivery Network(CDN)과 같은 서비스를 사용하여 사용자와 가까운 위치에서 콘텐츠를 제공할 수 있도록 하세요.
- **네트워크 지연:** 인터넷 연결이 느리거나 불안정한 경우 백엔드 구간의 로딩 속도에 영향을 끼칠 수 있습니다. 이 문제는 사용자의 네트워크 상황에 따라 달라질 수 있습니다.
- **파일 크기:** 큰 파일을 다운로드할 때 속도가 느려질 수 있습니다. 이미지, 스크립트, 스타일 시트 등의 리소스를 최적화하고 압축하여 파일 크기를 줄이세요.
- **동시 다운로드 제한:** 브라우저는 동시에 다운로드할 수 있는 리소스의 수를 제한합니다. 동시에 다운로드되는 리소스 수를 줄이기 위해 여러 리소스를 하나로 병합하거나 비동기 로딩 기법을 사용할 수 있습니다.

프론트 엔드 구간에서 최적화하기

- **복잡한 DOM 구조:** 웹 페이지의 Document Object Model(DOM) 구조가 복잡한 경우 브라우저가 페이지를 렌더링하는 데 시간이 오래 걸릴 수 있습니다. DOM 구조를 간소화하고 불필요한 중첩 요소를 제거하여 렌더링 속도를 개선하세요.
- **무거운 CSS:** 많은 수의 CSS 규칙과 복잡한 선택자를 사용한다면 브라우저가 스타일 계산에 시간을 많이 소모할 수 있습니다. CSS 최적화 및 규칙 간소화, 불필요한 스타일 제거 작업을 통해 렌더링 속도를 개선하세요.
- **JavaScript 실행 지연:** 웹 페이지에 사용된 JavaScript 코드가 많거나 실행 시간이 오래 걸린다면 렌더링 속도에 영향을 줄 수 있습니다. JavaScript 코드를 최적화하고 실행 시간을 줄이며, 필요에 따라 비동기 로딩 방식을 사용하여 렌더링 속도를 개선하세요.

- **이미지 최적화:** 무거운 이미지 파일은 렌더링 속도를 느리게 만들 수 있습니다. 이미지를 압축하고 적절한 크기와 포맷으로 최적화하세요. 필요에 따라 레이지 로딩 기법을 사용하여 렌더링 속도를 개선하세요.
- **웹 폰트 사용:** 웹 폰트는 사용자에게 다양한 폰트 스타일을 제공할 수 있지만 로딩 시간에 영향을 줄 수 있습니다. 웹 폰트를 최적화하고 필요한 것만 사용하세요. 폰트 로딩 전에 텍스트를 표시하는 방식으로 렌더링 속도를 개선할 수도 있습니다.

리소스 목록 확인하기



브라우저가 서버로부터 다운로드하는 리소스는 페이지 로드 성능에 큰 영향을 줄 수 있습니다. **페이지 로드 상세** 창의 **리소스 목록**에서는 로딩 속도가 느리거나 파일 사이즈가 큰 리소스를 빠르게 파악할 수 있습니다.

각 리소스의 시작 시간을 기준으로 타임라인 차트를 제공해 로딩 속도가 느린 리소스를 파악할 수 있습니다. 각 리소스의 파일 크기를 같이 참조해 성능에 영향을 끼치는 부분을 개선하세요. 각 리소스를 선택하면 나타나는 **리소스 상세** 창을 통해 상세 시간 정보를 확인할 수 있습니다.

AJAX 모니터링 및 분석

Asynchronous JavaScript and XML(AJAX)는 비동기 방식으로 데이터를 교환하며 웹 페이지를 업데이트하는 기술입니다. AJAX는 사용자 경험을 개선하고 서버로의 요청을 줄이는 등의 장점을 가지고 있습니다. 그러나 AJAX는 브라우저에서 실행되는 JavaScript 코드로 구현되기 때문에 모니터링이 필요합니다. 다음과 같은 이유에서 AJAX 모니터링은 필요합니다.

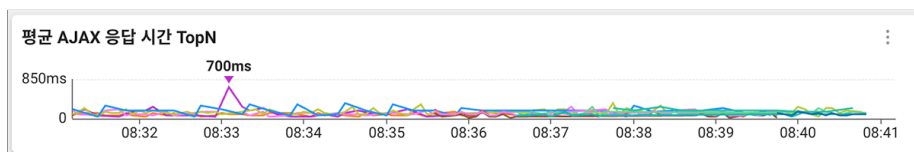
- AJAX 요청이 제대로 처리되는지 확인하기 위해

- AJAX 요청으로 인한 성능 저하를 파악하기 위해
- AJAX 요청이 보안상 문제를 유발할 가능성이 있는지 확인하기 위해
- AJAX 요청이 네트워크 대역폭을 낭비하고 있는지 확인하기 위해

브라우저 애플리케이션의 AJAX 요청 상태 파악하기

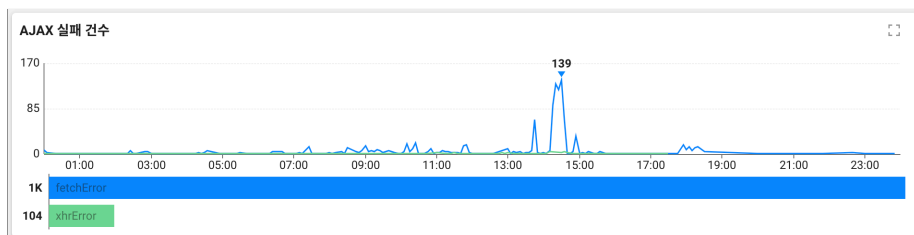
브라우저 모니터링 대시보드 메뉴의 AJAX 관련 위젯으로 브라우저 애플리케이션의 AJAX 요청 상태를 간단히 확인할 수 있습니다.

1. AJAX 응답 시간 확인하기



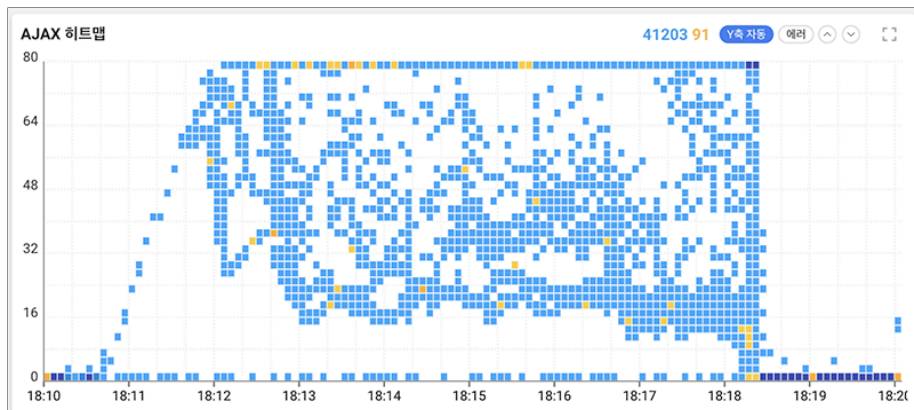
브라우저 애플리케이션에서 발생하는 AJAX의 평균 로드시간을 확인할 수 있습니다. 로드가 오래 걸리는 호스트 또는 path를 파악하는데 용이합니다.

