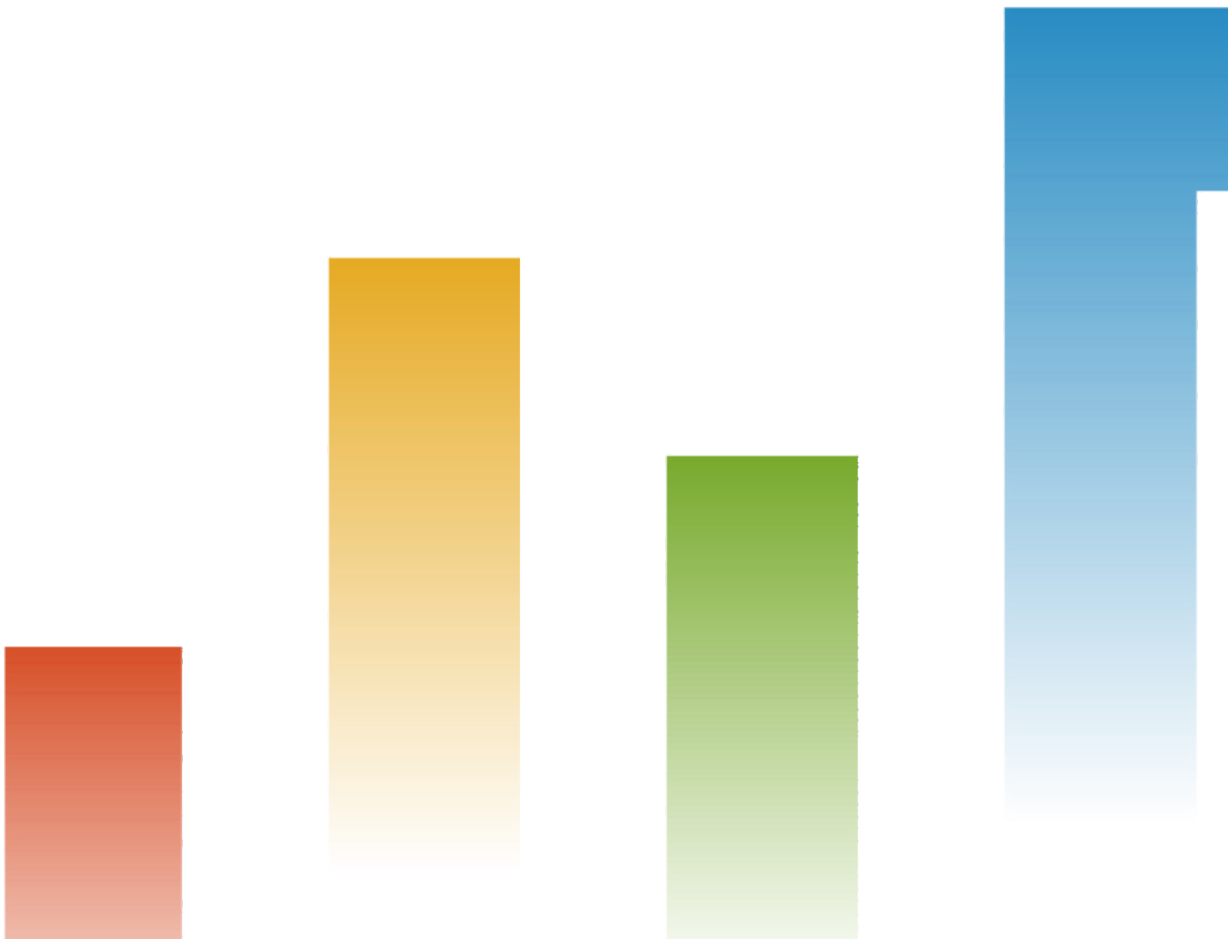
WhaTap

Java Monitoring

release date. 2026. 09. 10.



Java 문서 목차

- JAVA 모니터링 7
  - 시작하기 9
  - 지원 환경 11
  - 에이전트 설치 25
    - Tomcat 31
    - JBoss 35
    - JEUS 38
    - WebLogic 42
    - WebSphere 45
    - Resin 49
    - Liberty 53
    - Jetty 56
    - Spring Boot 59
    - Play2 64
    - IBM Bluemix 67
    - Elastic Beanstalk 71
    - Docker 75
    - 배치 프로그램 79
    - 이전 Java 버전 97
  - 오픈소스 추적
    - Java Helper 100
    - Weaving 104
  - 설치 점검 112
  - 문제 해결 114
  - 설정하기 120

- 에이전트 이름 132
- 기능 제어 136
- 에이전트 사용량 140
- 통신 설정 141
- 성능 설정 143
- 로그 설정 144
- 트랜잭션 설정 149
- DB/SQL 166
- HTTPC/API Call 171
- 스크립트 플러그인 173
- 사용자 수 186
- 부하량 제어 188
- 알림 설정 191
- Apdex 194
- 통계 195
- 에러 스택 198
- 추가 옵션 201
- 비동기 추적 203

◦ 관리하기

- 에이전트 업데이트 206
- 에이전트 삭제 207
- MSA 그룹 208
- AES 256 암호화 209
- 스케일 인 자동 삭제 210

◦ 대시보드 211

- 애플리케이션 대시보드 212
- 트랜잭션 맵 239

- 액티브 트랜잭션 249
- 히트맵/트레이스 254
- 자원 대시보드 282
- 비즈니스 대시보드 288
- Flex 보드 294
  - Flex 보드 가이드 297
  - 위젯 관리 307

◦ 분석 312

- 일일 현황 314
- 성능 트렌드 316
- 큐브 분석 319
- 큐브 330
- 히트맵 분석 334
- 멀티 트랜잭션 분석 360
- TX 프로파일 365
- MSA 분석 375

◦ 메트릭스 383

- 애플리케이션 메트릭 386
- 성능 카운터 406
- 메트릭 차트 v2 414
- 메트릭 차트 435
- 메트릭 검색 438
- 이상 탐지 440

◦ 트랜잭션 443

- 트랜잭션 조회 445
- 트랜잭션 프로파일 455

- 액티브 트랜잭션 481
- 멀티 트레이스 484
- 엔드포인트 설정 488
- 스택 수집 492

◦ 애플리케이션 토폴로지 494

◦ 트랜잭션 분석 502

◦ 보고서 515

- APM 보고서 517
- 통합 보고서 525

◦ 경고 알림 533

- 이벤트 설정 0
  - 메트릭스 539
  - 복합 메트릭스 560
  - 히트맵 패턴 567
  - 이상 탐지 572
  - 실시간 로그 576
  - 복합 로그 589
  - 트랜잭션 603
- 애플리케이션 (클래식) 614
- 이벤트 수신 설정 618
  - 알림 포맷 641
- 이벤트 기록 및 모니터링 650
  - 실시간 알림 658
  - 시스템 이벤트 661

◦ 인스턴스 성능 분석 663

◦ 연결된 프로젝트 681

◦ Log 모니터링 696
◦ 실험실 922

Java 애플리케이션 모니터링

Web Application Server(WAS), 배치 애플리케이션 등 JVM 환경에서 동작하는 모든 애플리케이션에 적용할 수 있습니다.
애플리케이션 장애 분석을 위해 Thread Dump(스레드 덤프)를 확보하던 번거로움에서 벗어나, 이미 확보된 스택을 통해 장애 상황의 재현 없이 정확한 원인을 분석할 수 있습니다.

와탭의 애플리케이션 모니터링은 장애 재현 없이 실시간으로 모니터링이 가능합니다.

주요 특장점

- **실시간 트랜잭션 모니터링**

  와탭은 현재 진행 중인 트랜잭션의 수행 현황을 실시간으로 보여줍니다. 모든 애플리케이션 문제는 진행 중인 트랜잭션 증가로 나타납니다. 증가한 순간에 수행 중인 SQL, 외부 호출, Method를 즉시 확인할 수 있다면 문제를 가장 빠르게 인지하고 분석할 수 있습니다.

- **트랜잭션 성능 Deep Dive**

  와탭은 모든 트랜잭션의 수행이력을 수집해 분석합니다. 트랜잭션 성능은 SQL, Http Call뿐 아니라 Method 레벨까지 추적할 수 있습니다.

  특히 **액티브 스택** 기능으로 번거로운 Method 트레이스 설정 없이 Method 레벨 분석이 가능합니다. **액티브 스택**은 와탭의 특허 기술로 기존 제품에서 발견하지 못했던 숨겨진 문제 원인을 찾아 개선 포인트를 제공합니다.

  > ⓘ 액티브 스택 기능은 Java, Python, .NET에서만 지원합니다.

- **MSA 호출 패턴 분석**

  MSA 구조의 복잡한 애플리케이션간의 호출 관계를 트랜잭션 URL 기준으로 통계적으로 시각화하여 제공합니다. 시점별로 변화하는 호출 관계와 호출 비중의 변화를 통계적으로 파악할 수 있습니다.

- **AI 기반 응답 분포 패턴 분석**

  와탭은 머신러닝으로 특정 패턴을 자동 인식하여 알림 경고를 발생, 운영자가 인지하지 못한 이슈까지 알려드립니다.
  온프레미스 환경에서도 제공하며, 적용 범위는 별도 계약으로 정합니다.
  출원번호: 10-2020-0037381

- **멀티 프로젝트 트랜잭션 연계 분석**

MSA 환경에서 트랜잭션 연계 추적 기능을 제공합니다. 전체 트랜잭션을 추적하여 지연이 발생하는 애플리케이션을 식별할 수 있습니다.

- **스택 통계 분석**

  통계 분석으로 실시간 모니터링이나 트레이스 정보에서 발견하기 힘든 튜닝 포인트를 찾아낼 수 있습니다. 거시적인 문제의 추이 또한 그래프와 표를 통하여 개선 대상을 선별하거나 특정 시간대의 응답 지연 현상을 확인할 수 있습니다.

- **사후 분석(Cube)**

  시간을 축으로 다양한 지표를 한번에 확인할 수 있습니다. 응답시간, 에러 건수 및 처리량 등이 높은 시간대만을 특정할 수 있어 동 시간대 문제 요소를 식별하거나 서비스 현황을 정확하게 파악할 수 있습니다.

- **서비스 폭주 시 부하량 제어**

  시스템이 감당할 수 없는 과도한 요청이 들어오는 경우, 일부 요청을 거절하거나 잘못된 서비스 요청은 차단하여 안정적인 서비스 운영을 지원합니다.

- **비동기 트랜잭션 트레이스**

  와탭만의 고유한 BCI(Byte Code Instrumentation) 기술로 비동기 트랜잭션의 성능을 효과적으로 추적합니다. WebFlux, Reactor, Hystrix 등을 사용하는 애플리케이션의 트랜잭션 트레이스 정보를 수집합니다.

Java 모니터링 시작하기

Java 애플리케이션 모니터링을 시작하려면 와탭 프로젝트를 생성하고, 애플리케이션 서버에 Java APM 에이전트를 설치합니다. JVM 옵션으로 에이전트를 적용하면 트랜잭션, SQL, 외부 호출까지 자동으로 추적합니다.

와탭 모니터링은 다음 절차로 시작합니다.

1단계. 회원 가입하기
2단계. 제품 선택하기
3단계. 프로젝트 생성하기
4단계. 액세스 키 확인하기

이후 지원 환경을 확인한 후 에이전트 설치 단계로 이동합니다.

회원 가입하기

WhaTap 모니터링 서비스를 사용하려면 먼저 회원 가입이 필요합니다. 회원 가입에 관한 자세한 내용은 계정 관리 > 회원 가입 문서를 참고하세요.

제품 선택하기

1. 와탭 모니터링 서비스에 로그인합니다.
2. **전체 프로젝트** > **+ 프로젝트** 버튼을 클릭합니다.
3. ❶ **상품 선택** 화면에서 모니터링할 제품의 **생성하기** 버튼을 클릭하세요.

프로젝트 생성하기

모니터링할 제품을 선택했다면, 에이전트를 설치하기 전에 먼저 프로젝트를 생성합니다.

1. ❷ **프로젝트 생성** 화면에서 프로젝트를 생성하려면 다음 항목을 입력하거나 선택하세요.
    - **프로젝트 이름**(필수): 프로젝트의 이름을 입력합니다.

- **데이터 서버 지역**(필수): 데이터 서버가 위치한 리전(지역)을 선택합니다. 리전은 클라우드 서비스를 제공하는 데이터 센터의 묶음입니다. 선택한 리전에 사용자의 데이터가 저장됩니다.
- **타임 존**(필수): 알림 및 보고서 생성 시 기준이 되는 시간을 설정합니다.
- **알림 언어 설정**(필수): 프로젝트에서 발생하는 경고 알림 메시지의 언어를 설정합니다. (한글, 영어 지원)
- **프로젝트 그룹**: 여러 프로젝트를 그룹으로 묶어 관리할 수 있습니다. 소속될 그룹이 있으면 선택하세요. 없다면 그룹 추가 버튼을 클릭해 그룹을 추가할 수 있습니다.
- **프로젝트 설명**: 프로젝트에 대한 추가 설명을 입력합니다.

2. 모든 설정을 완료하면 **프로젝트 생성하기** 버튼을 클릭하세요.

> ⓘ 조직을 선택한 상태에서 프로젝트를 추가할 경우 **조직 하위 그룹**을 필수로 설정해야 합니다. 그룹에 대한 자세한 설명은 다음 문서를 참고하세요.

액세스 키 확인

액세스 키는 와탭 서비스 활성화를 위한 고유 ID입니다.

설치 안내 섹션에서 **프로젝트 액세스 키 발급받기** 버튼을 선택하세요. **액세스 키**를 자동으로 발급받은 후 다음 단계를 진행합니다.

> ⓘ 프로젝트를 생성한 후, 자동으로 **에이전트 설치** 페이지로 이동합니다. **에이전트 설치** 페이지로 이동하지 않는다면 화면 왼쪽 메뉴에서 **관리** > **에이전트 설치**를 선택하세요.

Java 지원 환경

Java 에이전트 지원 환경

에이전트를 설치하기 전에 JVM 애플리케이션 서버가 다음의 사양을 충족하는지 확인하세요.

Web Application Server (WAS), 데몬, 배치 애플리케이션 등 JVM 기반 모든 애플리케이션에 적용할 수 있습니다.

> ⓘ 최종 업데이트: 2025-11-18

> ⓘ Java 1.6 미만은 support@whatap.io로 문의하거나 다음 문서를 참고하세요.

지원 버전

구분 / JDK 버전	Java 6 ~ 21	Java 22	Java 23
지원 여부	✅	✅	✅
지원 범위	Limited Support	Fully support	

운영체제

OS 계열	지원 버전	아키텍처
Unix	HP-UX 11.x	32bit, 64bit
	Oracle Solaris 2.8, 2.9, 10, 11	32bit, 64bit
Linux	Red Hat	Itanium 64bit

OS 계열	지원 버전	아키텍처
	x86 Intel Linux	32bit
Windows	Windows Server	2000 이상
	Windows	XP 이상
기타	Itanium	64bit

ⓘ JDK 6를 설치할 수 있는 모든 운영체제를 지원합니다.

애플리케이션 서버

벤더	제품	지원 버전
Apache	Jakarta Tomcat	6.x ~ 10.x
Red Hat	JBoss Application Server	6.x, 7.x
TmaxSoft	JEUS	6.x, 7.x
Oracle	WebLogic Server	9.x ~ 12.x
	GlassFish	3.x ~ 5.x
	SUN Application Server	8.x, 9.x
IBM	WebSphere, WebSphere Liberty	-
	BlueMix	-
Caucho	Resin	3.x, 4.x
Eclipse	Jetty	-

벤더	제품	지원 버전
Fujitsu	Interstage	5.x ~ 7.x
Hitachi	Cosminexus	7.x ~ 9.x
AWS	Elastic Beanstalk	-
Docker	Container Environment	-

ⓘ Java EE Spec을 준수하는 모든 애플리케이션 서버를 지원합니다. 다음 링크를 참고하세요.

프레임워크 / 라이브러리

카테고리	프레임워크	지원 버전
웹 프레임워크	Spring Boot	1.5.x ~ 3.x
비동기 프레임워크	Netty	-
	Akka HTTP	-
	Play Framework	-
마이크로서비스	Quarkus	-
	Quarkus Reactive	1.x ~ 2.x
통합 프레임워크	Mule Framework	3.9.5, 4.5

ⓘ Java EE Spec을 준수하는 Application Server에서 동작하는 모든 라이브러리를 지원합니다.

데이터베이스 드라이버

카테고리	데이터베이스	드라이버	지원 버전
RDBMS	Generic	JDBC	any compliant driver
	MySQL	JDBC	5.x ~ 8.x
	AWS MySQL	JDBC	1.x
	MariaDB	JDBC	1.x ~ 3.x
	PostgreSQL	JDBC	-
	Oracle	ojdbc	5, 6, 7, 8, 14
	Microsoft SQL Server	JDBC	-
	DB2	JDBC	-
	Derby	JDBC	-
	H2	JDBC	-
	jTDS	JDBC	-
	Tibero	-	-
NoSQL	Redis	Jedis	-
	Redis	Lettuce	5.x ~ 6.x

애플리케이션 서버별 호환성 및 데이터 소스

Tomcat

호환성

WAS Ver.	Java SE 6 (jdk1.6.0)	Java SE 7 (jdk1.7.0_80)	Java SE 8 (jdk1.8.0_91)
Tomcat 6	O	O	O
Tomcat 7	O	O	O
Tomcat 8	X	O	O
Tomcat 9	X	X	O

WAS Ver.	OpenJDK 6 (1.6.0_45)	OpenJDK 7 (1.7.0_80)	OpenJDK 8 (1.8.0_91)
Tomcat 6	O	O	O
Tomcat 7	O	O	O
Tomcat 8	X	O	O
Tomcat 9	X	X	O

WAS Ver.	ibm-java-x86_64-60	ibm-java-x86_64-71	ibm-java-x86_64-80
Tomcat 6	O	O	O
Tomcat 7	O	O	O
Tomcat 8	X	O	O

WAS Ver.	ibm-java-x86_64-60	ibm-java-x86_64-71	ibm-java-x86_64-80
Tomcat 9	X	X	O

DataSource

DB	JDBC Driver 파일명	JDBC Ver.	JDK Ver.	호환성
MySQL	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
MariaDB	mariadb-java-client-1.4.6.jar	1.4.6	JDK 1.7.0_80	O
PostgreSQL	postgresql-9.4.1209.jre7.jar	9.4.1209	JDK 1.7.0_80	O
Amazon Aurora	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
Oracle	ojdbc6-11.2.0.2.0.jar	11.2.0.2.0	JDK 1.7.0_80	O
DB2	db2jcc.jar, db2jcc_license_cu.jar	1.4.2	JDK 1.7.0_80	O

JBoss

호환성

WAS Ver.	Java SE 6 (jdk1.6.0)	Java SE 7 (jdk1.7.0_80)	Java SE 8 (jdk1.8.0_91)
JBOSS EAP 7.0 (standalone)	X	X	O
JBOSS EAP 6.1.1 (standalone)	X	O	X
JBOSS EAP 6.2 (standalone)	X	O	O
JBOSS EAP 6.3 (standalone)	X	O	O
JBOSS EAP 6.4 (standalone)	X	O	O

WAS Ver.	Java SE 6 (jdk1.6.0)	Java SE 7 (jdk1.7.0_80)	Java SE 8 (jdk1.8.0_91)
JBOSS EAP 7.0 (domain)	X	X	O
JBOSS AS 5.1.0 (default)	O	O	O

WAS Ver.	OpenJDK 6 (1.6.0_45)	OpenJDK 7 (1.7.0_80)	OpenJDK 8 (1.8.0_91)
JBOSS EAP 7.0 (standalone)	X	X	O
JBOSS EAP 6.1.1 (standalone)	X	O	X
JBOSS EAP 6.2 (standalone)	X	O	O
JBOSS EAP 6.3 (standalone)	X	O	O
JBOSS EAP 6.4 (standalone)	X	O	O
JBOSS EAP 7.0 (domain)	X	X	O
JBOSS AS 5.1.0 (default)	O	O	O

WAS Ver.	ibm-java-x86_64-60	ibm-java-x86_64-71	ibm-java-x86_64-80
JBOSS EAP 7.0 (standalone)	X	X	O
JBOSS EAP 6.1.1 (standalone)	X	O	X
JBOSS EAP 6.2 (standalone)	X	O	O
JBOSS EAP 6.3 (standalone)	X	O	O
JBOSS EAP 6.4 (standalone)	X	O	O
JBOSS EAP 7.0 (domain)	-	-	-
JBOSS AS 5.1.0 (default)	O	O	O

DataSource

DB	JDBC Driver 파일명	JDBC Ver.	JDK Ver.	호환성
MySQL	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
MariaDB	mariadb-java-client-1.4.6.jar	1.4.6	JDK 1.7.0_80	O
PostgreSQL	postgresql-9.4.1209.jre7.jar	9.4.1209	JDK 1.7.0_80	O
Amazon Aurora	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
Oracle	ojdbc6-11.2.0.2.0.jar	11.2.0.2.0	JDK 1.7.0_80	O
DB2	db2jcc.jar, db2jcc_license_cu.jar	1.4.2	JDK 1.7.0_80	O

JEUS

호환성

WAS Ver.	Java SE 6 (jdk1.6.0)	Java SE 7 (jdk1.7.0_80)	Java SE 8 (jdk1.8.0_91)
JEUS 7	O	O	O
JEUS 6	O	O	O

WAS Ver.	OpenJDK 6 (1.6.0_45)	OpenJDK 7 (1.7.0_80)	OpenJDK 8 (1.8.0_91)
JEUS 7	O	O	O
JEUS 6	O	O	O

WAS Ver.	ibm-java-x86_64-60	ibm-java-x86_64-71	ibm-java-x86_64-80
JEUS 7	O	O	O

WAS Ver.	ibm-java-x86_64-60	ibm-java-x86_64-71	ibm-java-x86_64-80
JEUS 6	O	O	X

DataSource

DB	JDBC Driver 파일명	JDBC Ver.	JDK Ver.	호환성
MySQL	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
MariaDB	mariadb-java-client-1.4.6.jar	1.4.6	JDK 1.7.0_80	O
PostgreSQL	postgresql-9.4.1209.jre7.jar	9.4.1209	JDK 1.7.0_80	O
Amazon Aurora	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
Oracle	ojdbc6-11.2.0.2.0.jar	11.2.0.2.0	JDK 1.7.0_80	O
DB2	db2jcc.jar, db2jcc_license_cu.jar	1.4.2	JDK 1.7.0_80	O

WebSphere

호환성

WAS Ver.	ibm-java-x86_64-60
8.5.5.10	O

DataSource

DB	JDBC Driver 파일명	JDBC Ver.	JDK Ver.	호환성
Mysql	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O

DB	JDBC Driver 파일명	JDBC Ver.	JDK Ver.	호환성
Oracle	ojdbc6-11.2.0.2.0.jar	11.2.0.2.0	JDK 1.7.0_80	O

JETTY

호환성

WAS Ver.	Java SE 6 (jdk1.6.0)	Java SE 7 (jdk1.7.0_80)	Java SE 8 (jdk1.8.0_91)
Jetty 8.1.21	X	O	O
Jetty 9.2.18	X	O	O
Jetty 9.3.12	X	X	O

WAS Ver.	ibm-java-x86_64-60	ibm-java-x86_64-71	ibm-java-x86_64-80
Jetty 8.1.21	X	O	O
Jetty 9.2.18	X	O	O
Jetty 9.3.12	X	X	O

DataSource

DB	JDBC Driver 파일명	JDBC Ver.	JDK Ver.	호환성
MySQL	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O
Oracle	ojdbc6-11.2.0.2.0.jar	11.2.0.2.0	JDK 1.7.0_80	O

공통 지원 환경

서비스 제공 형태

와탭 모니터링은 SaaS 또는 온프레미스(설치형) 방식으로 이용할 수 있습니다. 어느 쪽을 선택해도 같은 에이전트로 같은 모니터링 기능을 사용합니다.

Deployment	Support	Description
SaaS	✅	와탭이 운영하는 수집 서버에 연결. 리전별 주소는 방화벽 항목 참조
온프레미스(설치형)	✅	수집 서버를 직접 구축해 운영(폐쇄망 환경 지원). 도입 범위는 영업 담당자와 협의

> ⓘ 인터넷에 연결하지 않는 폐쇄망 환경에서도 온프레미스로 구축할 수 있습니다.

제공 형태에 따른 기능 차이

어느 형태를 선택해도 모니터링 기능은 같습니다. 모바일 앱만 예외로 service.whatap.io SaaS에서만 사용할 수 있습니다.

> ⓘ 온프레미스에서 일부 기능을 사용하려면 별도 계약이 필요합니다.

브라우저 지원

와탭 모니터링 서비스는 웹브라우저와 모바일 앱에서 이용할 수 있습니다.

브라우저	권장 여부	지원 버전
Google Chrome	O	111 이상
Mozilla FireFox	X	최신 버전

브라우저	권장 여부	지원 버전
Edge	X	최신 버전
Safari	X	최신 버전

지원 안내

- 브라우저 호환성과 성능을 이유로 Chrome 최신 버전 사용을 권장합니다.
- 사용자 인터페이스(User Interface, UI)는 HTML5 표준 기술로 구현되어 Internet Explorer는 지원하지 않습니다.

제약 사항

와탭의 웹 인터페이스는 모바일 브라우저를 지원하지 않습니다. 모바일 기기에서 와탭에 접속하려면 Android 또는 iOS용 와탭 앱을 설치하세요. 와탭 모바일 앱은 모바일 환경에 최적화되어 있습니다. WhaTap 모바일 앱에 대한 자세한 내용은 다음 문서를 참고하세요.

방화벽

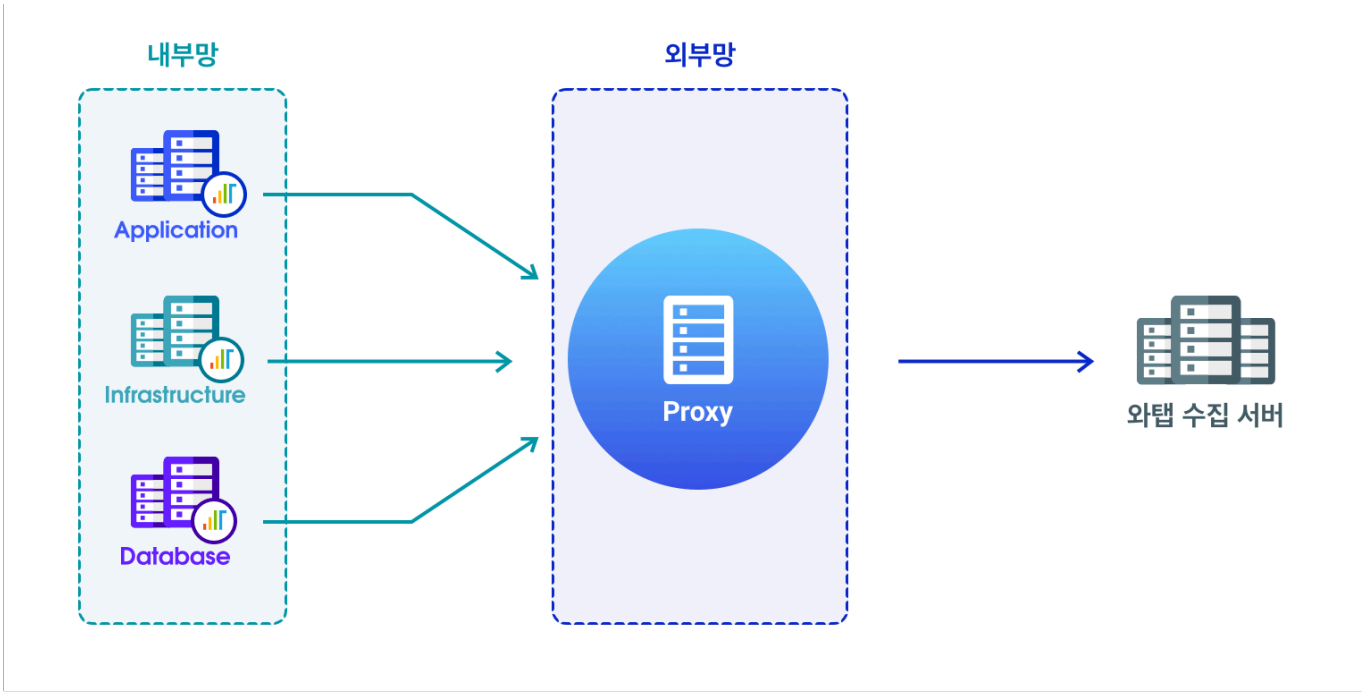
와탭 에이전트는 수집 서버 **TCP 6600** 포트로 접근할 수 있어야 합니다. 모니터링 대상과 가까운 수집 서버 주소를 허용하세요.

출발지: 와탭 에이전트

클라우드	목적지	목적지 IP	포트
AWS	와탭 서울 수집 서버	13.124.11.223 / 13.209.172.35	TCP 6600
	와탭 도쿄 수집 서버	52.68.36.166 / 52.193.60.176	TCP 6600
	와탭 싱가포르 수집 서버	18.138.0.93 / 18.139.67.236	TCP 6600
	와탭 자카르타 수집 서버	108.136.91.69 / 108.137.158.44	TCP 6600
	와탭 태국 수집 서버	43.209.109.219 / 43.209.48.86	TCP 6600

클라우드	목적지	목적지 IP	포트
	와탭 뭄바이 수집 서버	13.127.125.69 / 13.235.15.118	TCP 6600
	와탭 캘리포니아 수집 서버	52.8.223.130 / 52.8.239.99	TCP 6600
	와탭 버지니아 수집 서버	107.23.220.101 / 54.236.221.105	TCP 6600
	와탭 프랑크프루트 수집 서버	3.125.142.162 / 3.127.76.140	TCP 6600
Azure	와탭 서울 수집 서버	52.231.66.38 / 20.194.5.115	TCP 6600
	와탭 도쿄 수집 서버	52.246.169.54 / 20.210.27.232	TCP 6600
Kakao	와탭 서울 수집 서버	61.109.237.237 / 61.109.238.166	TCP 6600

에이전트에서 수집 서버로 직접 접속할 수 없다면 제공하는 Proxy 모듈을 이용해 경유하세요.



모바일 앱

와탭 모바일 앱은 안드로이드와 iOS 환경을 지원합니다. 링크로 이동하거나 QR 코드를 스캔해 와탭 앱을 설치할 수 있습니다.
모바일 앱에 대한 자세한 설명은 다음 문서를 참고하세요.

iOS	Android
	
iOS 12 버전 이상	Android 5.0 버전 이상

Java 에이전트 설치

에이전트 다운로드

액세스 키를 발급 받으면 **에이전트 다운로드** 섹션에서 **다운로드** 버튼이 활성화됩니다.

1. **다운로드** 버튼을 선택해 와탭 에이전트 파일을 다운로드하세요.

   > 다음 명령어를 이용해 서버에서 직접 다운로드할 수도 있습니다.
   >
   > ```
   > wget https://api.whatap.io/agent/whatap.agent.java.tar.gz
   > ```

2. 모니터링 대상 서버에 접속하세요.
3. 모니터링 대상 서버의 원하는 경로에 에이전트 파일의 압축을 해제하세요. 압축을 해제한 경로가 `$WHATAP_HOME`이 됩니다.

   ```
   tar -zxvf whatap.agent.java.tar.gz
   ```

에이전트 설치 파일 구성

에이전트 파일은 애플리케이션 모니터링에 필요한 정보를 추출해 와탭 수집 서버로 전달하는 트레이서와 트레이서가 데이터를 보낼 수 있도록 돕는 요소로 구성되어 있습니다.



whatap.conf 파일 설정하기

기본 정보 입력하기

압축을 푼 `$WHATAP_HOME` 경로로 이동하세요. $WHATAP_HOME/whatap.conf 파일에 액세스 키와 데이터 수집 서버 IP 주소를 입력하세요. 와탭 모니터링 서비스 화면으로 이동해 **설치 안내** 섹션의 **에이전트 다운로드** 항목에서 **액세스 키**, **수집 서버 IP 주소** 값을 복사한 다음 붙여넣기하세요.

```
license={액세스 키}
whatap.server.host={수집 서버 IP}
```

> ⓘ whantap.conf 파일에 대한 자세한 내용은 다음 문서를 참조하세요.

보안키 설정하기

SQL 변수, HTTP 쿼리를 조회하거나 Thread 중지 기능 이용하려면 보안키를 설정하세요.

- Java 에이전트 2.2.2 버전 이상

  `$WHATAP_HOME` 경로에 security.conf 파일을 생성한 후 다음과 같이 6자리 영어, 숫자로 구성된 암호를 입력하세요.

  security.conf

  ```
  paramkey=ABCDEF # SQL 변수 및 HTTP 쿼리 조회
  threadkill=ABCDEF # Thread 중지 기능
  ```

- Java 에이전트 2.2.2 버전 미만

  `$WHATAP_HOME` 경로에 paramkey.txt 파일을 생성한 후 다음과 같이 6자리 영어, 숫자로 구성된 암호를 입력하세요.

  paramkey.txt

  ```
  ABCDEF # SQL 변수 및 HTTP 쿼리 조회, Thread 중지 기능
  ```

Java 에이전트 2.2.2 버전 미만에서 2.2.2 버전으로 업데이트하는 경우 기존 paramkey.txt 파일의 키값은 seuciry.conf 파일의 키값으로 자동 적용됩니다. 예를 들어, paramkey.txt에 `FEDCBA` 를 사용한 상태에서 2.2.2 버전으로 업데이트했다면, security.conf 파일에는 다음과 같이 적용됩니다.

```
paramkey=FEDCBA
threadkill=FEDCBA
```

paramkey.txt 파일이 존재하지 않을 경우 security.conf 파일의 키값은 **WHATAP**이라는 키값으로 자동 생성됩니다.

- SQL 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
- HTTP 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.

JVM 옵션 추가

사용하는 애플리케이션 서버 환경에 따른 설정 방법을 다음과 같이 제공합니다. 에이전트를 설치하고 적용하기 전에 지원 환경을 먼저 확인하세요.

Tomcat

Tomcat 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

JBoss

JBoss 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

JEUS

JEUS 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

WebLogic

WebLogic 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

WebSphere

WebSphere 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

Resin

Resin 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

Liberty

Liberty 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

Jetty

Jetty 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

Spring Boot

Spring Boot 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

Play2

Play2 프레임워크에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

IBM Bluemix

IBM Bluemix 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

Elastic Beanstalk

AWS Elastic Beanstalk 환경에서 Java 에이전트를 적용하기 위한 설정 방법을 안내합니다.

Docker

Docker 환경에서 Java 에이전트를 적용하기 위한 JVM 옵션 설정 방법을 안내합니다.

배치 프로그램

배치 애플리케이션에 와탭 에이전트를 적용하기 위한 설치 및 설정 절차를 안내합니다.

이전 Java 버전

Java 5~8 이하 환경에서 에이전트를 적용할 때 필요한 JVM 옵션을 안내합니다.

Tomcat JVM 옵션 추가

환경 변수 추가

Tomcat 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다. 사용하는 운영체제에 맞게 설정하세요.

Linux

catalina.sh 파일 상단에 `JAVA_OPTS` 환경 변수를 추가하세요.

{Tomcat_PATH}/bin/catalina.sh

```
########## WHATAP START #############
WHATAP_HOME=/whatap
WHATAP_JAR=`ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1`
JAVA_OPTS="${JAVA_OPTS} -javaagent:${WHATAP_JAR} "
########## WHATAP END #############
```

> ⓘ `ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1` 항목은 향후 에이전트 업데이트를 진행할 경우 최신 와탭 에이전트를 적용하기 위해서입니다.

Windows

- setup.bat 파일을 통해 실행하는 경우

  catalina.bat 파일 상단에 `JAVA_OPT` 환경 변수를 추가하세요.

  {Tomcat_PATH}/bin/catalina.bat

  ```
  rem ########## WHATAP START #############
  set WHATAP_HOME=C:\whatap
  for /f %%f in ('dir /b /on "%WHATAP_HOME%\whatap.agent-*.jar"') do set last=%%f
  ```

```
set WHATAP_JAR=%last%
set WHATAP_OPTS=-javaagent:%WHATAP_HOME%\%WHATAP_JAR%

if "x%JAVA_OPTS%"=="x" goto setWhatap

set JAVA_OPTS_TMP=%JAVA_OPTS:"=%
if not "x%JAVA_OPTS_TMP:whatap=%"=="x%JAVA_OPTS_TMP%" goto endWhatap

:setWhatap
set JAVA_OPTS=%JAVA_OPTS% %WHATAP_OPTS%

:endWhatap
rem ########## WHATAP END #############
```

- Tomcat을 Windows 인스톨러로 설치한 경우

    1. **시작** > **Apache Tomcat** `X.Y.Z.` > **Configure Tomcat**을 선택하세요.
    2. **Java** 탭을 선택하세요.
    3. **Java Options** 텍스트 박스에 `-javaagent` 옵션을 추가하세요.

```
-javaagent:{와탭 설치 경로}/whatap.agent-X.Y.Z.jar
```

Apache Tomcat

Java 17 버전 이상

Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는

기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

> ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

JBoss JVM 옵션 추가

환경 변수 추가

JBoss 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다.

JVM 옵션에 `-javaagent` 및 `-Djboss.modules.system.pkgs` 옵션을 추가하세요.

JBOSS 7.0 이상 EAP 6.0 이상

설정 위치: (JBoss 설치 경로)/bin/standalone.conf(domain.conf)

```
#!/bin/sh
########## WHATAP #############
WHATAP_HOME=/home/ec2-user/whatap
WHATAP_JAR=`ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1`
JAVA_OPTS="${JAVA_OPTS} -javaagent:${WHATAP_JAR} -Djboss.modules.system.pkgs=whatap"
########## WHATAP #############
```

> ⓘ JBoss 5.0 버전 이하는 (JBoss 설치 경로)/bin/run.conf 경로에서 설정하세요. 에러가 발생하면 `-Djboss.platform.mbeanserver=true` 옵션을 추가하세요.

Java 17 버전 이상

Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

> ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어,

프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

JEUS JVM 옵션 추가

JVM 옵션 추가

JEUS 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다.

1. JEUS 버전에 따른 시작 옵션의 설정 파일의 경로를 참고해 `-javaagent` 옵션을 추가하세요.

- JEUS 7 JEUS 6

$JEUS_HOME/domains/jeus_domain/config.xml

```
<domain>
  <servers>
    <server>
      <name>server1</name>
      <jvm-config>
        <jvm-option>
          -Xmx1024m -XX:MaxPermSize=128m
          -javaagent:/whatap/whatap.agent-X.Y.Z.jar
        </jvm-option>
      </jvm-config>
    </server>
  </servers>
  ...
</domain>
```

$JEUS_HOME/config/$hostname/JEUSMain.xml

```
<node>
  <name>node01</name>
  <engine-container1>
    <name>container1</name>
    ...
    <command-option>-Xmx1024m -XX:MaxPermSize=128m
      -javaagent:/whatap/whatap.agent-#.#.#.jar
```

```
    </command-option>
    ...
  </engine-container1>
</node>
```

1. 애플리케이션 서버를 다시 시작하세요.

   ```
   jdown && jboot
   ```

2. 애플리케이션 서버 로그 파일과 에이전트 로그 파일을 통해 에이전트가 정상 작동하는지, 에러가 발생하지 않았는지 확인하세요. 로그 파일의 위치는 다음을 참고하세요.

   - 에이전트: $WHATAP_HOME/logs/whatap-{SERVER_NAME}-{DATE}.log
   - JEUS 7: $JEUS_HOME/domains/$HOST_NAME/servers/$NODE_NAME/logs/JeusServer.log
   - JEUS 6: $JEUS_HOME/logs/$NODE_NAME/JeusServer.log

3. 에이전트가 애플리케이션 서버의 종류와 서비스 컨테이너 명을 인식했는지 확인하세요.

ⓘ 와탭 모니터링 서비스에 로그인한 다음 프로젝트를 선택하세요. **애플리케이션** > **인스턴스 성능 분석** 메뉴를 선택한 다음 **실행 환경 변수** 탭을 선택하세요. **whatap.name**과 **whatap.type** 항목을 확인하세요. **whatap.name** 항목의 마지막 요소가 컨테이너 이름이어야 합니다. **whatap.type** 항목에는 애플리케이션 서버의 종류가 명시되어야 합니다.



ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

> ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

WebLogic JVM 옵션 추가

JVM 옵션 추가

WebLogic 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다.

사용하는 운영체제를 확인 후 설정을 완료하세요.

- Linux Windows

{WebLogic_Path}/user_projects/domains/{User_Domain}/bin/startWebLogic.sh

```
########## WHATAP START ############
WHATAP_HOME=/path/to/whatap
WHATAP_JAR=`ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1`
JAVA_OPTIONS="${JAVA_OPTIONS} -javaagent:${WHATAP_JAR} "
########## WHATAP END ############
```

{WebLogic_Path}/user_projects/domains/{User_Domain}/bin/startWebLogic.bat

```
rem ########## WHATAP START ############
set WHATAP_HOME=\path\to\whatap
for /f %%f in ('dir /b /on "%WHATAP_HOME%\whatap.agent-*.jar"') do set last=%%f
set WHATAP_JAR=%last%
set WHATAP_OPTS=-javaagent:%WHATAP_HOME%\%WHATAP_JAR%
if "x%JAVA_OPTIONS%"=="x" goto setWhatap
set JAVA_OPTIONS_TMP=%JAVA_OPTIONS:"=%
if not "x%JAVA_OPTIONS_TMP:whatap=%"=="x%JAVA_OPTIONS_TMP%" goto endWhatap
:setWhatap
set JAVA_OPTIONS=%JAVA_OPTIONS% %WHATAP_OPTS%
:endWhatap
rem ########## WHATAP END ############
```

> ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

WebSphere JVM 옵션 추가

JVM 옵션 추가

WebSphere 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다. JVM 옵션을 추가하려면 관리 콘솔(Admin console)로 접속하세요.

1. 웹브라우저에서 관리 콘솔에 로그인하세요.
2. **Environment** > **Virtual Hosts** > **default host** > **Host Aliases** 항목을 선택한 다음 에이전트를 설치할 서버의 포트 번호를 확인하세요.

   WebSphere port
3. **Servers** > **Server Types** > **Websphere application servers** 항목으로 이동한 다음 에이전트를 설치할 서버를 선택하세요.

   WebSpherer application servers
4. 선택한 서버 화면에서 **Configuration** 탭을 선택하고 **Server Infrastructure** 섹션의 **Java and Process Management** > **Process definition** 메뉴를 선택하세요.

   WebSphere Configuration
5. 오른쪽 메뉴에서 **Additional Properties** > **Java Virtual Machine** 메뉴를 선택하세요.
6. **Configuration** 탭에서 **Generic JVM arguments** 텍스트 박스에 `-javaagent` 와 `-Dwhatap.port` 를 추가하세요.

   ```
   -javaagent:/whatap/whatap.agent-X.Y.Z.jar
   -Dwhatap.port={포트번호} ## 2번 항목 포트 번호 ##
   ```

   Add JVM arguments

> ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.
>
> ```
> ```

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

Java 5 버전 이하, IBM Java는 지원하지 않습니다.

클라이언트 IP 주소가 수집되지 않을 때

대시보드나 기타 다른 메뉴에서 클라이언트 IP 주소가 정상 수집되지 않을 경우 다음을 참조하세요.

IBM HTTP 서버에서 백엔드의 WebSphere 애플리케이션 서버로 실제(original) 클라이언트 IP 주소로 전달해야 Java 에이전트에서 유효한 클라이언트 IP 주소를 수집할 수 있습니다. 실제 클라이언트 IP 주소를 전달하려면 IBM HTTP 서버에서 HTTP 헤더(header)에 `X-Forwarded-For` 추가해야 합니다.

이에 대한 자세한 내용은 IBM 지원 문서에서 다루고 있습니다. 다음 링크를 참조하세요. Passing the real client IP from IBM HTTP Server 9.0 to WebSphere Application Server

> ⓘ IBM 지원 문서를 확인하려면 IBM 계정이 필요할 수 있습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음

문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Resin JVM 옵션 추가

JVM 옵션 추가

Resin을 사용하는 애플리케이션 서버 환경에서 JVM 옵션을 추가하는 방법을 확인하세요. Java 버전에 따라 적용해야할 JVM 옵션이 다를 수 있습니다. 사용자의 환경에 맞는 JVM 옵션을 적용하세요. 옵션을 적용한 다음 애플리케이션 서버를 다시 시작하세요.

- 옵션 1: resin.properties 파일에 `jvm_args` 속성을 사용해 `javaagent` 인수를 추가하세요.
- 옵션 2: `conf/resin.conf` 또는 `conf/resin.xml` 파일에 `<jvm-args>` 섹션을 추가해 `-javaagent` 인수를 설정하세요.

> ⓘ
> - 애플리케이션 서버 로그 파일과 에이전트 로그 파일을 통해 에이전트가 정상 작동하는지, 에러가 발생하지 않았는지 확인하세요. 로그 파일의 위치는 다음을 확인하세요.
>   - 에이전트: $WHATAP_HOME/logs/whatap-`{SERVER_NAME}`-`{DATE}`.log
>   - RESIN 4.x: $RESIN_HOME/log/jvm-app-#.log
> - 에이전트가 정상 작동하지 않거나 에러가 발생한다면 다음 문서를 확인하세요.

Java 6 ~ 8 버전

resin.properties

```
jvm_args : -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar
```

resin.conf | resin.xml

```
<jvmarg>-javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar</jvm-arg>
```

Java 9 ~ 15 버전

resin.properties

```
jvm_args : -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true
```

resin.conf | resin.xml

```
<jvmarg>-javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar</jvm-arg>
<jvmarg>-Djdk.attach.allowAttachSelf=true</jvm-arg>
```

Java 16 버전 이상

resin.properties

```
jvm_args : -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

resin.conf | resin.xml

```
<jvmarg>-javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar</jvm-arg>
<jvmarg>-Djdk.attach.allowAttachSelf=true</jvm-arg>
<jvmarg>--add-opens=java.base/java.lang=ALL-UNNAMED</jvm-arg> <!-- Java 17 or later -->
<jvmarg>--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED</jvm-arg>
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

  ```
  -Djdk.attach.allowAttachSelf=true
  ```

  example

  ```
  java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
  ```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Liberty JVM 옵션 추가

JVM 옵션 추가

Liberty 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다.

jvm.options 파일에 에이전트 옵션을 추가하세요. 파일이 없는 경우 신규로 생성하세요.

```
-javaagent:/app/whatap/whatap.agent-2.1.1.jar
-Dorg.osgi.framework.bootdelegation=whatap.* ## OSGI 프레임워크 설정을 위한 추가 옵션 ##
```

ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Jetty JVM 옵션 추가

JVM 옵션 추가

Jetty 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다.

- jetty.sh 파일에 JVM 옵션 추가

  $JETTY_HOME/bin/jetty.sh

```
...
JAVA_OPTIONS+=("-javaagent:/home/vagrant/whatap/whatap.agent-X.Y.Z.jar -Dwhatap.port=8080")
usage()
{
    echo "Usage: ${0##*/} [-d] {start|stop|run|restart|check|supervise} [ CONFIGS ... ] "
    exit 1
}
[ $# -gt 0 ] || usage
...
```

- 쉘 스크립트 없이 Jetty 시작하기

```
$ java -javaagent:/home/vagrant/whatap/whatap.agent-X.Y.Z.jar -Dwhatap.port=8080 -jar start.jar &
```

ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

> ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어,

프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Spring Boot JVM 옵션 추가

Spring Boot를 사용하는 애플리케이션 서버 환경에서 JVM 옵션을 추가하는 방법을 확인하세요.

JVM 옵션 추가

- Java 17 버전 이상의 경우 reflection 관련 JVM 옵션 추가로 애플리케이션을 기동할 수 있습니다.

```
# Java 17 이상만 지원
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Dwhatap.oname={Agent_Name} --add-opens=java.base/
java.lang=ALL-UNNAMED -jar {application.jar}
```

- 시작 스크립트에 JVM 옵션으로 `-javaagent` 를 추가하세요.

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -jar {application.jar}
```

JVM 옵션을 추가한 다음 사용하는 Spring Boot의 버전과 에이전트 버전에 맞춰 적합한 에이전트 설정을 진행하세요.

Spring Boot 3

Spring Boot 3.0.0 버전 이상

- Java 에이전트 v2.2.9 버전 이상(whatap.agent-2.2.9.jar)

  whatap.conf

```
weaving=spring-boot-3.0
```

- Java 에이전트 v2.2.5 버전 이상(whatap.agent-2.2.5.jar)

  whatap.conf

```
weaving=spring-boot-3.0,tomcat10
```

- Java 에이전트 v2.2.4 버전 이하(whatap.agent-2.2.4.jar)
  - `jakarta.servlet.http.HttpServlet` 추적(동기 방식)

    whatap.conf

    ```
    weaving=tomcat10
    ```

  - WebClient 추적(비동기 방식)

    whatap.conf

    ```
    weaving=webflux-6.0
    ```

Spring Boot 2

Spring Boot 2.7.0 버전 이상

- Java 에이전트 v2.2.9 버전 이상(whatap.agent-2.2.9.jar)

  whatap.conf

  ```
  weaving=spring-boot-2.7
  ```

- Java 에이전트 v2.2.5 버전 이상(whatap.agent-2.2.5.jar)

  whatap.conf

  ```
  weaving=spring-boot-2.7,tomcat9
  ```

- Java 에이전트 v2.2.4 버전 이하(whatap.agent-2.2.4.jar)

  WebClient 추적(비동기 방식)

whatap.conf

```
weaving=webflux-5.3
```

Spring Boot 2.5.0 버전 이상

- Java 에이전트 v2.2.9 버전 이상(whatap.agent-2.2.9.jar)

  whatap.conf

  ```
  weaving=spring-boot-2.5
  ```

- Java 에이전트 v2.2.5 버전 이상(whatap.agent-2.2.5.jar)

  whatap.conf

  ```
  weaving=spring-boot-2.5,tomcat9
  ```

- Java 에이전트 v2.2.4 버전 이하(whatap.agent-2.2.4.jar)

  WebClient 추적(비동기 방식)

  whatap.conf

  ```
  weaving=webflux-5.3
  ```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

  ```
  -Djdk.attach.allowAttachSelf=true
  ```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

> ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Play2 JVM 옵션 추가

JVM 옵션 추가

Play2 애플리케이션 서버 환경에서 Java 모니터링 에이전트를 효과적으로 사용하기 위해 필요한 JVM 옵션을 추가하는 방법을 안내합니다.

$PLAY_HOME/framework/build 파일에 에이전트 옵션을 설정하세. 별도로 이용하는 시작 또는 중지 스크립트가 있다면 해당 파일에 적용하세요.

```
#! /usr/bin/env sh

########## WHATAP START #############
WHATAP_HOME=/apps/whatap
WHATAP_JAR=`ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1`
JAVA_OPTS="${JAVA_OPTS} -Dwhatap.play2=2.2.6 -javaagent:${WHATAP_JAR}"
########## WHATAP END #############

if [ -z "${PLAY_VERSION}" ]; then
  PLAY_VERSION="2.2.6"
...
```

ⓘ Play는 기본 설정 외 `-Dwhatap.play2=2.2.6` 옵션을 추가해야 합니다.

ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

> ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어,

프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

IBM Bluemix JVM 옵션 추가

에이전트 적용하기

IBM BlueMix를 이용해 배포할 경우 와탭 에이전트를 적용하는 방법입니다.

IBM BlueMix는 컨테이너로 **WebSphere Liberty** 환경을 제공합니다. **Liberty**는 **WebSphere Application Server**와 다른 경량화 환경으로 **Spring Boot**가 동작하는 방식과 유사합니다.

> ⓘ **Platform as a Service(PaaS)** 환경에서는 배포 대상 애플리케이션에 와탭 에이전트의 설정을 포함해 배포합니다.

다음은 설정 환경 예제입니다. 가이드에 따른 환경을 구성할 경우 로컬 개발 환경에서 생성하는 파일들입니다.

```
whatap@vmwas01:/apps/bluemix/java-helloworld$ ls -alrt
합계 64
drwxrwxr-x 3 whatap whatap  4096 10월 29 13:13 ..
-rw-rw-r-- 1 whatap whatap  1079 10월 29 13:13 .classpath
-rw-rw-r-- 1 whatap whatap  1184 10월 29 13:13 .project
-rw-rw-r-- 1 whatap whatap    39 10월 29 13:13 .gitignore
-rw-rw-r-- 1 whatap whatap   151 10월 29 13:13 CONTRIBUTING.md
drwxrwxr-x 2 whatap whatap  4096 10월 29 13:13 .settings
-rw-rw-r-- 1 whatap whatap  2823 10월 29 13:13 pom.xml
-rw-rw-r-- 1 whatap whatap   122 10월 29 13:13 manifest.yml
-rw-rw-r-- 1 whatap whatap  3522 10월 29 13:13 README.md
-rw-rw-r-- 1 whatap whatap 11323 10월 29 13:13 LICENSE
drwxrwxr-x 3 whatap whatap  4096 10월 29 13:13 src
drwxrwxr-x 2 whatap whatap  4096 10월 29 13:13 target
drwxrwxr-x 8 whatap whatap  4096 10월 29 13:13 .git
drwxrwxr-x 6 whatap whatap  4096 10월 29 15:26 .
```

1. `${APP_HOME}` 에서 src/main/resources/whatap-agent/ 디렉터리를 생성하고 jar 파일, conf 파일을 복사하세요.

   ```
   $ mkdir -p src/main/resources/whatap-agent/
   ```

```
$ cp /apps/whatap/whatap.agent.tracer-1.5.4.jar src/main/resources/whatap-agent/
$ cp /apps/whatap/whatap.conf src/main/resources/whatap-agent/
```

2. ${APP_HOME}/manifest.yml 파일에 옵션을 추가하세요. yml 파일이므로 공백, 들여쓰기 기준을 잘 맞춰 작성하세요.

```
1  ---
2  applications:
3  - name: sample-java-helloworld
4  random-route: true
5  memory: 256M
6  path: target/JavaHelloWorldApp.war
7  # 여기서부터 추가합니다.
8  env:
9    JAVA_OPTS: "-javaagent:/{APPLICATION_DIR}/WEB-INF/classes/whatap-agent/whatap.agent-X.Y.Z.jar
   -Dorg.osgi.framework.bootdelegation=whatap.* "
```

ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

ⓘ whatap.conf 설정은 PaaS가 아닌 환경과 동일하게 적용합니다. 적용 후 에이전트명 식별이 어려울 수 있으니 상황에 맞는 에이전트명을 적용하세요. 에이전트 네이밍에 관한 자세한 내용은 다음 문서를 참조하세요.

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Elastic Beanstalk JVM 옵션 추가

에이전트 적용하기

Elastic Beanstalk를 이용해 배포할 경우 와탭 에이전트를 적용하는 방법입니다.

> ⓘ **Platform as a Service(PaaS)** 환경에서는 배포 대상 애플리케이션에 와탭 에이전트의 설정을 포함해 배포합니다.

1. Spring Boot 결과물을 jar로 배포할 경우 .ebextensions의 내용을 적용할 수 없습니다. 다음 파일들을 압축해 zip 형태로 배포하세요.
   - .elasticbeanstalk/config.yml: eb 명령을 실행하는 디렉터리 하위에 자동 생성
   - .ebextensions/`{config_name}`.config: eb 설정 파일
   - Procfile: JVM command line 옵션을 설정하기 위한 파일
   - `{application}`.jar: 실행할 applicaion.jar 파일
2. Service에 적용할 와탭 에이전트의 파일을 압축해 S3(혹은 다운로드할 수 있는 public 경로)에 업로드하세요.
   - paramkey.txt
   - whatap.agent-2.0_25.jar
   - whatap.conf

   ```
   # 디렉터리 압축하기
   zip -r whatap-agent.zip agent
   ```
3. .ebextensions/`{config_name}`.config 파일에 다운로드할 와탭 에이전트 경로를 입력하세요.

   다운로드할 수 있도록 압축 파일 형태로 만들어 둔 경우 eb를 실행해 자동으로 다운로드한 다음 압축을 풉니다.

   ```
   sources:
     target directory:
       S3경로
   ```

Example

```
sources:
  /home/webapp:
    http://s3.ap-northeast-2.amazonaws.com/{bucket-name}/whatap-agent.zip
```

4. 와탭 에이전트 옵션을 추가한 JVM command를 입력해 Procfile을 작성하세요.

Procfile

```
web: java -javaagent:${WHATAP_JAR_FILE_PATH} -Dwhatap.name=${WHATAP_NAME}
-Dwhatap.okind=${WHATAP_OKIND_NAME} -Dwhatap.server.home=${APPLICATION_PATH}
-Dwhatap.conf.path=${WHATAP_HOME} -jar ${APPLICATION}.jar
```

Example

```
web: java -javaagent:/home/webapp/agent/whatap.agent-X.Y.Z.jar -Dwhatap.name=bootTest
-Dwhatap.okind=test -Dwhatap.server.home=/var/app/current -Dwhatap.conf.path=/home/webapp/agent -jar
whatap-boot-test.jar
```

5. Procfile 파일과 .ebextensions 파일, `{application}` .jar 파일을 압축하세요.

```
zip -r {application}.zip Procfile .ebextensions/ {application}.jar
```

6. .elasticbeanstalk/config.yml 파일에 배포할 zip파일 경로를 추가하세요.

```
deploy:
  artifact: /path/to/{application}.zip
```

7. eb 배포 명령어를 실행하세요.

```
eb deploy
```

ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```

```

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

ⓘ 자세한 내용은 AWS 가이드 문서를 참조하세요.

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

ⓘ Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Docker JVM 옵션 추가

Docker는 컨테이너 기반의 가상화 플랫폼으로 Java 애플리케이션을 Docker 컨테이너 내에서 실행할 때 JVM 옵션을 추가하여 애플리케이션의 성능 및 자원 사용을 조정할 수 있습니다. 이 문서에서는 Dockerfile을 수정하여 JVM 옵션을 추가하는 방법을 안내합니다.

> ⓘ Docker가 설치되어 있어야 하며, Java 애플리케이션의 Docker 이미지를 빌드할 수 있어야 합니다.

에이전트 적용하기

기존 Docker 이미지를 바탕으로 와탭 에이전트 설정을 추가한 이미지를 빌드하세요.

> ⓘ 이 문서에서는 Java 에이전트를 설치하는 경우만을 다룹니다. 와탭 쿠버네티스 모니터링과 함께 적용하는 경우 다음 문서를 참조하세요.

Docker 빌드 디렉터리 생성하기

Docker 빌드 디렉터리를 생성하세요.

```
mkdir -p {Docker build dir}
```

에이전트 설정 파일 생성하기

whatap.conf 파일을 생성하세요.

```
1 cat >{Docker build Dir}/whatap.conf <<EOL
2 # 액세스 키를 입력하세요.
3 license=XXXXXXXXXXXXX-XXXXXXXXXXXXX-XXXXXXXXXXXXX
4 # 수집 서버 IP 정보를 입력하세요.
5 whatap.server.host=xx.xx.xx.xx/yy.yy.yy.yy
6 EOL
```

Dockerfile 생성하기

이미지를 빌드하려면 와탭 이미지에서 `-javaagent` 옵션에 적용할 jar 파일을 복사할 수 있습니다.

```
cat >/home/silver/whatap/docker/Dockerfile <<EOL
FROM whatap/kube_mon as build
## 실제 이미지 생성 (기존 이미지에 Whatap 추가)
## $Image_Name(이미지명)
FROM $Image_Name
RUN mkdir -p /whatap
COPY --from=build /data/agent/micro/whatap.agent-*.jar /whatap
COPY ./whatap.conf /whatap/
#...
EOL
```

`JAVA_OPTS` 적용하기

`JAVA_OPT` 에 다음 내용을 추가하세요.

```
WHATAP_HOME=/whatap
WHATAP_JAR=ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1
export JAVA_OPTS="-javaagent:${WHATAP_JAR} "
```

Docker 빌드하기

Docker를 빌드하세요.

```
cd docker
docker build -t $Image_Name
```

ⓘ Java 17 버전 이상의 경우 reflection 관련한 다음 옵션을 추가하세요.

```
--add-opens=java.base/java.lang=ALL-UNNAMED
```

Java 에이전트 파일 이름은 `Rename` 기능을 활용해 수정할 수 있습니다. Java 에이전트의 이름을 수정했다면 `JAVA_OPTS`에 새로운 Java 에이전트 이름을 등록하세요.

Java 에이전트 이름 수정 방법 예시

```
java -cp whatap.agent-X.Y.Z.jar whatap.agent.setup.Rename -from whatap.agent-X.Y.Z.jar -to whatap.agent.jar
```

힙 히스토그램 조회

와탭의 Java 모니터링에서는 JVM 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있는 기능을 기본으로 제공합니다. **애플리케이션** > **인스턴스 성능 분석** 메뉴에서 **힙 히스토그램** 탭을 선택하세요.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

Java 5 버전 이하, IBM Java는 지원하지 않습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

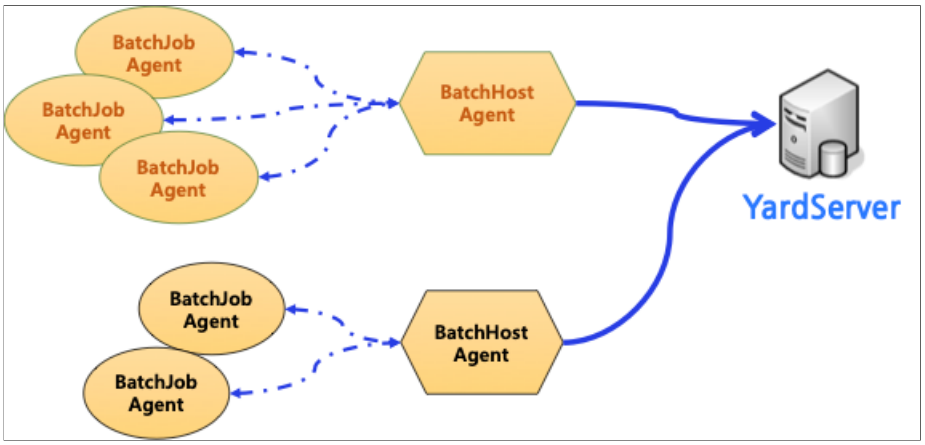
에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

배치 프로그램 모니터링

배치 애플리케이션에 와탭 에이전트를 적용하기 위한 설치 및 설정 절차를 안내합니다.



배치 모니터링 구성

배치 애플리케이션은 WAS와 달리 실행과 중단을 반복합니다. 따라서 배치 모니터링은 다음 두 개의 프로세스로 나뉩니다.

프로세스	역할	실행 단위
배치 호스트(Batch Host)	배치 에이전트에서 받은 데이터를 수집 서버로 전송	호스트당 1개
배치 에이전트(Batch Agent)	배치 잡의 데이터를 수집해 배치 호스트로 전송	배치 잡당 1개

두 프로세스는 UDP로 통신합니다. 이 구조 덕분에 여러 배치 잡을 병렬로 실행할 수 있습니다. 배치 호스트와 배치 에이전트는 배치 잡을 실행하는 서버에 함께 설치하세요.

통신 요건

경로	프로토콜/포트
배치 호스트 → 수집 서버	TCP 6600(아웃바운드)
배치 에이전트 → 배치 호스트	UDP 6611



배치 프로그램 모니터링은 Java 모니터링의 부가 기능이지만 Java 모니터링 에이전트와 별개로 동작하는 독립된 시스템입니다. Java 에이전트의 설정 옵션은 배치 모니터링에 적용되지 않습니다. 이 문서에 기재한 옵션만 사용하세요. Java 에이전트에서 `profile_` 접두어를 쓰는 옵션은 배치 프로그램에서 `trace_` 접두어를 씁니다.

에이전트 다운로드

다음 명령어를 실행해 배치 애플리케이션 모니터링용 에이전트를 내려받으세요.

```
wget https://api.whatap.io/agent/whatap.agent.batch.tar.gz
tar xzf whatap.agent.batch.tar.gz
```

압축을 해제하면 다음과 같이 구성됩니다.

```
{BATCH_HOME}/
 ├─── whatap.agent.batch.host-X.Y.Z.jar   배치 호스트
 ├─── whatap.agent-X.Y.Z.jar           Java 에이전트
 ├─── whatap.conf                  배치 호스트 설정 파일
 └─── agent/
    ├─── whatap.agent.batch-X.Y.Z.jar    배치 에이전트
    └─── whatap.conf                  배치 에이전트 설정 파일
```

whatap.conf 파일이 없다면 새로 만든 다음, **설정** > **에이전트 설치** 메뉴에서 확인할 수 있는 license 와 whatap.server.host 항목을 추가하세요.

설정 파일의 위치

> 도입할 때 가장 많이 실패하는 부분입니다. 반드시 확인하세요.
>
> 와탭은 설정 파일과 로그의 위치를 whatap.home 기준으로 결정합니다. whatap.home 이 결정되는 방식은 기동 방법에 따라 다릅니다. 설정 파일을 찾지 못하면 기동할 때 배너에 PCODE(0) 이 표시되고 데이터를 수집하지 않습니다.

whatap.home 결정 방식

우선순위는 다음과 같습니다.

1. -Dwhatap.home 을 지정하면 항상 그 값이 우선합니다.
2. 지정하지 않으면 -javaagent 로 기동한 프로세스(배치 에이전트)는 에이전트 jar이 있는 디렉터리를 whatap.home 으로 씁니다.
3. -cp 로 메인 클래스를 지정해 기동한 프로세스(배치 호스트)는 위 자동 설정이 적용되지 않으므로, 기동 시점의 작업 디렉터리가 whatap.home 이 됩니다.

프로세스	기동 방법	whatap.home	읽어들이는 설정 파일
배치 호스트	-cp + 메인 클래스	작업 디렉터리	작업 디렉터리의 whatap.conf
배치 에이전트	-javaagent	에이전트 jar이 있는 디렉터리	jar과 같은 디렉터리의 whatap.conf

> 배치 호스트는 반드시 설치 디렉터리로 이동한 다음 기동하세요. 작업 스케줄러나 잡 관리 제품에서 기동하면 작업 디렉터리가 예상과 다를 수 있습니다. 기동 스크립트 안에서 작업 디렉터리를 명시적으로 이동하세요.

배치 호스트 설정

`license` 와 `whatap.server.host` 옵션값은 프로젝트의 **설정** > **에이전트 설치** 메뉴에서 확인할 수 있습니다.

{BATCH_HOME}/whatap.conf

```
license=
whatap.server.host=
whatap.server.port=6600
```

Option	Default	Description
`license`	-	프로젝트 액세스 키
`whatap.server.host`	-	수집 서버 주소. 이중화한 경우 슬래시(/)로 구분. 예) `10.0.0.1/10.0.0.2`
`whatap.server.port`	`6600`	수집 서버 포트

위 세 항목만으로 동작합니다. 이후 설명하는 옵션은 필요할 때만 추가하세요.

UDP 수신 설정

배치 호스트가 배치 에이전트로부터 데이터를 받을 주소와 포트입니다.

{BATCH_HOME}/whatap.conf

```
net_udp_listen_ip=0.0.0.0
net_udp_listen_port=6611
```

Option	Default	Description
`net_udp_listen_port`	`6611`	UDP 데이터를 받을 포트. 기본 포트를 다른 용도로 쓰는 경우 변경
`net_udp_listen_ip`	`0.0.0.0`	UDP 데이터를 받기 위해 바인딩할 IP 주소. 변경 가능하나 기본값 권장

UDP 포트를 변경하면 배치 호스트와 배치 잡 양쪽을 모두 변경해야 합니다.

변경 대상	설정 파일	옵션
수신 측(배치 호스트)	{BATCH_HOME}/whatap.conf	net_udp_listen_port
송신 측(배치 잡)	배치 잡의 whatap.conf	net_udp_port

한쪽만 변경하면 오류 없이 트랜잭션 데이터를 잃습니다. 배치 잡은 기존 포트로 계속 보내고 배치 호스트는 변경한 포트에서만 기다리기 때문입니다. 변경한 다음에는 배치 호스트와 배치 잡을 모두 다시 기동하세요.

트레이스 스텝 수집 설정

개별 트랜잭션 트레이스는 스텝을 무한정 수집할 수 없습니다. 트랜잭션별로 길이가 제한된 버퍼에 트레이스 스텝을 저장하며, 기본 설정에서는 **한 배치 잡당 1,000개**까지 수집합니다.

{BATCH_HOME}/whatap.conf

```
split_trace_enabled=false
```

Option	Default	Description
split_trace_enabled	false	트레이스를 분할해 전송

상한을 초과한 스텝은 **트레이스에 별도 표시 없이 누락됩니다.** 트레이스 분석 화면의 스텝 번호가 1,000 부근에서 끊겨 있으면 이 경우입니다.

스텝 수가 많은 배치 잡의 전체 처리 내역이 필요하면 기술지원팀으로 문의하세요.

프로파일링 종료 대기 시간

배치 호스트는 실행 중인 잡을 주기적으로 점검해, 일정 시간 동안 스텝이 도착하지 않은 잡의 프로파일링을 강제로 종료합니다.

{BATCH_HOME}/whatap.conf

```
profile_expired_time=30000
```

Option	Default	Description
profile_expired_time	30000	마지막 스텝을 받은 시점부터 이 시간(밀리초)이 지나면 프로파일링 강제 종료

배치 잡이 비정상 종료됐을 때 트랜잭션이 실행 중인 상태로 남지 않도록 자동 정리하는 기능입니다. 강제 종료한 트랜잭션에는 `lost-profile` 스텝을 기록합니다.

외부 시스템 응답 대기처럼 계측 대상 처리가 발생하지 않는 구간이 이 시간을 넘으면, 잡이 정상 실행 중이어도 강제 종료합니다. 이런 구간이 있는 배치 잡은 실제 무입력 구간보다 여유 있는 값으로 설정하세요.

배치 에이전트 설정

{BATCH_HOME}/agent/whatap.conf

```
net_udp_ip=127.0.0.1
net_udp_port=6611

job_name=sample-batch-job
time_limit=300000

trace_httpc_limit=1000000
trace_sql_limit=1000000

active_stack_enabled=true
active_stack_start_wait_time=1000
active_stack_time1=5000
active_stack_time2=10000
```

Option	Default	Description
net_udp_ip	127.0.0.1	데이터를 보낼 배치 호스트의 IP 주소. 같은 서버에서 실행하면 변경 불필요
net_udp_port	6611	데이터를 보낼 배치 호스트의 UDP 포트. 배치 호스트의 `net_udp_listen_port`와 같은 값

Option	Default	Description
job_name	-	배치 잡을 식별하는 이름. 트랜잭션 이름으로 표시. 잡마다 고유한 값 설정
time_limit	300000	지연 경고를 표시하는 임계치(밀리초). 잡을 중단시키는 값 아님
trace_httpc_limit	1000000	배치 잡 1건에서 수집할 HTTP 호출 최대 건수
trace_sql_limit	1000000	배치 잡 1건에서 수집할 SQL 최대 건수
active_stack_enabled	true	액티브 스택 수집 여부
active_stack_start_wait_time	1000	배치 잡 시작 후 첫 액티브 스택을 수집하기까지 대기 시간(밀리초)
active_stack_time1	5000	최초 5회까지의 액티브 스택 수집 간격(밀리초)
active_stack_time2	10000	6회 이후의 액티브 스택 수집 간격(밀리초)

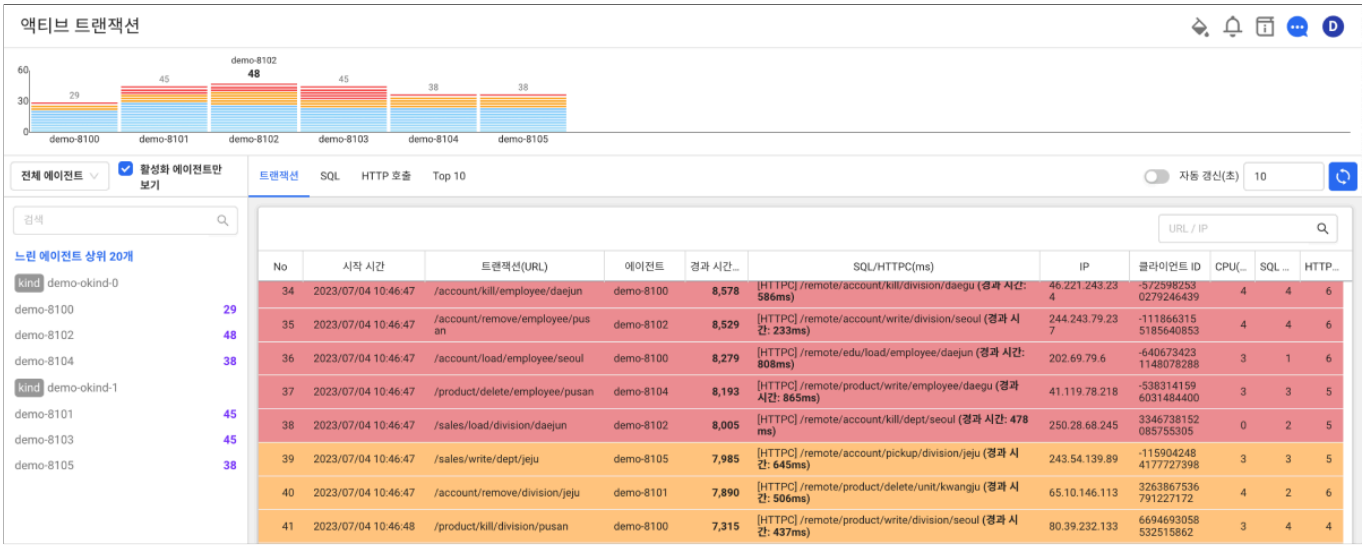
net_udp_ip 와 net_udp_port 는 배치 호스트와 같은 서버에서 실행하는 일반적인 구성이라면 기본값으로 동작합니다. 그 외 옵션은 필요할 때만 추가하세요.

time_limit 설정

배치 잡은 보통 장시간 실행하며 잡마다 예상 처리 시간이 다릅니다. 잡별로 time_limit 을 설정하세요.

time_limit 을 설정하면 액티브 트랜잭션 화면의 경고 표시 임계치를 자동으로 계산합니다. 개별 설정은 필요하지 않습니다.

경과 시간	표시
time_limit 의 70%까지	기본 색상
time_limit 의 70% 초과	노란색
time_limit 초과	빨간색



색상 판정은 잡 전체의 경과 시간을 기준으로 합니다. 트레이스를 분할한 경우에도 구간별이 아니라 잡 전체의 경과 시간으로 판정합니다.

액티브 스택 수집 타이밍

배치 에이전트는 실행 중인 배치 잡의 스택 트레이스를 일정 간격으로 수집합니다. 수집한 액티브 스택은 트레이스 분석 화면에서 **ActiveStack** 스텝으로 확인할 수 있습니다.

`active_stack_time1` 과 `active_stack_time2` 는 임계치가 아니라 수집 간격입니다. 기본값 기준 수집 타이밍은 다음과 같습니다.

회차	수집 시점	적용 옵션
1회	배치 잡 시작 후 1초	`active_stack_start_wait_time`
2회 ~ 5회	직전 수집으로부터 5초 간격	`active_stack_time1`
6회 이후	직전 수집으로부터 10초 간격	`active_stack_time2`

기본값으로 90초 실행하는 배치 잡에서는 약 12회의 액티브 스택을 수집합니다.

수집 간격이 전환되는 기준인 5회는 고정값이며 변경할 수 없습니다. 변경할 수 있는 항목은 `active_stack_start_wait_time` , `active_stack_time1` , `active_stack_time2` 세 개의 시간 설정입니다.

`active_stack_time1` 과 `active_stack_time2` 의 최솟값은 100밀리초입니다. 100 미만을 설정해도 100으로 적용합니다.

액티브 스택을 수집하지 않으려면 `active_stack_enabled=false` 로 설정하세요.

수집 건수 상한

배치 잡 1건에서 수집할 HTTP 호출과 SQL의 최대 건수입니다. 기본값은 각각 100만 건입니다.

Option	Default
`trace_httpc_limit`	`1000000`
`trace_sql_limit`	`1000000`

설정한 건수를 초과하면 해당 종류의 데이터 수집이 그 지점에서 멈추고, 초과분은 통계에 반영하지 않습니다. 배치 잡 자체는 계속 실행하며 트랜잭션도 정상적으로 끝납니다.

> 수집이 멈춰도 트레이스에 별도 표시가 남지 않습니다. 실제 처리 건수가 상한을 넘길 가능성이 있다면 미리 충분히 큰 값을 설정하세요.
>
> 트랜잭션 목록에 표시되는 SQL 건수 또는 HTTP 호출 건수가 설정한 상한값과 정확히 일치하면 상한에 도달했을 가능성이 있습니다.
>
> 두 옵션에는 제품상 상한값이 없습니다. 다만 값을 크게 설정하면 배치 프로세스의 메모리 사용량이 늘어나므로, 검증 환경에서 확인한 다음 운영 환경에 적용하세요.

잡마다 설정 분리하기

`job_name` 과 `time_limit` 은 잡마다 다른 값을 설정합니다. 설정 파일은 `whatap.home` 을 기준으로 읽어들이므로, `-Dwhatap.home` 으로 잡별 디렉터리를 지정하는 방식을 권장합니다.

에이전트 jar은 한곳에 두고 설정 파일만 잡별로 준비하세요.

```
{BATCH_HOME}/agent/whatap.agent.batch-X.Y.Z.jar   모든 잡이 공유

{JOB_HOME}/
```

```
├── job-a/
│   ├── job-a.jar
│   └── whatap.conf     job_name=job-a, time_limit=600000
└── job-b/
    ├── job-b.jar
    └── whatap.conf     job_name=job-b, time_limit=120000
```

설정 파일을 읽어들인 디렉터리에는 `logs` 디렉터리를 만듭니다. 잡 디렉터리에 `logs`가 없다면 그 디렉터리의 설정을 쓰지 않은 것입니다. 문제를 확인할 때 지표로 활용하세요.

실행

실행 순서

배치 호스트는 모든 배치 잡에 대해 하나만 기동합니다.

1. 모든 배치 잡을 실행하기 전에 배치 호스트를 1회 기동하세요.
2. 각 배치 잡을 `-javaagent` 옵션과 함께 실행하세요. 여러 잡을 병렬로 실행할 수 있습니다.
3. 모든 배치 잡이 끝난 다음 배치 호스트를 종료하세요.

> 설정 파일은 기동할 때 1회만 읽어들입니다. 설정을 변경하면 배치 에이전트는 다음에 실행하는 잡부터 반영하고, 배치 호스트는 다시 기동해야 합니다.
>
> 배치 호스트 프로세스를 강제 종료하면 실행 중인 잡에서 보낸 데이터를 잃을 수 있습니다. 모든 잡이 끝난 다음 정지하세요.

배치 호스트 기동

- Linux Windows

Batch Host

```
cd {BATCH_HOME}
java -cp {BATCH_HOME}/whatap.agent.batch.host-X.Y.Z.jar:{BATCH_HOME}/whatap.agent-X.Y.Z.jar \
```

```
    -Dwhatap.name=batch whatap.agent.batch.App
```

Java 17 이상이라면 리플렉션 관련 옵션을 추가하세요.

Batch Host, Java 17 or later

```
cd {BATCH_HOME}
java --add-opens=java.base/java.lang=ALL-UNNAMED \
    -cp {BATCH_HOME}/whatap.agent.batch.host-X.Y.Z.jar:{BATCH_HOME}/whatap.agent-X.Y.Z.jar \
    -Dwhatap.name=batch whatap.agent.batch.App
```

Batch Host

```
cd /d {BATCH_HOME}
java -cp "{BATCH_HOME}\whatap.agent.batch.host-X.Y.Z.jar;{BATCH_HOME}\whatap.agent-X.Y.Z.jar" -Dwhatap.name=batch
whatap.agent.batch.App
```

Java 17 이상이라면 리플렉션 관련 옵션을 추가하세요.

Batch Host, Java 17 or later

```
cd /d {BATCH_HOME}
java --add-opens=java.base/java.lang=ALL-UNNAMED -cp "{BATCH_HOME}\whatap.agent.batch.host-
X.Y.Z.jar;{BATCH_HOME}\whatap.agent-X.Y.Z.jar" -Dwhatap.name=batch whatap.agent.batch.App
```

윈도우에서는 클래스패스 구분자가 세미콜론(;)입니다. 명령 프롬프트는 세미콜론을 인자 구분자로 해석하므로 클래스패스 전체를 큰따옴표로 감싸야 합니다.