2. AJAX 실패 건수 확인하기



브라우저 애플리케이션에서 AJAX 요청이 정상적으로 이루어지지 않는 개수입니다. 브라우저 애플리케이션 전반에서 발생하고 있는 AJAX 실패 건수에 대해 실시간으로 확인할 수 있습니다.

3. AJAX 히트맵 확인하기



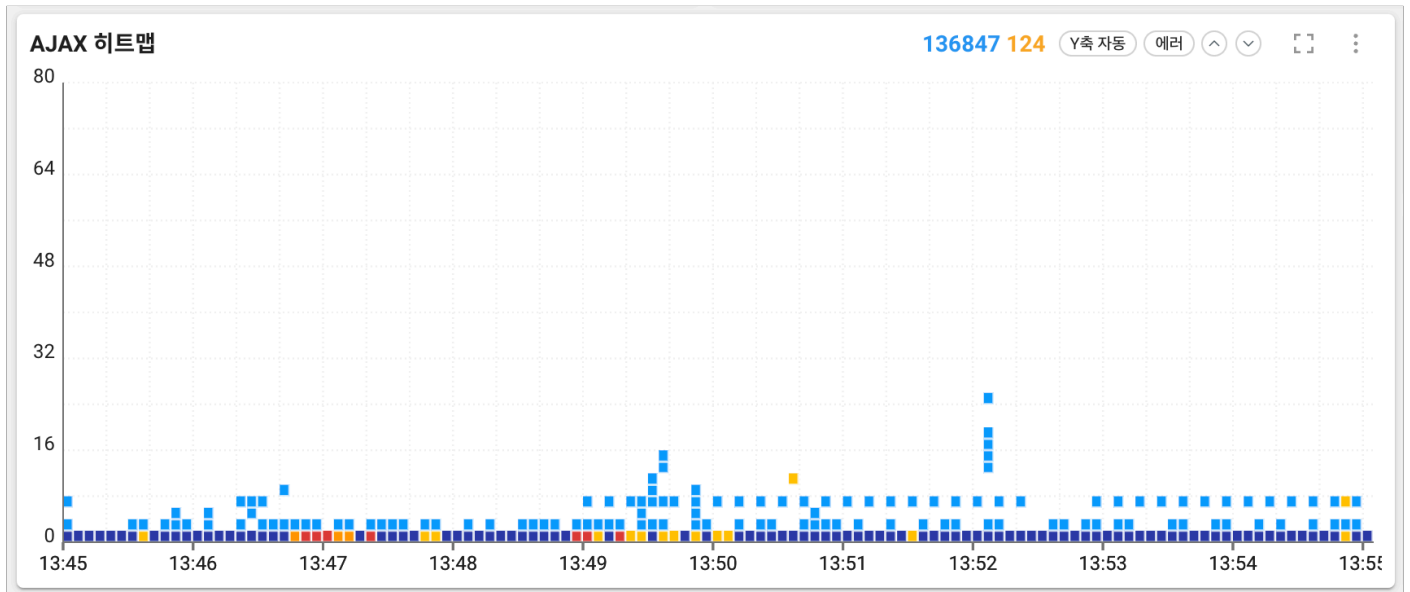
AJAX 요청을 히트맵 차트 형태로 제공합니다. 실시간으로 발생하는 AJAX 요청을 표시합니다.

AJAX가 느릴수록 차트 상단에 위치하며 문제가 있는 AJAX에 대해선 황색 계열로 표시됩니다.

AJAX 히트맵으로 성능 저하 원인 파악 및 해결하기

AJAX 요청이 느리거나 문제가 있다면 각 AJAX가 어떤 상태인지 확인할 필요가 있습니다.

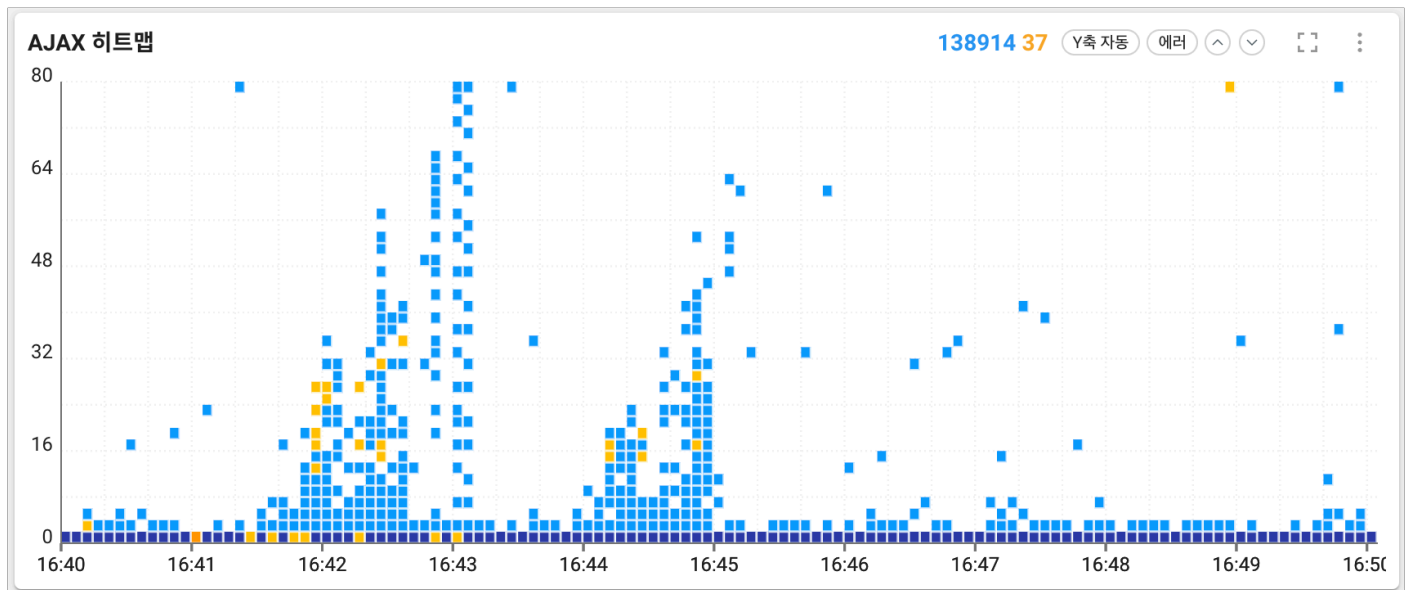
특정 AJAX 요청 에러



AJAX 요청 시 응답값이 400 이상이거나 요청 자체를 하지 못한 경우 황색 계열로 히트맵 차트에 표시됩니다. 이 색상이 나타나면 다음과 같은 상황이 발생한 것으로 추측할 수 있습니다.

- 서버에 문제가 발생하여 요청을 처리하지 못한 경우
- 자바스크립트 코드상 요청이 적절하지 않은 경우
- 인증 문제로 인해 요청이 실패한 경우
- 서버에서 응답을 제공하지 않은 경우

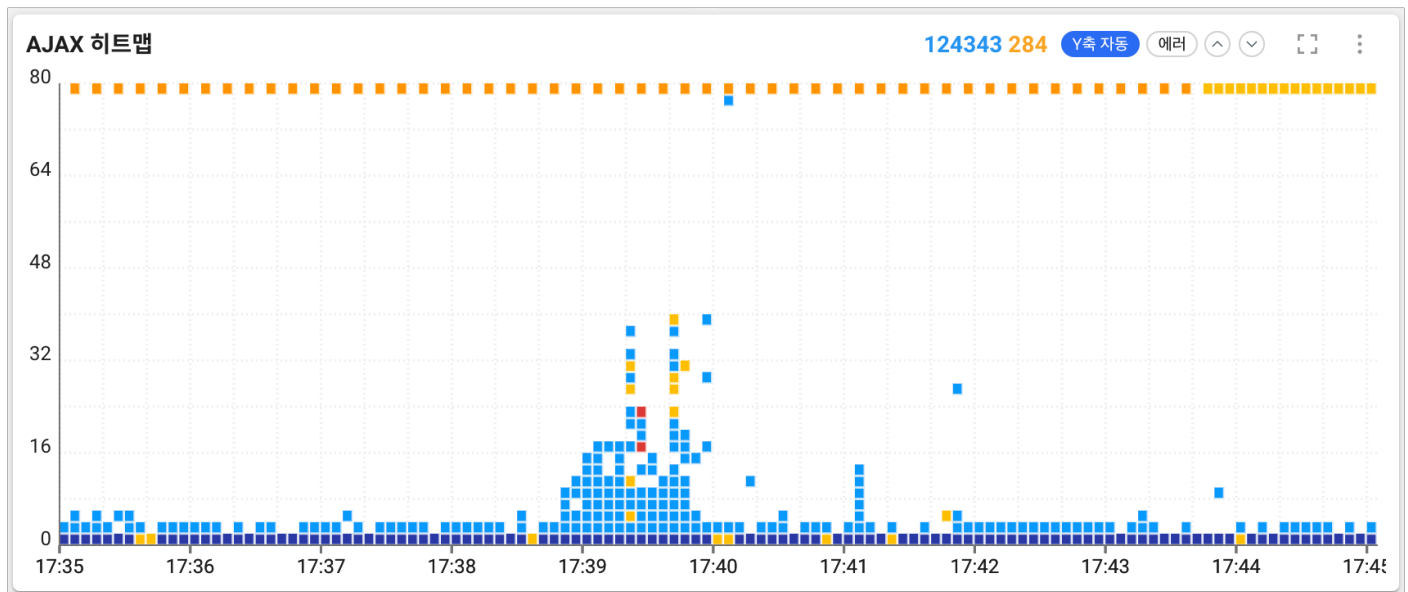
병목 현상



브라우저 애플리케이션 전반에서 AJAX 요청에 병목 현상이 발생한다면 다음과 같은 상황인지 확인하세요.

- **서버 성능 제한:** 서버 측에서 처리할 수 있는 요청의 수가 제한되거나 서버 자원이 부족한 경우 병목 현상이 발생할 수 있습니다.
- **동시 요청 수 제한:** 웹 브라우저는 동시에 처리할 수 있는 요청 수에 제한이 있습니다. 너무 많은 요청이 동시에 발생하면 병목 현상이 발생할 수 있습니다.
- **대량의 데이터 전송:** 전송되거나 받아야 하는 데이터의 크기가 매우 크다면 요청 처리에 시간이 오래 걸려 병목 현상이 발생할 수 있습니다.
- **자바스크립트 실행 성능:** AJAX 요청을 처리하는 자바스크립트 코드의 성능이 좋지 않거나 다른 스크립트와 충돌이 발생하는 경우 병목 현상이 발생할 수 있습니다.
- **응답 처리 지연:** 서버로부터 받은 응답을 처리하는 데 시간이 오래 걸리는 경우 병목 현상이 발생할 수 있습니다.
- **네트워크 대역폭 제한:** 인터넷 연결의 대역폭이 제한되어 있는 경우 동시에 처리할 수 있는 요청 수가 제한되어 병목 현상이 발생할 수 있습니다.