배치 잡 실행

- Linux Windows

Batch Job

```
java -javaagent:{BATCH_HOME}/agent/whatap.agent.batch-X.Y.Z.jar \
    -Dwhatap.home={JOB_HOME}/job-a \
    -jar {JOB_HOME}/job-a/job-a.jar
```

Java 17 이상이라면 `--add-opens=java.base/java.lang=ALL-UNNAMED` 를 추가하세요.

Batch Job

```
java -javaagent:{BATCH_HOME}\agent\whatap.agent.batch-X.Y.Z.jar ^
    -Dwhatap.home={JOB_HOME}\job-a ^
    -jar {JOB_HOME}\job-a\job-a.jar
```

Java 17 이상이라면 `--add-opens=java.base/java.lang=ALL-UNNAMED` 를 추가하세요.

기동 확인

기동할 때 표시되는 배너로 설정 적용 여부를 확인합니다. 배치 호스트와 배치 에이전트의 배너는 서로 다릅니다.

배치 호스트

```
WhaTap vX.Y.Z YYYYMMDD PCODE(프로젝트코드)
Java <JDK 버전> asmlib <버전> asm <버전>
whatap.agent-X.Y.Z.jar
./whatap.conf
```

Display	Description
PCODE(숫자)	정상. 숫자는 프로젝트 코드
PCODE(0)	license 또는 설정 파일이 적용되지 않음. 문제 해결 참조

세 번째 줄에는 배치 호스트 jar이 아니라 Java 에이전트 jar 파일명을 표시합니다. 정상 동작입니다.

배치 에이전트

```
WhaTap vX.Y.Z YYYYMMDD
Java <JDK 버전> asmlib <버전> asm <버전>
whatap.agent.batch-X.Y.Z.jar
{JOB_HOME}/job-a/whatap.conf
```

배치 에이전트 배너에는 `PCODE` 를 표시하지 않습니다. **네 번째 줄에 출력한 설정 파일 경로**가 잡별 설정을 적용했는지 확인하는

기준입니다. 의도한 잡 디렉터리의 경로가 아니라면 `whatap.home`이 예상과 다르게 결정된 것입니다.

배너의 Java·asm 값은 프로세스를 실행한 JDK에 따라 결정합니다. 두 프로세스를 다른 JDK로 기동하면 서로 다른 값을 표시합니다.

화면에서 확인하기

배치 잡은 히트맵과 트랜잭션 목록에서 트랜잭션으로 확인할 수 있습니다. 트랜잭션 이름은 `job_name`에 설정한 값입니다.

가상 트랜잭션

`split_trace_enabled=true`인 경우, 스텝 수가 많은 잡은 트레이스가 일정 단위로 분할되어 전송됩니다. 분할한 각 구간은 가상 트랜잭션으로 표시하며 이름에 일련번호가 붙습니다. **기본값은 `false` 이므로 기본 설정에서는 가상 트랜잭션이 표시되지 않습니다.**

`job_name=sample-job`이고 스텝 수가 약 5,000인 경우의 예입니다.

트랜잭션 이름	내용	표시되는 소요 시간
`sample-job-1` ~ `sample-job-4`	분할한 구간(가상 트랜잭션)	각 구간의 처리 시간
`sample-job`	실제 트랜잭션	잡 전체의 소요 시간

잡 전체의 소요 시간은 일련번호가 붙지 않은 트랜잭션에서 확인하세요. 스텝 번호는 잡 전체에서 연속됩니다.

- 가상 트랜잭션은 히트맵과 TPS에 반영하지만 통계에는 잡 1건으로 집계합니다. 트랜잭션 건수를 많게 계상하지 않습니다.
- 가상 트랜잭션은 CPU 사용 시간이 0, 클라이언트 IP 주소가 `0.0.0.0`으로 표시됩니다.
- 트레이스 분석 화면에서 스텝 실행 내역을 파일로 내려받을 때 트레이스가 분할되어 있으면 파일도 분할 단위로 나뉩니다. 잡 전체의 스텝 내역이 필요하면 가상 트랜잭션을 각각 내려받으세요.

배치 에이전트의 메트릭 지표 수집

대시보드에는 배치 호스트(Batch Host)의 Heap Memory를 수집하고, 배치 애플리케이션의 성능 지표는 메트릭(TagCount: `batch_job_counter`)으로 수집합니다.

batch_job_counter

- batch job: `pid`, `hostname`, `job_name`
- gc: `gc_count`, `gc_time_sum`, `gc_oldgen_count`
- heap: `heap_tot`, `heap_use`, `heap_max`, `heap_pending_final`, `heap_perm`
- cpu: `cputime`
- thread count: `thread_total_started`, `thread_count`, `thread_daemon`, `thread_peak_count`

문제 해결

배치 호스트 기동 배너에 PCODE(0)이 표시됩니다

1. 배너 네 번째 줄에 출력한 경로에 whatap.conf가 있는지 확인하세요.
2. 없다면 `whatap.home`이 예상과 다르게 결정된 것입니다. 설정 파일의 위치를 확인하고, 필요하면 `-Dwhatap.home`으로 명시적으로 지정하세요.
3. 설정 파일이 있다면 `license` 값이 올바른지 확인하세요. 값에 문제가 있으면 로그에 `WhaTap AccessKey Error`를 기록합니다.

java.net.BindException 오류

UDP 6611을 이미 사용 중입니다. 배치 호스트를 강제 종료한 경우 Java 프로세스가 남아 포트를 계속 점유할 수 있습니다.

- Linux Windows

```
ss -anu | grep 6611
ps -ef | grep whatap.agent.batch.App
```

```
netstat -ano | findstr ":6611"
```

남아 있는 프로세스를 종료한 다음 배치 호스트를 다시 기동하세요.

잡마다 설정한 내용이 반영되지 않습니다

해당 디렉터리를 `whatap.home`으로 인식하지 않은 것입니다. 기동 배너 네 번째 줄에 출력한 경로를 확인하세요.

가장 흔한 원인은 에이전트 jar을 공유 위치에 두고 설정 파일만 잡 디렉터리에 둔 경우입니다. `-javaagent`로 지정한 jar의 디렉터리가 `whatap.home`이 되므로 잡 디렉터리의 whatap.conf는 읽어들이지 않습니다. `-Dwhatap.home`으로 잡 디렉터리를 지정하세요.

모니터링 데이터가 표시되지 않습니다

다음 순서로 확인하세요.

1. 배치 호스트가 기동되어 있는지 확인하세요.
2. 기동 배너의 `PCODE`가 0이 아닌지 확인하세요.
3. 배치 호스트에서 수집 서버로 TCP 6600을 허용했는지 확인하세요.
4. 배치 에이전트에서 배치 호스트로 UDP 6611이 통신되는지 확인하세요.
5. UDP 포트를 변경했다면 배치 호스트의 `net_udp_listen_port`와 배치 잡의 `net_udp_port`가 같은 값인지 확인하세요.
6. `{whatap.home}/logs/whatap.log`에 오류를 기록했는지 확인하세요.

스텝이 도중에 끊겨 있습니다

수집 상한에 도달한 경우입니다. 상한에 도달해도 트레이스에 별도 표시가 남지 않으므로 다음을 확인하세요.

확인 항목	관련 설정
스텝 번호가 1,000 부근에서 끊겼는지	트레이스 스텝 수집 설정
SQL 건수가 `trace_sql_limit`와 일치하는지	수집 건수 상한
HTTP 호출 건수가 `trace_httpc_limit`와 일치하는지	수집 건수 상한
트레이스 마지막에 `lost-profile` 스텝이 있는지	프로파일링 종료 대기 시간

트랜잭션에 lost-profile이 기록됩니다

`profile_expired_time` (기본 30초) 동안 스텝이 전달되지 않아 프로파일링을 강제 종료한 상태입니다.

- 잡이 `kill -9` 등으로 비정상 종료된 경우에는 정상적인 자동 정리 결과입니다.
- 잡이 정상 실행 중이었다면, 외부 시스템 응답 대기처럼 계측 대상 처리가 발생하지 않는 구간이 30초를 넘었을 가능성이 있습니다. 프로파일링 종료 대기 시간을 참조해 값을 조정하세요.

자동으로 계산되는 설정

다음 값은 `time_limit` 에서 자동으로 계산합니다. 설정 파일에 직접 기재할 필요가 없습니다.

Option	Calculation
`warning_time`	`time_limit` 의 70%
`yellow_time`	`time_limit` 의 70%
`red_time`	`time_limit` 의 100%

적용 확인 체크리스트

#	항목	확인 방법
1	에이전트 압축 해제	`{BATCH_HOME}` 하위와 agent 디렉터리에 jar이 있습니다.
2	호스트 설정	`license` 와 `whatap.server.host` 를 기재했습니다.
3	통신 요건	TCP 6600 아웃바운드를 허용했습니다.
4	배치 호스트 기동	배치 호스트 배너의 `PCODE` 가 0이 아닙니다.
5	에이전트 설정	잡마다 `job_name` 과 `time_limit` 을 설정했습니다.

#	항목	확인 방법
6	잡 실행	`-Dwhatap.home`으로 잡 디렉터리를 지정했습니다.
7	설정 적용	잡 디렉터리에 logs를 만들었습니다.
8	데이터 확인	히트맵에 `job_name`의 트랜잭션이 표시됩니다.
9	소요 시간 확인	일련번호가 붙지 않은 트랜잭션에서 잡 전체 시간을 확인했습니다.
10	병렬 실행	여러 잡을 동시 실행해 `job_name`별로 구분되어 표시됩니다.
11	종료 처리	모든 잡 종료 후 배치 호스트를 정지했습니다.

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

이전 Java 버전 JVM 옵션

Java 1.6 버전 미만에서 와탭 에이전트를 적용할 수 있습니다. 하지만 일부 기능에 제약이 있습니다.

Java 1.5 버전

Java 1.5 버전은 `-javaagent` 옵션을 제공합니다. 다만 외부 스레드의 정보를 조회하는데 제약이 있어 일부 기능에 제약이 있습니다.

다음의 기능들은 동작하지 않습니다.

- 액티브 트랜잭션에서 실시간 트레이스
- 스택 분석 (탑 스택, 유니크 스택, 액티브 스택)
- 스레드 목록의 상세 스택

Java 1.4 버전

Java 1.4 버전은 `-javaagent` 옵션이 없습니다. 다른 방식으로 와탭 에이전트를 설치해야 합니다. 물론 whatap.agent.jar 파일은 Java 1.4 버전을 사용해야 합니다.

whatap.java14.tracer-`X.Y.Z`.jar

boot.jar 생성

setup.sh 파일을 이용해 boot.jar 파일을 생성하세요. 만약 [JAVA_HOME]을 입력하지 않으면 현재 경로를 JDK의 위치를 자동으로 인식합니다.

```
$ setup.sh [JAVA_HOME]
```

setup.sh 실행 결과: whatap.java14.boot-`X.Y.Z`.jar

bootclasspath에 설치

`bootclasspath` 옵션을 사용해 빌드된 jar와 tracer jar를 prepend로 추가하세요.

```
-Xbootclasspath/p:${WHATAP_HOME}/whatap.java14.boot-1.0.2.jar:${WHATAP_HOME}/whatap.java14.tracer-1.0.2.jar
```

제약 사항

jdk 1.4 버전에서는 다음의 기능들이 동작하지 않습니다.

- 액티브 트랜잭션에서 실시간 트래이스
- 스택 분석 (탑 스택, 유니크 스택, 액티브 스택)
- 컴포넌트 버전
- 스레드 목록, 힙히스토그램
- 로드된 클래스
- 오픈 소켓
- 에이전트 덤프

다음 단계

오픈 소스 추적

애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리에 따라 에이전트 옵션을 추가해야 할 수 있습니다. 예를 들어, 프레임워크나 오픈소스로 spring-boot-3.x, feign-client-11, okhttp3-4.4 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

사용자의 애플리케이션 환경에 맞는 프레임워크나 오픈소스 라이브러리를 설정하는 방법은 다음 문서를 참조하세요.

설치 점검하기

프로젝트 생성 및 에이전트 설치, JVM 옵션까지 모두 적용했다면 다음 문서에서 점검 사항을 확인하세요.

설치 문제 해결

에이전트 설치 시 발생할 수 있는 다양한 문제들과 이를 해결하기 위한 구체적인 지침을 제공합니다. 방화벽 설정, SpringBoot 및 Tomcat JMX 설정, OSGI 프레임워크 사용 서버 설정 등 다양한 상황에 대한 해결책을 안내합니다. 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정

에이전트 설정(whatap.conf) 파일에 옵션을 적용해 모니터링을 위한 다양한 기능을 제공합니다. 기본 설정부터 서버 연결 및 데이터 전송 설정, 여러 애플리케이션 서버의 설정 파일 관리 방법, 트랜잭션 추적 등을 포함합니다. 자세한 내용은 다음 문서를 참조하세요.

모니터링 시작하기

모든 설정을 완료한 다음 애플리케이션 서버를 다시 시작하세요. 에이전트가 정보를 수집하기 시작합니다. 먼저 **애플리케이션 대시보드** 메뉴에서 모니터링 데이터가 수집되는지 확인하세요. **애플리케이션 대시보드**에 대한 자세한 내용은 다음 문서를 참조하세요.

Java Helper

Java Helper는 애플리케이션의 라이브러리를 자동으로 분석하여 최적의 `weaving` 설정을 생성하는 CLI 도구입니다. Java 애플리케이션에서 사용하는 프레임워크나 오픈소스 라이브러리를 와탭 에이전트가 추적하려면 `weaving` 설정이 필요합니다. Java Helper는 에이전트 패키지에 `whatap.javahelper.jar`로 포함되어 있으며, **JDK 8 이상**에서 동작합니다.

`weaving` 옵션을 직접 지정하려면 수동 설정 문서를 참조하세요.

지원 포맷

JAR, Spring Boot Fat JAR, Maven Shade JAR, WAR 포맷을 지원합니다. EAR은 루트 레벨 JAR 모듈까지 분석합니다.

Java Helper 명령어

```
java -jar whatap.javahelper.jar conf generate <입력> [옵션]
```

입력

Input	Description	Example
<JAR 경로>	JAR 파일을 직접 지정하여 분석	/app/myapp.jar
--pid <PID>	실행 중인 Java 프로세스의 PID로 분석	--pid 12345

JAR 경로와 `--pid` 중 하나는 반드시 지정해야 합니다.

Options

Option	Description	Default
-o <파일>	매칭 결과를 whatap.conf 파일로 저장	미지정 시 화면 출력만

Option	Description	Default
-v	분석 과정의 상세 로그 출력 (verbose)	off

사용 시나리오

신규 에이전트 설치

JAR 파일 경로를 지정하여 whatap.conf를 바로 만듭니다.

```
java -jar whatap.javahelper.jar conf generate /app/myapp.jar -o whatap.conf
```

생성된 whatap.conf 예시:

whatap.conf

```
weaving=spring-boot-3.2,tomcat10,mongodb-4.8

# 추천 모듈 (필요시 주석 해제)
#weaving+=,reactor-kafka-1.3
#weaving+=,resilience4j
```

만든 파일에 라이선스 키와 서버 정보를 추가하면 설치가 완료됩니다.

운영 중 설정 점검

실행 중인 프로세스를 PID로 분석하여 매칭 결과를 확인합니다.

```
java -jar whatap.javahelper.jar conf generate --pid 12345
```

출력 예시:

```
========================================
 WhaTap Weaving Module 매칭 결과
========================================
```

```
탐지된 라이브러리: 25개
매칭된 모듈:     8개
버전 불일치:     1개
자동 포함:       3개

LIBRARY              VERSION   WEAVING MODULE      STATUS
spring-boot          3.2.5     spring-boot-3.2     MATCHED
mongodb-driver-sync  4.11.1    mongodb-4.8         MATCHED
lettuce-core         6.3.2     lettuce-6.1         AUTO
...
```

현재 whatap.conf와 비교하여 빠진 모듈이 있는지 확인할 수 있습니다. 설정 파일로 저장하려면 `-o` 옵션을 추가하세요.

라이브러리 업그레이드 후

Spring Boot 버전 업그레이드 등 라이브러리 변경 시, 업그레이드한 JAR로 다시 분석하면 새 `weaving` 설정을 받을 수 있습니다.

```
java -jar whatap.javahelper.jar conf generate /app/myapp-upgraded.jar
```

매칭 상태

Status	Description	Action
`MATCHED`	artifactId, groupId, version 모두 일치하여 weaving 모듈 적용 가능	자동 적용
`AUTO`	자동 포함 대상 모듈 (weaving 설정 없이 기본 적용)	자동 적용
`VERSION_MISMATCH`	라이브러리는 탐지되었으나 버전이 weaving 모듈 지원 범위 밖	에이전트 업데이트 확인
`NO_MATCH`	weaving 레지스트리에 해당하는 모듈이 없음	해당 없음

참고

의존성 분석 방식

JAR 파일에서 라이브러리 버전을 확인하는 과정은 3단계 우선순위로 동작합니다. 상위 단계에서 추출한 정보는 하위 단계에서 덮어쓰지 않습니다.

1. **Maven pom.properties**: JAR 내부의 `META-INF/maven/{groupId}/{artifactId}/pom.properties`에서 `groupId`, `artifactId`, `version`을 추출합니다. 가장 정확한 방법입니다.
2. **MANIFEST.MF**: pom.properties가 없으면 `META-INF/MANIFEST.MF`에서 `Implementation-Title`, `Implementation-Version` 등 표준 속성을 읽습니다.
3. **파일명 파싱**: 위 두 방법이 실패하면 JAR 파일명에서 추론합니다. (예: `jackson-databind-2.14.0.jar` → `version="2.14.0"`)

Weaving 플러그인 설정

whatap.conf 파일에 `weaving` 옵션을 직접 지정하여 프레임워크와 오픈소스 라이브러리를 추적합니다. 라이브러리를 자동으로 분석하여 설정을 만들려면 자동 설정 (Java Helper) 문서를 참조하세요.

예를 들어, `spring-boot-3.x` , `feign-client-11` , `okhttp3-4.4` 사용 시 다음과 같이 옵션을 설정하세요.

whatap.conf

```
weaving=spring-boot-3.0,feign-11,okhttp3-4.4
```

추적 가능한 오픈소스 목록

Apache Camel

CXF

camel-cxf Version	설정 값	에이전트 최소 버전
3.15 이상	weaving=camel-cxf-3.15	v2.2.27

Netty

camel-netty4 Version	설정 값	에이전트 최소 버전
2.25 이상	weaving=camel-netty4-2.25	v2.2.42

SEDA

camel-seda Version	설정 값	에이전트 최소 버전
2.22 이상	weaving=camel-seda-2.22	v2.2.20

camel-seda Version	설정 값	에이전트 최소 버전
3.2 이상	weaving=camel-seda-3.2	v2.2.20

DynamoDB

DynamoDB Version	설정 값	에이전트 최소 버전
1.11	weaving=dynamodb-1.11	v2.2.39
2.25	weaving=dynamodb-2.25	v2.2.39

Feign Client

Feign Version	설정 값	에이전트 최소 버전
11 이상	weaving=feign-11	v2.2.6

Hystrix

Hystrix Version	설정 값	에이전트 최소 버전
1.5 이상	weaving=hystrix-1.5	v2.0_21

Kafka

Kafka Version	설정 값	에이전트 최소 버전
kafka-clients 2.4.0 이상	weaving=kafka-clients-2.4.0	v2.2.15
reactor-kafka 1.3 이상	weaving=reactor-kafka-1.3	v2.2.5

Logging

Log4j2

Log4j2 Version	설정 값	에이전트 최소 버전
2.17 이상	weaving=log4j-2.17	v2.2.28

ⓘ 특정 appender 제외하는 에이전트 옵션 제공

Logback

Logback Version	설정 값	에이전트 최소 버전
1.2.8 이상	weaving=logback-1.2.8	v2.2.28

ⓘ 특정 appender 제외하는 에이전트 옵션 제공

MongoDB

MongoDB Version	설정 값	에이전트 최소 버전
3.8.2 이상	weaving=mongodb-3.8	v2.2.11
4.0.3 이상	weaving=mongodb-4.0	v2.2.11
4.4 이상	weaving=mongodb-4.4	v2.2.11
4.8 이상	weaving=mongodb-4.8	v2.2.11
4.11 이상	weaving=mongodb-4.11	v2.2.45

MongoDB Version	설정 값	에이전트 최소 버전
5.0 이상	weaving=mongodb-5.0	v2.2.45

Mule Framework

Mule Version	설정 값	에이전트 최소 버전
3.9.5 이상	weaving=mule-3.9.5	v2.2.23
4.5 이상	weaving=mule-4.5	v2.2.23

OkHttp

OkHttp Version	설정 값	에이전트 최소 버전
2.7 이상	weaving=okhttp-2.7	v2.0_15
3.x	weaving=okhttp3	v2.0_15
3.x (4.4 이상)	weaving=okhttp3-4.4	v2.2.9

Quarkus Reactive

Quarkus Reactive Version	설정 값	에이전트 최소 버전
1.13 이상	weaving=quarkus-reactive-1.13	v2.2.19
2.10 이상	weaving=quarkus-reactive-2.10	v2.2.19

RabbitMQ

reactor-rabbitmq Version	설정 값	에이전트 최소 버전
1.2 이상	weaving=reactor-rabbitmq-1.2	v2.0_06

Redis

Jedis

Jedis Version	설정 값	에이전트 최소 버전
2.9.0 이상	weaving=jedis-2.9	v2.2.43
2.9.3 이상	weaving=jedis-2.9	v2.0_33
3.2 이상	weaving=jedis-3.2	v2.0_09

Lettuce

Lettuce Version	설정 값	에이전트 최소 버전
5.1 이상	weaving=lettuce-5.1	v2.2.7
6.2 이상	weaving=lettuce-6.2	v2.2.16

Retrofit

Retrofit Version	설정 값	에이전트 최소 버전
2.5 이상	weaving=retrofit-2.5	v2.2.39

okhttp 설정과 동시 사용 불가

Ribbon

Ribbon Version	설정 값	에이전트 최소 버전
전체 버전	weaving=ribbon	v2.2.10

Spring Boot

Spring Boot Version	설정 값	에이전트 최소 버전
2.1 이상	weaving=spring-boot-2.1	v2.2.23
2.5 이상	weaving=spring-boot-2.5	v2.2.9
2.7 이상	weaving=spring-boot-2.7	v2.2.9
3.0 이상	weaving=spring-boot-3.0	v2.2.9
3.2 이상	weaving=spring-boot-3.2	v2.2.38
4.0 이상	weaving=spring-boot-4.0	v2.2.69

포함 기술 스택 보기

- **spring-boot-2.1**: r2dbc-mysql-0.8.2, spring-cloud-gateway-2.1, spring-kafka-2.7, spring-webflux-5.1, tomcat9
- **spring-boot-2.5**: r2dbc-mysql-0.8.2, spring-cloud-gateway-3.0, spring-kafka-2.7, spring-webflux-5.3, mongodb-4.0.3, reactor-kafka-1.3, rxjava2, tomcat9
- **spring-boot-2.7**: jasync-r2dbc-mysql-2.1.23, r2dbc-mysql-0.9.3, spring-cloud-gateway-3.1, spring-kafka-3.0, spring-webflux-5.3, mongodb-4.4, reactor-kafka-1.3, rxjava2, tomcat9
- **spring-boot-3.0**: jasync-r2dbc-mysql-2.1.23, r2dbc-mysql-1.0.2, spring-cloud-gateway-4.0, spring-

kafka-3.0, spring-webflux-6.0, mongodb-4.8, redis(lettuce-6.2), reactor-kafka-1.3, rxjava3, tomcat10

- **spring-boot-3.2**: jasync-r2dbc-mysql-2.1.23, r2dbc-mysql-1.1.3, spring-cloud-gateway-4.1, spring-kafka-3.1, spring-webflux-6.1, redis(lettuce-6.2), reactor-kafka-1.3, rxjava3, tomcat10
- **spring-boot-4.0**: RestTemplate, RestClient, WebClient, R2DBC

Tomcat

Tomcat Version	설정 값	에이전트 최소 버전
9.x	weaving=tomcat9	v2.2.5
10.x	weaving=tomcat10	v2.2.5

Undertow

Undertow Version	설정 값	에이전트 최소 버전
2.2 이상	weaving=undertow-2.2	v2.2.39
2.3 이상	weaving=undertow-2.3	v2.2.14

Vert.x

Vert.x Version	설정 값	에이전트 최소 버전
3.5.3	weaving=vertx-3.5	v2.2.42
3.9	weaving=vertx-3.9	v2.2.44
4.5	weaving=vertx-4.5	v2.2.39

Java 에이전트의 CVE 오탐 방지

Java 에이전트에 포함된 라이브러리 클래스는 애플리케이션에서 실제 사용할 때만 로딩됩니다. CVE 취약점 검사 시 라이브러리가 검출되더라도, 사용하지 않는다면 실제 취약점은 없습니다.

불필요한 추적 클래스를 삭제하려면 다음 명령어를 실행하세요.

```
java -cp whatap.agent-X.Y.Z.jar whatap.agent.setup.RemoveWeaving -remove [weaving jar filename]
```

Java 에이전트 업데이트 시마다 삭제 작업을 다시 수행해야 합니다.

특정 라이브러리 클래스를 제거하는 명령어는 **Java 에이전트 2.2.37 버전** 이상에서 지원합니다.

사용법:

```
java -cp whatap.agent-X.Y.Z.jar whatap.agent.setup.RemoveWeaving [arguments] [weaving jar filename]
```

Example

```
java -cp whatap.agent-2.2.37.jar whatap.agent.setup.RemoveWeaving -remove spring-boot-2.5.jar spring-boot-2.7.jar
```

Arguments:

- `-r` 또는 `-remove` : Java 에이전트에서 weaving jar 파일을 삭제하고 새로운 Java 에이전트를 생성합니다.
- `-l` 또는 `-list` : weaving jar 파일 목록을 출력합니다.
- `-d` 또는 `-debug` : 디버그 로깅(debug logging)을 활성화합니다.
- `-e` 또는 `-error` : 에러에 대한 풀 스택 트레이스 정보를 표시합니다.
- `-h` 또는 `-help` : 도움말을 출력하고 종료합니다.

Java 설치 점검

에이전트를 설치한 다음 확인해야할 사항입니다. 만약 설치에 문제가 생겼다면 다음 문서를 참조하세요.

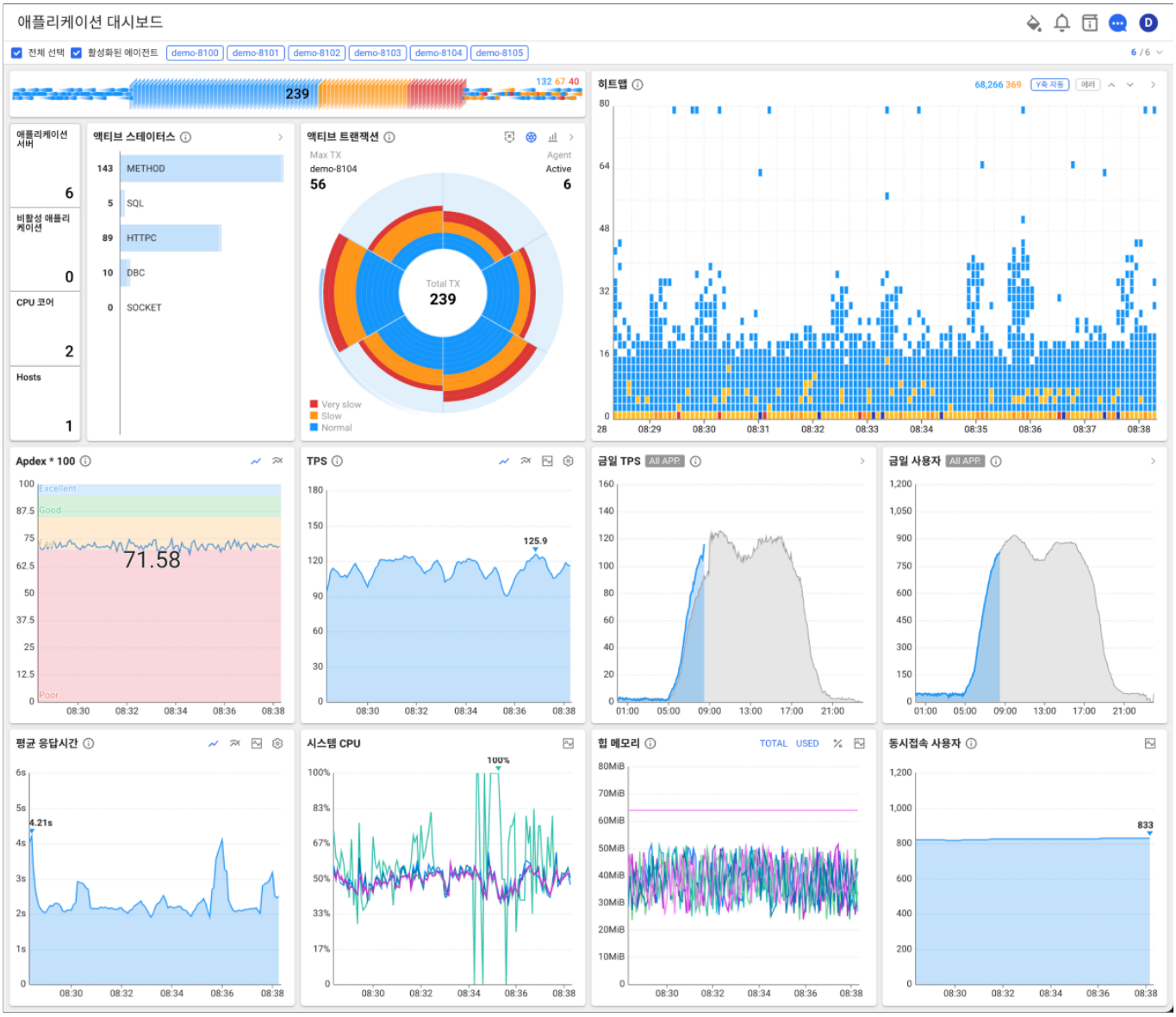
WAS 다시 시작

애플리케이션 서버를 다시 시작하세요. 구동 로그에서 다음과 같은 와탭 캐릭터 로그가 보인다면 `JAVA_OPTS` 설정을 완료한 것입니다.

```
Nov 16, 2016 3:06:40 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deployment of web application directory /var/lib/tomcat7/webapps/ROOT has finished in 577 ms
Nov 16, 2016 3:06:40 AM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["http-bio-8080"]
Nov 16, 2016 3:06:40 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 3984 ms
_        ____      _____
| | /| / / /  ___ /_  __/__ ____
| |/ |/ / / _ \/ _ `// / / _ `/ _ \
|__/|__/_//_/\_,_//_/  \_,_/ .__/
                          /_/
Just Tap, Always Monitoring
WhaTap Agent version 0.3.9 20161115
```

에이전트 연결 확인

와탭 모니터링 서비스로 이동하세요. 로그인 후 프로젝트 목록에서 생성한 프로젝트를 선택하세요. **애플리케이션 대시보드** 메뉴에서 연결한 서버의 데이터가 위젯을 통해 표시된다면 모든 설정을 완료한 것입니다.



ⓘ 에이전트 이름의 기본값은 `{type}-{ip2}-{ip3}-{port}` 입니다. 에이전트의 이름을 변경하거나 속성을 추가하려면 다음 문서를 참조하세요.

Java 에이전트 문제 해결

방화벽 설정 확인

와탭의 데이터 수집 서버에 대한 TCP 아웃바운드 정책을 설정하지 않으면 모니터링 정보를 전송할 수 없습니다. **방화벽 차단**을 **해제**하세요.

`telnet` 명령을 사용해 "Connected to `IP`"를 확인하세요.

```
$ telnet 52.193.60.176 6600
Trying 52.193.60.176...
Connected to 52.193.60.176.
Escape character is '^]'.
```

> ⓘ 수집 서버 정보는 와탭 모니터링 서비스에서 해당 프로젝트를 선택한 다음 **관리** > **에이전트 설치** 메뉴에서 확인할 수 있습니다.

SpringBoot 2.2 버전 이상, Tomcat JMX가 off 상태

SpringBoot 2.2 버전 이상에서 Embedded Tomcat을 사용하는 경우, 기본값으로 JMX 관련 기능이 동작하지 않습니다. JMX 기능을 활성화하세요. 관련 문서를 참고하세요.

application.properties

```
spring.jmx.enabled=true
server.tomcat.mbeanregistry.enabled=true # tomcat embedded인 경우 #
```

application.yml

```
spring:
  jmx:
    enabled: true
server:
```

```
tomcat:
  mbeanregistry:
    enabled: true #tomcat embedded인 경우
```

OSGI 프레임워크 사용 애플리케이션 서버의 경우

OSGI 프레임워크 구조의 애플리케이션 서버인 경우 JVM 옵션에 에이전트 패키지 prefix(whatap)를 등록하세요.

Jboss EAP 6.0 버전 이상, Jboss AS 7.0 버전 이상, Wildfly 8.0 버전 이상, IBM WebSphere AS 7.0 버전 이상 등이 해당합니다.

JBoss AS, Wildfly, JBoss EAP 6.0 이상

다음 파일에 prefix를 등록하세요.

$JBOSS_HOME/bin/standalone.conf(domain.conf)

```
-Djboss.modules.system.pkgs=whatap
```

JBOSS EAP 7.0

WebSphere

JVM 옵션으로 다음 내용을 추가하세요.

```
-Dcom.ibm.ws.classloader.server.alwaysAllowedPackages=whatap
```

- 기본값으로 '*'로 지정된 경우 별도 설정이 필요 없습니다.
- 설정 위치는 WebSphere를 참조합니다.

security.policy 권한을 다음과 같이 추가하세요.

$WEBSPHERE_HOME/properties/server.policy || $WEBSPHERE_PROFILE_HOME/properties/server.policy

```
grant codeBase "file:$WHATAP_HOME/-"
{
  permission java.security.AllPermission;
```

```
};
```

Log Manager 관련 에러

JBoss AS 7.0 버전 이상, JBoss EAP 6.0 버전 이상에서 Log Manager 관련 에러가 발생한다면 JVM 옵션을 추가합니다.

- `-Djava.util.logging.manager` 항목에 Log Manager package 명을 설정하세요.
- `-Xbootclassloader` 항목에 JBoss Log Manager JAR file을 설정하세요.

(JBoss_Path)/bin/standalone.conf(domain.conf)

```
# Specify the exact Java VM executable to use.
#
if [ "x$JBOSS_MODULES_SYSTEM_PKGS" = "x" ]; then
   JBOSS_MODULES_SYSTEM_PKGS="org.jboss.byteman,org.jboss.logmanager,whatap"
fi

...
# Specify options to pass to the Java VM.
#
if [ "x$JAVA_OPTS" = "x" ]; then
   JAVA_OPTS="-Xms1303m -Xmx1303m -Djava.net.preferIPv4Stack=true"
   JAVA_OPTS="$JAVA_OPTS -Djboss.modules.system.pkgs=$JBOSS_MODULES_SYSTEM_PKGS -Djava.awt.headless=true"
   JAVA_OPTS="$JAVA_OPTS -Djava.util.logging.manager=org.jboss.logmanager.LogManager -Xbootclasspath/p:/home/vagrant/EAP-7.0.0/modules/system/layers/base/org/jboss/logmanager/main/jboss-logmanager-2.0.3.Final-redhat-1.jar"
else
   echo "JAVA_OPTS already set in environment; overriding default settings with values: $JAVA_OPTS"
fi
```

MBeanServerBuilder 에러가 발생하는 경우

JBoss 5.0 버전 이하에서 MBeanServerBuilder 관련 에러가 출력된 경우 JVM 옵션을 추가하세요.

(JBoss_Path)/bin/run.conf

```
# Specify options to pass to the Java VM.
#
if [ "x$JAVA_OPTS" = "x" ]; then
  JAVA_OPTS="-Xms128m -Xmx512m -XX:MaxPermSize=256m -Dorg.jboss.resolver.warning=true
-Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000"
  JAVA_OPTS="$JAVA_OPTS -Djboss.platform.mbeanserver"
fi
```

Permission 오류가 발생하는 경우

Java Security Policy 관련 오류가 발생하면 $JAVA_HOME/jre/lib/security/java.policy 파일에 권한 설정을 추가하세요.

모든 권한을 일괄 적용하려면 다음과 같이 java.policy 파일에 설정을 추가하세요.

$JAVA_HOME/jre/lib/security/java.policy

```
grant {
  permission java.security.AllPermission;
};
```

java.io.FilePermission 오류가 발생하는 경우

```
org.hyperic.sigar.SigarException: Failed to load sigar-ppc64-aix-7: Access denied (java.io.FilePermission
        at org.hyperic.sigar.Sigar.loadLibrary(Sigar.java:170)
        at org.hyperic.sigar.Sigar.<clinit>(Sigar.java:100)
        at java.lang.J9VMInternals.initializeImpl(Native Method)
        at java.lang.J9VMInternals.initialize(J9VMInternals.java:200)
        at whatap.xtra.sigar.SigarMain.<clinit>(SigarMain.java:31)
        at java.lang.J9VMInternals.initializeImpl(Native Method)
        at java.lang.J9VMInternals.initialize(J9VMInternals.java:200)
        at java.lang.Class.forNameImpl(Native Method)
        at java.lang.Class.forName(Class.java:217)
        at whatap.agent.proxy.SigarFactory.create(SigarFactory.java:62)
        at whatap.agent.counter.task.SystemPerf.<init>(SystemPerf.java:35)
        at whatap.agent.counter.CounterExecutingManager.run(CounterExecutingManager.java:78)
```

다음과 같이 java.policy 파일에 설정을 추가하세요.

$JAVA_HOME/jre/lib/security/java.policy

```
grant {
  ...
  permission java.io.FilePermission {오류 메시지에서 확인된 패키지 경로}, "read"
```

```
};
```

java.util.PropertyPermission 오류가 발생하는 경우

다음과 같이 java.policy 파일에 설정을 추가하세요.

$JAVA_HOME/jre/lib/security/java.policy

```
grant {
  ...
  permission java.util.PropertyPermission {오류 메시지에서 확인된 패키지 경로}, "read"
};
```

Sigar library를 로딩하지 못하는 경우

$WHATAP_HOME/lib1/*.so 파일에 실행 권한이 있는지 확인하세요. 권한이 없다면 실행 권한을 부여하세요.

```
$ sudo chmod +x *.so
```

AIX 7에서 $WHATAP_HOME/lib1 하위에 libsigar-ppc64-aix-7.so 파일이 없어 오류가 발생한 경우 파일을 복사하세요.

```
org.hyperic.sigar.SigarException: Failed to load sigar-ppc64-aix-7: Access denied (java.io.FilePermission
        at org.hyperic.sigar.Sigar.loadLibrary(Sigar.java:170)
        at org.hyperic.sigar.Sigar.<clinit>(Sigar.java:100)
        at java.lang.J9VMInternals.initializeImpl(Native Method)
        at java.lang.J9VMInternals.initialize(J9VMInternals.java:200)
        at whatap.xtra.sigar.SigarMain.<clinit>(SigarMain.java:31)
        at java.lang.J9VMInternals.initializeImpl(Native Method)
        at java.lang.J9VMInternals.initialize(J9VMInternals.java:200)
        at java.lang.Class.forNameImpl(Native Method)
        at java.lang.Class.forName(Class.java:217)
        at whatap.agent.proxy.SigarFactory.create(SigarFactory.java:62)
        at whatap.agent.counter.task.SystemPerf.<init>(SystemPerf.java:35)
        at whatap.agent.counter.CounterExecutingManager.run(CounterExecutingManager.java:78)
```

```
$ cp libsigar-ppc64-aix-5.so libsigar-ppc64-aix-7.so
```

CPU 지표가 -1로 표시되는 경우

최신 버전의 Java 에이전트를 사용하는 환경에서 IBM Java를 실행할 때, CPU 지표가 **-1**로 표시되는 경우가 있습니다. 이 문제는

특정 플랫폼에서 **System Load Average** 값을 지원하지 않기 때문에 발생합니다.

Java 에이전트는 기본적으로 Java Management Extensions(JMX)를 활용하여 지표를 수집합니다. `OperatingSystemMXBean` 인터페이스의 `getSystemLoadAverage()` 메소드를 사용합니다. 이 메소드는 운영 체제에 따라 시스템의 최근 1분간의 평균 부하를 반환합니다. 그러나 일부 플랫폼에서는 해당 값을 지원하지 않아 **-1**을 반환할 수 있습니다.

> ⓘ `getSystemLoadAverage()` 메서드는 OpenJDK 소스 코드에서 확인할 수 있습니다. 예를 들어, OpenJDK 7u 버전의 소스 코드는 GitHub 저장소에서 확인할 수 있습니다.

권고 사항: 내장 라이브러리를 통한 메트릭 수집 설정

CPU 지표가 **-1**로 표시되는 현상을 해결하려면, Java 에이전트 설정을 통해 내장 라이브러리(OSHI 또는 SIGAR)를 활성화하세요.

- **Java 8 이상 환경**: **OSHI** 라이브러리 활용

  운영 체제 수준의 지표를 안정적으로 수집하기 위해 **OSHI** 라이브러리를 사용할 수 있습니다. 에이전트 설정 방법은 다음을 참조하세요.

  whatap.conf

  ```
  # default false
  oshi_enabled=true
  ```

- **Java 8 미만 환경**: **SIGAR** 라이브러리 활용

  Java 8 미만 환경에서는 **SIGAR** 라이브러리를 활용해 지표를 수집할 수 있습니다. 에이전트 설정 방법은 다음을 참조하세요.

  whatap.conf

  ```
  # default false
  sigar_enabled=true
  ```

> ⓘ **OSHI**(`oshi_enabled`)와 **SIGAR**(`sigar_enabled`) 옵션은 기본값으로 비활성화되어 있습니다. 필요한 경우 명시적으로 설정해야 합니다.
>
> - 두 설정을 동시에 활성화하지 않도록 주의하세요. 환경에 맞는 라이브러리를 선택하여 설정하세요.

Java 에이전트 설정

와탭 에이전트는 에이전트 별 필요한 설정을 whatap.conf 파일에 작성합니다. 에이전트는 환경변수를 통해 설정 파일의 위치를 파악하고 로딩합니다.

별도 설정이 없는 경우 에이전트는 자기 자신(whatap.agent-X.Y.Z.jar)이 위치한 디렉터리에서 whatap.conf 파일을 찾도록 구현했습니다. 5초에 한번씩 whatap.conf 파일의 변경 여부를 체크하고 리로딩합니다.

사용자의 편의성을 위해 와탭 모니터링 서비스에서도 에이전트 설정 기능을 제공하고 있습니다.

기본 설정

whatap.conf

```
whatap.home=.
whatap.config=whatap.conf
```

- whatap.home : 기본값은 whatap.agent.jar 파일의 경로를 입력하세요.
- whatap.config : 파일명만 입력할 것을 권고합니다. 상대 패스를 같이 입력할 수는 있습니다.

에이전트가 데이터를 수집하고, 서버로 데이터를 전송하기 위해서는 최소한 액세스 키와 서버 주소는 설정돼 있어야 합니다.

whatap.conf

```
license=x220g160hgd29-x3qpff0garcus7-z4p0kao58un11a
whatap.server.host=10.10.0.1
```

와탭 에이전트는 최초 실행시 다음과 같이 어떤 jar 파일이 에이전트로 사용됐는지, 어떤 경로에서 whatap.conf 파일을 로딩하는지 확인할 수 있습니다.



서버 연결 및 데이터전송

와텝 에이전트는 whatap.conf 파일에 설정한 서버 주소로 연결합니다.

whatap.conf

```
whatap.server.host=10.10.1.1
whatap.server.port=6600 # 기본값
```

- TCP 세션을 연결하면 `license`를 이용해 서버로부터 통신키를 전달 받습니다. 잘못된 `license`를 설정하면 서버는 세션을 종료합니다.
- TCP 연결이 반복적으로 종료되면 방화벽 문제 확인하거나 `license` 값이 올바른지 확인하세요.
- TCP 세션을 연결하면 서버로부터 받은 비밀키를 기반으로 보안 통신으로 데이터를 전송합니다.

여러 애플리케이션 서버의 설정 파일 관리

한 서버에 여러 애플리케이션을 운영할 경우, 각 애플리케이션 별 설정 파일을 별도로 관리하려면 whatap.conf 파일명을 변경하고 시작 스크립트에 JVM 옵션을 추가하세요.

Service A

```
-Dwhatap.config=whatap_Aservice.conf
```

Service B

```
-Dwhatap.config=whatap_Bservice.conf
```

> whatap.conf 설정 파일은 반드시 `${WHATAP_HOME}` 경로에 위치해야 합니다.

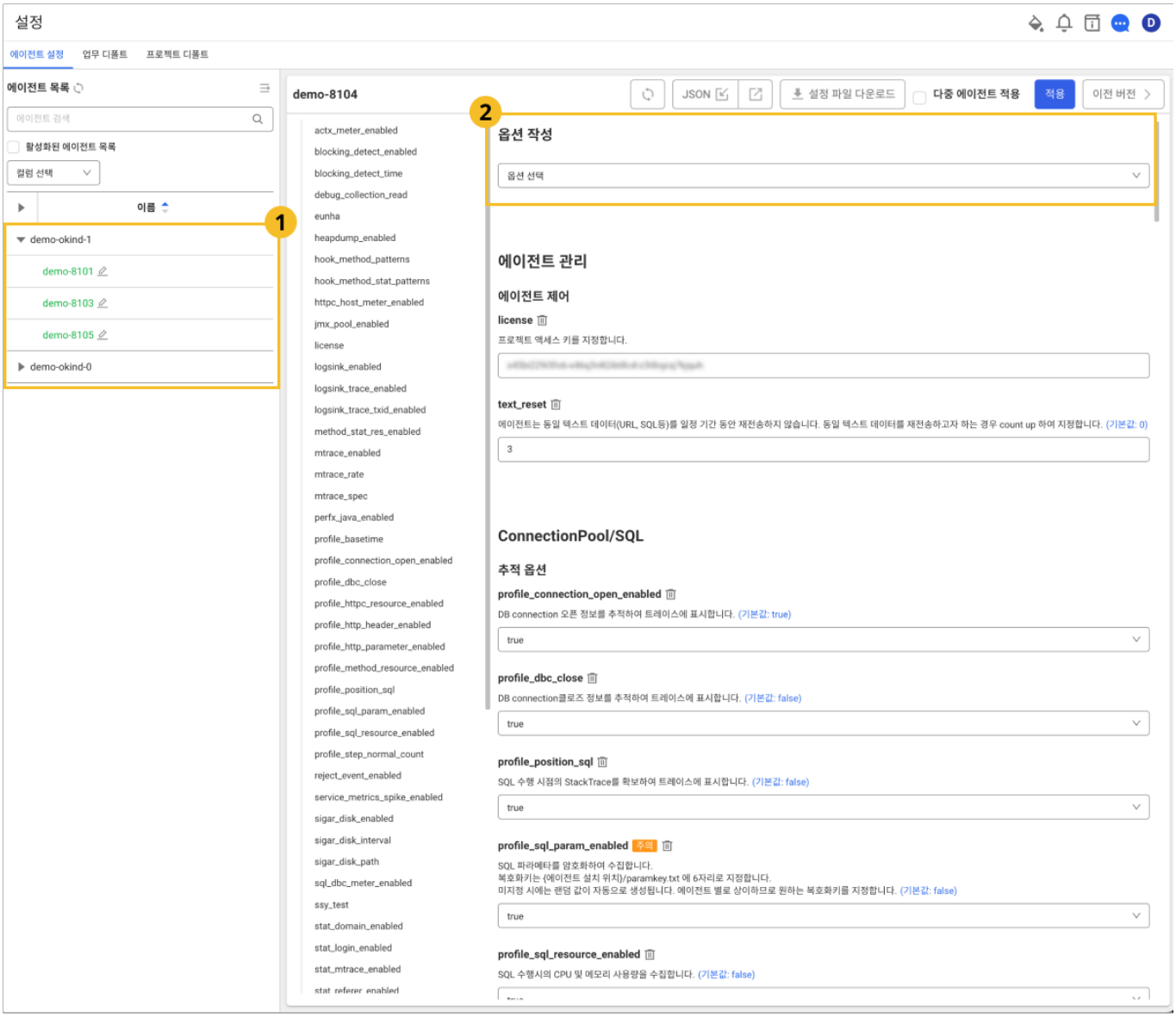
서비스 화면에서 에이전트 설정하기

홈 화면 > 프로젝트 선택 > **관리** > **에이전트 설정**

모니터링 대상 서버에 위치한 whatap.conf 파일을 직접 수정하지 않고 와탭 모니터링 서비스에서 에이전트 설정 옵션을 추가하거나 수정, 삭제할 수 있습니다. 또한 **설정 파일 다운로드** 버튼을 클릭해 whatap.conf 파일을 다운로드할 수 있습니다.

- 이 기능은 **수정** 권한을 가진 멤버만 이용할 수 있습니다. **수정** 권한이 없는 멤버는 설정 내용을 조회만 할 수 있습니다.
- 옵션 값으로 설정할 수 있는 형식은 다음과 같습니다.
  - Boolean 형식의 값은 `true` 또는 `false`를 선택하세요.
  - 숫자 형식의 값은 숫자만 입력할 수 있습니다.
  - 텍스트(String) 형식의 값을 입력 또는 수정할 경우 옵션 설명을 자세히 확인하세요.
- 수정할 수 없는 옵션은 선택할 수 없습니다. (예, `license`)
- 추가 또는 수정, 삭제한 옵션에 따라 에이전트를 재시작해야할 수 있습니다.
- 애플리케이션 종류 및 에이전트의 버전에 따라 적용할 수 있는 옵션 키는 다를 수 있습니다.

옵션 추가하기



1. **에이전트 목록**에서 옵션을 추가하려는 1 에이전트를 선택하세요.
2. 2 **옵션 작성**에서 추가할 옵션 항목을 선택하세요.
   - **검색**에서 추가할 옵션을 찾을 수 있습니다. 텍스트를 입력하면 일치하는 옵션을 필터링합니다.
   - **직접 입력**을 선택하면 옵션 키와 값을 입력할 수 있습니다.

3. 선택한 옵션 키에 대한 설명과 기본값을 확인한 다음 설정값을 입력하세요.



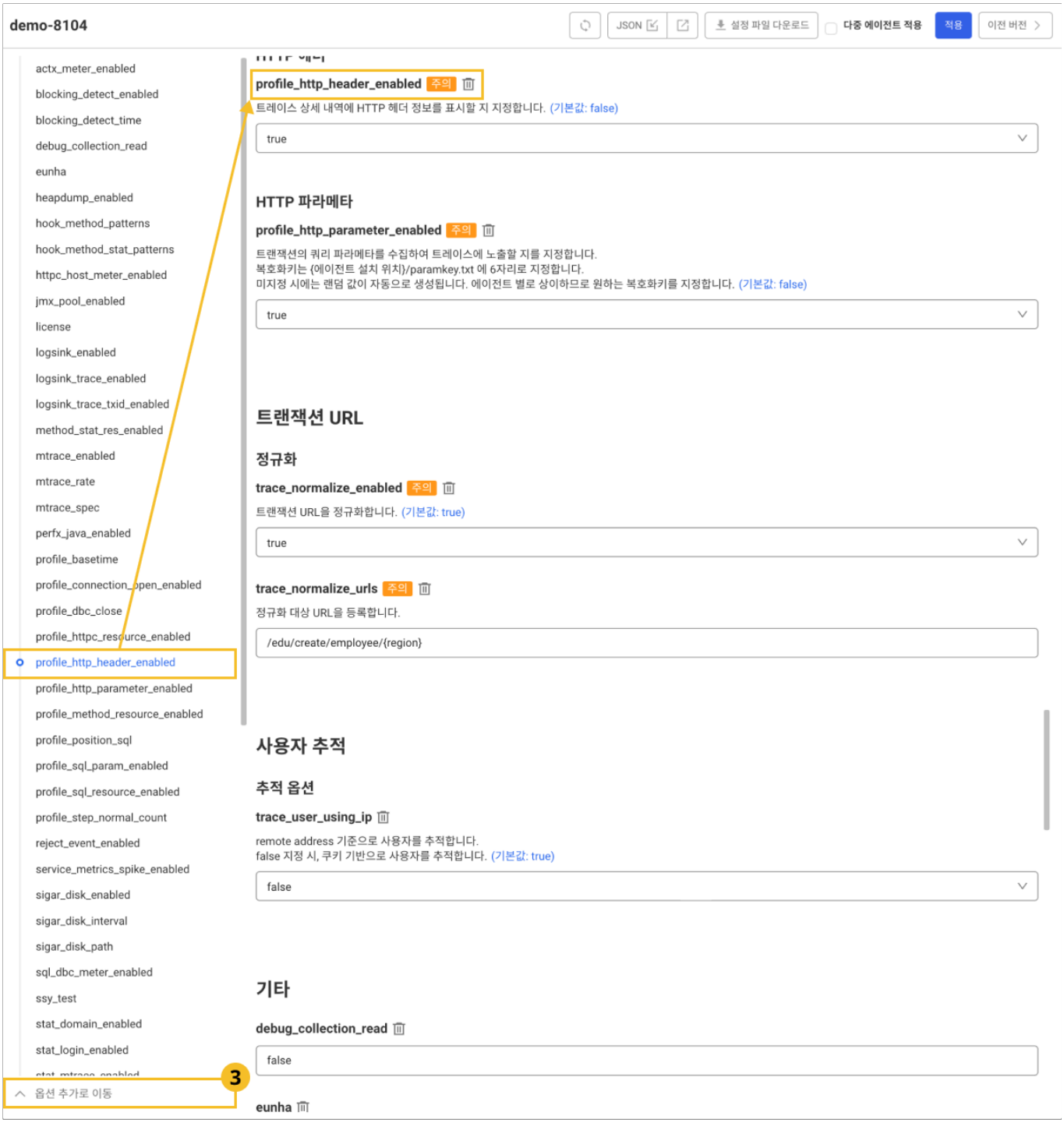
   - 선택한 옵션을 취소하려면 🗑 버튼을 선택하세요.
   - 옵션을 추가 설정하려면 **추가하기** 버튼을 선택하고 2번의 과정을 반복하세요.

4. 원하는 모든 옵션을 추가했으면 화면 오른쪽 위에 **적용** 버튼을 선택하세요.

선택한 옵션 및 설정값을 에이전트에 적용합니다.

ⓘ
- 수정 중인 내용을 초기화하려면 화면 오른쪽 위에 ↻ 버튼을 선택하세요.
- 옵션값으로 아무것도 입력하지 않은 상태에서 **적용** 버튼을 선택하면 해당 옵션을 삭제합니다.
- 이미 추가한 옵션은 옵션 목록에서 선택할 수 없습니다.
- 애플리케이션 종류 및 에이전트의 버전에 따라 적용할 수 있는 옵션 키는 다를 수 있습니다.
- 예시로 제공한 이미지는 애플리케이션 종류 및 에이전트에 따라 다를 수 있습니다.

옵션 수정 또는 삭제하기


demo-8104
JSON
설정 파일 다운로드
다중 에이전트 적용
적용
이전 버전
actx_meter_enabled
blocking_detect_enabled
blocking_detect_time
debug_collection_read
eunha
heapdump_enabled
hook_method_patterns
hook_method_stat_patterns
httpc_host_meter_enabled
jmx_pool_enabled
license
logsink_enabled
logsink_trace_enabled
logsink_trace_txid_enabled
method_stat_res_enabled
mtrace_enabled
mtrace_rate
mtrace_spec
perfx_java_enabled
profile_basetime
profile_connection_open_enabled
profile_dbc_close
profile_httpc_resource_enabled
profile_http_header_enabled
profile_http_parameter_enabled
profile_method_resource_enabled
profile_position_sql
profile_sql_param_enabled
profile_sql_resource_enabled
profile_step_normal_count
reject_event_enabled
service_metrics_spike_enabled
sigar_disk_enabled
sigar_disk_interval
sigar_disk_path
sql_dbc_meter_enabled
ssy_test
stat_domain_enabled
stat_login_enabled
옵션 추가로 이동
3
profile_http_header_enabled 주의
트레이스 상세 내역에 HTTP 헤더 정보를 표시할 지 지정합니다. (기본값: false)
true
HTTP 파라메타
profile_http_parameter_enabled 주의
트랜잭션의 쿼리 파라메타를 수집하여 트레이스에 노출할 지를 지정합니다.
복호화키는 {에이전트 설치 위치}/paramkey.txt 에 6자리로 지정합니다.
미지정 시에는 랜덤 값이 자동으로 생성됩니다. 에이전트 별로 상이하므로 원하는 복호화키를 지정합니다. (기본값: false)
true
트랜잭션 URL
정규화
trace_normalize_enabled 주의
트랜잭션 URL을 정규화합니다. (기본값: true)
true
trace_normalize_urls 주의
정규화 대상 URL을 등록합니다.
/edu/create/employee/{region}
사용자 추적
추적 옵션
trace_user_using_ip
remote address 기준으로 사용자를 추적합니다.
false 지정 시, 쿠키 기반으로 사용자를 추적합니다. (기본값: true)
false
기타
debug_collection_read
false
eunha

1. 화면을 위 또는 아래로 스크롤하거나 왼쪽의 옵션 목록에서 수정 또는 삭제하려는 옵션을 선택하세요.
2. 변경하려는 옵션에서 원하는 값을 선택하거나 수정하세요. 옵션을 삭제하려면 🗑 버튼을 선택하세요.
3. 변경한 사항을 적용하려면 **적용** 버튼을 선택하세요.

> ⓘ
> - 화면 가장 위로 이동하려면 3 **옵션 추가로 이동** 버튼을 선택하세요.
> - 옵션값으로 아무것도 입력하지 않은 상태에서 **적용** 버튼을 선택하면 해당 옵션을 삭제합니다.

여러 에이전트에 동시 적용하기

프로젝트에 소속된 여러 개의 에이전트에 변경한 옵션을 동시에 적용할 수 있습니다.

1. 화면 오른쪽 위에 **다중 에이전트 적용** 체크박스를 선택하면 각 옵션 항목에 체크박스가 생성됩니다.
2. 동시에 적용하길 원하는 옵션의 체크박스를 선택하세요. 여러 개를 선택할 수 있습니다.
3. 화면 오른쪽 위에 **적용** 버튼을 선택하세요.
4. **에이전트 적용** 창이 나타나면 변경한 옵션을 적용할 에이전트를 선택하세요. 모두 선택하려면 **전체 선택** 체크박스를 선택하세요.



5. **적용** 버튼을 선택하세요.

에이전트 기본값 설정하기

프로젝트에 새로운 에이전트를 추가할 경우 기존의 설정값을 반복해서 적용하는 번거로움을 피하고 싶다면 프로젝트별, 업무별 기본 설정값을 만들어 적용할 수 있습니다.



- **업무 디폴트**: whatap.conf 파일에서 `whatap.okind` 항목으로 분류한 에이전트들에 옵션을 공통 적용할 수 있고, 적용된 옵션을 확인할 수 있습니다.
- **프로젝트 디폴트**: 프로젝트에 소속된 모든 에이전트들에 옵션을 공통 적용할 수 있고, 적용된 옵션을 확인할 수 있습니다.

> ⓘ
> - **업무 디폴트** 또는 **프로젝트 디폴트** 탭을 선택한 다음 옵션을 적용하는 것은 에이전트를 공통으로 관리하는 데 유용합니다.
> - 옵션의 적용 우선 순위는 **업무 디폴트**, **프로젝트 디폴트**, **에이전트 설정** 순입니다.
>   - **프로젝트 디폴트**에서 b 옵션값을 1000을 적용하고, **업무 디폴트**에서 b 옵션값을 2000으로 적용하면, **업무 디폴트**에서 설정한 옵션값 2000을 우선 적용합니다.
>   - **프로젝트 디폴트**에서 b 옵션값을 아무것도 입력하지 않았고 **업무 디폴트**에 b 옵션값에 2000이 적용된 경우에도 **업무 디폴트**에서 설정한 옵션값을 우선 적용합니다.

에이전트 설정 공유하기

에이전트 설정 내용을 json 형식의 파일로 저장하고 파일을 불러와 다른 에이전트에 적용할 수 있습니다.

1. **에이전트 목록**에서 설정 내용을 json 파일로 내보낼 에이전트를 선택하세요.
2. 화면 오른쪽 위에 ↗ 버튼을 선택하세요.
3. json 파일을 다운로드하세요.
4. **에이전트 목록**에서 다른 에이전트를 선택하세요.
5. 화면 오른쪽 위에 ↙ 버튼을 선택하세요.

json 파일을 통해 가져온 에이전트 설정을 확인하세요.

에이전트 설정 옵션 안내

다음 링크를 통해 설정할 수 있는 옵션들에 대한 설명을 제공합니다.

에이전트 이름

Java 에이전트 이름을 설정하여 모니터링 대상을 식별하는 방법을 안내합니다.

기능 제어

Java 에이전트의 기능을 활성화하거나 비활성화하는 설정 옵션을 안내합니다.

에이전트 사용량

Java 에이전트가 사용하는 시스템 자원량을 확인하는 방법을 안내합니다.

통신 설정

Java 에이전트와 수집 서버 간 네트워크 통신을 설정하는 방법을 안내합니다.

성능 설정

Java 에이전트의 성능 관련 옵션을 설정하여 오버헤드를 최적화하는 방법을 안내합니다.

로그 설정

Java 에이전트의 로그 파일 위치와 로그 레벨 설정 옵션을 안내합니다.

트랜잭션 설정

Java 에이전트의 트랜잭션 추적과 프로파일링 관련 설정 옵션을 안내합니다.

DB/SQL

Java 에이전트의 데이터베이스 연결과 SQL 추적 관련 설정 옵션을 안내합니다.

HTTPC/API Call

Java 에이전트의 외부 HTTP 호출과 API 연동 추적 설정 옵션을 안내합니다.

스크립트 플러그인

Java 에이전트에 사용자 정의 스크립트를 추가하여 모니터링을 확장하는 방법을 안내합니다.

사용자 수

Java 에이전트의 동시 사용자 수 집계 방식과 관련 설정 옵션을 안내합니다.

부하량 제어

Java 에이전트의 쓰로틀링과 부하량 제어 관련 설정 옵션을 안내합니다.

알림 설정

Java 에이전트에서 발생하는 이벤트의 알림 관련 설정 옵션을 안내합니다.

Apdex

Java 에이전트의 Apdex 점수 설정 옵션을 안내합니다.

통계

Java 에이전트의 통계 데이터 수집과 관련된 설정 옵션을 안내합니다.

에러 스택

Java 에이전트의 트랜잭션 에러 발생 시 스택 트레이스 수집 설정을 안내합니다.

추가 옵션

Java 에이전트의 추가 설정 옵션에 대해 안내합니다.

비동기 추적

Java 비동기 호출의 트랜잭션 연결과 추적 설정 방법을 안내합니다.

에이전트 이름 식별

와탭은 모니터링 정보 수집 대상인 애플리케이션 서버 식별을 위한 정보로 기본적으로 애플리케이션 서버로부터 수집한 정보를 활용합니다. 기본 활용 정보는 애플리케이션 서버의 종류, IP 주소 등을 조합해 애플리케이션 서버를 고유 식별자로 사용합니다. 필요에 따라 사용자가 설정한 명칭을 이용하거나 패턴을 변경해 고유한 값으로 변경해 이용합니다. 에이전트의 이름은 반드시 고유한 값이어야 합니다.

애플리케이션 서버로부터 추출한 정보를 활용하는 이유는 애플리케이션 서버 정지 또는 네트워크 단절, 에이전트 문제로 인한 수집 서버와 에이전트의 통신 두절 상태가 복구되었을 경우, 재접속된 에이전트로부터 송신되는 정보가 기존 에이전트로부터 송신된 정보와의 연속성을 유지하기 위해서입니다.

- 에이전트 이름은 프로젝트 단위로 고유해야 합니다.
- 에이전트 아이디/이름을 변경하면 이전 데이터와 연결되지 않습니다.

- 와탭 모니터링 서비스에서는 모니터링 대상을 오브젝트 또는 에이전트로 혼용해서 사용합니다.

기본 이름 결정 방식

JVM 옵션으로 `whatap.oname` 설정이 없는 경우 에이전트 이름을 다음과 같은 우선 순위로 자동 부여합니다.

1. `whatap.type` 을 식별합니다.
2. `whatap.type` 이 서블릿 컨테이너면 애플리케이션이 사용하는 포트를 식별합니다.
3. 포트를 식별할 수 없다면 process ID(PID)를 검색합니다.
4. 식별한 정보에 IPv4 주소 3, 4번째를 추가해 최종 이름을 결정합니다.

예를 들어, IP 주소가 10.11.**12.13**인 서버에서 **Tomcat**이 **8080** 포트로 시작한 경우 `TC-12-13-8080` 명칭으로 에이전트 이름을 자동 부여합니다.

에이전트 이름 변수

사용할 수 있는 이름 변수를 다음과 같이 제공합니다.

변수	설명
문자열	입력한 문자열을 사용합니다.
`{type}`	컨테이너 타입을 자동 식별해 부여합니다. 자동 식별 대상: JAVA, TOMCAT, JBOSS, RESIN, SPRINGBOOT, JETTY, JEUS, WEBLOGIC, WEBSPHERE, UNDERTOW, PLAY
`{ip0}`	IPv4 주소 중 첫 번째 단위를 사용합니다. (예시, **10**.11.12.13 중 **10**)
`{ip1}`	IPv4 주소 중 두 번째 단위를 사용합니다. (예시, 10.**11**.12.13 중 **11**)
`{ip2}`	IPv4 주소 중 세 번째 단위를 사용합니다. (예시, 10.11.**12**.13 중 **12**)
`{ip3}`	IPv4 주소 중 네 번째 단위를 사용합니다. (예시, 10.11.12.**13** 중 **13**)
`{pid}`	Java의 Process ID(PID)를 사용합니다.
`{port}`	애플리케이션이 Listen 하는 포트를 사용합니다.

에이전트 이름 설정

시작 스크립트에 JVM 옵션을 추가해 자동으로 부여한 이름 대신 직접 이름을 설정할 수 있습니다.

옵션	설정 위치	설명
`-Dwhatap.name`	JVM Options	에이전트 이름을 패턴으로 설정합니다. 기본값 `{type}-{ip2}-{ip3}-{port}`
`-Dwhatap.oname`	JVM Options	에이전트 이름을 고정값으로 설정합니다.

옵션	설정 위치	설명
		-Dwhatap.oname=java-sales-01 옵션을 설정하지 않으면 -Dwhatap.name 옵션에 설정된 패턴으로 적용됩니다.

IP 주소가 10.11.12.13이고, 포트 번호가 8000인 환경에서

- **-Dwhatap.oname 옵션을 적용하지 않은 경우**: **whatap.oname**은 -Dwhatap.name 에 설정된 패턴에 따라 JAVA-12-13-8000 으로 기본 설정됩니다.
- **-Dwhatap.oname 옵션을 적용한 경우**: -Dwhatap.name 옵션을 무시하고 -Dwhatap.oname 옵션에 설정한 값으로 에이전트 이름이 적용됩니다.

- 클라우드의 Scale Out 환경에서는 VM이 복사되어 사용될 수 있습니다. 따라서 오브젝트 이름이 자동 부여되는 것이 중요합니다.
- 만약 2개의 VPC 환경을 만들고 동일한 private IP를 사용하도록 네트워크를 구성한 경우 두 개의 VPC가 중복되면 안 됩니다. 예를 들어, 별도의 프로젝트로 분리하거나 -Dwhatap.name 옵션과 이름 변수를 서로 다른 방식으로 조합해 설정할 수 있습니다.

다음은 whatap.conf 파일에 에이전트 이름으로 적용할 수 있는 옵션입니다.

- **whatap.name** String

  기본값 {type}-{ip2}-{ip3}-{port}

  에이전트 이름을 패턴으로 설정합니다.

- **auto_oname_enabled** Boolean

  기본값 false

  서버에 등록할 에이전트 이름(whatap.oname)을 서버로부터 자동 부여받는 기능을 활성화합니다. 해당 옵션 적용 시 JVM 옵션으로 설정한 -Dwhatap.name , -Dwhatap.oname 값은 무시합니다. 수집 서버와의 통신을 통해 whatap.oname 값을 부여 받은 다음 에이전트의 일반적인 동작을 시작합니다.

- **auto_oname_prefix** String

  기본값 agent

  에이전트 이름을 서버로부터 자동 부여할 때 에이전트 이름의 prefix, 보통 업무명을 사용합니다. prefix 일련번호 '1~)'을

부여합니다.

> ⓘ 에이전트 이름을 자동으로 부여하는 `auto_oname_enabled` 옵션이 활성화 상태여야(`true`) 해당 옵션을 사용할 수 있습니다.

- **auto_oname_reset** Int

  기본값 `0`

  서버로부터 새로운 에이전트 이름을 부여받기 위해 수정합니다. 에이전트 이름을 자동 부여하면 `whatap.oname` 이라는 시스템 환경 변수로 설정됩니다. 한번 정해진 시스템 환경 변수는 자바 인스턴스가 다시 시작할 때까지 유지합니다. 리셋을 원할 때 `auto_oname_reset` 값을 수정하세요. 현재 설정 값과 다른 값으로 변경하면 적용됩니다.

- **whatap.okind** String

  여러 개의 에이전트를 '종류별' 그룹으로 묶어 모니터링 단위를 설정할 수 있습니다. 이 옵션을 통해 여러 에이전트를 그룹 단위로 모니터링하고 관리할 수 있으며, 성능 지표를 분석하거나 알림을 설정할 수도 있습니다.

  > ⓘ **대시보드** 및 **분석**, **통계** 메뉴 등에서 **종류별**로 분류된 그룹 옵션으로 확인할 수 있습니다.

- **whatap.onode** String

  여러 개의 에이전트를 '노드별' 그룹으로 묶어 모니터링 단위를 설정할 수 있습니다. 이 옵션을 통해 여러 에이전트를 그룹 단위로 모니터링하고 관리할 수 있으며, 성능 지표를 분석하거나 알림을 설정할 수도 있습니다.

  > ⓘ **대시보드** 및 **분석**, **통계** 메뉴 등에서 노드별로 분류된 그룹 옵션으로 확인할 수 있습니다.

에이전트 기능 제어

자바(Java) 애플리케이션 서버의 모니터링을 위한 에이전트 기능을 세밀하게 조정하는 방법을 안내합니다. 에이전트의 활성화/비활성화, 트랜잭션 및 성능 카운터 추적, OS 정보 수집 등의 기능을 포함하며, 각 기능을 제어하는 다양한 설정 옵션들에 대해 확인할 수 있습니다.

- **shutdown** Boolean

  기본값 `false`

  `true` 인 경우 에이전트의 모든 동작을 중지하고 서버와의 연결을 종료합니다.

- **enabled** Boolean

  기본값 `true`

  전체 기능을 활성화합니다. `false` 인 경우에도 서버와 최소한의 통신을 유지하기 위한 정보는 전송됩니다.

- **transaction_enabled** Boolean

  기본값 `true`

  트랜잭션 추적 기능을 활성화합니다. 히트맵에 기록되는 트랜잭션 정보가 해당합니다. `enabled` 옵션이 `false` 인 경우 비활성화합니다.

- **counter_enabled** Boolean

  기본값 `true`

  성능 카운터 추적 기능을 활성화합니다. 액티브 트랜잭션 수 및 사용자 수, JVM 자원 사용량, Process CPU 사용량, DB Pool 사용량 정보 등이 해당합니다. `enabled` 옵션이 `false` 인 경우 비활성화합니다.

- **stat_enabled** Boolean

  기본값 `true`

  통계 정보 추적 기능을 활성화합니다. 5분 단위로 수집하는 트랜잭션, SQL, HTTPCALL, UserAgent, Client IP 등의 통계 데이터 등이 해당합니다. `enabled` 옵션이 `false` 인 경우 비활성화합니다.

- **sigar_enabled** Boolean

  기본값 `false`

  sigar 라이브러리를 통한 OS 정보 수집을 활성화합니다. sigar 라이브러리를 통해 5초 단위로 수집하는 CPU, Memory, Disk 등의 OS 자원 데이터가 해당합니다. `enabled` 옵션이 `false` 인 경우 비활성화합니다.

> ◦ whatap.agent.2.1.0 버전 미만에서 기본값은 `true` 이며, whatap.agent.2.1.0 버전 이상에서는 기본값이 `false` 입니다.
>
> ◦ Java 1.6 이하 버전 환경에서는 `sigar_enabled=true` 옵션을 추가하여 sigar 라이브러리를 활성화하고 성능 지표를 수집할 수 있습니다.

- **active_stack_enabled** Boolean

  기본값 `true`

  액티브 스택 추적을 활성화합니다. 스택 메뉴의 탑 스택 및 유니크 스택, 액티브 스택이 해당합니다. `enabled` 또는 `counter_enabled` 옵션이 `false` 인 경우 비활성화합니다.

- **license** String

  에이전트 설치 시 서버로부터 부여받은 액세스 키를 입력합니다. 액세스 키에는 에이전트가 속한 프로젝트와 보안 통신을 위한 암호 키를 포함합니다.

- **cypher_level** aes_bit [128, 256]

  기본값 `128`

  AES 보안 알고리즘에 대한 암호 레벨을 설정합니다. 256 비트까지 확장할 수 있습니다. AES 256 암호화 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

- **encrypt_level** encrypt_level [1, 2, 3]

  기본값 `2`

  와탭 에이전트는 서버로 데이터를 전송할 때 데이터 속성에 따라 선택적으로 암호화합니다. 높은 보안을 유지하면서도 성능상 이점을 가지고 있습니다. 데이터 유형에 상관 없이 일괄적인 암호화 정책을 적용하려면 다음 옵션을 참조하세요.

  ◦ `1` : 암호화 전송 기능 사용 안함

  ◦ `2` : SQL 파라미터, Plain Text와 같은 민감한 속성에 대한 암호화 전송

  ◦ `3` : 모든 항목에 대한 암호화 전송

- **stat_ip_enabled** Boolean

  기본값 `true`

  프로젝트의 사이드 메뉴 중 **통계** > **클라이언트 IP** 메뉴에서 IP 통계 사용 여부를 활성화합니다.

> 다음 문서를 참조하세요.

- **stat_ipurl_enabled** Boolean

  기본값 false

  프로젝트 사이드 메뉴 중 **통계** > **IP 트랜잭션** 통계 수집 여부를 설정합니다.

  > 다음 문서를 참조하세요.

- **realtime_user_enabled** Boolean

  기본값 true

  프로젝트의 사이드 메뉴 중 **대시보드** > **애플리케이션 대시보드** 메뉴에서 확인할 수 있는 지표 중 **동시접속 사용자** 지표 수집 여부를 설정합니다.

- **hook_direct_patch_classes** ClassFile_FullPath

  직접적으로 특정 클래스를 로딩 타임에 바꿔치기할 때 사용합니다. 클래스를 컴파일한 후에 별도 파일을 만들고 그 파일의 전체 경로를 설정합니다.

- **active_stack_second** Seconds

  기본값 10

  액티브 스택을 추적하는 간격을 설정합니다.

  > 주어진 기본값을 변경하지 않을 것을 권장합니다.

- **boot_redefine_size** Int

  기본값 100

  에이전트를 Attach 방식 또는 Watcher 방식으로 설치했을 경우 이미 로딩된 클래스의 추적을 위해 BCI를 새로 수행합니다. 이때 동시 redefine하는 클래스의 개수를 설정합니다.

- **trace_component_enabled** Boolean

  기본값 true

  프로젝트의 사이드 메뉴 중 **인스턴스 성능 관리** > **라이브러리 버전** 메뉴 기능을 활성화합니다.

- **realtime_user_thinktime_max / wclient_thinktime_max** MiliSeconds

  기본값 `300000`

  실시간 브라우저 사용자 수를 측정할 경우 동일 사용자로 인정되는 최대 호출 간격을 설정합니다.

  > ⓘ Java 에이전트 2.2.0 버전 이상의 경우 `wclient_thinktime_max` 옵션을 이용하세요.

- **time_sync_interval_ms** MiliSeconds

  기본값 `300000`

  에이전트와 서버 간 동기화하는 시간 간격을 설정합니다. 동기화하지 않을 경우 `0` 으로 설정하세요.

- **detect_deadlock_enabled** Boolean

  기본값 `false`

  Java 스레드의 교착 상태(DeadLock) 여부를 체크해 감지할 경우 이벤트를 발생합니다. 발생 간격은 5초 단위입니다. 같은 DeadLock 건에 대한 이벤트는 한 시간에 한 번만 발생합니다.

- **text_reset** Int

  기본값 `0`

  와탭 에이전트는 한번 보낸 텍스트 유형 데이터를 hash 처리하므로 다음 날까지 다시 전송하지 않습니다. 이전 설정 값과 다른 값을 입력하는 경우 다시 전송합니다.

  > ⓘ 트랜잭션 URL, SQL String 등이 텍스트 유형 데이터에 해당합니다.

에이전트 사용량

Java 에이전트가 CPU, 메모리, 디스크 사용량을 수집하는 방법을 안내합니다. 자바(Java)의 JMX 라이브러리를 기본적으로 활용하며, 필요에 따라 Linux의 proc 디렉터리나 oshi, sigar 라이브러리를 통한 성능 지표 수집 방법을 확인할 수 있습니다.

- **linux_proc_stat_enabled** Boolean

  기본값 `false`

  CPU, 메모리, 디스크 사용량을 Linux의 proc 디렉터리에서 수집할지 여부를 설정합니다.

- **oshi_enabled** Boolean Java Agent v2.1.0 or later

  기본값 `false`

  CPU, 메모리, 디스크 사용량을 측정할 때 oshi 라이브러리 사용 여부를 설정합니다.

  > ⓘ Java 8 이상의 환경에서 사용할 것을 권장합니다.

- **oshi_netstat_enabled** Boolean Java Agent v2.2.31 or later

  기본값 `false`

  netstat 지표 수집 여부를 설정합니다. Maximum Transmission Unit(MTU) 지표가 추가됩니다.

- **sigar_enabled** Boolean

  기본값 `false`

  sigar 라이브러리를 통한 OS 정보 수집을 활성화합니다. sigar 라이브러리를 통해 5초 단위로 수집하는 CPU, Memory, Disk 등의 OS 자원 데이터가 해당합니다. `enabled` 옵션이 `false` 인 경우 비활성화합니다.

  > ⓘ
  > - whatap.agent.2.1.0 버전 미만에서 기본값은 `true` 이며, whatap.agent.2.1.0 버전 이상에서는 기본값이 `false` 입니다.
  > - Java 8 미만 또는 Java 1.6 이하 버전 환경에서는 `sigar_enabled=true` 옵션을 추가하여 sigar 라이브러리를 활성화하고 성능 지표를 수집할 수 있습니다.
  > - 애플리케이션 환경에 따라 CPU 지표가 **-1** 표시되는 현상이 있을 수 있습니다. 와탭에서 제공하는 권고 사항을 확인하고 현상을 해결할 수 있습니다. 다음 문서를 참조하세요.

에이전트 통신 설정

에이전트가 애플리케이션 서버로부터 수집한 데이터를 서버로 전송하기 위한 네트워크 설정 방법을 안내합니다. 주요 설정 옵션으로는 수집 서버 호스트, 포트, 타임아웃 값, 네트워크 전송 크기, 데이터 전송 큐 크기 등이 있으며, 에이전트의 효율적인 데이터 관리 및 안정적인 서버 통신을 보장하는 데 중요합니다.

- **whatap_server_host** ip_address

  기본값 `127.0.0.1,127.0.0.1`

  에이전트가 수집한 데이터를 전송할 서버를 설정합니다. 수집 서버 이중화로 2개 이상의 IP 주소를 가진 경우 쉼표(,)를 구분자로 이용하세요. 설정한 IP 주소에는 수집 서버 proxy 서버가 리스닝(Listening) 상태로 서비스돼야 합니다.

- **whatap_server_port** tcp_port

  기본값 `6600`

  수집 서버 포트를 설정합니다. 포트는 하나만 설정할 수 있으므로 `whatap_server_host` 옵션에 설정한 수집 서버들은 동일 포트를 사용해야 합니다.

- **tcp_so_timeout** MiliSecond

  기본값 `60000`

  수집 서버와 통신하는 TCP 세션의 Socket Timeout 값을 설정합니다.

- **tcp_connection_timeout** MiliSecond

  기본값 `5000`

  수집 서버와 통신하는 TCP 세션의 Connection Timeout 값을 설정합니다.