타임아웃 현상



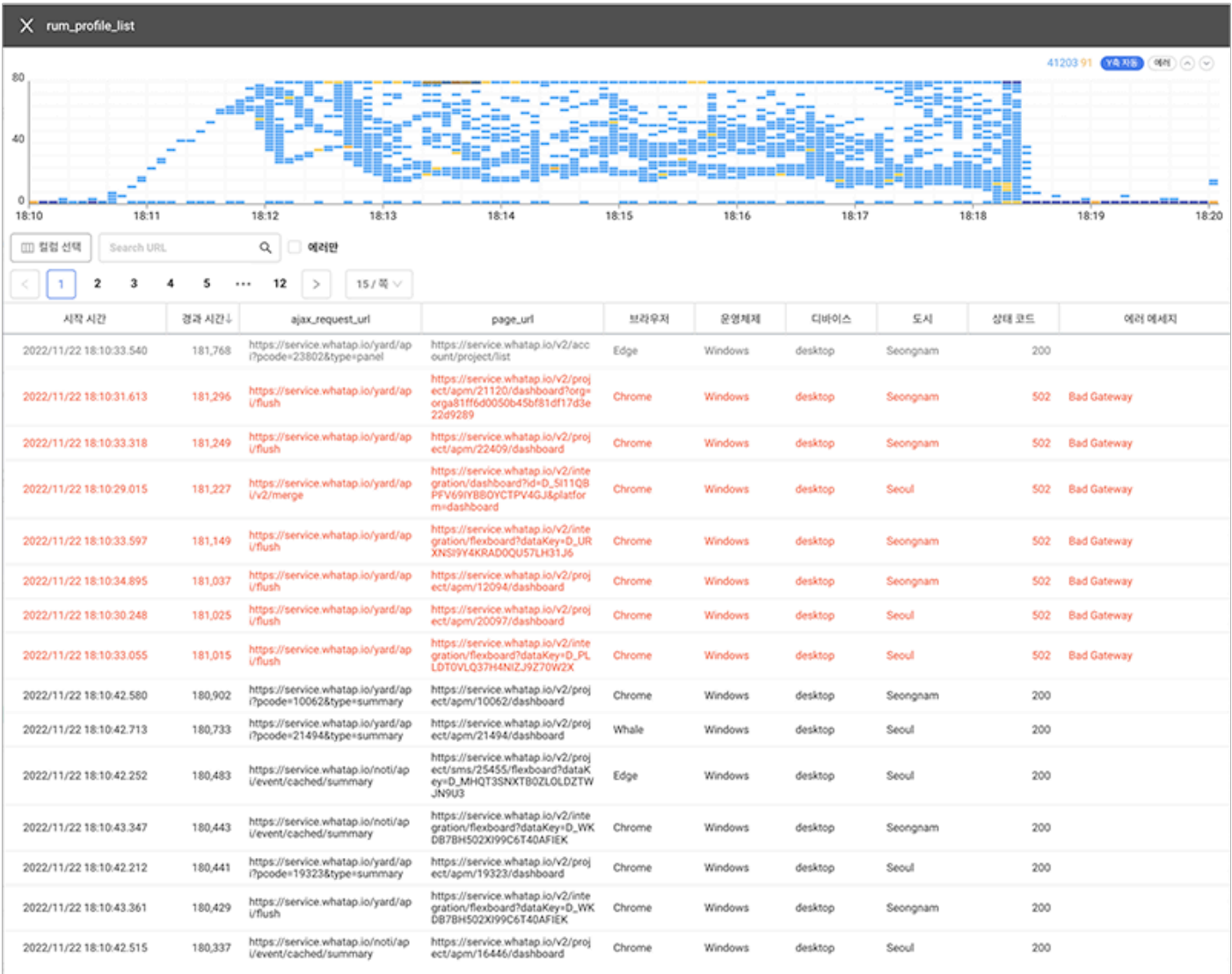
브라우저 애플리케이션 전반에서 AJAX 요청이 타임아웃 오류가 발생한다면 다음과 같은 상황인지 확인하세요.

- 서버 응답 지연: 서버 측에서 처리하는 데 시간이 오래 걸리거나 서버가 다운된 경우 타임아웃이 발생할 수 있습니다.
- 네트워크 지연: 인터넷 연결이 불안정하거나 지연이 발생하면 AJAX 요청이 제때 완료되지 않을 수 있습니다.
- 요청 처리 지연: 브라우저에서 요청을 처리하는 데 시간이 오래 걸리는 경우 타임아웃이 발생할 수 있습니다.
- 대량의 데이터: 전송되거나 받아야 하는 데이터의 크기가 매우 크다면 요청 처리에 시간이 오래 걸려 타임아웃이 발생할 수 있습니다.
- 자바스크립트 실행 오류: AJAX 요청을 처리하는 자바스크립트 코드에 문제가 있거나 다른 스크립트와 충돌이 발생하는 경우 타임아웃이 발생할 수 있습니다.
- 브라우저 호환성 문제: AJAX 요청을 처리하는 데 문제가 있는 특정 브라우저에서만 오류가 발생할 수 있습니다.
- 요청 타임아웃 설정: AJAX 요청의 타임아웃 값을 너무 낮게 설정한 경우 요청이 완료되기 전에 타임아웃이 발생할 수 있습니다.

AJAX 상세 정보 확인하기

AJAX로 인한 성능 저하 현상을 발견했다면 어떤 페이지에서 어떤 AJAX가 문제를 일으키고 있는지 확인해야 합니다. AJAX 히트맵 위젯에서 차트의 특정 영역을 드래그하면 선택한 영역의 AJAX에 대한 상세

정보를 확인할 수 있습니다.



문제가 발생한 AJAX의 경과시간, AJAX URL, 페이지 URL, 브라우저, 상태 코드, 에러 메시지 등을 확인할 수 있습니다. AJAX URL을 통해 어떤 AJAX 호출이 문제를 일으켰는지, 브라우저 정보를 통해 어떤 환경의 사용자가 문제를 겪고 있는지 파악할 수 있습니다. 상태 코드와 에러 메시지는 문제의 원인을 파악할 수 있는 중요한 정보입니다. 빠른 원인 파악은 문제 해결과 서비스 개선을 위한 지름길입니다.