- **net_send_max_bytes** Byte

  기본값 `5242880`

  에이전트가 데이터를 수집하고 네트워크로 한 번에 전송할 수 있는 최대 byte 크기입니다.

- **net_send_queue1_size** Int

  기본값 `256`

  트레이스 정보와 액티브 스택을 제외한 나머지 데이터 전송에 사용할 Queue의 크기를 설정합니다. 설정한 크기가 10 이하인 경우 10으로 설정합니다.

- **net_send_queue2_size** Int

기본값 512

트레이스 정보와 액티브 스택 데이터 전송에 사용할 Queue의 크기를 설정합니다. 설정한 크기가 10 이하인 경우 10으로 설정합니다.

에이전트 성능 설정

자바(Java) 에이전트의 성능 관련 옵션을 안내합니다. 동시에 처리할 수 있는 최대 트랜잭션 수와 트랜잭션 데이터를 저장하는 버퍼의 초기 크기 설정을 확인할 수 있습니다. 이는 시스템의 부하와 자원 사용을 최적화하는 데 중요한 역할을 합니다.

whatap.agent.2.1.0 버전 미만

- **whatap.tx_max_count** Int

  기본값 `5000`

  트랜잭션을 최대 저장하는 개수입니다. 동시에 처리하는 트랜잭션이 많을 경우 해당 크기를 더 늘려주세요.

- **whatap.tx_default_capacity** Int

  기본값 `101`

  트랜잭션을 저장하는 버퍼의 시작 크기입니다.

에이전트 로그 설정

자바(Java) 애플리케이션 환경에서 발생하는 로그 데이터를 에이전트를 통해 관리하는 방법을 안내합니다. 로그 파일의 경로 및 이름 설정, 보관 기간 설정, 로그 모니터링, 사용자 정의 로그 추적 활성화 방법 등을 포함합니다. 로그 관리를 위한 다양한 설정 옵션을 제공해 시스템의 효율적인 로그 관리를 도와줍니다.

경로 및 이름 설정

에이전트 로그의 경로와 이름은 whatap.conf 파일에 설정할 수 있습니다. `log_root`의 기본값으로 `${WHATAP_HOME}/logs` 경로를 사용합니다.

whatap.conf

```
log_root=whatap/logs
log_name=whatap
```

로그 관리 옵션

whatap.conf 파일에 로그 관련 옵션을 작성해 로그 내용을 관리할 수 있습니다.

whatap.conf

```
watchlog_enabled=true
log_rotation_enabled=true
log_oname_enabled=true
log_keep_days=7
```

- **watchlog_enabled** Boolean

  기본값 `false`

  값이 `true`이면 로그 모니터링을 활성화합니다.

- **watchlog_check_interval** Int

  기본값 `2000`

로그를 감시하는 간격입니다. 매 간격으로 로그 파일에 기록을 추가했는지 확인합니다.

- **watchlog_read_count** Int

  기본값 `8`

  로그를 읽는 최대 횟수입니다. 한 번에 `watchlog_buffer_size` 크기 만큼 읽습니다.

- **watchlog_buffer_size** Int

  기본값 `131072(128k)`

  로그를 한 번에 읽는 로그 사이즈입니다.

- **watchlog.LOGKEY**

  하나의 로그 파일을 감시하기 위한 설정입니다. `file` 이름을 입력하고 감시한 `word` 여러 개를 쉼표(,)로 구분해 설정합니다. `check_interval` 간격으로 로그를 감시하다가 키워드를 발견하면 경고합니다. 한번 경고가 나가면 `silent` 만큼 경고를 멈춥니다.

  - **watchlog.LOGKEY.enabled** Boolean 기본값 `true`
  - **watchlog.LOGKEY.file** String
  - **watchlog.LOGKEY.words** String
  - **watchlog.LOGKEY.silent** Int 기본값 `10000`
  - **watchlog.LOGKEY.check_interval** Int 기본값 `1000`

- **log_root** String

  기본값 `${WHATAP_HOME}` /logs

  에이전트 로그 경로를 설정합니다.

- **log_datasource_lookup_enabled** Boolean

  기본값 `true`

  InitialContext Lookup 시 DataSource인 경우 로그를 기록하는 기능을 활성화합니다.

- **log_rotation_enabled** Boolean

  기본값 `true`

  에이전트 로그 파일을 날짜 별로 저장하는 기능을 활성화합니다. 로그 파일명은 whatap- `yyyymmdd` .log 형태로 저장합니다. 값이 `false` 이면 로그 파일명은 whatap.log 형태로 저장합니다.

- **log_keep_days** Int

기본값 `7`

로그 파일 보관 기간을 설정합니다. `log_rotation_enabled` 값이 `true`인 경우에만 동작합니다.

- **log_oname_enabled** Boolean

  기본값 `false`

  값이 true이면 각 로그마다 에이전트 이름을 추가합니다.

```
20220728 09:28:48(GMT) [ote-proxy] logsink send zip-mode enabled=true
20220728 09:28:48(GMT) [ote-notihu][BCI] httpc org/apache/http/impl/client/AbstractHttpClient
20220728 09:28:50(GMT) [ote-notihu][InnerMod] whatap.tracer.http.jar loaded 12962 bytes
20220728 09:28:51(GMT) [ote-notihu] logsink send zip-mode enabled=true
20220728 09:28:53(GMT) [ote-yard] logsink send zip-mode enabled=true
20220728 09:31:00(GMT) [ote-keeper][UrlNorm] Class Scan tot=9292 real=2897 classes 11ms done.
20220728 09:31:00(GMT) [ote-keeper][UrlNorm] Service PathTree done. BuildBy=FullBuild PathTree
20220728 09:31:00(GMT) [ote-gatewa][UrlNorm] Class Scan tot=13269 real=5816 classes 25ms done.
```

로그싱크(LogSink)

whatap.agent-2.1.0 버전 이상부터는 애플리케이션 서버를 통합 모니터링하는 로그싱크(LogSink) 기능을 제공합니다.

> LogSink 옵션은 whatap.agent-2.1.0 버전 이상에서 사용 가능합니다.

로그 모니터링 기능 활성화

- **logsink_enabled** Boolean

  기본값 `false`

  로그 모니터링 기능을 활성화합니다.

- **logsink_trace_enabled** Boolean

  기본값 `false`

  Log에 트랜잭션 ID를 삽입하여, 트랜잭션 트레이스의 로그 탭을 노출할지 여부를 지정합니다.

프레임워크 로그내용 수집

프레임워크에서 수집하는 로그를 모니터링하려면 모듈에서 로그를 가로채기 위한 설정과 플러그인을 추가해야 합니다.
프레임워크에서 수집하는 로그의 기본 카테고리 이름을 'AppLog'로 설정합니다. 기본값은 `hooklog_enabled` 옵션의 설정과 동일합니다.

- **hooklog_enabled** Boolean

  기본값 `logsink_enabled`

  로그 추적 활성화 여부를 설정합니다.

  > ◦ Java 에이전트 2.2.4 버전 이상에서는 애플리케이션의 재기동 없이 옵션의 변경 사항을 적용할 수 있습니다.
  >
  > ◦ Java 에이전트 2.2.4 버전 미만에서는 `hooklog_enabled` 값을 변경하면 재기동이 필요합니다. `hooklog_enabled` 기본값은 `logsink_enabled` 입니다. 따라서 `logsink_enabled=true` 로 설정하고 자바 애플리케이션을 재시작하면 `hooklog_enabled` 는 `true` 로 설정됩니다.

- **hooklog_custom_methods** String

  사용자 정의 로그를 등록합니다. 임의의 로그 프레임워크 내용을 전달합니다. 사이트에서 개별로 만든 로그 모듈의 로그를 추적할때 사용하세요.

  Java

```java
package io.home.test;

public class MyLog {
  public void customLog(String log) { ... }
}
```

  whatap.conf

```
hooklog_custom_methods=io.home.test.MyLog.customLog
```

특정 로그 Appender 제외 설정하기

특정 로그 appender를 제외하는 에이전트 옵션을 안내합니다. 다음 제공하는 에이전트 옵션을 통해 모니터링 대상에서 특정

appender를 제외할 수 있습니다.

- **ignore_logback_appender_set** String Java Agent v2.2.28 or later

  logback-1.2.8을 추적할 때 특정 appender를 제외하는 에이전트 옵션입니다. 예를 들어, `CONSOLE`과 `ACCESS_APPENDER`를 제외하려면 다음 예제를 참조하세요.

  whatap.conf

  ```
  weaving=logback-1.2.8
  # example: ignore 'CONSOLE', 'ACCESS_APPENDER' appender
  ignore_logback_appender_set=CONSOLE,ACCESS_APPENDER
  ```

  ◦ 이 에이전트 옵션을 적용할 경우 `weaving=logback-1.2.8` 옵션을 추가 적용해야 합니다.

  ◦ `ACCESS_APPENDER`는 예시로 사용한 값입니다. 사용자 환경에 맞는 appender 이름을 지정해야 합니다.

- **ignore_log4j_appender_set** String Java Agent v2.2.28 or later

  log4j-2.17을 추적할 때 특정 appender를 제외하는 에이전트 옵션입니다. 예를 들어, `CONSOLE`과 `ACCESS_APPENDER`를 제외하려면 다음 예제를 참조하세요.

  whatap.conf

  ```
  weaving=log4j-2.17
  # example: ignore 'CONSOLE', 'ACCESS_APPENDER' appender
  ignore_log4j_appender_set=CONSOLE,ACCESS_APPENDER
  ```

  ◦ 이 에이전트 옵션을 적용할 경우 `weaving=log4j-2.17` 옵션을 추가 적용해야 합니다.

  ◦ `ACCESS_APPENDER`는 예시로 사용한 값입니다. 사용자 환경에 맞는 appender 이름을 지정해야 합니다.

에이전트 트랜잭션 설정

트랜잭션은 사용자 브라우저의 요청을 처리하는 서버 사이드의 Logical Unit of Work(LUW)입니다. 웹 애플리케이션에서 하나의 HTTP 요청을 받아 응답을 반환하는 전체 과정이 트랜잭션에 해당합니다.

애플리케이션의 성능은 트랜잭션 처리 성능으로 평가할 수 있습니다. 트랜잭션 성능은 트랜잭션 시작부터 종료까지의 응답 시간, 자원 사용량, 호출자 속성 등의 정보가 포함됩니다.

트랜잭션 성능은 응답 시간 분포와 통계 데이터를 기반으로 분석할 수 있습니다.

트랜잭션 트레이싱

다음은 애플리케이션 내의 각 트랜잭션의 실행 경로를 상세하게 추적할 수 있는 에이전트 옵션입니다. 최소 응답 시간부터 트랜잭션의 실행 스텝 수까지 다양한 측면을 조정할 수 있습니다.

- **profile_basetime** MiliSecond

  기본값 `500`

  트레이스의 상세 정보를 수집할 최소 응답 시간을 설정합니다. 설정한 값 이하의 시간 내에 트랜잭션이 종료된 경우 트레이스 정보를 수집하지 않습니다. 단, 5분 단위로 최초 호출된 URL과 에러가 발생한 트랜잭션에 대한 트레이스 정보는 수집합니다.

  > ⓘ `profile_basetime` 옵션에 상관없이 에러를 수집하려면 `profile_concern_error_enabled` 옵션을 `true` 로 설정하세요. 기본값 `false` 로 설정하면 에러를 샘플링하게 됩니다

- **profile_step_normal_count** Int

  기본값 `1000`

  트랜잭션 트레이스의 최대 스텝 수를 설정합니다.

- **profile_step_heavy_count** Int

  기본값 `1020`

  Heavy한 스텝의 경우 트레이스 기본 스텝 수를 초과하더라도 정해진 값만큼 기록합니다.

- **profile_step_max_count** Int

  기본값 `1024`

트레이스 스텝의 최대 수를 설정합니다. 수집된 트레이스 스텝 수가 이 값을 초과하면 이후 수집되는 스텝들은 모두 버려집니다. `profile_step_heavy_count`을 최대 1000으로 설정하면 `profile_step_max_count` 만큼 액티브 스택이 수집됩니다.

- **profile_step_heavy_time** MiliSecond

  기본값 `100`

  Heavy한 스텝의 기준을 설정합니다. 설정한 값보다 수행 시간이 긴 경우 `profile_step_normal_count` 값을 초과하더라도 `profile_step_heavy_count`에 설정한 값만큼 기록합니다.

- **active_stack_count** Int

  기본값 `100`

  트랜잭션 내에서 수집하는 액티브 스택의 최대 수를 설정합니다.

- **profile_method_resource_enabled** Boolean

  기본값 `false`

  트레이스에서 method 스텝을 수집할 때 해당 스텝에서 사용한 CPU와 메모리 사용량을 추적합니다.

- **profile_position_method** String

  설정한 메소드를 수행하는 시점의 StackTrace를 기록합니다.

- **profile_position_depth** Int

  기본값 `50`

  position 추적을 위해 StackTrace를 기록할 때 최대 라인 수를 설정합니다.

- **trace_error_callstack_depth** Int

  기본값 `50`

  Error 발생 시 수집하는 StackTrace의 최대 라인 수를 설정합니다.

- **trace_active_callstack_depth** Int

  기본값 `50`

  액티브 스택에서 수집하는 StackTrace의 최대 라인수를 설정합니다.

- **trace_active_transaction_yellow_time** MiliSecond

  기본값 `3000`

  액티브 트랜잭션의 아크이퀄라이저에서 노란색으로 표현할 기준을 설정합니다.

- **trace_active_transaction_red_time** MiliSecond

  기본값 8000

  액티브 트랜잭션의 아크이퀄라이저에서 빨간색으로 표현할 기준을 설정합니다.

- **hook_method_patterns** String

  응답시간을 측정할 메소드를 설정합니다. 마지막 마침표(.)를 구분자로 클래스 FullName과 메소드로 구분되며 별표(*)를 와일드 카드로 사용할 수 있습니다. 대상이 여러 개인 경우 쉼표(,)로 구분합니다.

  whatap.conf

```
hook_method_patterns=a.b.C1.*
```

  ◦ 이 옵션을 통해 등록한 메소드는 **트레이스 분석** 또는 **트랜잭션 정보** 창에서 Method 스텝으로 확인할 수 있습니다.

  ◦ 가급적 국소 범위에 적용할 것을 권장합니다. 광범위한 범위로 적용할 경우 애플리케이션의 성능 저하를 유발할 수 있습니다.

- **hook_method_supers** String

  특정 클래스를 상속받은 메소드의 응답시간을 측정하고자 할 때 Super Class를 설정합니다. 클래스 FullName을 설정하며 대상이 여러 개인 경우 쉼표(,)로 구분합니다.

  whatap.conf

```
hook_method_supers=a.b.C1
```

- **hook_method_interfaces** String

  특정 인터페이스를 구현한 메소드의 응답시간을 측정하고자 할 때 Interface를 설정합니다. 인터페이스 FullName을 설정하며 대상이 여러 개인 경우 쉼표(,)로 구분합니다.

- **hook_method_ignore_classes** String

  메소드 트레이스를 설정할 때 트레이스에서 제외하고 싶은 클래스들을 설정합니다.

- **hook_method_access_public_enabled** Boolean

  기본값 true

  메소드 트레이스를 설정할 때 public 메소드에 대해서만 별도 대상으로 처리할지 여부를 설정합니다.

- **hook_method_access_private_enabled** Boolean

  기본값 `false`

  메소드 트레이스를 설정할 때 private 메소드에 대해서만 별도 대상으로 처리할지 여부를 설정합니다.

- **hook_method_access_protected_enabled** Boolean

  기본값 `true`

  메소드 트레이스를 설정할 때 protected 메소드에 대해서만 별도 대상으로 처리할지 여부를 설정합니다.

- **hook_method_access_none_enabled** Boolean

  기본값 `true`

  메소드 트레이스를 설정할 때 no access indicated 메소드에 대해서만 별도 대상으로 처리할지 여부를 설정합니다.

- **stacklog_socket_port** TCP_PortNumber

  기본값 `0`

  목적지의 TCP 포트를 설정하면 Socket.connect() 시점 StackTrace를 에이전트 로그로 기록합니다. 기본 설정으로 확인되지 않는 DB 연결, HTTPC 연결 등을 추적할 때 사용할 수 있습니다.

  > 설정한 목적지로 연결 시마다 매번 StackTrace를 기록합니다. 성능 저하를 유발할 수 있습니다. 디버깅 용도로 선별한 에이전트에 한시적으로만 사용해야 합니다.
  >
  > DB 연결 상태 추적 예시, `stacklog_socket_port=1521`

- **trace_concern_error_enabled** Boolean

  기본값 `true`

  `trace_basetime` 또는 `profile_basetime` 옵션과 무관하게 에러가 있는 상세 트레이스에 대한 수집 여부를 설정합니다.

예외 처리

다음은 애플리케이션에서 발생하는 예외를 관리하고 모니터링하기 위한 옵션입니다. 비즈니스 예외를 등록하고 관련 통계를 수집하여 애플리케이션의 안정성을 확인할 수 있습니다. 또한 특정 예외를 무시하거나 HTTP 상태 코드가 에러인 경우 에러로 처리할지 여부를 설정할 수도 있습니다. 예외로 처리된 항목들은 **히트맵** 또는 **트레이스 분석** 창에서 정상 트랜잭션으로 표시됩니다.

> Java 에이전트의 예외 처리 기준에 대한 자세한 내용은 다음 문서를 참조하세요.

- **biz_exceptions** String

  Business Exception을 등록합니다.

  whatap.conf

```
# example
biz_exceptions=com.home.test.exception.BizException
```

> 에러 클래스 수집 / 히트맵 표시 레벨: INFO(파란색) / 에러 통계 포함

> Java 에이전트 v2.2.20 이상에서는 등록한 비즈니스 Exception을 에러 통계로 수집합니다. v2.2.20 미만에서는 등록한 비즈니스 Exception을 에러 통계에서 제외합니다. 다만 트레이스 상세에서는 나타납니다.

- **biz_exceptions_status** String Java Agent v2.2.20 or later

  Business Exception 발생 시 이벤트 알림을 억제할 Business Exception과 Status를 설정합니다. 이 옵션에서 설정한 **Business Exception:Status**에 대해서는 이벤트 레벨이 INFO로 변경됩니다. Business Exception과 Status는 콜론(:)으로 구분하며, 하나의 Business Exception에 여러 개의 Status를 사용하는 경우 앰퍼샌드(&)로 구분합니다. 쉼표를 구분자로 이용해 여러 개를 설정할 수 있습니다.

  whatap.conf

```
# example
biz_exceptions_status=java.util.concurrent.TimeoutException:400&404&408,java.lang.Exception:200
```

> 에러 클래스 수집 / 히트맵 표시 레벨: INFO(파란색) / 에러 통계 포함

- **ignore_exceptions** String

  등록한 Exception의 에러 자체를 무시합니다.

  whatap.conf

```
# example
ignore_exceptions=com.home.test.exception.BizException
```

에러 클래스 무시 / 히트맵 표시 레벨: INFO(파란색) / 에러 통계 미포함

- **transaction_status_error_enable** Boolean

  기본값 `true`

  HTTP 401, 403과 같이 정상 응답이 아닌 HTTP 상태 코드를 반환하는 경우 에러로 처리할지 여부를 설정합니다.

- **status_ignore** String

  무시하려는 HTTP 상태 코드를 설정할 수 있습니다. 여러 값을 대상으로 할 경우 쉼표(,)를 구분자로 사용하세요.

  whatap.conf

```
# example
status_ignore=408,500
```

Status 에러 무시 / 히트맵 표시 레벨: INFO(파란색) / 에러 통계 미포함

- **status_ignore_set** String

  `whatap.error.STATUS_ERROR`를 무시합니다. 여러 값을 대상으로 할 경우 쉼표(,)를 구분자로 사용하세요.

  whatap.conf

```
# example
status_ignore_set=/api/test/timeout/{time}:408,/error:500
```

Status 에러 세트 무시 / 히트맵 표시 레벨: INFO(파란색) / 에러 통계 미포함

- **httpc_status_error_enable** Boolean

  기본값 `true`

  HTTP 상태 코드가 에러인 경우 수집 여부를 설정합니다. 클라이언트 에러 응답(`400` 이상), 서버 에러 응답(`500` 이상)이

해당됩니다.

> HTTP 상태 코드에 대한 자세한 내용은 다음 링크를 참조하세요.

- **httpc_status_ignore** String

  HTTP 상태 코드가 에러(`HTTPC_ERROR`)인 경우 무시할 수 있습니다. 여러 값을 대상으로 할 경우 쉼표(,)를 구분자로 사용하세요.

  whatap.conf

```
# example
httpc_status_ignore=404,500
```

> HTTP 상태 에러 코드 무시 / 에러 통계 미포함

- **httpc_status_url_ignore_set** String

  `whatap.error.HTTPC_ERROR` 를 무시합니다. 여러 값을 대상으로 할 경우 쉼표(,)를 구분자로 사용하세요.

  `httpc_status_url_ignore_set=/a/b/c:400,/a/ab/c:404` 와 같이 설정하세요. 이때 url은 통계/에러 분석에서 `HTTPC_URL` 값을 입력합니다.

  whatap.conf

```
# example
httpc_status_url_ignore_set=/a/b/c:404,/d/e/f:500
```

> HTTP 상태 에러 코드 무시 / 에러 통계 미포함

- **trace_sql_exception_enabled** Boolean

  기본값 `true`

  JDBC 드라이버에서 SqlException 발생 시 추적 여부를 설정합니다. 옵션 값을 `false` 로 설정하면 사용자 정의 예외 처리로 `biz_exceptions` 처리할 수 있습니다.

- **trace_exception_handler_overwrite_enabled** Boolean

기본값 `true`

spring-boot ExceptionHandler를 통과하는 예외(exception)가 발생하면 이전에 발생한 예외를 대체할 수 있습니다. **트레이스 분석** 창의 **레코드 요약** 탭에서 예외(exception)를 spring-boot의 ExceptionHandler에 등록한 예외로 표시할 수 있습니다.

HTTP 트랜잭션 추적

다음은 HTTP 요청 및 응답에 관련된 다양한 정보를 추적하고 기록하는 에이전트 옵션입니다. 이를 통해 사용자는 애플리케이션의 HTTP 트랜잭션을 세밀하게 모니터링하고 분석할 수 있습니다. 트랜잭션 이름에 파라미터 값을 추가하거나 특정 URL이나 HTTP 메소드를 제외할 수 있습니다. 또한 HTTP 헤더 정보와 파라미터 정보를 포함하여 트랜잭션을 보다 상세하게 기록할 수 있습니다.

- **trace_normalize_enabled** Boolean

  기본값 `true`

  트랜잭션 URL을 파싱해 정규화하는 기능을 활성화합니다.

  > ⓘ `false`로 값을 변경하면 패스 파라미터 파싱을 비활성화합니다. 이 경우 통계 데이터의 의미가 약화됨으로 디버그 용도로만 잠시 사용하는 것을 권장합니다.

- **trace_auto_normalize_enabled** Boolean

  기본값 `true`

  트랜잭션 URL 정규화할 때 패턴 값을 어노테이션에서 추출해 자동으로 파싱하는 기능을 활성화합니다.

- **trace_normalize_urls** String

  정규화할 트랜잭션 URL 패턴을 설정합니다. 호출 URL 패턴을 파싱해 패스 파라미터를 제거합니다.

  > ⓘ **예시**
  >
  > `/a/{v}/b` 라고 선언하면 `a/123/b` → `a/{v}/b` 로 치환합니다. 여러 개를 등록할 때는 쉼표(,)를 구분자로 사용하세요. 치환 패턴 정리 후 보완이 필요합니다.

- **web_static_content_extensions** String

  기본값 `js, htm, html, gif, png, jpg, css, swf, ico`

스태틱 콘텐츠임을 판단하는 확장자를 설정합니다. 이 옵션에 설정한 확장자를 가진 트랜잭션들은 트레이스 추적과 카운팅에서 제외합니다.

- **trace_transaction_name_header_key** String

  설정한 HTTP 헤더 키에서 추출한 값을 트랜잭션의 이름 마지막에 추가합니다.

- **profile_http_header_enabled** Boolean

  기본값 `false`

  트레이스 내역에 http 헤더 정보를 기록하려면 `true` 로 설정하세요.

- **profile_http_parameter_enabled** Boolean

  기본값 `false`

  트레이스 내역에 http 파라미터 정보를 기록하려면 `true` 로 설정하세요. 파라미터는 별도 보안키를 입력해야 조회할 수 있습니다.

  > ◦ **Java 에이전트 2.2.2 버전 미만**: 보안 키는 WAS 서버 `${WHATAP_AGENT_HOME}` /paramkey.txt 파일 내에 6자리로 작성합니다. paramkey.txt 파일이 존재하지 않는 경우 랜덤 값으로 자동 생성합니다.
  >
  > ◦ **Java 에이전트 2.2.2 버전 이상**: 보안 키는 WAS 서버 `${WHATAP_AGENT_HOME}` /security.conf 파일 내에 `paramkey` 키값을 확인하세요. security.conf 파일이 존재하지 않을 경우 `paramkey` 키값을 **WHATAP**으로 자동 생성합니다.
  >
  > ◦ 보안키 설정 파일에 대한 자세한 내용은 다음 문서를 참조하세요.

- **profile_http_header_url_prefix** String

  트레이스 내역에 http 헤더 정보를 기록할 대상 URL의 prefix를 정의할 때 사용합니다.

- **profile_http_parameter_url_prefix** String

  트레이스 내역에 http 파라미터 정보를 기록할 대상 URL의 prefix를 정의할 때 사용합니다.

- **trace_transaction_name_key** String

  HTTP request parameter 값을 해당 옵션에 설정하면, 파라미터 값을 추출하여 트랜잭션의 이름 마지막에 추가합니다.

  예를 들어, HTTP 파라미터로 `paramKey` 를 가진 `/api/test` URL을 호출하는 경우 다음 예제와 같이 설정하면 트레이스 내역에 `/api/test$paramKey={value}` 로 트랜잭션 이름을 표시합니다. 다만 여러 개의 파라미터를 등록할 수는 없습니다.

  whatap.conf

```
# example
trace_transaction_name_key=paramKey
```

- **recursive_max** Int

  기본값 `1000000`

  트랜잭션의 재귀 호출 여부 검출을 위한 옵션입니다. 단일 트랜잭션으로부터 파생되는 재귀 호출 횟수를 카운트하여 이벤트 알림을 발행하기 위한 기준을 설정합니다.

  > ⓘ HTTP URL 재귀 호출을 대상으로 합니다. jsp:forward를 통해 재호출하는 케이스도 카운트에 포함합니다.

- **hook_httpservlet_classes** String

  HTTP 트랜잭션의 END POINT를 추가로 설정합니다. 메소드의 첫 번째 2개의 파라미터는 `HttpServletRequest`와 `HttpServletResponse`만 설정할 수 있습니다.

- **hook_jsp_patterns** String

  기본값 `org.apache.jasper.servlet.JspServlet.serviceJspFile`

  JSP 파일을 로딩하는 메소드를 설정합니다. 트랜잭션 호출 결과로 반환하는 JSP 정보를 트레이스에 표시합니다. 이 옵션을 통해 추가한 설정에 기본값이 자동 추가됩니다.

- **trace_ignore_url_set** String

  트랜잭션 추적에서 제외할 URL을 설정합니다. 2개 이상의 값을 설정하려면 쉼표(,)를 구분자로 이용하세요.

  > ⓘ 이 옵션을 통해 등록한 URL은 **히트맵**, **트레이스 분석** 또는 **트랜잭션 정보** 창에서 수집 제외됩니다.

- **trace_ignore_active_url_set** String

  액티브 트랜잭션 목록에서 제외할 URL을 설정합니다. 2개 이상의 값을 설정하려면 쉼표(,)를 구분자로 이용하세요.

  > ⓘ 이 옵션을 통해 등록한 URL은 **액티브 트랜잭션** 창에서만 수집 제외되며, 히트맵, 트레이스 분석, 트랜잭션 정보 등 다른 트랜잭션 데이터에는 영향을 주지 않습니다.

- **trace_ignore_url_prefix** String

  트랜잭션 추적에서 제외할 URL prefix를 설정합니다. 2개 이상의 값을 설정하려면 쉼표(,)를 구분자로 이용하세요.

> ⓘ 이 옵션을 통해 등록한 URL은 **히트맵**, **트레이스 분석** 또는 **트랜잭션 정보** 창에서 수집 제외됩니다.

- **ignore_http_method** String

  기본값 `PATCH,OPTIONS,HEAD,TRACE`

  설정한 HTTP 메소드(Method)로 요청된 트랜잭션 정보는 수집하지 않습니다. 여러 개를 설정하려면 쉼표(,)를 구분자로 이용하세요.

  whatap.conf

```
# e.g. http_method가 OPTIONS, HEAD인 트랜잭션 "추적"
ignore_http_method=PATCH,TRACE

# e.g. http_method가 OPTIONS, HEAD인 트랜잭션 "무시"
ignore_http_method=OPTIONS,HEAD
```

> ⓘ 이 옵션을 통해 등록한 HTTP 메소드는 **히트맵**, **트레이스 분석** 또는 **트랜잭션 정보** 창에서 수집 제외됩니다.

- **trace_tx_name_with_method_enabled** Boolean

  기본값 `false`

  통계 데이터에서 URL을 변경한 URL+method로 수집할 수 있습니다. **통계** 또는 **트랜잭션 검색** 메뉴에서 URL+method로 확인할 수 있습니다.

  예시, /api/user+GET, /api/user+POST

NON HTTP 트랜잭션 추적

다음은 HTTP 이외의 트랜잭션을 추적하고 모니터링하기 위한 옵션입니다. 이 옵션을 통해 사용자는 백그라운드 작업이나 다른 프로토콜에 대한 트랜잭션을 세밀하게 추적할 수 있습니다.

- **trace_auto_transaction_enabled** Boolean

  기본값 `false`

  트레이스 대상 메소드가 트랜잭션 시작점(`Javax.http.httpservlet` , `hook_service_*`) 내에서 수행하는 경우가 아니라면

수집되지 않습니다. 이 경우 트레이스 대상 메소드가 트랜잭션 시작점이 되도록 설정합니다.

> ⓘ 주로 개발 환경에서 백그라운드 트랜잭션의 END POINT를 찾을 때 사용합니다.

- **trace_auto_transaction_backstack_enabled** Boolean

  기본값 `true`

  `trace_auto_transaction_enabled` 옵션의 값이 `true` 인 경우 트랜잭션 시작 시 StackTrace를 기록합니다. 이를 통해 트랜잭션의 시작점을 찾아낼 수 있습니다.

- **trace_background_socket_enabled** Boolean

  기본값 `true`

  트랜잭션이 아닌 백그라운드 스레드에 의한 소켓이 오픈될 때도 이를 기록합니다.

- **async_stack_enabled** Boolean

  기본값 `false`

  백그라운드 스레드에 대한 Active Stack 기능 사용 여부를 설정합니다.

- **async_thread_match** String

  액티브 스택을 덤프할 백그라운드 스레드 이름을 설정합니다. 여러 개를 설정하려면 쉼표(,)를 구분자로 사용하세요. 이름을 설정할 때 'Thread-*' 처럼 별표(*)를 사용해 비교 패턴을 사용할 수 있습니다.

- **async_thread_parking_class** String

  기본값 `sun.misc.Unsafe`

  스택의 Top 메소드가 `async_thread_parking` 에 등록한 클래스/메소드일 때 스레드가 파킹 상태에 있다고 판단하고 덤프를 생성하지 않습니다.

- **hook_service_patterns** String

  NON-Http 트랜잭션 추적을 위한 시작점 패턴을 설정합니다. 와일드카드 문자를 사용할 수 없으며, 2개 이상의 값을 설정하려면 쉼표(,)를 구분자로 이용하세요.

- **hook_serivce_ignore_methods** String

  `hook_service_patterns` 에서 설정한 내역 중 시작점으로 불필요한 메소드를 추가할 수 있습니다.

- **hook_service_supers** String

NON-HTTP 트랜잭션 추적을 위한 시작점의 공통 분모가 특정 클래스의 메소드를 상속 받은 경우라면 공통 분모의 메소드를 설정합니다. 메소드 전체 경로를 입력하세요. 와일드 카드(`*`)를 사용할 수 있으며, 쉼표(,)를 구분자로 이용해 여러 개의 메소드를 등록할 수 있습니다.

whatap.conf

```
hook_service_supers=a.b.C.method,a.b.C.*
```

- **hook_service_interfaces** String

  NON-Http 트랜잭션 추적을 위한 시작점의 공통 분모가 특정 인터페이스를 구현한 경우라면 이를 설정합니다.

- **hook_service_access_public_enabled** Boolean

  기본값 `true`

  Non Http Demon 프로세스의 트랜잭션을 설정할 때 public 메소드에 대해서만 Access 권한 기준을 on/off 설정합니다.

- **hook_service_access_private_enabled** Boolean

  기본값 `true`

  Non Http Demon 프로세스의 트랜잭션을 설정할 때 private 메소드에 대해서만 Access 권한 기준을 on/off를 설정합니다.

- **hook_service_access_protected_enabled** Boolean

  기본값 `true`

  Non Http Demon 프로세스의 트랜잭션을 설정할 때 protected 메소드에 대해서만 Access 권한 기준을 on/off를 설정합니다.

- **service_name_mode** String

  기본값 `full`

  트랜잭션 명으로 다음의 옵션을 사용할 수 있습니다.

  - `full` : Full Class 이름 사용
  - `class` : 서비스 명칭을 Class 이름으로 사용
  - `method` : 서비스 명칭을 Method 이름으로 사용
  - `string` : 서비스 명칭을 문자열 중에서 첫 번째 파라미터로 사용
  - `arg` : 파라미터 중에서 `service_name_index` 옵션에 설정한 인덱스에 파라미터를 서비스 명칭으로 사용

멀티 트랜잭션 추적

다음은 애플리케이션 간 호출을 추적하고 모니터링하기 위한 옵션입니다. 이 옵션을 통해 사용자는 여러 애플리케이션 간의 상호작용을 이해하는 데 도움이 받을 수 있습니다. 제공하는 다양한 옵션을 통해 멀티 트랜잭션 추적의 세부적인 동작을 조정할 수 있습니다.

- **mtrace_enabled** Boolean

  기본값 `true`

  멀티 트랜잭션 추적 기능(Multi Transaction ID, 이하 MTID) 사용 여부를 설정합니다. MTID를 추적하면 등록한 모든 애플리케이션 간의 호출을 확인할 수 있습니다.

- **mtrace_rate** Percentage

  기본값 `10`

  최초 트랜잭션이 발생할 때 발급하는 MTID의 발급 비율을 설정하는 옵션입니다.

- **mtrace_caller_key** String

  기본값 `x-wtap-mst`

  MTID 추적에 사용할 Caller Key Name을 설정합니다.

- **mtrace_callee_key** String

  기본값 `x-wtap-tx`

  MTID 추적에 사용할 Callee Key Name을 정합니다.

- **mtrace_send_url_length** Int

  기본값 `80`

  Http Caller는 Callee에게 자신의 URL을 넘겨줍니다. 이때 URL 길이를 제한하는데, 이 길이의 값을 설정합니다.

- **mtrace_callee_id_send_enabled** Boolean

  기본값 `false`

  MTID 추적 시 HTTPC 호출과 함께 발행된 Callee ID를 트레이스에 표현합니다.

- **mtrace_callee_id_recv_enabled** Boolean

  기본값 `false`

MTID 추적 시 수신한 Callee ID를 트레이스에 표현합니다.

- **mtrace_alltx_enabled** Boolean

  기본값 `false`

  log4j와 같은 로깅 시스템과 연계하기 위해서 모든 트랜잭션의 MTID를 추적할 수 있습니다.

- **mtrace_basetime** Miliseconds

  기본값 `100`

  `mtrace_alltx_enabled` 옵션의 값이 `true`이면 너무 많은 로그가 남을 수 있습니다. 이때는 트레이스 로깅량을 줄일 필요가 있습니다.

- **stat_mtrace_enabled** Boolean

  기본값 `false`

  Caller와 Callee의 상관관계 통계를 수집합니다. Caller에 적용하면 Caller의 상세 정보를 보내주고 Callee에 적용하면 url 단위 Caller-Callee 호출 통계를 수집 서버로 전송합니다.

트레이스 데이터 샘플링

다음은 트레이스 데이터의 샘플링을 조절하고, 데이터의 전송 여부를 설정하는 옵션입니다. 이 옵션을 통해 사용자는 애플리케이션의 성능과 데이터 전송량을 관리할 수 있습니다.

에이전트 지원 범위: Java Agent v2.2.4 or later

- **trace_sampling_enabled** Boolean

  기본값 `false`

  트레이스 데이터 샘플링 수집 여부를 설정할 수 있습니다.

- **trace_sampling_tps** Int

  기본값 `10000`

  트레이스 데이터 샘플링 수를 설정할 수 있습니다. 설정한 샘플링 수를 초과하는 데이터는 전송하지 않으며, 5초마다 초기화합니다.

- **trace_send_enabled** Boolean

  기본값 `true`

트레이스 데이터 전송 여부를 설정할 수 있습니다.

- **debug_trace_samling** Boolean

  기본값 `false`

  트레이스 데이터 샘플링 디버그 옵션을 켜거나 끌 수 있습니다.

트레이스 URL 이름 수정하기

- **hook_tx_name_patterns / hook_tx_name_mode** String Java Agent v2.2.4 or later

  특정 URL 및 특정 메소드(method)를 호출 시 트레이스 URL의 이름을 수정할 수 있습니다. 예를 들어 `io.home.test.TestController.test1` 에 등록한 URL 호출 시 `test100000()` 메소드를 호출하는 경우, 서비스 화면에서 `/api/100000+test1+test100000` 으로 확인할 수 있습니다.

  `hook_tx_name_mode` 옵션을 통해 **class**, **method**, **string**, **return** 중 하나를 선택할 수 있습니다.

  whatap.conf

```
# 호출 메소드의 전체 경로 (* 사용 가능)
hook_tx_name_patterns=ab.cd.ef.GH.ij

# class, method, string, return 중 택1
hook_tx_name_mode=method
```

특정 exception 무시하기

- **ignore_exception_tx_pattern** String Java Agent v2.2.4 or later

  특정 서비스에서 특정 exception을 무시할 수 있도록 설정할 수 있습니다. 여러 개를 등록할 때는 쉼표(,)를 구분자로 이용하세요.

  whatap.conf

```
ignore_exception_tx_pattern=exception:service_url
# e.g. java.util.concurrent.TimeoutException:/api/test/timeout,org.springframework.web.util.NestedServletException:/api/posts/test/cexception
```

HttpURLConnection 추적하기

다음은 `HttpURLConnection` 클래스를 사용하여 HTTP 호출을 추적하기 위한 옵션입니다.

- **HttpURLConnection** Boolean

  기본값 `true`

  `HttpURLConnection` 클래스의 메소드를 추적하기 위한 옵션입니다.

- **hook_HttpURLConnection_startup_enabled** Boolean Java Agent v2.2.4 or later

  기본값 `false`

  `HttpURLConnection` 클래스의 메소드를 최초 호출 시에도 추적할 수 있도록 설정할 수 있습니다.

- **HttpURLConnection_weblogic** Boolean Java Agent v2.2.4 or later

  기본값 `true`

  Weblogic에서 `HttpURLConnection`으로 http 호출을 하는 경우 `weblogic.net.http.HttpURLConnection` 클래스의 메소드를 추적하기 위한 옵션입니다.

- **hook_HttpURLConnection_weblogic_startup_enabled** Boolean Java Agent v2.2.4 or later

  기본값 `false`

  `weblogic.net.http.HttpURLConnection` 클래스의 메소드를 최초 호출 시에도 추적할 수 있도록 설정할 수 있습니다.

메모리 할당량 수집하기

- **trace_malloc_enabled** Boolean

  기본값 `false`

  트랜잭션별 평균 메모리 할당량을 수집합니다

  > ⓘ 해당 옵션의 값을 `true` 로 설정하여 활성화 시 **트레이스 분석** 창의 **레코드 요약** 탭에서 **메모리 할당량**을 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

DB/SQL 설정

데이터베이스 및 SQL 성능 데이터 수집을 위한 자바(Java) 에이전트의 다양한 옵션을 안내합니다. DBCP, Hikari, Tomcat 등 다양한 DB 연결 풀(Connection Pool) 정보 추적 및 SQL 실행 세부 사항을 기록할 수 있습니다. 또한, SQL 파라미터 정보 기록, DB 연결 누수 추적 등 성능 최적화에 필수적인 정보 수집 옵션을 제공합니다.

- **dbcp_pool_enabled** Boolean

  기본값 `true`

  JMX를 사용하지 않고 DBCP의 DB Connection 정보를 추적하기 위해 사용합니다.

- **hikari_pool_enabled** Boolean

  기본값 `false`

  JMX를 사용하지 않고 hikari pool의 DB Connection 정보를 추적하기 위해 사용합니다.

- **tomcat_ds_enabled** Boolean

  기본값 `false`

  JMX를 사용해 Tomcat DB Connection Pool 정보를 추적하는 기능을 활성화합니다.

- **tomcat_pool_enabled** Boolean

  기본값 `true`

  JMX를 사용하지 않고 Tomcat DB Connection Pool 정보를 추적하는 기능을 활성화합니다.

- **weblogic_ds_enabled** Boolean

  기본값 `false`

  JMX를 사용해 Weblogic DB Connection Pool 정보를 추적하는 기능을 활성화합니다.

- **weblogic_pool_enabled** Boolean

  기본값 `true`

  JMX를 사용하지 않고 Weblogic DB Connection Pool 정보를 추적하는 기능을 활성화합니다.

- **jeus_pool_enabled** Boolean

  기본값 `true`

  JMX를 사용하지 않고 JEUS DB Connection Pool 정보를 추적하는 기능을 활성화합니다.

- **profile_connection_open_enabled** Boolean

  기본값 `true`

  트레이스 내역에 DBConnection 오픈 정보를 기록합니다.

- **trace_dbc_close_enabled** Boolean Java Agent v2.2.3 or later

  기본값 `false`

  `profile_connection_open_enabled`의 값이 `true`인 경우에만 동작합니다. 트레이스 내역에 DBConnection 클로즈 정보를 기록합니다.

  > ⓘ **Java 에이전트 2.2.3 버전 이전**에서는 `profile_dbc_close` 옵션명을 사용하세요.

- **profile_sql_param_enabled** Boolean

  기본값 `false`

  트레이스 내역에 SQL 파라미터 정보를 기록할 때 사용합니다. 파라미터는 별도 보안 키를 입력해야 조회할 수 있습니다.

  > ⓘ
  > - **Java 에이전트 2.2.2 버전 이전**: 보안 키는 WAS 서버 `${WHATAP_AGENT_HOME}` /paramkey.txt 파일 내에 6자리로 작성합니다. paramkey.txt 파일이 존재하지 않는 경우 랜덤 값으로 자동 생성합니다.
  > - **Java 에이전트 2.2.2 버전 이후**: 보안 키는 WAS 서버 `${WHATAP_AGENT_HOME}` /security.conf 파일 내에 `paramkey` 키값을 확인하세요. security.conf 파일이 존재하지 않을 경우 `paramkey` 키값을 **WHATAP**으로 자동 생성합니다.
  > - 보안키 설정 파일에 대한 자세한 내용은 다음 문서를 참조하세요.

- **profile_sql_resource_enabled** Boolean

  기본값 `false`

  트레이스에서 SQL을 수집할 때 해당 스텝에서 사용한 CPU와 메모리 사용량을 추적합니다.

- **profile_update_count** Boolean

  기본값 `false`

  `executeUpdate()` 메소드를 통해 SQL UPDATE 문을 수행한 경우 UPDATE 건수를 수집합니다.

- **custom_pool_classes** String

  pre-define되지 않는 별도의 Connection Pool을 사용하는 경우 해당 클래스 명을 설정합니다.

- **ds_update_interval** MiliSeconds

  기본값 5000

  DB Connection 정보 Count 간격을 설정합니다.

- **profile_position_sql** Boolean

  기본값 false

  SQL을 수행하는 시점의 StackTrace를 기록합니다.

- **profile_sql_param_length** Int

  기본값 40

  SQL 파라미터의 길이를 설정합니다.

  > 최대값은 128이며 그 이상의 값을 입력하더라도 파라미터는 128까지 저장합니다.

- **trace_dbc_leak_enabled** Boolean

  기본값 false

  DBConnection Leak을 추적하는 기능을 활성화합니다.

  > Connection Wrapper를 사용해 Leak을 추적하기에 운영 서비스에 영향을 미칠 수 있으므로 반드시 테스트 후 적용하세요.

- **trace_dbc_leak_fullstack_enabled** Boolean

  기본값 false

  DBConnection Leak이 감지되는 경우 해당 시점 StackTrace를 수집합니다.

  > 피크 타임(Peak Time)에는 가급적 적용하지 마세요. 옵션을 적용하면 CPU 사용량이 다소 증가할 수 있습니다. 문제 해결 용도로만 한시적으로 적용할 것을 권고합니다.

- **trace_sql_normalize_enabled** Boolean

  기본값 true

  SQL 문에서 리터럴 부분을 추출해 SQL 문을 정규화하는 기능을 활성화합니다.

- **profile_error_jdbc_fetch_max** Int

  기본값 `10000`

  SQL Fetch Count(`ResultSet.next()` 호출 건 수)가 설정한 값을 초과하면 **TOO_MANY_RECORDS** 에러로 처리합니다. `0` 으로 설정하면 에러 처리하지 않습니다.

- **profile_error_sql_time_max** Int

  기본값 `30000`

  SQL 수행 시간이 설정한 값을 초과하면 **SLOW_SQL** 에러로 처리합니다. `0` 으로 설정하는 경우 에러 처리를 하지 않습니다.

- **hook_connection_open_patterns** String

  DB Connection Open 시 호출하는 메소드를 등록합니다. 미리 설정하지 않은 Connection Pool의 getConnection을 등록하는 것이 일반적입니다.

  Example

  ```
  hook_connection_open_patterns=mypool.ConPool.getConnection
  ```

- **hook_jdbc_con_classes** String

  미등록한 JDBC Connection 클래스를 지정합니다.

  Example

  ```
  hook_jdbc_con_classes=mypool.ConPool
  ```

- **hook_jdbc_pstmt_classes** String

  미등록한 jdbc PreparedStatement 클래스를 설정합니다. 생성자 파라미터에 SQL 문자열이 전달되는 구조여야 한다는 것을 주의하세요.

  Example

  ```
  hook_jdbc_pstmt_classes=org.apache.derby.impl.jdbc.EmbedPreparedStatement
  ```

- **hook_jdbc_cstmt_classes** String

  미등록한 jdbc CallableStatement 클래스를 설정합니다.

  Example

```
hook_jdbc_cstmt_classes=org.apache.derby.impl.jdbc.EmbedCallableStatement
```

- **hook_jdbc_stmt_classes** String

  미등록한 JDBC Statement 클래스를 설정합니다.

  Example

```
hook_jdbc_stmt_classes=org.apache.derby.impl.jdbc.EmbedStatement
```

- **hook_jdbc_rs_classes** String

  미등록한 JDBC ResultSet 클래스를 설정합니다.

  Example

```
hook_jdbc_rs_classes=org.apache.derby.impl.jdbc.EmbedResultSet
```

- **hook_jdbc_wrapping_driver_patterns** String

  DB2 드라이버처럼 난독 처리한 JDBC 드라이버는 `hook_jdbc_xxx` 옵션으로 직접 BCI가 어렵습니다. 이런 경우 Wrapper 방식으로 SQL를 추적할 수 있습니다. 이때 `Driver.connect` 를 설정해 추적합니다.

- **debug_dbc_stack_enabled** Boolean

  기본값 `false`

  DB Connection 시점의 StackTrace를 트레이스에 저장합니다. 애플리케이션에서 사용하는 Connection Pool 정보를 얻기 위해 사용합니다.

- **ignore_sql_hash_set** String Java Agent v2.2.4 or later

  해당 옵션에 입력한 SQL 해시값과 일치하는 SQL 문은 무시하도록 설정할 수 있습니다.

HTTPC/API Call 설정

HTTP 외부 호출과 API 호출 관련 데이터를 수집, 분석을 위한 자바(Java) 에이전트의 옵션 설정 방법을 안내합니다. **SLOW_HTTPC** 에러 처리 및 CPU와 메모리 사용량 추적, 호출 시점의 스택 트레이스 기록, URL 정규화 등 세밀한 모니터링을 위한 다양한 옵션을 제공합니다. 이를 통해 개발자는 애플리케이션의 외부 의존성을 효율적으로 관리하고 성능 문제를 식별할 수 있습니다.

- **profile_error_httpc_time_max** Int

  기본값 `10000`

  HTTPC 수행 시간이 지정한 값을 초과하면 **SLOW_HTTPC** 에러로 처리합니다. `0` 으로 설정하면 에러 처리하지 않습니다.

- **profile_httpc_resource_enabled** Boolean

  기본값 `false`

  트레이스에서 HTTP Call 스텝을 수집할 때 해당 스텝에서 사용한 CPU와 메모리 사용량을 추적합니다.

- **profile_position_httpc** Boolean

  기본값 `false`

  HTTPC가 수행하는 시점의 StackTrace를 기록합니다.

- **trace_httpc_normalize_enabled** Boolean

  기본값 `true`

  트랜잭션 내 HTTPC URL을 파싱해 정규화하는 기능을 활성화합니다.

- **trace_httpc_normalize_urls** String

  정규화할 HTTPC URL 패턴을 설정합니다. 호출 URL 패턴을 파싱해 패스 파라미터를 제거합니다.

  > ⓘ 예시, `/a/{v}/b` 라고 선언하면 `a/123/b` → `a/{v}/b` 로 치환합니다. 여러 개를 등록할 때는 쉼표(,)를 구분자로 사용합니다. 치환 패턴 정리 후 보완이 필요합니다.

- **hook_httpc_patterns** String

  HTTP outbound 호출을 수행하는 `full package 클래스명` . `메소드` 를 설정합니다. HTTP Call을 수행하는 메소드(method)의 전체 경로(full path)를 등록해 사용합니다. 등록 후 클래스 재정의(class redefine)를 하거나 에이전트를 재시작해야 합니다.

whatap.conf

```
hook_httpc_patterns=io.home.test.baseapp.app.post.service.HookHttpcService.*
```

- 클래스를 여러 개 등록하려면 쉼표(,)를 구분자로 사용하세요.
- 와일드 카드는 별표(*)만 사용할 수 있으며 정규식은 사용할 수 없습니다.

스크립트 플러그인

Java 에이전트에서 제공하는 플러그인 옵션을 통해 사용자가 원하는 코드를 트레이스 데이터에 주입하거나 메소드 수행 전후로 부가 정보를 추가하는 방법을 제공합니다. 메소드 시작/종료 시점에 특정 코드를 실행할 위치를 설정하는 옵션과 사용자 정의 pool을 모니터링하기 위한 클래스 설정 방법을 포함합니다. 실제 플러그인 적용 사례를 통해 구체적인 활용 방안을 제시하며, 모니터링의 유연성과 세밀함을 높일 수 있는 다양한 설정 예시와 API 사용법을 안내합니다.

플러그인 옵션

Java 에이전트 플러그인은 메소드 시작/종료 부분에 삽입해 실행합니다. 플러그인을 적용할 위치(클래스, 메소드)는 에이전트 옵션으로 설정할 수 있습니다.

- `hook_trace_helper_start_patterns`

  메소드 시작 부분에 트레이스 플러그인을 삽입할 지점(클래스, 메소드)을 설정합니다. 플러그인 코드는 `$WHATAP_HOME` /plugin/TraceHelperStart.x 파일에 작성하세요.

- `hook_trace_helper_end_patterns`

  메소드 종료 부분에 트레이스 플러그인을 삽입할 지점(클래스, 메소드)를 설정합니다. 플러그인 코드는 `$WHATAP_HOME` /plugin/TraceHelperEnd.x 파일에 작성하세요.

- `hook_trace_helper_patterns`

  메소드 시작/종료 양쪽에 트레이스 플러그인을 삽입할 지점(클래스, 메소드)을 설정합니다. 플러그인 코드는 `$WHATAP_HOME` /plugin/TraceHelperStart.x, `$WHATAP_HOME` /plugin/TraceHelperEnd.x 파일에 각각 작성하세요.

- `custom_pool_classes`

  사용자 정의 pool을 모니터링 하기 위해 pool 사용량 정보를 가진 클래스를 설정합니다. 플러그인 코드는 `$/WHATAP_HOME` /plugin/CustomPool.x에 작성합니다.

플러그인 설정 예시

여러 개의 클래스를 설정하는 경우 쉼표(,)를 구분자로 이용하세요. 패키지 이름의 문자열 일부 혹은 전부를 '*'로 치환할 수 있습니다.

- **패키지와 메소드 이름을 설정하는 경우**

```
whatap.bytecode.instrument.PluginTestA.testA,whatap.bytecode.instrument.PluginTestB.testB
```

- **패키지와 메소드 이름을 `*`로 치환하는 경우**

```
*PluginTestA.testA,whatap.bytecode.instrument.PluginTestB.*
```

```
*.testA, *PluginTestB.testB
```

- **전체를 대상으로 하는 경우**

```
*.*
```

- **Custom Pool을 설정하는 경우**

  Custom Pool은 식별자@패키지 명 형태로 설정합니다.

  예시, Class 명 앞에 `whatap_plugin_guide` 라는 식별자를 설정니다. 식별자와 클래스는 @로 구분합니다.

```
whatap_plugin_guide@com.ibm.ws.connectionpool.monitor.ConnectionPoolStats
```

플러그인 코드 작성 예시

대부분의 경우 오브젝트를 반환하므로 명시적 type casting 처리가 필요합니다. 다음은 플러그인 코드 작성 예시입니다.

설정한 메소드 시작 시간을 트레이스에 기록하기

1. Java 에이전트 `hook_trace_helper_patterns` 옵션에 적용할 메소드를 whatap.conf 파일에 작성하세요.

   $WHATAP_HOME/whatap.conf

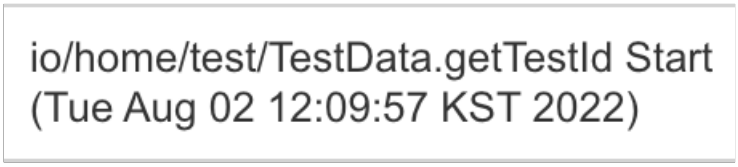
   ```
   hook_trace_helper_patterns=org.apache.catalina.connector.RequestFacade.*
   ```

2. TraceHelperStart.x 파일에 플러그인 코드를 작성하세요.

$WHATAP_HOME/plugin/TraceHelperStart.x

```
// String prefix에 클래스명, 메소드명을 대입하세요.
String prefix = $point.class1 + "." + $point.method;
// 와탭 트레이스 속성 명칭을 `prefix` + "st" 로 현재 시간을 설정하세요.
$ctx.setAttribute(prefix + "st", new Long(System.currentTimeMillis()));
// 트레이스 정보에 현재 시간을 추가하세요.
$ctx.profile(prefix + " Start", new java.util.Date().toString());
```

트레이스 내역에 메소드 시작시간을 다음과 같이 표시합니다.



설정한 메소드 종료시간과 수행시간을 트레이스에 기록하기

1. Java 에이전트 `hook_trace_helper_patterns` 옵션에 적용할 메소드를 whatap.conf 파일에 작성하세요.

   $WHATAP_HOME/whatap.conf

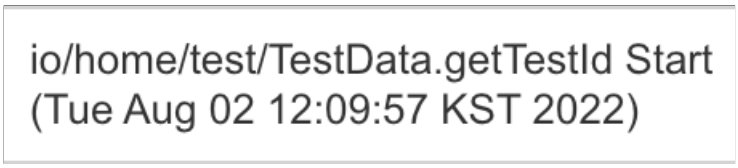
   ```
   hook_trace_helper_patterns=org.apache.catalina.connector.RequestFacade.*
   ```

2. TraceHelperEnd.x 파일에 플러그인 코드를 작성하세요.

   $WHATAP_HOME/plugin/TraceHelperEnd.x

   ```
   1  String prefix = $point.class1 + "." + $point.method;
   2  // TraceHelperStart.x 파일에서 추가한 트레이스 속성을 가져오세요.
   3  long st = ((Long) $ctx.getAttribute(prefix + "st")).longValue();
   4  long gap = System.currentTimeMillis() - st;
   5  StringBuilder sb = new StringBuilder();
   6  sb.append(new java.util.Date().toString() + " (Gap:" + gap + " milliseconds)");
   7  // 트레이스 정보에 시작 시간과의 현재 시간과의 Gap(수행시간)을 추가하세요.
   8  $ctx.profile(prefix + " End", sb.toString());
   ```

트레이스 내역에 메소드 종료시간과 수행시간을 다음과 같이 표시합니다.



플러그인 API

플러그인 API는 공통 항목인 `$ctx` , `$point` , `$req` , `$res` 로 구분합니다. 트랜잭션 시작/종료시, HTTPC 구간, 특정 메소드 실행 구간 등에 적용할 수 있습니다.

> 플러그인 API를 사용해 개인식별정보 저장을 금지하며, 개인정보 수집 시 해당 내용을 개인정보처리방침에 명시해야합니다.

파라미터 제공 - 공통 플러그인 파일

파라미터	메소드/변수	설명
$ctx	속성관리	
	void setAttribute(String key, Object value)	Attribute를 설정합니다.
	Object getAttribute(String key)	Attribute의 값을 가져옵니다.
	트레이스 데이터	
	void profile(String desc)	트레이스 메시지 기록
	void profile(String name, String msg)	트레이스 Step명, 메시지 기록
	void profile(String name, String msg, int elapsed)	트레이스 Step명, 메시지, 수행시간 기록
	void profile(String name, String msg, int elapsed, int value)	트레이스 Step명, 메시지, 수행시간, 순서 기록

파라미터	메소드/변수	설명
	HTTP Request	
	String service()	서비스 이름 반환
	void service(String name)	서비스 이름 설정
	int serviceHash()	서비스이름의 Hash값 반환
	String remoteIp()	서비스 요청 IP 반환
	void remoteIp(String ip)	서비스 요청 IP 설정
	boolean isError()	에러 여부 반환
	void login(String id)	로그인 ID 설정
	String login()	로그인 ID 반환
	String httpMethod()	HTTP Method 반환
	String httpQuery()	HTTP Query 반환
	String httpContentType()	HTTP ContentType 반환
	String userAgent()	User-Agent 반환
	int status()	HTTP status 반환
$point	String class1	클래스명
	String method	메소드명
	Object this1	Hooking 대상 클래스/메소드
	Object[] args	인자

파라미터	메소드/변수	설명
	Object return1	리턴

파라미터 미제공 - 공통 플러그인 파일

파라미터	메소드/변수	설명
없음	void log(Object c)	Logger를 통한 Logging
	void println(Object c)	System.out.println() 을 통한 출력
	Object field(Object o, String field)	Field 값 반환
	Object method(Object o, String method)	Invoke 메소드
	Object method(Object o, String method, String param)	Invoke 메소드
	String toString(Object o)	Object를 toSting() 반환
	String toString(Object o, String def)	Object를 toSting() 반환, null 인경우 def 반환
	int syshash(Object o)	hash 값 반환
	int syshash(HookArgs hook, int x)	x 번째 argument의 hash 값 반환
	int syshash(HookArgs hook)	argument의 hash 값 반환
	int cint(Object o)	int로 반환
	float cfloat(Object o)	float 으로 반환
	String cString(Object o)	String 으로 반환
	long clong(Object o)	long으로 반환

파라미터	메소드/변수	설명
	double cdouble(Object o)	double로 반환
	String desc(Object o)	Class signature 반환
	String toJson(Object o)	json으로 반환
	void shell(final String cmd, final String env)	shell 실행

데몬 및 배치

데몬, 배치와 같은 애플리케이션 서버를 모니터링할 때 시작점으로 설정한 메소드에 적용하는 API입니다. 다음 파일에 코드를 작성하세요.

- 서비스 시작 부분 적용: `$WHATAP_HOME` /plugin/AppServiceStart.x
- 서비스 종료 부분 적용: `$WHATAP_HOME` /plugin/AppServiceEnd.x

플러그인 파일	파라미터	메소드/변수	설명
AppServiceStart.x	$ctx	공통 속성 참조	
	$point	공통 속성 참조	
AppServiceEnd.x	$ctx	공통 속성 참조	

HTTP 서비스

`javax.servlet.http.HTTPServlet` 클래스를 사용하는 일반적인 웹 애플리케이션에 적용할 수 있는 API입니다. 다음 파일에 코드를 작성하세요.

- 서비스 시작 부분 적용: `$WHATAP_HOME` /plugin/HttpServiceStart.x
- 서비스 종료 부분 적용: `$WHATAP_HOME` /plugin/HttpServiceEnd.x

플러그인 파일	파라미터	메소드/변수	설명
HttpServiceStart.x/ HttpServiceEnd.x	$ctx	공통 속성 참조	
	$req	String getCookie(String key)	Cookie 값 반환
		String getRequestURI()	RequestURI 반환
		String getRemoteAddr()	RemoteAddr 반환
		String getMethod()	HTTP Method 반환
		String getQueryString()	HTTP QueryString 반환
		String getParameter(String key)	Parameter 값 반환
		Object getAttribute(String key)	Attribute 값 반환
		String getHeader(String key)	Header 값 반환
		Enumeration getParameterNames()	getParameterNames 반환
		Enumeration getHeaderNames()	getHeaderNames 반환
		WrSession getSession()	Session Wrapper 반환
		Set getSessionNames()	getSessionNames 반환
		Object getSessionAttribute(String key)	getSessionAttribute 반환
		boolean isOk()	Plugin 상태 반환
		Throwable error()	Error 반환
	$res	String getContentType()	ContentType 반환
		String getCharacterEncoding()	CharacterEncoding 반환

플러그인 파일	파라미터	메소드/변수	설명
		boolean isOk()	Plugin 상태 반환
		Throwable error()	Error 반환

HTTP Outbound

`HttpClient`와 같은 라이브러리를 이용해 HTTP Outbound Call 수행 할 때 적용할 수 있는 API입니다. 다음 파일에 코드를 작성하세요.

- 서비스 시작 부분 적용: `$WHATAP_HOME`/plugin/HttpCallStart.x
- 서비스 종료 부분 적용: `$WHATAP_HOME`/plugin/HttpCallEnd.x

플러그인 파일	파라미터	메소드/변수	설명
HttpCallStart.x	$ctx	공통 속성 참조	
	$req	String url()	URL 반환
		String host()	Hostname 반환
		int port()	Port 반환
		boolean isOk()	Plugin 상태 반환
		Throwable error()	Error 반환
HttpCallEnd.x	$ctx	공통 속성 참조	
	$res	String getContentType()	ContentType 반환
		String getCharacterEncoding()	CharacterEncoding 반환
		boolean isOk()	Plugin 상태 반환
		Throwable error()	Error 반환

특정 메소드

자바 에이전트 옵션(`hook_trace_helper_*`)을 통해 설정한 메소드에 대해 적용하는 API 입니다.

플러그인 파일	파라미터	메소드/변수	설명
TraceHelperStart.x	$ctx	공통 속성 참조	
	$point	공통 속성 참조	
TraceHelperEnd.x	$ctx	공통 속성 참조	
	$point	공통 속성 참조	

사용자 정의 Pool

사용자 정의 Pool을 모니터링 하기 위한 API입니다. Pool 사용량 정보를 가진 메소드를 설정하면 와탭 모니터링 서비스에서 관찰하고 통계 정보를 확인할 수 있습니다.

플러그인 파일	파라미터	메소드/변수	설명
CustomPool.x	$id	custom_pool_classes에 설정한 id 값	
	$pool	custom_pool_classes에 설정한 class	
	$result	active(Object o)	Active Pool Count 설정
		int idle(Object o)	Idle Pool Count 설정

플러그인 적용 사례

다음은 실제 플러그인 적용 사례입니다.

Elasticsearch 엔진의 검색 요청 모니터링

Elasticsearch(이하 ES) 엔진은 서블릿 엔진이 아니기 때문에 비정형 모니터링을 해야 합니다. 타제품이 지표 중심의 모니터링을 한다면 와탭의 플러그인을 활용해 ES의 요청 및 처리시간, 검색 키워드를 트레이싱할 수 있습니다.

whatap.conf

```
# 트랜잭션 EndPoint로 org.elasticsearch.search.SearchService.executeQueryPhase를 설정합니다.
hook_service_patterns=org.elasticsearch.search.SearchService.executeQueryPhase

# hook_trace_helper_start_patterns에 등록된 메소드가 실행될 때마다 TraceHelperStart.x 파일의 코드를 실행합니다.
hook_trace_helper_start_patterns=org.elasticsearch.search.query.QueryPhase.execute
```

${WHATAP_HOME}/plugin/TraceHelperStart.x

```
1  if ($ctx.inner() == null) {
2      return;
3  }
4
5  try {
6      String tclass = "org.elasticsearch.search.query.QueryPhase";
7      String tmethod = "execute";
8
9      if (tclass.equals($point.class1) && tmethod.equals($point.method)) {
10         // 첫번째 argument의 query 메소드 호출 결과를 String query에 저장,
11         // org.elasticsearch.search.query.QueryPhase.execute(SearchContext searchContext)입니다.
12         // 즉, SearchContext.query() 메소드를 invoke 한 결과를 String query에 저장합니다.
13         String query = " " + method($point.args[0], "query");
14         // String query를 트레이스 정보로 출력
15         $ctx.profile(query);
16     }
17 } catch(Exception e) {
18     $ctx.profile(e.toString());
19 }
```

특정 IP로 유입되는 트랜잭션 수집 제외

Health Check, 내부 사용자들이 유발한 트랜잭션을 수집 제외하려면 플러그인을 활용할 수 있습니다.

${WHATAP_HOME}/plugin/HttpServiceStart.x

```
// ignoreIP 값을 수집 제외할 IP로 설정하세요.
String ignoreIP = "123.234.123.234";
String remoteIP = $req.getRemoteAddr();

if ($ctx.ok() && remoteIP != null) {
    if (remoteIP.equals(ignoreIP)) {
        $ctx.ignore();
    }
}
```

Atomikos Pool 모니터링

API 문서를 참고해 적용하세요.

whatap.conf

```
custom_pool_classes=atomikos@com.atomikos.jdbc.internal.AbstractDataSourceBean
```

${WHATAP_HOME}/plugin/CustomPool.x

```
int total = cint(method($pool,"poolTotalSize"));
int avail = cint(method($pool,"poolAvailableSize"));

$result.active(total - avail);
$result.idle(avail);
```

트레이스에 쿠키값 표현

쿠키값을 트레이스에서 확인할 수 있습니다.

${WHATAP_HOME}/plugin/HttpServiceStart.x

```
// $req.getCookie() 파라미터로 확인하려는 쿠키를 설정하세요.
String cookie = $req.getCookie("AWSALB");

if ($ctx.ok()) {
    if (cookie != null) {
```

```
6         $ctx.profile(cookie);
7     }
8 }
```

사용자 수 설정

자바(Java) 웹 애플리케이션 서버에 연결된 사용자 수를 에이전트 설정을 통해 집계하는 방법을 안내합니다. 실시간 사용자 집계 활성화 여부 설정, 사용자 수를 집계하기 위한 쿠키 제한 설정, IP 주소 또는 HTTP 헤더 특정 값을 기준으로 한 사용자 집계 방법, 클라이언트 IP 정보 추적을 위한 HTTP 헤더 설정 등을 포함합니다.

- **trace_user_enabled** Boolean

  기본값 `true`

  실시간 사용자 집계 여부를 설정합니다. 사용자 추적 옵션이 중복 설정된 경우 동작 우선 순위는 다음과 같습니다.

  1. **trace_user_using_ip / wclient_using_ip**
  2. **trace_user_using_jsession**
  3. **user_header_ticket / wclient_header_ticket**

  > ⓘ Java 에이전트 2.2.0 버전 이상의 경우 `wclient_using_ip` , `wclient_header_ticket` 옵션을 이용하세요.

- **trace_user_cookie_limit / wclient_cookie_limit** Int

  기본값 `2048`

  사용자 집계를 위해 쿠키를 발행하는 경우 기존 쿠키가 너무 많다면 쿠키 오버플로어가 발생할 수 있습니다. 이를 방지하기 위해 limit을 설정합니다.

  > ⓘ Java 에이전트 2.2.0 버전 이상의 경우 `wclient_cookie_limit` 옵션을 이용하세요.

- **trace_user_using_ip / wclient_using_ip** Boolean

  기본값 `true`

  실시간 사용자를 IP 주소 기반으로 집계합니다. IP 주소가 아닌 쿠키를 기반으로 사용자를 구분하고 싶으면 값을 `false` 로 변경하세요.

  > ⓘ Java 에이전트 2.2.0 버전 이상의 경우 `wclient_using_ip` 옵션을 이용하세요.

- **user_header_ticket / wclient_header_ticket** String

HTTP 헤더의 특정 값으로 사용자 수를 집계하고자 하는 경우 해당 Key 값을 설정합니다. 모바일 클라이언트인 경우 사용자 구분을 위해 header를 사용한다면 다음 옵션을 이용할 수 있습니다.

whatap.conf

```
user_header_ticket=login
```

> ⓘ Java 에이전트 2.2.0 버전 이상의 경우 `wclient_header_ticket` 옵션을 이용하세요.

- **trace_http_client_ip_header_key** String

  클라이언트 IP(Remote IP)의 정보를 특정 HTTP 헤더의 값으로 변경해 설정하는 기능입니다. 프록시(Proxy) 환경에서 `X-Forwarded-For` 헤더 값을 클라이언트 IP로 설정할 수 있습니다.

  WEB/WAS 앞에 L4와 같은 로드밸런서가 위치한 경우 클라이언트의 IP 주소가 아닌 L4의 IP 주소가 Remote Address가 되는 경우가 있습니다. 이 상황에서 실제 클라이언트 IP 정보가 http 헤더에 특정 키 값으로 기록되는 경우라면 해당 키 값으로 대체할 수 있습니다.

- **trace_user_using_jsession** Boolean

  기본값 `false`

  실시간 사용자를 SESSIONID 기반으로 집계합니다.

부하량 제어 설정

자바(Java) 애플리케이션 서버의 트래픽 쓰로틀링 옵션을 설정하는 에이전트 옵션을 안내합니다. 애플리케이션의 최대 동시 요청 수를 제한하고, 특정 사용자나 URL을 기준으로 서비스 접근을 제어하는 기능을 포함합니다. 사용자 정의 메시지 전송, URL 리다이렉션 설정, 이벤트 알림 활성화 등 다양한 옵션을 제공하여 과부하 상황에서 애플리케이션의 안정성을 보장할 수 있습니다.

> ⓘ Java 에이전트의 쓰로틀링 옵션은 Servlet 기반 애플리케이션에만 적용됩니다. 이 옵션을 지원하는 서버는 Tomcat, Weblogic, Undertow 등의 Servlet이며, 최신 버전의 Spring은 해당 옵션을 지원하지 않습니다.

- **throttle_enabled** Boolean

  기본값 `false`

  애플리케이션의 최대 동시 처리 수를 제한하는 쓰로틀링 기능을 활성화합니다. throttle_으로 시작하는 모든 옵션은 `throttle_enabled=true` 상태에서만 동작합니다.

  쓰로틀링 제어 관련 정책의 우선 순위는 다음과 같습니다.

  1. Block: URL, 사용자 IP 주소 기준으로 서비스를 차단하며 가장 우선 적용합니다.
  2. Passing: Passing에 적용된 URL들은 Reject 정책보다 우선합니다.
  3. Reject: Block, Passing 정책 이후에 Reject 정책을 적용합니다.

- **throttle_limit** Int

  기본값 `10000`

  에이전트 별 동시 처리하는 요청(트랜잭션) 수가 설정한 값을 초과하면 추가로 도달하는 요청은 reject 합니다.

- **throttle_rejected_message** String

  기본값 `too many request!!`

  쓰로틀링을 제한할 경우 사용자에게 전달할 메시지를 설정합니다.

- **throttle_rejected_forward** String

  사용자 요청이 limit 값을 초과해 reject 될 때 사용자에게 전달하는 안내 페이지 URL을 설정합니다.
  `throttle_rejected_message` 를 동시 설정한 경우 `throttle_rejected_forward` 를 우선 적용합니다.

> 안내 페이지를 동일한 컨테이너에서 서비스한다면, 이 역시 동시 처리 수로 산정하므로 재귀 호출로 인한 장애 요소가 될 수 있습니다. 그러므로 안내 페이지는 static html 페이지로 만들거나 외부에 있어야 합니다.

- **reject_event_enabled** Boolean

  기본값 `false`

  사용자 요청을 Reject할 경우 이벤트 알림 발생 여부를 설정합니다.

- **reject_event_interval** MiliSecond

  기본값 `30000`

  Reject에 대한 이벤트 알림 발생 후 설정한 시간 동안 중복 이벤트에 대한 알림은 발생하지 않습니다.

- **throttle_blocking_url** String

  `throttle_limit`을 초과하지 않는 경우라도 블록킹(처리 거부)할 URL을 설정합니다. 시스템 장애를 유발하는 URL을 긴급하게 블록킹하기 위해 사용할 수 있습니다.

- **throttle_blocking_ip** ip_address

  사용자 IP 주소를 기준으로 블록킹하고자 할 때 설정합니다. 디도스 공격이나 잘못된 사용자를 IP 주소 기반으로 차단할 때 사용할 수 있습니다. 여러 개인 경우 쉼표(,)를 구분자로 사용하세요.

- **throttle_target_urls** String

  등록한 URL을 대상으로 쓰로틀링 기능을 적용합니다. 여러 개인 경우 쉼표(,)를 구분자로 사용하세요.

- **throttle_passing_url** String

  `throttle_limit`을 초과하는 경우라도 처리해야 하는 URL을 설정합니다. 여러 개인 경우 쉼표(,)를 구분자로 사용하세요.

- **throttle_passing_url_prefix** String

  `throttle_limit`을 초과하는 경우라도 처리해야 하는 URL들을 prefix로 설정합니다. 여러 개인 경우 쉼표(,)를 구분자로 사용하세요.

- **throttle_blocked_message** String

  기본값 `request blocked!!`

  요청을 블록킹한 사용자에게 전달할 메시지를 설정합니다.

- **throttle_blocked_forward** String

요청을 블록킹한 사용자에게 전달할 URL을 설정합니다. `throttle_blocked_message` 를 동시 설정한 경우 `throttle_blocked_forward` 를 우선 적용합니다.

에이전트 알림 설정

자바(Java) 에이전트를 통해 애플리케이션 서버에서 발생하는 다양한 이벤트에 대한 알림 설정 방법을 제공합니다. 트랜잭션 재귀 호출, 서비스 거절, HTTPC 연결 오류, 힙 및 디스크 사용량 초과, CPU 사용량 임계치 도달, DB 커넥션 중복 할당 및 예외 발생 시 이벤트 알림을 설정하는 옵션을 포함합니다. 각 이벤트별로 발행 간격, 발행 여부, 임계치 설정 등 세밀한 조정을 할 수 있습니다.

- **recursive_event_interval** MiliSeconds

  기본값 `300000`

  트랜잭션의 재귀 호출에 대한 이벤트 알림 발행 간격을 설정합니다.

- **reject_event_enabled** Boolean

  기본값 `false`

  서비스 거절(호출 부하 제한/거절) 시 이벤트 알림 발행 여부를 설정합니다.

- **reject_event_interval** MiliSeconds

  기본값 `300000`

  서비스 거절(호출 부하 제한/거절) 시 이벤트 알림 발행 간격을 설정합니다.

- **httpc_event_enabled** Boolean

  기본값 `false`

  HTTPC 연결 오류 발생 시 이벤트 알림 발행 여부를 설정합니다.

- **httpc_event_interval** MiliSeconds

  기본값 `300000`

  HTTPC 연결 오류 발생 시 이벤트 알림 발행 간격을 설정합니다.

- **heap_event_enabled** Boolean

  기본값 `false`

  힙 사용량 임계 도달 시 이벤트 알림 발행 여부를 설정합니다.

- **heap_event_percent** Percentage

  기본값 `90`

  힙 사용량 이벤트 알림 발행 기준 임계치를 설정합니다.

- **heap_event_duration** MiliSeconds

  기본값 `30000`

  힙 사용량 이벤트 알림 발행 기준 지속 시간을 설정합니다.

- **heap_event_interval** MiliSeconds

  기본값 `300000`

  힙 사용량 이벤트 알림 발행 간격을 설정합니다.

- **heap_event_action** String

  기본값 `NONE`

  힙 사용량 이벤트 발생 시 실행할 동적 로딩 코드를 설정합니다.

  > ⓘ `$WHATAP_HOME` /plugin/ActionScript.x 파일에 작성한 Java 코드에 전달할 ID (`$id` 로 전달함)

- **disk_event_enabled** Boolean

  기본값 `false`

  디스크 사용량 임계치 도달 시 이벤트 알림 발행 여부를 설정합니다.

- **disk_event_percent** Percentage

  기본값 `90`

  디스크 사용량 이벤트 알림 발행 기준 임계치를 설정합니다.

- **disk_event_interval** MiliSeconds

  기본값 `300000`

  디스크 사용량 이벤트 알림 발행 간격을 설정합니다.

- **disk_event_action** String

  디스크 사용량 이벤트 발생 시 실행할 동적 로딩 코드를 설정합니다.

  > ⓘ `$WHATAP_HOME` /plugin/ActionScript.x 파일에 작성한 Java 코드에 전달할 ID (`$id` 로 전달함)

- **cpu_event_enabled** Boolean

기본값 `false`

CPU 사용량 임계 도달 시 이벤트 알림 발행 여부를 설정합니다.

- **cpu_event_percent** Percentage

  기본값 `90`

  CPU 사용량 이벤트 알림 발행 기준 임계치를 설정합니다.

- **cpu_event_duration** MiliSeconds

  기본값 `30000`

  CPU 사용량 이벤트 알림 발행 기준 지속시간을 설정합니다.

- **cpu_event_interval** MiliSeconds

  기본값 `300000`

  CPU 사용량 이벤트 알림 발행 간격을 설정합니다.

- **cpu_event_action** String

  CPU 사용량 이벤트 발생 시, 실행할 동적 로딩 코드에 전달할 ID를 설정합니다.

  > ⓘ `$WHATAP_HOME`/plugin/ActionScript.x 파일에 작성한 Java 코드에 전달할 ID (`$id` 로 전달함)

- **dbc_dup_event_enabled** Boolean

  기본값 `false`

  DB Connection이 중복 할당될 경우 이벤트 알림 발행 여부를 설정합니다.

- **dbc_dup_event_fullstack_enabled** Boolean

  기본값 `false`

  DB Connection이 중복 할당될 때 Stack 확보 여부를 설정합니다.

  > ⓘ `$WHATAP_HOME`/plugin/ActionScript.x 파일에 작성한 Java 코드에 전달할 ID (`$id` 로 전달함)

Apdex 설정

사용자 만족도를 측정하는 Apdex 점수를 설정하는 자바(Java) 에이전트 옵션을 안내합니다. Apdex 지표를 통해 애플리케이션의 성능을 객관적으로 평가하고 사용자 경험을 개선할 수 있습니다.

whatap.apdex_time millisecond

기본값 `1200`

만족 S 기본값은 에이전트 설정 메뉴에서 변경할 수 있습니다.

Application Performance Index(Apdex)는 애플리케이션 성능 지표입니다. 웹 애플리케이션의 고객 만족도를 측정합니다. 사용자 만족도에 대한 지표로 활용할 수 있으며, 0 ~ 1 사이의 값을 갖습니다.

(만족 횟수 + (허용 횟수 * 0.5)) / 전체 요청 수

- 만족 (Satisfied, S): 업무처리에 전혀 문제 없음 ≤ 1.2초 (만족 S 기본값)
- 허용 (Tolerating, T): 사용자가 지연을 느끼나 업무처리는 가능 ≤ 4.8초 (만족 S * 4)
- 불만 (Frustrated, F): 업무처리가 불가능 > 4.8초 (허용 T 초과 및 오류)

> ⓘ 에이전트 2.0 버전 이상부터 지원합니다.

통계 설정

자바(Java) 애플리케이션 모니터링을 위한 다양한 통계 수집 기능 관련 에이전트 옵션을 안내합니다. 성능 카운터 확장, 도메인별 트랜잭션 수집, 멀티 서버 트랜잭션 의존성 분석, 로그인 유형별 및 Referer 별 통계 수집 등을 포함합니다. 또한 SQL, HTTP Call, 오류 통계와 사용자 에이전트 정보 수집에 대한 최대 레코드 수 제한 설정도 확인할 수 있습니다.