사용자 접속 환경 분석

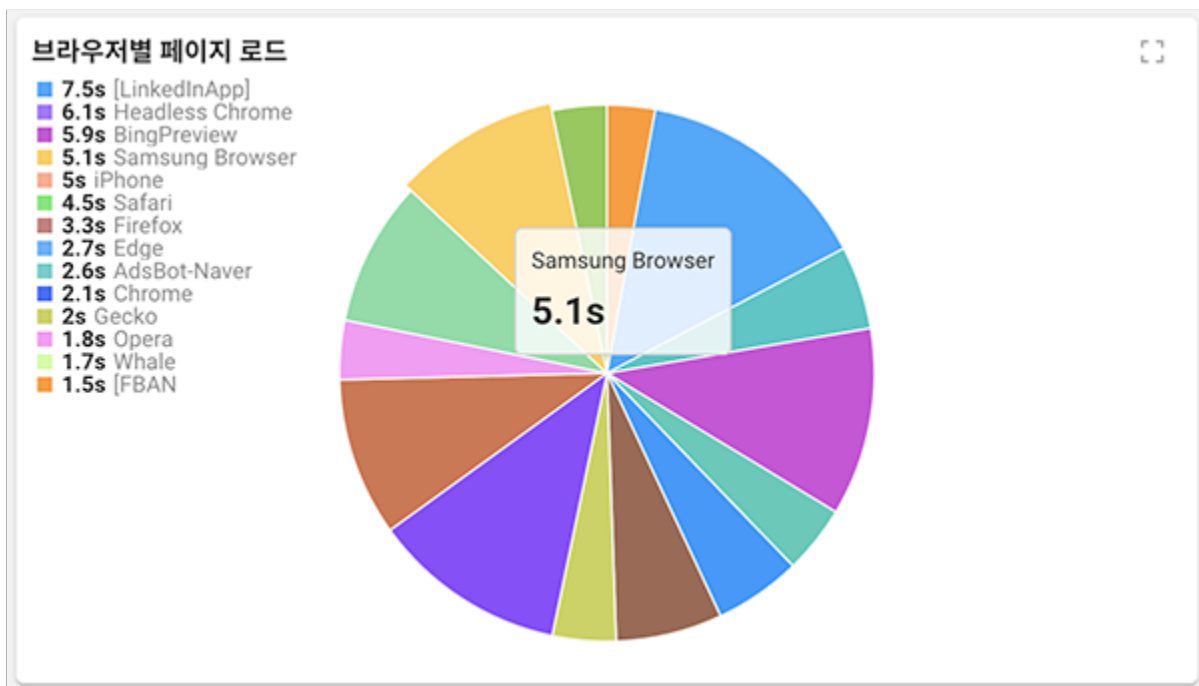
브라우저 애플리케이션을 이용하는 사용자의 환경은 다양합니다. 브라우저 애플리케이션을 모든 브라우저 및 접속 지역에 대응하여 개발하는 것은 불가능에 가깝습니다. 예상할 수 없는 문제를 모니터링하여 예방하고 최소화하는 기술적인 방법을 적용해야 합니다.

브라우저 애플리케이션을 사용자에게 안정적으로 제공할 수 있도록 아래와 같은 문제를 해결해야 합니다.

- 웹 브라우저 및 기기 호환성 문제 해결: 사용자들이 다양한 브라우저 및 기기를 사용하기 때문에 각각의 브라우저나 기기에 따라 호환성 문제가 발생할 수 있습니다. 사용자들이 사용하는 브라우저 및 기기 정보를 파악하면 해당 브라우저나 기기에서 발생하는 문제를 더욱 쉽게 해결할 수 있습니다.
- 지역별 브라우저 애플리케이션 성능 차이 문제 해결: 사용자들이 접속하는 지역에 따라 네트워크 속도가 다르기 때문에 브라우저 애플리케이션의 성능이 지연되는 경우가 발생할 수 있습니다. 이를 파악하면 해당 지역에서 발생하는 문제를 더욱 쉽게 해결할 수 있습니다.

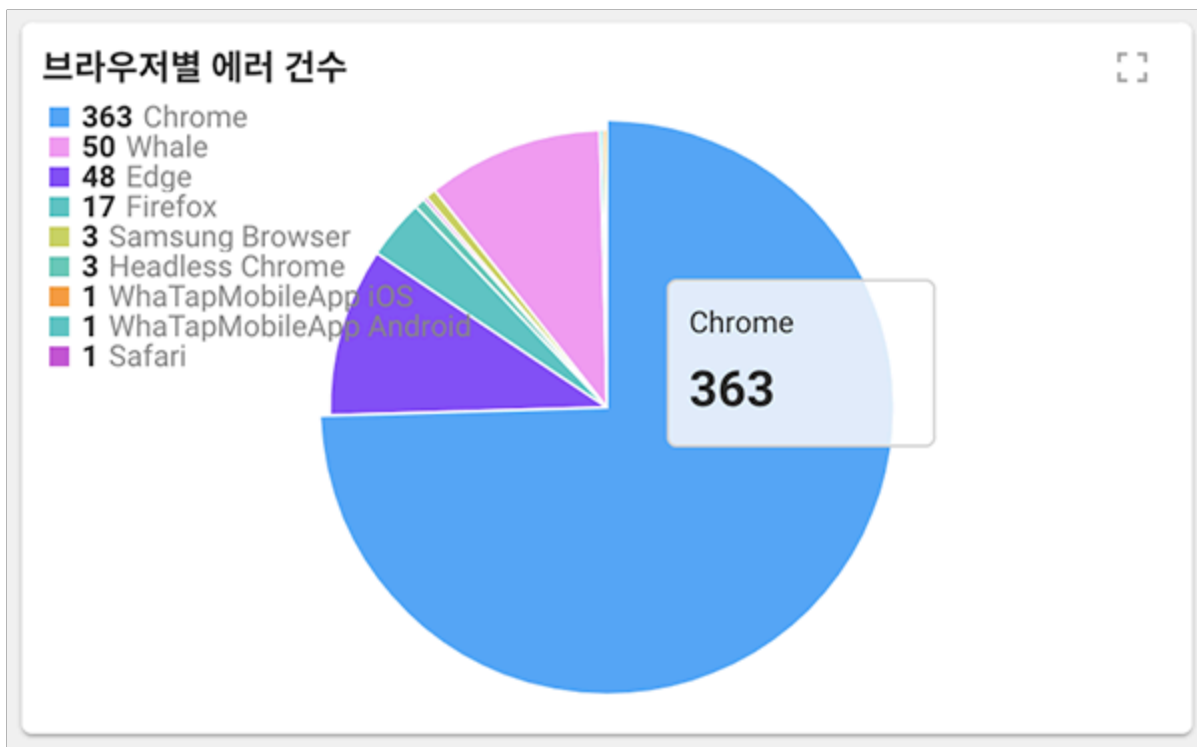
사용자의 접속 환경 파악하기

- 브라우저별 페이지 로드 시간 및 로드 수 확인하기



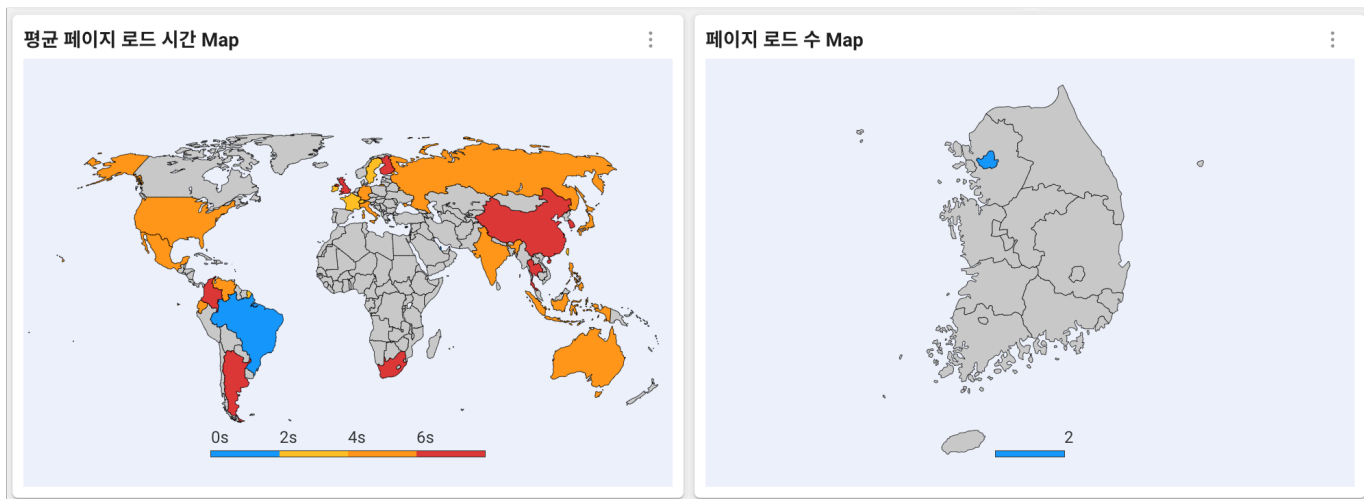
브라우저 애플리케이션의 브라우저별 페이지 로드 시간과 페이지 로드 수를 제공합니다. 이를 통해 브라우저별 페이지 로드 성능을 쉽게 비교할 수 있습니다.

- 브라우저별 에러 수 확인하기



브라우저별로 발생한 에러 건수를 파이 차트로 표시합니다. 브라우저가 어떤 에러를 자주 겪는지 쉽게 파악할 수 있습니다. 또한 특정 브라우저에서 발생하는 에러가 다른 브라우저에 비해 더 많다면 개발자는 해당 브라우저에 대한 최적화 작업을 진행할 수 있습니다.

• 지역 별 페이지 로드 시간 및 로드 수 확인하기



브라우저 애플리케이션의 지역별 페이지 로드 시간과 페이지 로드 수를 지도 차트로 표현합니다. 사용자들은 지역별로 브라우저 애플리케이션의 성능을 쉽게 비교할 수 있습니다. 페이지 로드 시간이 느린 지역에서 웹사이트의 성능을 향상시키기 위한 방법을 찾을 수도 있습니다. 예를 들어 로드 시간이 느린 지역에서 서버를 추가하거나 캐시를 더욱 효율적으로 이용할 수 있는 방법을 찾을 수 있습니다. 이렇게 함으로써 사용자들은 더욱 빠르고 안정적인 브라우저 애플리케이션을 이용할 수 있습니다.

성능 저하 원인 파악하기

- 브라우저마다 같은 리소스 파일을 불러오는데 로딩 시간이 상이합니다. [페이지 로드 상세](#)에서 특정 브라우저에서 느려지는 원인을 파악하고 다음의 상황을 확인해 보세요.
 - **CSS 처리:** 브라우저마다 다른 CSS 처리 방식으로 렌더링 속도에 차이가 발생할 수 있습니다. 브라우저별로 지원하는 CSS 속성이나 기능에도 차이가 있으므로 이를 고려하여 웹 페이지를 구현해야 합니다.
 - **최적화 수준:** 브라우저별로 페이지 로딩을 최적화하는 기술과 정도에 차이가 있을 수 있습니다. 이미지 로딩, 코드 실행, 리소스 요청 순서 등 최적화 작업은 브라우저마다 다르게 작동할 수 있습니다.
 - **웹 페이지의 복잡성:** 웹 페이지의 복잡성과 구성 요소가 브라우저별로 다른 성능을 나타낼 수 있습니다. 복잡한 자바스크립트 코드나 CSS 애니메이션은 일부 브라우저에서 성능 저하를 일으킬 수 있습니다.
- 특정 브라우저에서만 에러가 발생한다면 **웹 API 및 기능 지원**을 확인해 보세요.

브라우저마다 웹 API 및 기능 지원의 차이로 인해 일부 브라우저에서는 특정 기능이 느리게 동작하거나 전혀 작동하지 않을 수 있습니다. 이런 경우 폴리필(polyfill)을 사용해 해당 기능을 대체하거나 특정 브라우저에 대한 최적화를 고려해야 합니다.

에러 분석하기

다음과 같은 이유에서 브라우저 에러를 모니터링해야 합니다.

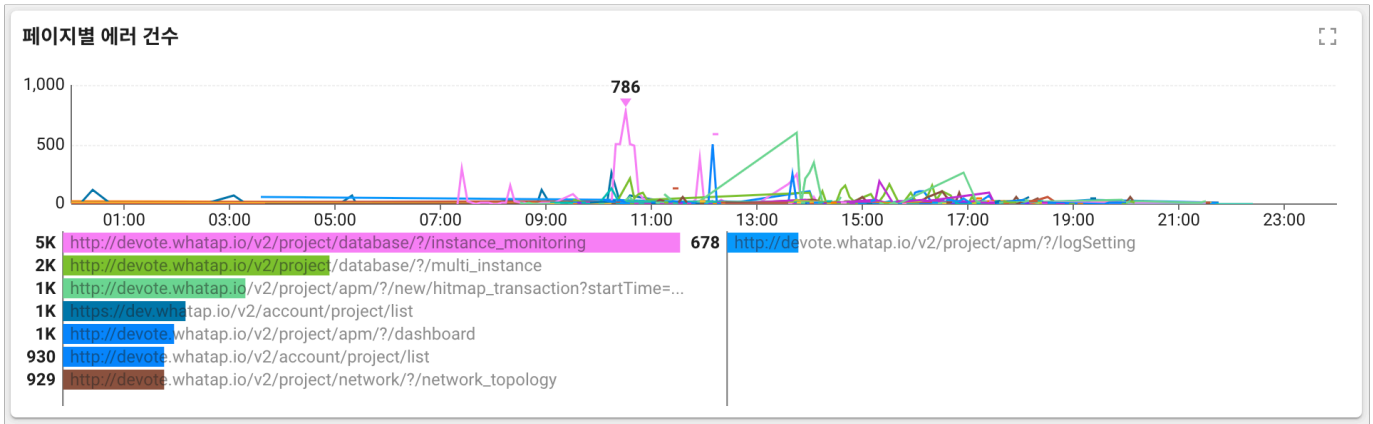
- **사용자 경험 개선:** 브라우저 에러가 발생하면 사용자가 애플리케이션을 제대로 사용하지 못할 수 있습니다. 따라서 브라우저 에러를 모니터링하고 가능한 빠르게 수정해야 합니다.
- **보안 취약점 방지:** 브라우저 에러는 웹사이트 보안에 큰 위협을 가할 수 있습니다. 에러를 모니터링하고 보안 취약점을 신속하게 수정해야 합니다.
- **성능 개선:** 브라우저 에러는 애플리케이션의 성능을 저하시킬 수 있습니다. 성능 저하는 서비스 품질과 사용자 만족도와 연관되어 있습니다. 성능 저하가 지속된다면 애플리케이션 사용자는 계속 줄어들 것입니다.