- **addin_monitor_num** Int

  기본값 `0`

  성능 카운터 확장을 위해 추가 선언할 클래스 수를 설정합니다. `addin_monitor_num` 에 설정한 값만큼 `addin_monitor_x` 를 선언합니다.

- **addin_monitor_0** Int

  기본값 `0`

  whatap.conf

  ```
  addin_monitor_num=2
  addin_monitor_0=a.a.Class1
  addin_monitor_1=a.a.Class2
  ```

- **stat_domain_enabled** Boolean

  기본값 `false`

  클라이언트의 접속 도메인별 트랜잭션 통계 수집 기능을 활성화합니다.

- **stat_domain_max_count** Int

  기본값 `7000`

  하나의 JVM에서 5분 동안 수집할 도메인별 트랜잭션 통계의 최대 레코드 수입니다.

- **stat_mtrace_enabled** Boolean

  기본값 `false`

  멀티 서버 트랜잭션에서 Caller & Callee 간에 버전 별 의존성 통계를 수집합니다.

- **mtrace_spec** String

  기본값 `v1`

  현재 인스턴스의 애플리케이션 버전을 설정합니다. 임의의 문자열을 설정할 수 있습니다. 이 데이터는 호출 통계를 위해 사용합니다.

- **stat_mtrace_max_count** Int

  기본값 `7000`

  멀티 서버 트랜잭션에서 Caller & Callee 간에 버전별 의존성 통계의 최대 레코드 수입니다.

- **stat_login_enabled** Boolean

  기본값 `false`

  로그인별 트랜잭션 통계를 수집합니다.

- **stat_login_max_count** Int

  기본값 `7000`

  하나의 JVM에서 5분 동안 수집할 로그인 별 트랜잭션 통계의 최대 레코드 수입니다.

- **stat_referer_enabled** Boolean

  기본값 `false`

  Referer별 트랜잭션 통계를 수집합니다.

- **stat_referer_max_count** Int

  기본값 `7000`

  하나의 JVM에서 5분 동안 수집할 Referer 별 트랜잭션 통계의 최대 레코드 수입니다.

- **stat_tx_max_count** Int

  기본값 `5000`

  트랜잭션 통계 정보의 개수를 제한합니다. 5분 동안 수집해 서버에 전송하는 통계 정보의 최대 레코드 수를 제한합니다.

- **stat_sql_max_count** Int

  기본값 `5000`

  SQL 통계 정보의 개수를 제한합니다. 5분 동안 수집해 서버에 전송하는 통계 정보의 최대 레코드 수를 제한합니다.

- **stat_httpc_max_count** Int

기본값 5000

Http Call 통계 정보의 개수를 제한합니다. 5분 동안 수집해 서버에 전송하는 통계 정보의 최대 레코드 수를 제한합니다.

- **stat_error_max_count** Int

  기본값 1000

  Error 통계 정보의 개수를 제한합니다. 5분 동안 수집해 서버에 전송하는 통계 정보의 최대 레코드 수를 제한합니다.

- **stat_useragent_max_count** Int

  기본값 500

  User Agent 통계 정보의 개수를 제한합니다. 5분 동안 수집해 서버에 전송하는 통계 정보의 최대 레코드 수를 제한합니다.

트랜잭션 에러 스택 설정

자바(Java) 애플리케이션에서 트랜잭션 중 발생하는 에러를 감지하고 이에 대한 스택 트레이스를 표시하는 에이전트 설정 옵션을 안내합니다. 트랜잭션, 메소드, SQL, HTTP 호출, SOCKET 호출 등 다양한 영역에서 에러 정보를 수집하고, 에러 스택의 길이, 제목 길이 등을 설정하는 방법을 포함합니다. 에이전트 설정을 통해 에러 관리를 개선하고, 애플리케이션의 오류 진단을 보다 효율적으로 수행할 수 있도록 지원합니다.

호출 서비스의 `status_code` 와는 무관하며, 여러 개의 에러 발생 시 에러 스택은 첫번째 에러 스택만 표시합니다.

> ⓘ 이 문서에서 안내하는 에이전트 설정은 **Java 에이전트 2.2.15** 버전 이상에서 지원합니다.

에이전트 설정

전체 설정

whatap.conf

```
# 에러 스택 표시 여부
trace_exception_stack=false

# 에러 스택 길이의 기본값
trace_exception_depth=50

# 에러 스택 메시지 제목의 길이
trace_exception_title_len=128
```

메소드

whatap.conf

```
# 메소드 에러 스택 표시 여부
# trace_exception_stack 값을 따라가며 별도로 true/false 값으로 기능을 켜거나 끌 수 있습니다.
trace_method_error_stack=trace_exception_stack
```

```
# 메소드 에러 스택 길이
# trace_exception_depth 값을 따라가며 별도로 설정할 수 있습니다.
trace_method_error_depth=trace_exception_depth
```

SQL

whatap.conf

```
# SQL 에러 스택 표시 여부
# trace_exception_stack 값을 따라가며 별도로 true/false 값으로 기능을 켜거나 끌 수 있습니다.
trace_sql_error_stack=trace_exception_stack

# sql 에러 스택 길이
# trace_exception_depth 값을 따라가며 별도로 설정할 수 있습니다.
trace_sql_error_depth=trace_exception_depth
```

HTTP Call

whatap.conf

```
# http call 에러 스택 표시 여부
# trace_exception_stack 값을 따라가며 별도로 true/false 값으로 기능을 켜거나 끌 수 있습니다.
trace_httpc_error_stack=trace_exception_stack

# http call 에러 스택 길이
# trace_exception_depth 값을 따라가며 별도로 설정할 수 있습니다.
trace_httpc_error_depth=trace_exception_depth
```

SOCKET

whatap.conf

```
# socket 에러 스택 표시 여부
# trace_exception_stack 값을 따라가며 별도로 true/false 값으로 기능을 켜거나 끌 수 있습니다.
trace_socket_error_stack=trace_exception_stack
```

```
# socket 에러 스택 길이
# trace_exception_depth 값을 따라가며 별도로 설정할 수 있습니다.
trace_socket_error_depth=trace_exception_depth
```

트랜잭션

whatap.conf

```
# 트랜잭션 에러 스택 표시 여부
# trace_exception_stack 값을 따라가며 별도로 true/false값으로 기능을 켜거나 끌 수 있습니다.
trace_endtx_error_stack=trace_exception_stack

# 트랜잭션 에러 스택 길이
# trace_exception_depth 값을 따라가며 별도로 설정할 수 있습니다.
trace_endtx_error_depth=trace_exception_depth
```

에이전트 추가 옵션

자바(Java) 애플리케이션의 모니터링 수준을 사용자의 필요에 맞게 조정할 수 있는 추가적인 에이전트 설정 옵션을 안내합니다. SQL 파라미터 수집, HTTP 파라미터 및 헤더 정보 수집, 사용자 IP 정보 추출, 사용자 수 추적 방법 선택, SAP Function 추적 등을 포함합니다. 이 옵션들을 통해 보다 세밀한 데이터 수집 및 분석이 가능하며, 특히 보안이 중요한 환경에서 필요에 따라 옵션을 조정할 수 있습니다.

다음은 자주 사용하지만 보안 등의 이유로 에이전트 기본 설정에서 제외한 옵션입니다. 필요하다면 아래 옵션을 추가해 적용할 수 있습니다.

- **profile_sql_param_enabled** Boolean

  기본값 `true`

  SQL 파라미터를 수집합니다.

- **profile_http_parameter_enabled** Boolean

  기본값 `true`

  HTTP 파라미터를 수집합니다.

- **profile_http_header_enabled** Boolean

  기본값 `false`

  HTTP 헤더 정보를 수집합니다.

- **trace_http_client_ip_header_key** String

  기본값 `X-Forwarded-For`

  사용자 IP 정보를 HTTP 헤더의 키에서 추출합니다.

- **trace_user_method / wclient_cookie_limit** Boolean

  기본값 `cookie`

  사용자 수 추적을 쿠키 기반으로 합니다.

  > ⓘ Java 에이전트 2.2.0 버전 이상의 경우 `wclient_trace_type` 옵션을 이용하세요.

- **hook_httpc_sap3_enabled** Boolean

기본값 `true`

SAP JCO을 사용하는 경우 SAP Function을 추적합니다.

비동기 추적

자바(Java) 에이전트 설정을 통해 비동기 애플리케이션 활동을 추적하는 방법을 안내합니다. `CompletableFuture` 메소드 수집 방법과 사용자 애플리케이션에 해당 메소드를 hooking하는 방식을 포함합니다. 설정 예시와 함께 whatap.conf 파일에 추가해야 할 옵션을 제공하여 비동기 호출의 성능 모니터링을 강화할 수 있도록 지원합니다.

CompletableFuture

Java 에이전트 설정을 통해 CompletableFuture 메소드를 수집할 수 있는 방법을 안내합니다.

CompletableFuture 메서드 추가

사용자의 Java 애플리케이션 프로젝트에 에이전트로 hooking할 CompletableFuture 메소드를 추가하세요.

- 메서드 이름: `trace` 로 통일
- ReturnType: `Supplier` , `Consumer` , `Runnable` , `Future` , `Function`

```
package io.home.test.util;

import java.util.concurrent.Future;
import java.util.function.Consumer;
import java.util.function.Supplier;

public class W {
    public static <T> Supplier<T> trace(Supplier<T> f) {
        return f;
    }
    public static <T> Consumer<T> trace(Consumer<T> f) {
        return f;
    }
    public static <T> Runnable trace(Runnable f) {
        return f;
    }
    public static <T> Future<T> trace(Future<T> f) {
        return f;
    }
```

```
    public static <T, R> Function<T, R> trace(Function<T, R> f) {
    return f;
}
}
```

에이전트 설정 추가

에이전트 설정을 위해 whatap.conf 파일에 다음 옵션을 추가하세요.

whatap.conf

```
hook_completablefuture_patterns=io.home.test.util.W.*
```

> ⓘ
> - Context가 있는 경우 `trace` 메소드를 추가해 트랜잭션을 연결하세요.
> - Context가 없는 경우 `hook_service_patterns` 옵션으로 서비스를 시작해야 합니다.

사용 예시

- **원본**

```
1  public CompletableFuture<String> serviceATimeout() {
2      return CompletableFuture.supplyAsync(() -> {
3          RestTemplate restTemplate = new RestTemplate();
4          return restTemplate.getForObject("http://localhost:8081/api/serviceB/timeout", String.class);
5      });
6  }
```

- `io.home.test.util.W.trace()` 적용

```
1  // io.home.test.util.W.trace() 적용
2  public CompletableFuture<String> serviceATimeout() {
3      return CompletableFuture.supplyAsync(W.trace(() -> {
4          RestTemplate restTemplate = new RestTemplate();
5          return restTemplate.getForObject("http://localhost:8081/api/serviceB/timeout", String.class);
6      }));
```

```
7 }
```

Java 에이전트 업데이트

최신 버전 에이전트 다운로드

1. 와탭 모니터링 서비스로 이동해 로그인하세요.
2. 에이전트를 업데이트할 프로젝트를 선택하세요.
3. 왼쪽 메뉴에서 **관리** > **에이전트 업데이트**를 선택하세요.
4. **최신 버전** 또는 **최신 버전으로 모든 에이전트에 다운로드** 버튼을 클릭하세요.
5. 에이전트를 다운로드한 다음 애플리케이션 서버를 다시 시작하세요. 업데이트를 자동 적용합니다.

에이전트 업데이트 자동 설정

자바 에이전트를 업데이트하려면 서버를 재시작해야 합니다. 재시작 이전에 whatap.agent.jar 파일이 변경되면 애플리케이션 서버가 재시작되는 과정에서 안정적인 시스템 종료(shutdown)를 실패할 수 있습니다.

원칙적으로 자바 에이전트 업데이트를 위해 다음의 순서로 진행해야 합니다.

1. app shutdown
2. agent jar 변경
3. app start

위 과정을 보다 쉽게 처리하기 위해 와탭의 에이전트 jar 파일명에는 버전 번호를 포함합니다. 업데이트 과정에서 최신 버전을 자동 적용하도록 설정하는 것이 중요합니다.

```
########## WHATAP #############
WHATAP_HOME=/whatap
WHATAP_JAR=`ls ${WHATAP_HOME}/whatap.agent-*.jar | sort -V | tail -1`
JAVA_OPTS="${JAVA_OPTS} -javaagent:${WHATAP_JAR} "
########## WHATAP #############
```

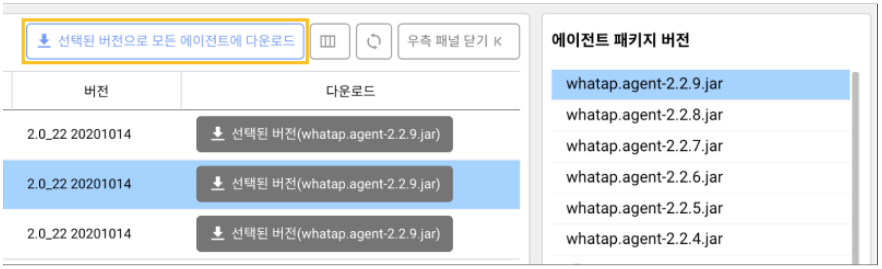
Java 에이전트 삭제

에이전트 삭제

애플리케이션 서버에서 JVM 옵션의 `-javaagent` 설정을 삭제한 다음 서버를 다시 시작하세요. 이후 `$WHATAP_HOME` 환경 변수를 삭제하세요.

에이전트 롤백

1. 와탭 모니터링 서비스에서 롤백을 원하는 프로젝트를 선택하세요.
2. 왼쪽 메뉴에서 **관리** > **에이전트 업데이트**로 이동하세요.
3. **다른 버전들** 버튼을 클릭하세요.
4. 롤백을 원하는 버전을 선택한 다음 **선택된 버전** 또는 **선택된 버전으로 모든 에이전트에 다운로드** 버튼을 클릭하세요.



5. 에이전트를 설치할 때 수정한 스크립트에서 `WHATAP_JAR` 환경 변수를 롤백하려는 버전으로 변경하세요.

```
########## WHATAP START ############
WHATAP_HOME=/Users/gyuseokjung/javatest/whatap
WHATAP_JAR=`ls ${WHATAP_HOME}/whatap.agent-2.0_25.jar`
JAVA_OPTS="${JAVA_OPTS} -javaagent:${WHATAP_JAR} "
########## WHATAP END ############
```

MSA 그룹

Auto Scale Out 환경에서 자동 등록된 에이전트의 IP / Port / PID 정보만을 가지고 업무 단위의 정보를 인지하기 어려운 경우가 있습니다. 이 경우 `okind` 속성을 JVM 옵션으로 부여해 쉽게 식별할 수 있습니다.

다음은 전형적인 Docker 환경에 대한 설명입니다.

- 각 에이전트는 잘게 쪼개어 놓은 MSA (Micro Service Architecture) 구성 요소입니다.
- 에이전트들은 자동으로 Scale In/Out 됩니다.
- 어떤 IP를 가진 서버에서 어떤 PID로 애플리케이션이 시작할지 예측할 수 없습니다.

위와 같은 Docker 환경에서 에이전트 이름과 더불어 업무를 식별할 수 있는 JVM 옵션으로 `whatap.okind` 를 추가해 `okind` 속성을 추가할 수 있습니다.

JAVA_OPTS

```
-Dwhatap.okind=mobile_ui
```

whatap.conf 파일에 설정을 추가할 수도 있습니다.

whatap.conf

```
whatap.okind=common_api
```

> MSA 환경에서 와탭이 제공하는 다양한 기능을 활용하려면 `okind` 속성을 추가하세요.

AES 256 암호화

와탭 애플리케이션 에이전트는 수집한 데이터를 암호화해 서버로 전송합니다. 이를 위해 XOR(Exclusive-OR) 연산과 고급 암호화 표준(Advanced Encryption Standard, 이하 AES) 암호화를 사용하며, 평문을 128비트 단위로 나누어 암복호화를 수행합니다. 256비트까지 확장해 보안을 강화할 수 있습니다.

기본적으로 Java Cryptography Extension(이하 JCE)은 128비트를 지원합니다. AES 256비트를 적용하려면 JCE를 업데이트해야 합니다.

기본 환경에서 AES 256 적용 시 다음과 같이 오류가 발생합니다.

```
Unsupported keysize or algorithm parameters.
##혹은,
Illegal key size or default parameters.
```

1. 다음 링크에서 버전에 맞는 파일을 다운로드하세요.

   > ⓘ **자바 버전 별 JCE 다운로드 링크**
   >
   > - JAVA 8
   > - JAVA 7
   > - JAVA 6
   > - JAVA 5
   > - JAVA 1.42

2. `$JAVA_HOME` /jre/lib/security 경로에 파일을 덮어쓰기하세요.
3. JCE를 적용했다면 `${WHATAP_HOME}` /whatap.conf 파일에 다음 설정을 추가하세요.

   whatap.conf

   ```
   cypher_level=256
   ```

4. 애플리케이션 서버를 다시 시작하면 AES 256 암호화를 적용합니다.

스케일 인 자동 삭제

Auto Scale 환경에서 에이전트 이름을 자동 부여하면 Scale Out이 발생할 경우 운영자 개입이 필요 없습니다. 시스템을 자동으로 확장합니다. 반면 Scale In이 발생하면 시스템이 정상적인 Shut down으로 인식해야 합니다.

에이전트 자동 삭제를 위해 와탭 서버는 자바 에이전트로부터 메시지를 받아야 합니다. 에이전트는 `${WHATAP_HOME}`에 Java 프로세스의 whatap_`{java-process-pid}`.shutdown 파일이 나타나면 **SILENT_SHUTDOWN** 이벤트를 와탭 서버에 전송합니다. 전송은 5초 이내로 완료합니다.

SILENT_SHUTDOWN 이벤트를 와탭 서버에 전송하고 1분 이내에 에이전트가 셧다운 되면 와탭 서버는 목록에서 해당 Java 에이전트를 바로 제거합니다. inactive 이벤트도 발생시키지 않습니다.

> ⓘ Tomcat의 경우 shutdown.sh 파일에 설정할 수 있습니다.
>
> ```
> touch $WHATAP_HOME/whatap_{java-process-pid}.shutdown
> ```

Java 대시보드

와탭 에이전트 설치를 완료했다면 와탭 모니터링 서비스에 로그인하세요. 초기 화면은 **프로젝트 목록**입니다. **프로젝트 목록**에서 에이전트를 설치한 애플리케이션 서버를 확인할 수 있습니다. 에이전트를 설치한 하나의 애플리케이션 서버는 프로젝트이며, 모니터링 단위입니다. 프로젝트 목록의 프로젝트를 선택하면 **애플리케이션 대시보드**로 진입할 수 있습니다.

대시보드란?

와탭 모니터링 서비스에서 대시보드는 프로젝트의 전체 현황을 한눈에 파악할 수 있는 기능입니다. 모니터링 중인 전체 자원 규모를 확인할 수 있고 실시간 대시보드에서 모니터링 대상 자원을 필터링할 수 있습니다. 진행 중인 트랜잭션과 종료된 트랜잭션 정보를 실시간으로 업데이트하고 시각화된 차트를 제공합니다. 운영자는 서비스 및 시스템의 정확한 현재 상태를 직관적으로 파악할 수 있습니다. 또한 즉각적인 장애 인지 및 잠재적 문제 요소를 빠르게 식별할 수 있습니다.

대시보드를 통해 파악할 수 있는 정보는 다음과 같습니다.

- 응답시간 분포 차트를 통해 응답시간 범위를 조정하거나 에러 데이터를 필터링할 수 있습니다.
- 진행 중 트랜잭션을 분석하고 종료한 트랜잭션을 실시간으로 업데이트합니다.
- TPS, 응답시간, CPU, Memory, 실시간 사용 등의 리소스 현황을 파악할 수 있습니다.
- 최신의 데이터를 자동 업데이트해 실시간 모니터링 서비스를 제공합니다.

다음 동영상을 통해 애플리케이션 대시보드에 대해 알아보세요.

애플리케이션 대시보드

홈 화면 > 프로젝트 선택 > **대시보드** > **애플리케이션 대시보드**



애플리케이션 대시보드에서는 애플리케이션의 주요 지표들을 에이전트를 통해 수집하고 차트를 구성해 실시간으로 모니터링하고 과거의 데이터를 조회할 수 있습니다. 대시보드에 배치된 위젯들을 통해 애플리케이션의 문제를 빠르게 파악하고 분석할 수 있습니다.

CPU, Memory를 제외한 일반적인 차트의 경우 안정적인 데이터는 파란색 계열로 표현되고, 문제로 식별되는 요소들은 붉은색 계열로 표시되어 현황을 쉽게 인지할 수 있습니다.

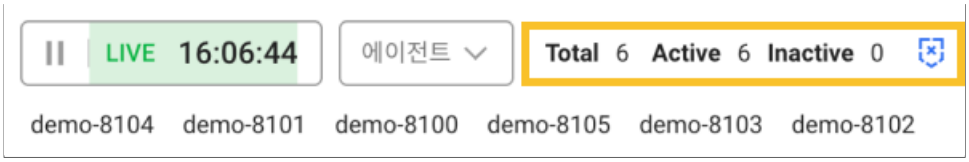
조회 시간 설정

대시보드에서는 실시간 모니터링 기능을 기본 제공하지만, 화면 상단의 시간 선택자를 이용해 과거 시간의 데이터를 조회할 수도 있습니다. 시간 선택자에서 ⏸ 버튼을 선택한 다음 시간을 설정하세요. 사용자가 설정한 시간을 기준으로 대시보드에 배치한

위젯의 데이터를 갱신합니다.

에이전트 확인

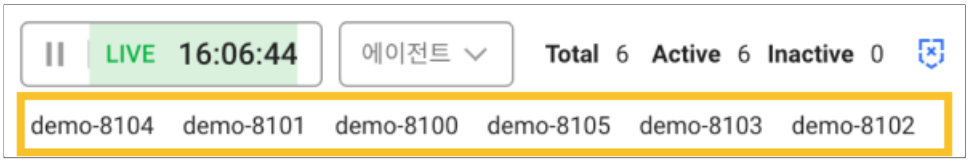
에이전트 연결 상태 확인하기



화면 왼쪽 위, 시간 선택자의 오른쪽에서는 해당 프로젝트와 연결된 에이전트의 상태를 확인할 수 있는 정보를 제공합니다. 이를 통해 모니터링 대상 서버의 동작 여부를 바로 확인할 수 있습니다.

- **Total**: 프로젝트와 연결된 모든 에이전트의 수
- **Active**: 활성화된 에이전트의 수
- **Inactive**: 비활성화된 에이전트의 수
- : 비활성화된 에이전트를 표시하거나 감출 수 있습니다.

에이전트별 모니터링



위 화면에서 노란 박스로 강조된 영역이 **에이전트 목록**입니다. 시간 선택자 아래에 위치한 에이전트 목록에서 하나 또는 둘 이상을 선택하세요. 기본적으로 대시보드는 모든 에이전트로부터 수집한 지표를 차트에 표시하지만, 특정 에이전트를 선택하면 해당 에이전트의 지표로만 위젯의 데이터를 갱신합니다.

> 에이전트를 하나 또는 둘 이상을 선택한 상태에서 다시 모든 에이전트를 선택하려면 선택을 해제하거나 **Total**을 클릭하세요.

> ⓘ 프로젝트에 연결된 에이전트의 수가 많을 경우 에이전트의 이름을 짧게 설정하는 것이 효율적입니다. 에이전트 이름 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

분류별 에이전트 모니터링

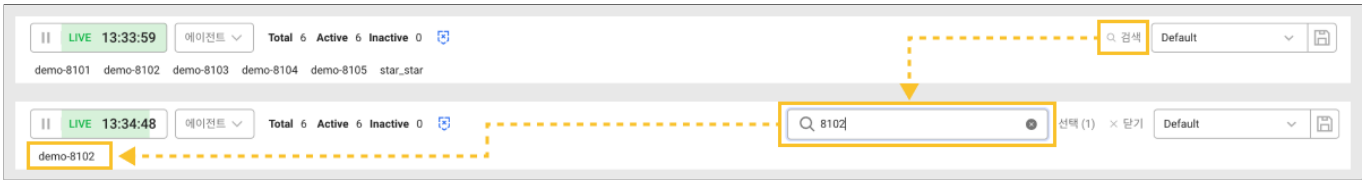
에이전트 설정에서 분류한 단위로 여러 에이전트를 쉽게 모니터링할 수 있습니다.

- **에이전트**: 개별 에이전트
- **종류별**: 에이전트의 종류(예: 웹서버, 데이터베이스 등)로 분류된 그룹, `whatap.okind` 로 설정
- **서버별**: 서버(노드)별로 분류된 그룹, `whatap.onode` 로 설정
- **종류별 에이전트**: `whatap.okind` 로 분류된 에이전트 목록
- **서버별 에이전트**: `whatap.onode` 로 분류된 에이전트 목록

> ⓘ 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

에이전트 검색하기

프로젝트에 연결된 에이전트의 목록이 많아 찾기 어렵다면 에이전트 검색 기능을 이용하세요. 화면 오른쪽 위에 **검색** 버튼을 클릭하면 문자열을 입력할 수 있는 상자가 나타납니다.



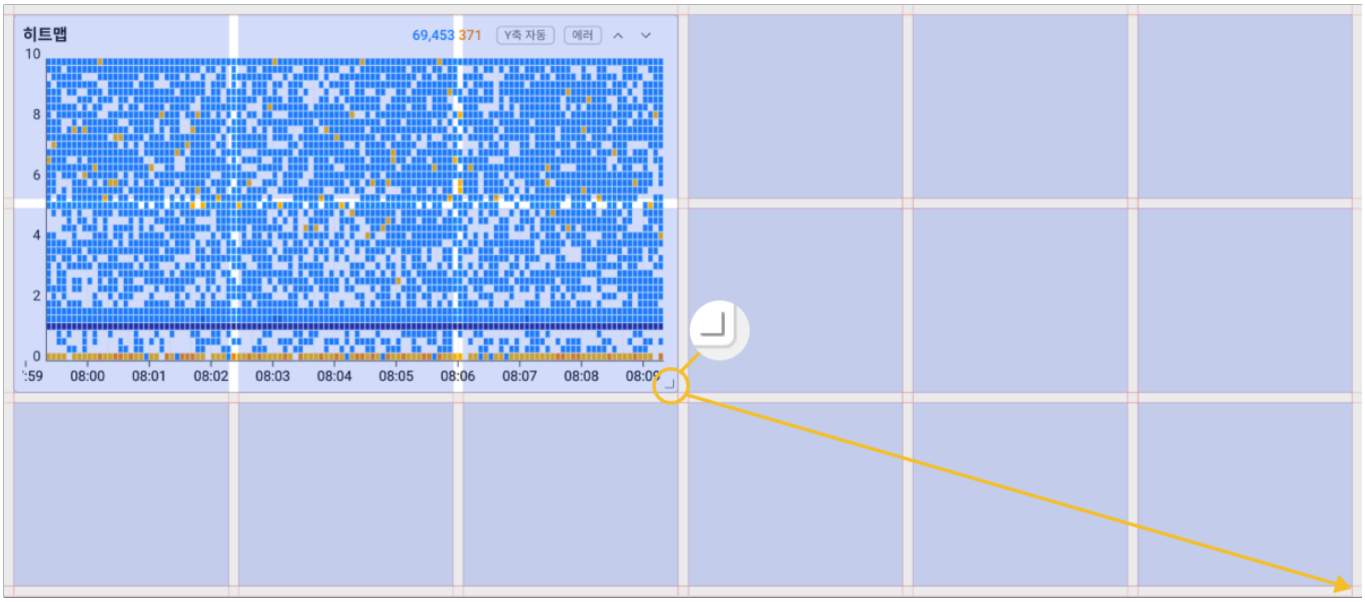
입력한 문자열과 일치하는 에이전트만 에이전트 목록에 표시됩니다. 검색한 에이전트를 기준으로 대시보드의 데이터를 필터링하려면 문자 입력 상자 오른쪽에 **선택** 버튼을 클릭하세요.

선택한 에이전트 항목을 초기화하려면 문자 입력 상자 오른쪽에 **닫기** 버튼을 클릭하세요.

위젯 편집

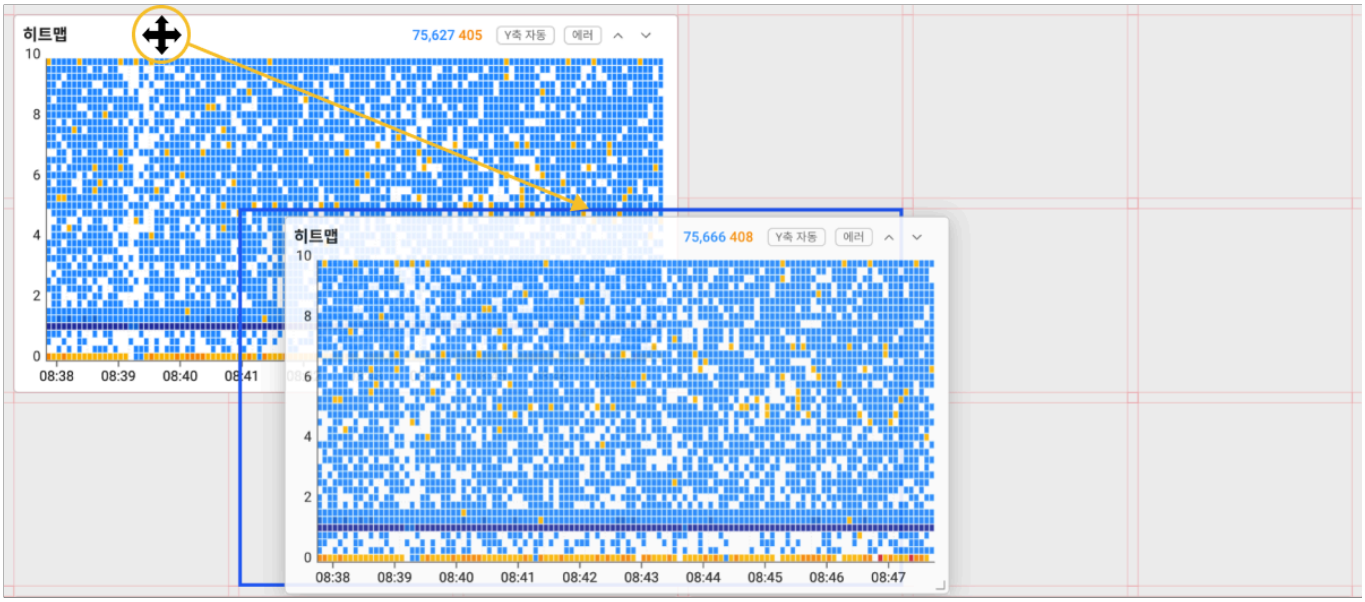
대시보드에 배치한 위젯은 사용자가 원하는 크기로 조절할 수 있고, 원하는 위치에 배치할 수 있습니다. 불필요하다고 생각되는 위젯은 삭제하고 다시 추가할 수도 있습니다.

위젯 크기 조절하기



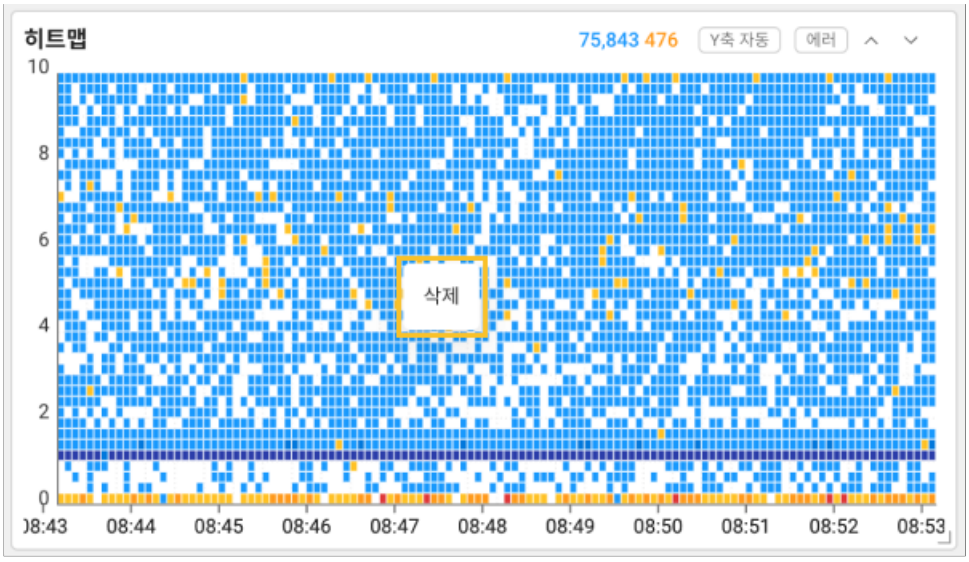
위젯의 오른쪽 아래에 ⌟ 요소를 마우스로 클릭한 상태에서 원하는 크기로 드래그하세요. 균일한 가로, 세로 비율의 격자가 표시되고, 격자 단위로 위젯의 크기를 조절할 수 있습니다.

위젯 이동하기



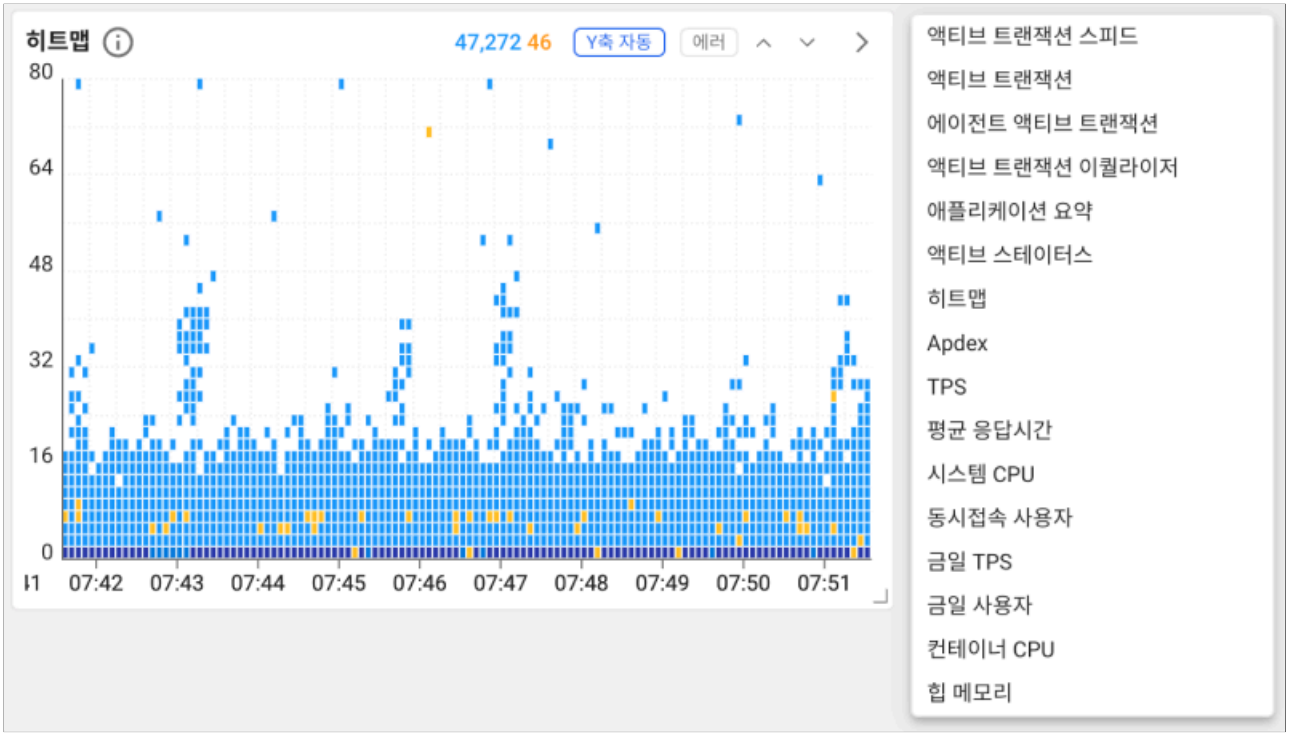
위젯의 윗부분으로 마우스 커서를 이동하면 커서 모양이 ✛ 모양으로 변경됩니다. 이때 마우스 왼쪽 버튼을 클릭한 상태로 원하는 위치로 드래그하여 위젯을 이동할 수 있습니다.

위젯 삭제하기



삭제할 위젯에서 마우스 오른쪽 버튼을 클릭하세요. **삭제** 버튼을 선택하면 해당 위젯이 대시보드에서 삭제됩니다.

위젯 추가하기



대시보드에서 빈 공간으로 마우스 커서를 이동한 다음 마우스 오른쪽 버튼을 클릭하세요. 팝업 메뉴에서 추가하려는 위젯을 선택하세요. 원하는 위치로 위젯을 배치하고 크기를 조절하세요.

- 대시보드에 배치할 수 있는 위젯에 대한 자세한 내용은 다음 문서를 참조하세요.
- 현재 추가할 수 있는 위젯은 고정적이지만 향후 업데이트를 통해 위젯 지원을 늘려갈 계획입니다.
- 대시보드에서 기본 제공되는 지표 외에 사용자가 원하는 지표를 추가할 수 있습니다. 사용자 정의 위젯에 대한 자세한 내용은 다음 문서를 참조하세요.

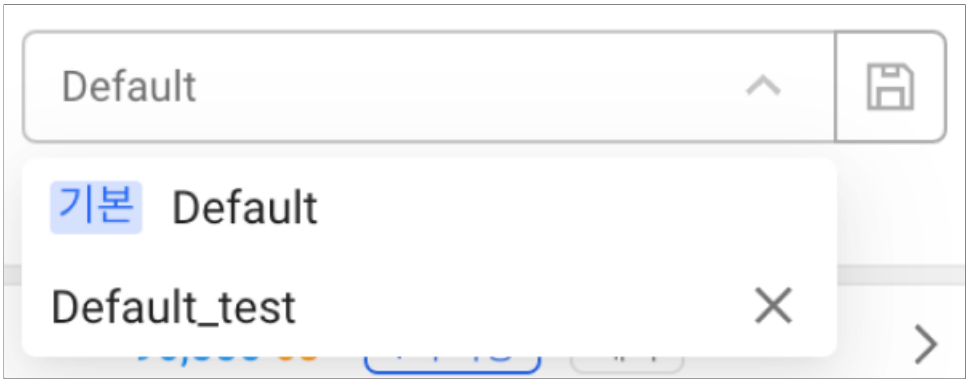
위젯 옵션

위젯에 표시된 아이콘 버튼의 기능은 다음과 같습니다.

- ⓘ : 주요 위젯에 대한 기능 및 정보를 확인할 수 있습니다.
- **병합 / 개별로 보기**: 해당 위젯 항목의 에이전트 데이터를 개별 또는 병합해 그래프로 표시합니다.
- **상세**: 해당 위젯 항목의 데이터를 에이전트별로 구분해 조회할 수 있는 모달 창이 나타납니다.

> ⓘ 위젯에 따라 제공되는 옵션은 다를 수 있습니다.

프리셋 설정



대시보드에서 사용자가 설정한 위젯의 설정과 레이아웃 상태를 저장하고 불러올 수 있습니다. 위젯의 크기를 조절하고, 원하는 위치에 배치해 새로운 프리셋을 만들 수 있습니다.

> ⓘ
> - **기본**으로 설정된 프리셋은 변경할 수 없습니다.
> - **프로젝트 수정** 권한이 있는 멤버만 프리셋을 저장하거나 수정할 수 있습니다. 멤버 권한에 대한 자세한 내용은 다음 문서를 참조하세요.

새로운 프리셋 만들기

1. 대시보드에서 원하는 형식으로 위젯을 배치해 보세요. 크기를 조절하고 자주 확인하는 위젯만 배치할 수도 있습니다.
2. 화면 오른쪽 위에 버튼을 선택하세요.
3. 새로운 프리셋 이름을 입력하세요.



에이전트 선택 내역을 같이 저장하려면 **에이전트 선택 내역 포함**을 선택하세요.

4. **저장** 버튼을 선택하세요.

프리셋 목록에서 새로 저장한 프리셋을 확인할 수 있습니다.