브라우저 에러 확인하기

[대시보드](#) > [브라우저 에러](#) 메뉴에서는 브라우저 애플리케이션에서 발생하는 에러를 모니터링합니다. 다음과 같은 에러 유형을 파악할 수 있습니다.

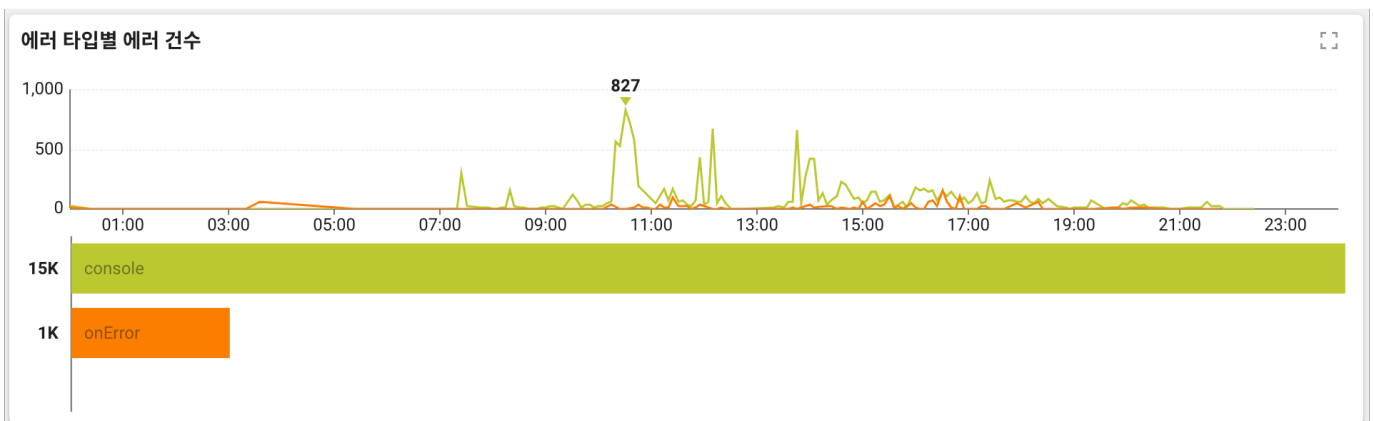
❗ 브라우저 에러 메뉴에 대한 자세한 내용은 [다음 문서](#)를 참조하세요.

- 페이지별 에러 건수 확인하기



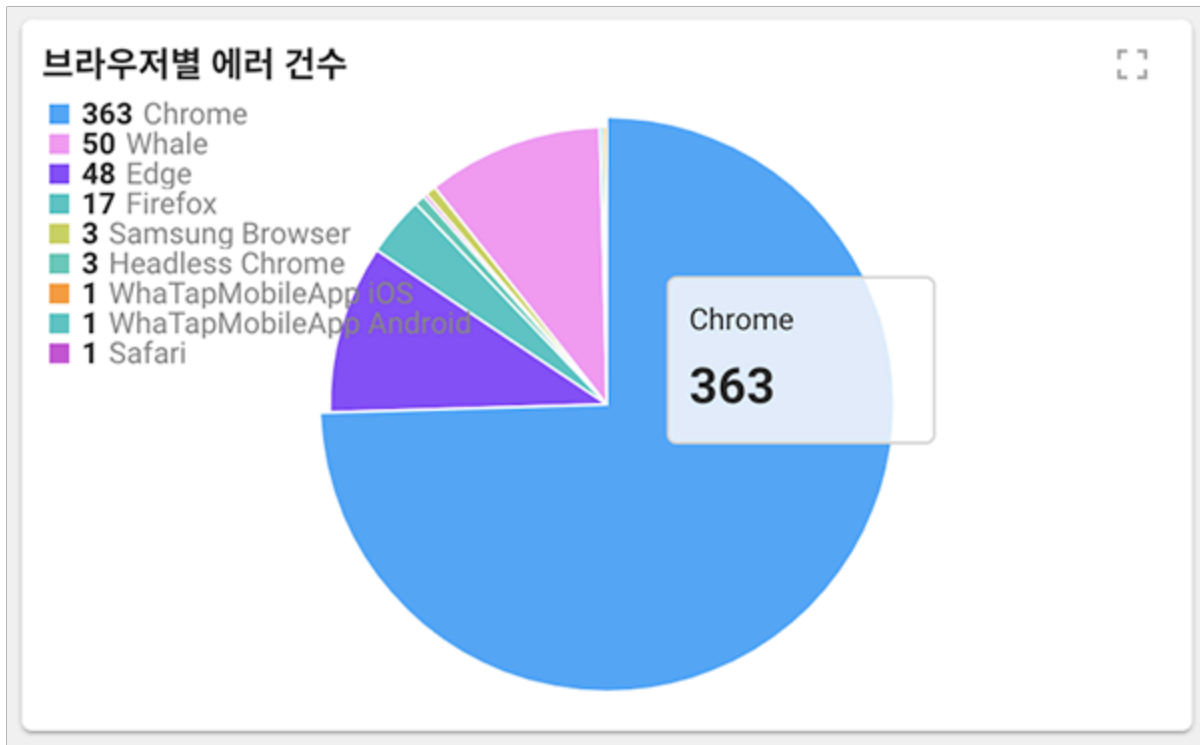
페이지별로 발생하는 에러 건수를 실시간으로 확인할 수 있습니다. 문제가 발생하는 페이지를 빠르게 파악하고 대처할 수 있습니다.

- 에러 타입별 에러 건수 및 AJAX 에러 건수 확인하기



브라우저 애플리케이션 전반에서 발생하는 에러 건수에 대해서 에러 타입별로 확인할 수 있습니다.

- 브라우저별 에러 건수 확인하기



브라우저별로 발생한 에러 건수를 파이 차트로 표시합니다. 브라우저가 어떤 에러를 자주 겪는지 쉽게 파악할 수 있습니다. 또한 특정 브라우저에서 발생하는 에러가 다른 브라우저에 비해 더 많다면 개발자는 해당 브라우저에 대한 최적화 작업을 진행할 수 있습니다.

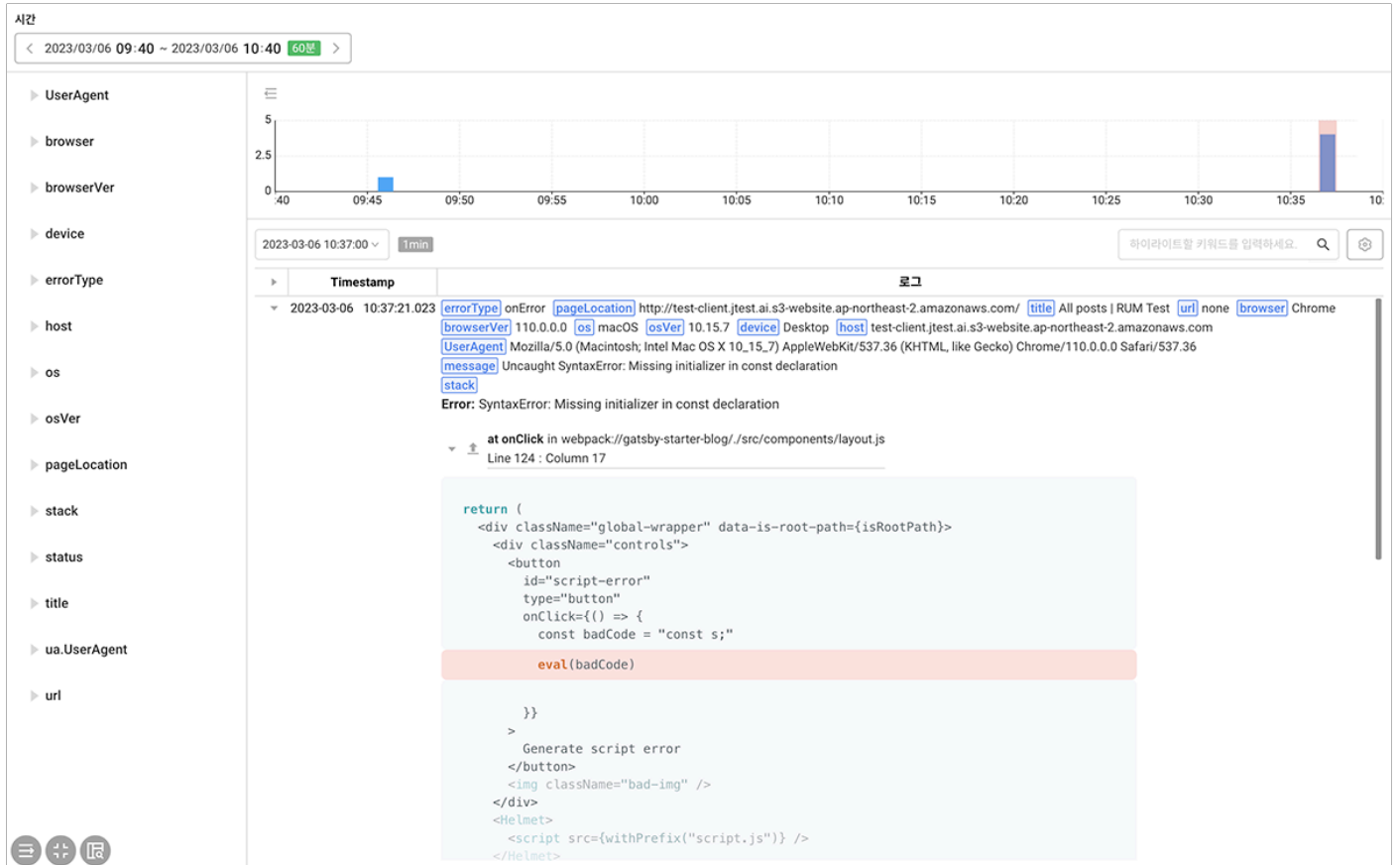
• 에러 메시지 테이블 확인하기

에러 메시지	에러 건수 ↓	브라우저 별 에러 수
basic	2K	1 7 16 96 2K
e.select is not a function	1K	46 126 127 1.5K
Lapply is not a function	1K	1 27 83 93 1.4K
Failed to fetch	1K	1 6 17 1.3K
The user aborted a request.	440	32 42 366
Unauthorized	121	121
Request Failed	117	1 26 90
Cannot read properties of undefined (reading 'textC...	114	14 100
Failed to execute 'fetch' on 'Window': The user abort...	73	73
Cannot read properties of null (reading 'getBoundin...	51	4 9 38
Forbidden	43	43
Script failed because the user didn't interact with th...	41	1 1 1 1 1

에러 메시지를 기준으로 한 에러 수와 브라우저 별로 발생한 에러 수를 테이블 목록으로 표시합니다. 어떤 브라우저에서 어떤 에러가 가장 많이 발생하는 지를 파악하여 문제점을 해결하는 데 이용할 수 있습니다. 에러 발생의 주요 원인이 무엇인 지를 파악하는 데에도 유용합니다. 새로운 에러가 발생하면 이 테이블을 참고하여 빠르게 대처할 수 있습니다.

에러 원인 파악하기

에러 메시지, 에러 발생 페이지를 찾았다면 [분석 > 에러 추적](#) 메뉴에서 에러에 대한 세부 정보를 확인하세요.



에러 추적 메뉴에서는 브라우저에서 발생한 에러 정보를 상세하게 확인할 수 있습니다. 에러 발생 페이지나 에러 메시지에 대해 필터를 적용해 검색할 수도 있습니다. 사용자가 발생시킨 에러는 사용자 환경 정보(브라우저, OS, 디바이스), 에러 스택 정보, 페이지 정보를 포함하고 있습니다.

에러 원인을 파악하려면 다음과 같은 단계로 진행할 것을 권장합니다.

1. 에러 메시지를 읽어보고 해당 페이지에서 어떤 작업을 수행했는지 확인하세요.
2. 해당 페이지에서 사용된 코드 및 라이브러리를 살펴봅니다.
3. 에러 스택 정보를 사용하여 어떤 함수나 라이브러리에서 에러가 발생했는지 확인하세요.
4. 브라우저, OS, 디바이스 정보를 사용하여 특정 환경에서만 발생하는 에러인지 확인하세요.
5. 페이지 정보를 사용하여 웹사이트의 특정 부분에서만 발생하는 에러인지 확인하세요.

에러 원인을 파악한 후에는 가능한 한 빠르게 수정하세요. 사용자 경험을 개선하고 브라우저 애플리케이션의 보안 취약점을 방지할 수 있습니다.