- 새로 만든 프리셋에 변경 사항이 생겼다면 다시 프리셋을 저장해야 합니다. 버튼을 선택한 다음 같은 이름으로 프리셋을 저장하세요. 기존의 프리셋에 변경 사항을 덮어쓰기합니다.
- 대시보드의 변경 사항을 저장하지 않고 다른 메뉴로 이동하면 변경 사항은 저장되지 않습니다.
- 프리셋은 프로젝트 단위로 저장되어 다른 사용자와 공유할 수 있습니다.

프리셋 삭제하기

사용하지 않는 프리셋이 있다면 프리셋 목록에서 삭제할 수 있습니다. 프리셋 목록에서 삭제하려는 항목의 오른쪽에 ✕ 버튼을 선택하세요.

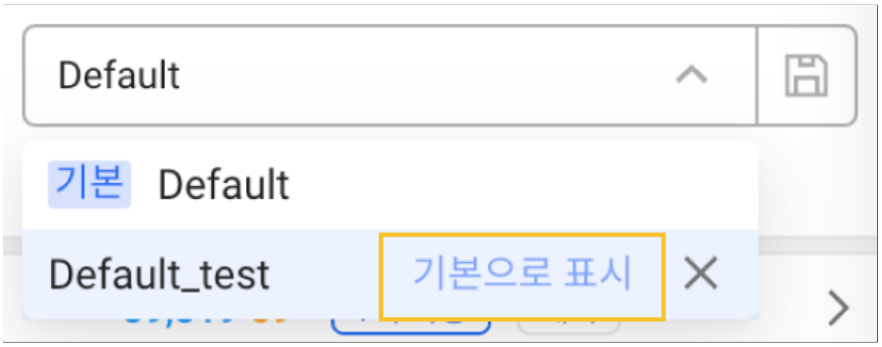
Default 프리셋은 삭제할 수 없습니다.

기본 프리셋 변경하기

애플리케이션 대시보드 메뉴로 처음 진입했을 때 보여지는 프리셋을 변경할 수 있습니다.

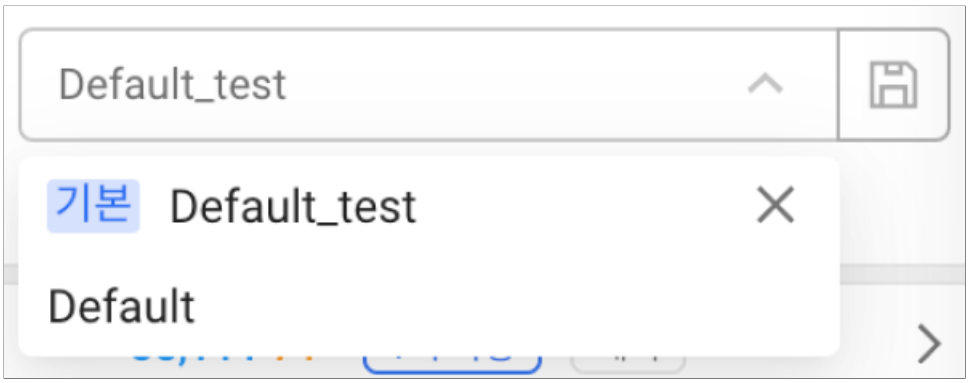
1. 대시보드의 오른쪽 위에 프리셋 선택 상자를 선택하세요.

2. 기본으로 설정할 프리셋에 마우스를 오버하세요.



3. **기본으로 표시** 버튼이 나타나면 선택하세요.

선택한 프리셋이 기본 프리셋으로 변경되며, 프리셋 목록에 **기본** 태그가 표시됩니다.



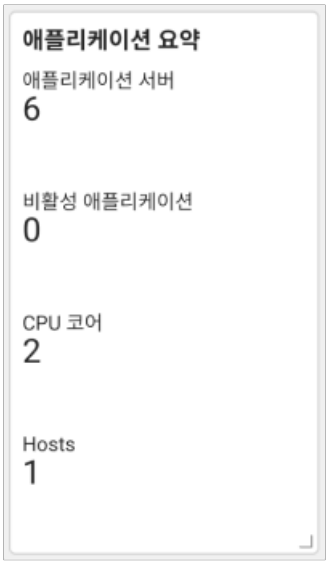
위젯 더 알아보기

> ⓘ 이 문서에서는 **애플리케이션 대시보드**에 배치된 위젯에 대한 설명을 주로 다룹니다. **애플리케이션 대시보드**를 통해 트랜잭션 및 사용자, 서비스, 리소스 분석 방법에 대한 자세한 내용은 다음 문서를 참조하세요.
>
> - 애플리케이션 대시보드 살펴보기
> - 히트맵 트랜잭션
> - DB 연결 지연과 커넥션 풀

에이전트 관련 위젯

애플리케이션에 설치한 에이전트를 통해서 애플리케이션의 정보를 확인할 수 있습니다.

애플리케이션 요약



프로젝트에 등록한 애플리케이션의 실행 상태 및 개수, CPU 코어 수, Hosts 수를 표시합니다.

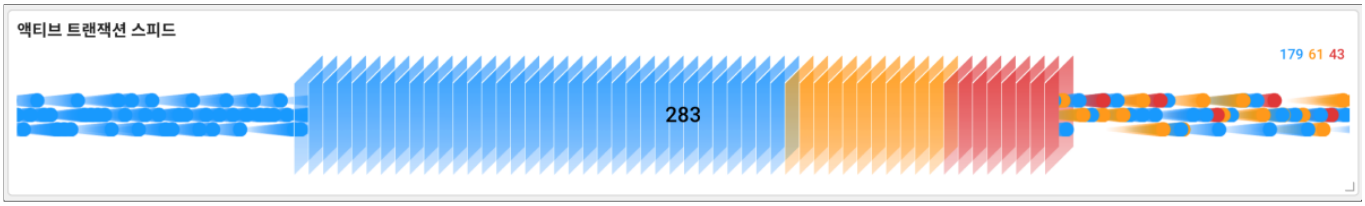
트랜잭션 관련 위젯

트랜잭션을 통해 확인할 수 있는 장애의 현황은 우선 응답시간을 통해 알 수 있습니다. 또한 진행 중인 트랜잭션이 종료되지 않는다면 이 또한 장애로 인식해야 합니다. 와탭은 진행 중인 상태의 시간에 따라 구간을 나누어 표시합니다. 파랑색(Normal)은 응답 시간이 정상인 트랜잭션, 주황색(Slow)은 응답 시간이 8초 정도의 느린 트랜잭션, 빨간색(Very slow)은 응답 시간이 보통의 2배 이상으로 느린 트랜잭션을 의미합니다. 이를 통해 사용자는 직관적으로 가장 빨리 장애를 인지할 수 있습니다.

> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면 다음 위젯에서 연계한 프로젝트의 성능 지표를 조회할 수 있습니다.
>
> - **액티브 트랜잭션 스피드**, **액티브 트랜잭션**: 현재 진행 중인 트랜잭션과 연계된 데이터베이스의 SQL 성능을 조회할 수 있습니다.
> - **히트맵**: 과거 시점의 트랜잭션과 연계된 데이터베이스의 성능 지표를 조회할 수 있습니다.

> 연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

액티브 트랜잭션 스피드



현재 진행 중인 트랜잭션(가운데 영역)과 종료된 트랜잭션(오른쪽 영역) 현황을 확인할 수 있습니다. 왼쪽에서 오른쪽으로 지나가는 물방울 모양의 아이콘은 트랜잭션을 의미합니다. 차트 영역을 클릭하면 진행 중인 트랜잭션 목록을 확인할 수 있는 팝업 창이 나타납니다. 트랜잭션 목록 중 하나를 선택하면 상세한 트랜잭션 정보와 SQL 쿼리, 콜 스택 정보를 확인할 수 있습니다.

- 왼쪽 영역

  요청 트랜잭션 수를 의미하며, 트랜잭션 수에 비례해 더 많이, 더 빠르게 표시합니다.

- 가운데 영역

  액티브 트랜잭션의 상태를 속도로 색상으로 분류하여 우측 상단에 개수로 표시합니다. 차트 안의 숫자는 전체 액티브 트랜잭션의 수입니다. 차트의 색상의 의미는 다음과 같습니다.

  - 파란색(Normal): 0초 ~ 3초
  - 주황색(Slow): 3초 ~ 8초
  - 빨간색(Very Slow): 8초 이상

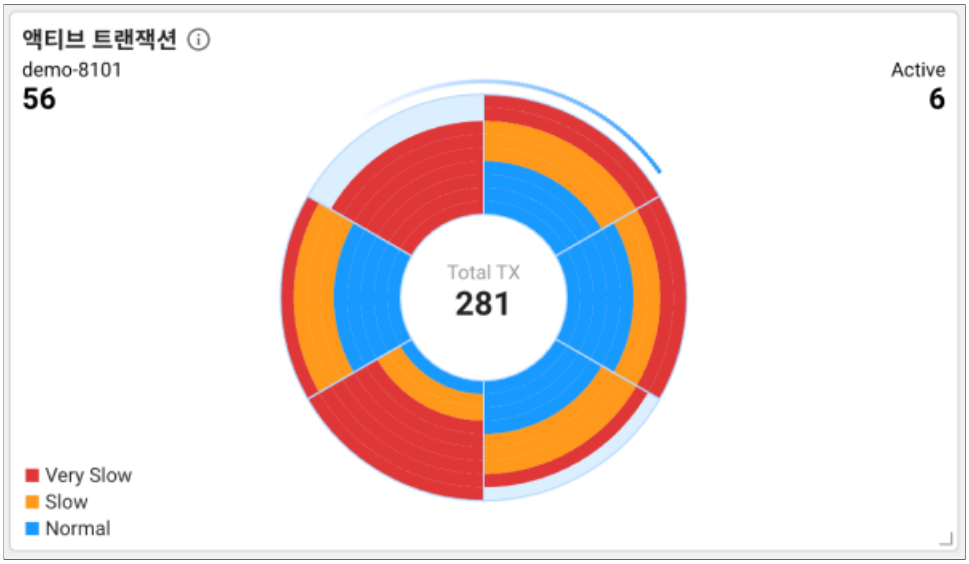
- 오른쪽 영역

  애플리케이션 성능 지표(Application Performance Index, Apdex)를 의미합니다. 웹 애플리케이션의 고객 만족도를 측정한 지표로, 0 ~ 1 사이의 값을 갖습니다. 물방울이 날아가는 속도는 Apdex 트랜잭션의 총 수에 비례하며, 물방울의 색상의 의미는 다음과 같습니다.

  - 파란색: Apdex 만족(Satisfied) 수
  - 주황색: Apdex 허용(Tolerating) 수
  - 빨간색: Apdex 불만(Frustrated) 수 = `APDEX 트랜잭션 총 수` - (`Apdex 만족 수` + `Apdex 허용 수`)

> ⓘ 애플리케이션 성능 지표(Application Performance Index, Apdex)에 대한 자세한 내용은 다음 링크를 참조하세요.

액티브 트랜잭션



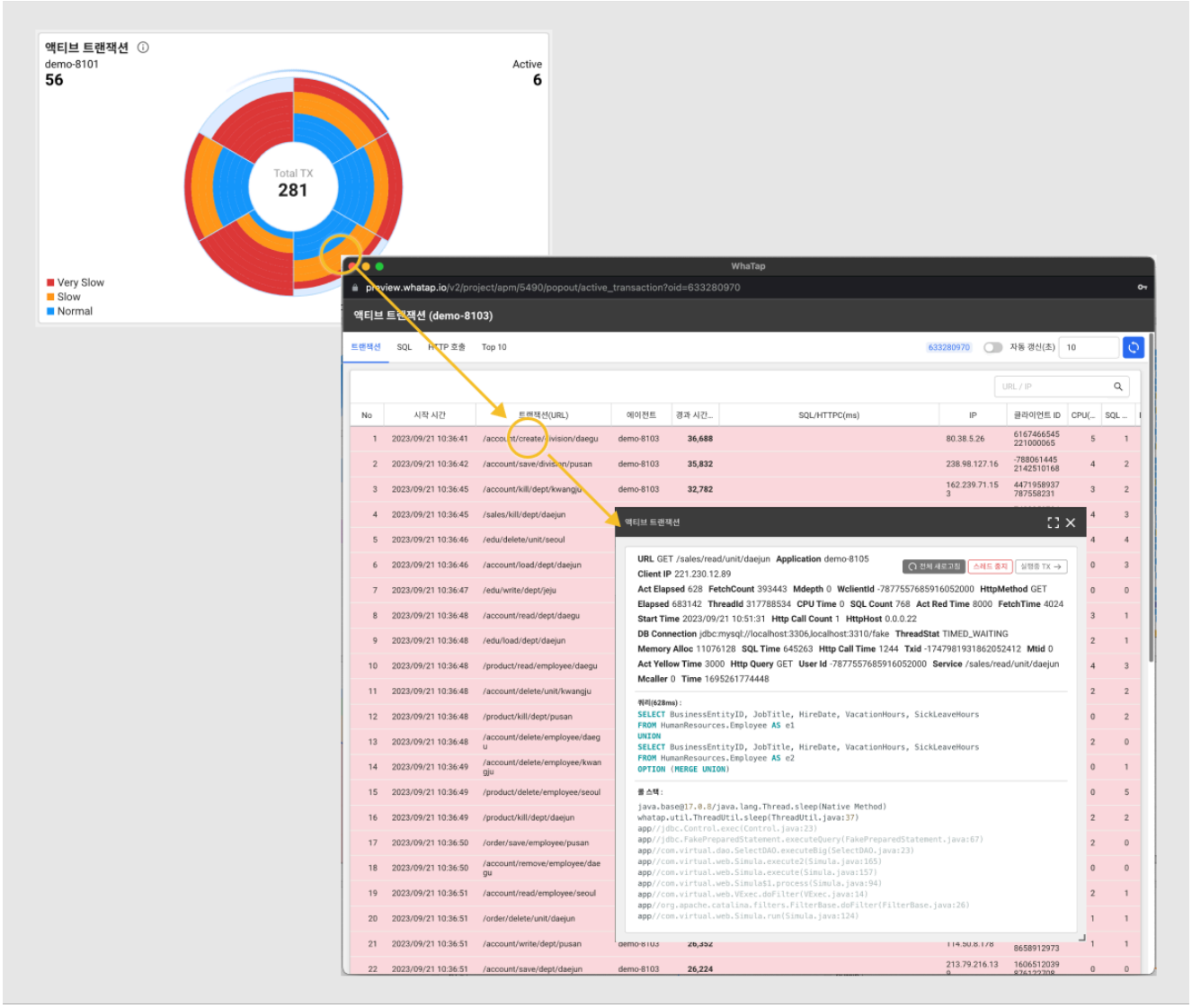
액티브 트랜잭션 위젯에서는 진행 중인 트랜잭션을 **아크 이퀄라이저** 차트로 표현합니다. 지연 발생 현황을 에이전트별로 확인할 수 있습니다. 에이전트 수만큼 아크가 분할됩니다. 5초마다 현재 서버에서 처리 중인 요청의 수를 표현해서 해당 요청이 각각 어느 정도의 시간 동안 처리 중인지 알 수 있습니다. 5초 간격의 시간에 감지된 요청들은 위험 여부를 파악할 수 있도록 다음과 같이 색으로 분류합니다.

- 빨간색(Very Slow): 8초 이상
- 주황색(Slow): 3초 ~ 8초
- 파란색(Normal): 0초 ~ 3초

파란색이 많이 표현되는 상황은 문제가 되지 않습니다. 이 중 일부가 주황색이나 빨간색으로 변경되는지 추이를 지켜봐야 합니다. 장애 상황에는 진행 중 트랜잭션 수가 증가하고 빨간색 비율이 높아집니다. 반면, 응답이 매우 빠른 시스템의 경우 처리하는 트랜잭션 수(TPS)는 높아도 진행 중인 트랜잭션 수는 낮을 수 있습니다.

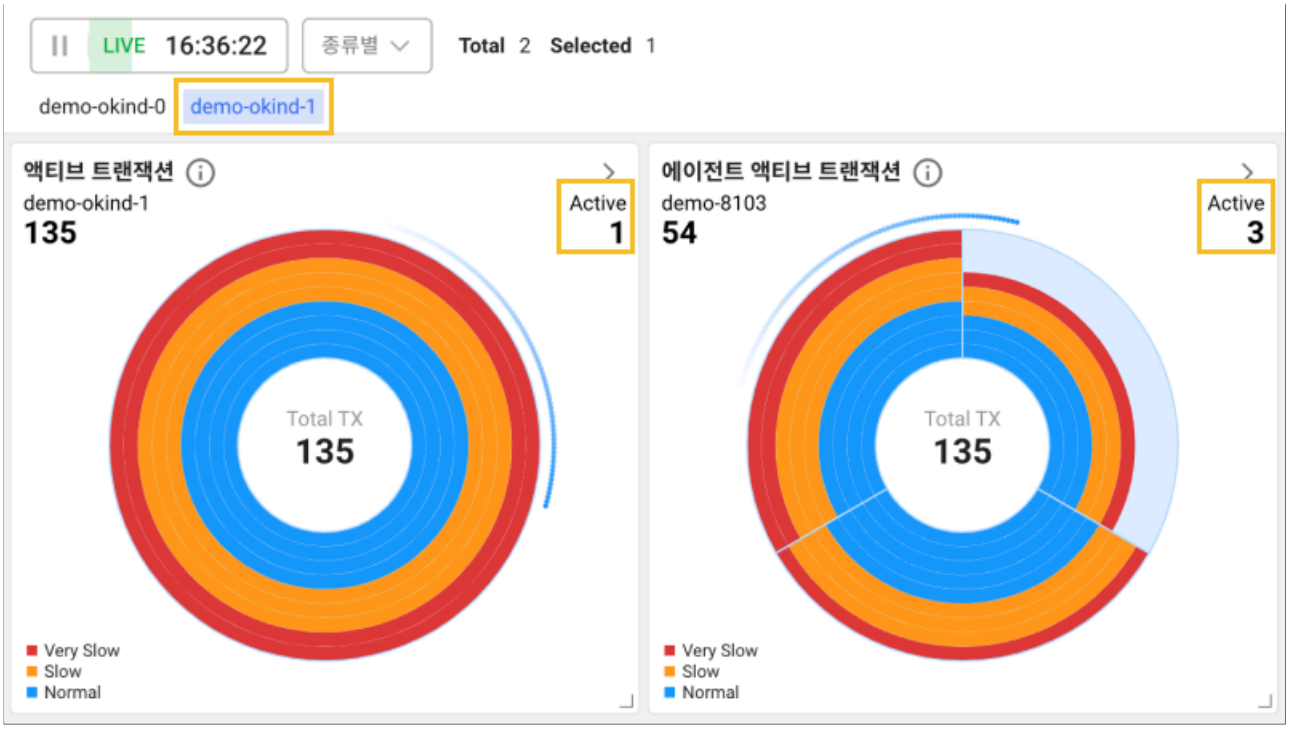
- 왼쪽 위에는 진행 중인 트랜잭션이 가장 많은 에이전트 이름을 표시합니다.
- 가운데 숫자는 전체 에이전트의 액티브 트랜잭션 수의 합계입니다.

- 액티브 트랜잭션 둘레에 두 개의 바는 처리량(TPS)에 따라 3단계 속도로 차트 주변을 회전합니다.
- 분할된 아크 영역을 선택하면 해당 에이전트에서 진행 중인 트랜잭션 목록 팝업창이 나타납니다. 상세한 트랜잭션 정보를 확인하려면 트랜잭션 목록에서 원하는 항목을 하나 선택하세요.



> ⓘ
> - 트랜잭션 목록에서 선택한 항목이 이미 종료된 트랜잭션일 경우 **트랜잭션 보기** 버튼을 선택하세요. **트랜잭션 정보** 창이 나타납니다. **트랜잭션 정보** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
> - 트랜잭션에 대한 자세한 내용은 다음 문서를 참조하세요.

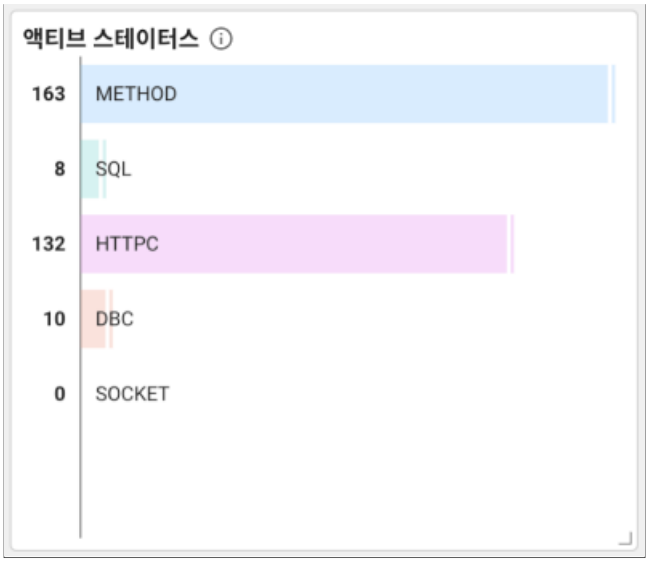
에이전트 액티브 트랜잭션



액티브 트랜잭션 위젯은 에이전트 선택 옵션에서 선택한 기준에 따라 병합된 데이터를 차트에 표시합니다. 반면 **에이전트 액티브 트랜잭션** 위젯은 선택한 에이전트 분류 기준에 포함된 에이전트의 현황을 개별로 확인할 수 있습니다.

대시보드에서 마우스 오른쪽 버튼을 클릭한 다음 위젯 목록에서 **에이전트 액티브 트랜잭션** 위젯을 선택하세요.

액티브 스테이터스



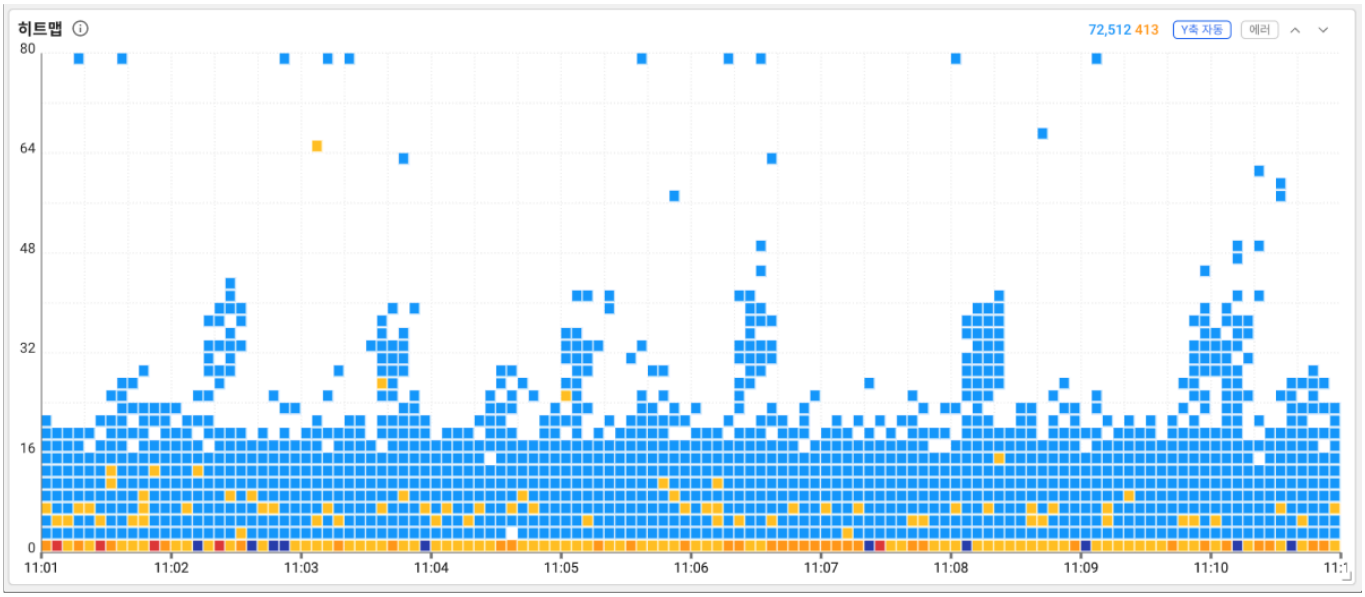
프로젝트 내의 모든 에이전트는 5초마다 액티브 트랜잭션 수를 수집함과 동시에 진행 상태를 수집합니다. 진행 상태는 **METHOD**, **SQL**, **HTTPC**, **DBC**, **SOCKET**으로 분류할 수 있습니다.

- **METHOD**: 메소드를 수행 중인 상태입니다. 일반적인 상황으로 파란색으로 표시됩니다.
- **SQL**: SQL을 수행 중인 상태입니다. **SQL** 수치가 증가한다면 데이터베이스 서버와의 연결에 문제를 확인해 보세요.
- **HTTPC**: 외부 API를 호출 중인 상태입니다. **HTTPC** 수치가 증가한다면 외부와 연결된 서버의 응답이 제대로 이루어지지 않는다고 봐야 합니다.
- **DBC**: 트랜잭션이 Connection Pool로부터 새로운 Connection을 획득(get)하려는 상태입니다. DB Connection Pool의 개수가 부족하면 새로운 연결 요청이 발생할 때마다 지연이 되면서 성능 장애의 원인이 됩니다. 이 경우 **DBC** 수치가 증가합니다.

  > ⓘ DB 연결 지연과 관련한 자세한 내용은 다음 문서를 참조하세요.

- **SOCKET**: 외부 시스템과의 TCP 연결 시도를 의미합니다. **SOCKET** 수치가 지속적으로 증가한다는 것은 외부 시스템과의 연결이 되지 않아 장애가 발생 중일 가능성이 높습니다.

히트맵

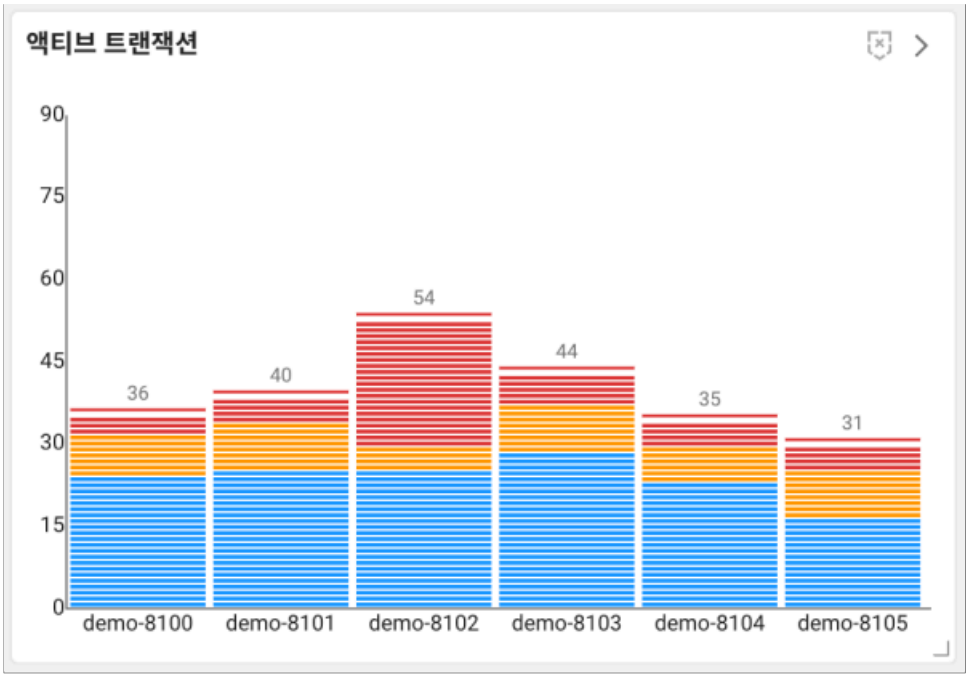


히트맵 차트는 종료된 트랜잭션 응답시간을 분포도로 표현합니다. 가로축은 트랜잭션 종료 시간입니다. 세로축은 수행 시간입니다. 수행 시간이 긴 트랜잭션은 분포도 상단에 위치합니다. **히트맵** 내의 차트 영역을 드래그하면 세부 트랜잭션 정보를 확인할 수 있는 **트레이스 분석** 창이 나타납니다. **트레이스 분석**에 대한 자세한 내용은 다음 문서를 참조하세요.

- 정상 트랜잭션은 하늘색 → 파란색 → 남색 순으로 표현됩니다.
- 에러 트랜잭션은 노란색 → 주황색 → 빨간색 순으로 표현됩니다.
- 오른쪽 위에 위치한 숫자는 차트 내 트랜잭션 건 수/에러 건수를 의미합니다.
- **Y축 자동** 버튼을 선택하면 수행 시간을 나타내는 Y축을 자동 조정합니다.
- **에러** 버튼을 선택하면 에러 트랜잭션만 표시합니다.
- ∧ 또는 ∨ 버튼을 선택해 5초 ~ 80초까지 차트를 확대/축소할 수 있습니다.

ⓘ
- 오류가 발생하지 않았고 수행 시간 500ms 이하 트랜잭션의 세부 정보는 url 당 5분에 1건만 수집됩니다.
- TPS나 평균 응답시간 같은 통계정보에는 영향을 미치지 않습니다.
- 에이전트 설정을 통해 정책을 설정할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- 히트맵 위젯을 통해 트랜잭션을 분석하는 과정은 다음 문서를 참조하세요.

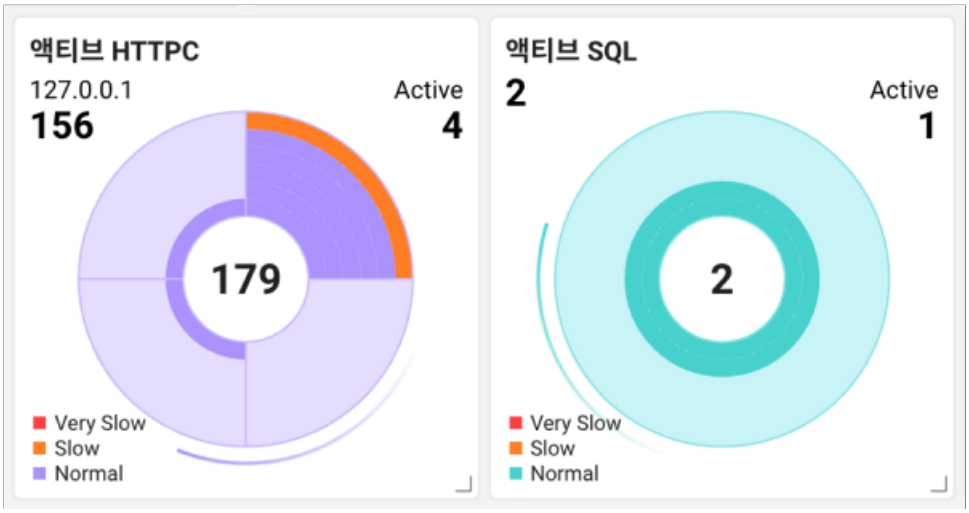
액티브 트랜잭션 이퀄라이저



전체 에이전트 목록과 진행 중인 트랜잭션의 총 개수를 표시합니다. 차트의 각 그래프에 마우스를 오버하면 트랜잭션 상태 속도를 확인할 수 있습니다. 트랜잭션 상태의 속도는 Normal, Slow, Very Slow로 분류합니다.

- 비활성화된 에이전트를 표시하거나 감추려면 버튼을 선택하세요.
- > 버튼을 선택하면 **액티브 트랜잭션** 메뉴로 이동합니다.

액티브 HTTPC/SQL



액티브 HTTPC/SQL 위젯은 애플리케이션에서 발생하는 외부 호출(HTTP 호출 및 데이터베이스 쿼리)의 성능을 실시간으로 모니터링합니다. 전체 트랜잭션을 보여주는 **액티브 트랜잭션** 위젯과 달리, 아웃바운드 호출에 특화되어 외부 시스템 병목을 즉각적으로 파악할 수 있습니다. 두 위젯을 함께 배치하면 트랜잭션 증가가 외부 시스템 지연 때문인지, 내부 로직 문제인지 구분할 수 있어 더욱 효과적인 다층적 분석이 가능합니다.

진행 중인 HTTPC/SQL을 **아크 이퀄라이저** 차트로 시각화하여 지연 발생 현황을 호스트/데이터베이스 URL별로 확인할 수 있습니다. 호스트/데이터베이스 URL 수만큼 아크가 분할되며, 5초마다 현재 서버에서 처리 중인 요청의 수와 처리 시간을 표현합니다. 5초 간격으로 감지된 요청들은 위험도에 따라 직관적인 색상으로 분류됩니다.

- 빨간색(Very Slow): 8초 이상
- 주황색(Slow): 3초~8초
- Normal: 0초~3초
  - 보라색(HTTPC Normal)
  - 민트색(SQL Normal)

Normal이 많이 표현되는 상황은 문제가 되지 않습니다. 일부가 주황색이나 빨간색으로 변경되는지 관찰해야 합니다. 장애 상황 시, 진행 중 트랜잭션 수가 증가하고 빨간색 비율이 높아집니다. 반면, 응답이 매우 빠른 시스템의 경우 처리하는 트랜잭션 수(TPS)는 높아도 진행 중인 트랜잭션 수는 적을 수 있습니다.

이 위젯은 특히 마이크로서비스 환경에서 서비스 간 HTTP 통신 지연이나 여러 데이터베이스를 사용하는 환경에서 DB 성능을 개별적으로 추적하는 데 유용합니다. 외부 API 호출이 많은 서비스는 API 제공자별 응답 시간을 모니터링할 수 있어 의존성 관리에

효과적입니다.

아크 이퀄라이저 차트

위젯의 핵심 시각화 요소로, 진행 중인 HTTPC/SQL을 호스트/데이터베이스 URL로 나누어 표시합니다. 전체 트랜잭션에서 문제를 발견했다면 이 차트를 통해 구체적인 외부 호출 병목을 확인할 수 있습니다.

- **왼쪽 위 표시**: 진행 중인 트랜잭션이 가장 많은 호스트/데이터베이스 URL을 표시하여 주요 관심 대상을 쉽게 식별
- **중앙 숫자**: 전체 액티브 HTTPC/SQL 수의 합계
- **TPS 기반 애니메이션**: 이퀄라이저 둘레의 바가 처리량(TPS)에 따라 3단계 속도로 회전하여 시스템 부하를 시각적으로 표현
- **5초 단위 갱신**: 실시간에 가까운 모니터링으로 즉각적인 대응 가능
- **아크 분할**: 호스트/데이터베이스 URL로 아크가 분할되어 어느 외부 시스템에서 지연이 발생하는지 한눈에 파악

호스트/데이터베이스 URL 상세 조회

분할된 아크 영역을 선택하면 해당 호스트/데이터베이스 URL에서 진행 중인 **트랜잭션 Top 100 목록** 팝업창이 나타납니다.

- **트랜잭션 목록**: 선택한 호스트/데이터베이스 URL로 진행 중인 트랜잭션 Top 100 표시
- **트렌드 차트**: 최근 10분간의 TPS/응답 시간/에러 건수/에러율을 차트로 표현
- **트랜잭션 상세 정보**: 목록에서 트랜잭션 선택 시 상세 정보 확인
  - 이미 종료된 트랜잭션의 경우 **트랜잭션 보기** 버튼을 선택하여 **트랜잭션 정보** 창으로 이동

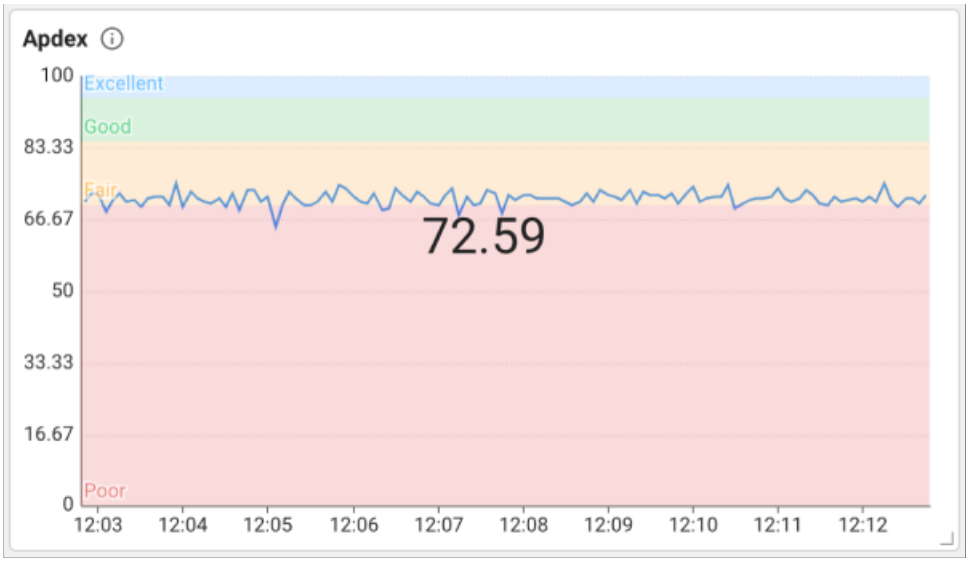
1. **액티브 HTTPC Top 100** 팝업창에서 트랜잭션 목록과 트렌드를 확인합니다.
2. 상세 분석이 필요한 트랜잭션을 선택합니다.

   a.아크 이퀄라이저에서 분석할 호스트 또는 데이터베이스 URL의 아크를 클릭합니다.
3. 트랜잭션 정보 창에서 상세 내용을 확인합니다.

서비스 관련 위젯

서비스 관련 위젯을 통해 애플리케이션 성능 튜닝을 위한 지표로 활용할 수 있습니다.

Apdex



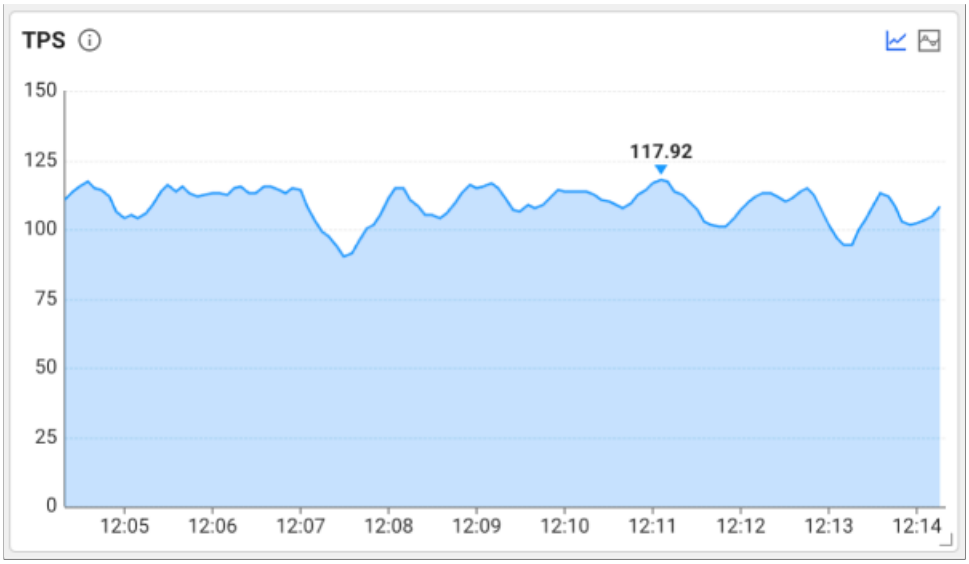
Application Performance Index(Apdex)는 애플리케이션 성능 지표입니다. 웹 애플리케이션의 고객 만족도를 측정합니다. 사용자 만족도에 대한 지표로 활용할 수 있으며, 0 ~ 1 사이의 값을 갖습니다.

(만족 횟수 + (허용 횟수 * 0.5)) / 전체 요청 수

- 만족 (Satisfied, S): 업무처리에 전혀 문제 없음 ≤ 1.2초 (만족 S 기본값)
- 허용 (Tolerating, T): 사용자가 지연을 느끼나 업무처리는 가능 ≤ 4.8초 (만족 S * 4)
- 불만 (Frustrated, F): 업무처리가 불가능 > 4.8초 (허용 T 초과 및 오류)

> ⓘ 만족 S 기본값은 **관리** > **에이전트 설정** 메뉴에서 변경할 수 있습니다. 설정 키 값은 `apdex_time` 입니다. 에이전트 버전 2.0 이상만 지원합니다.

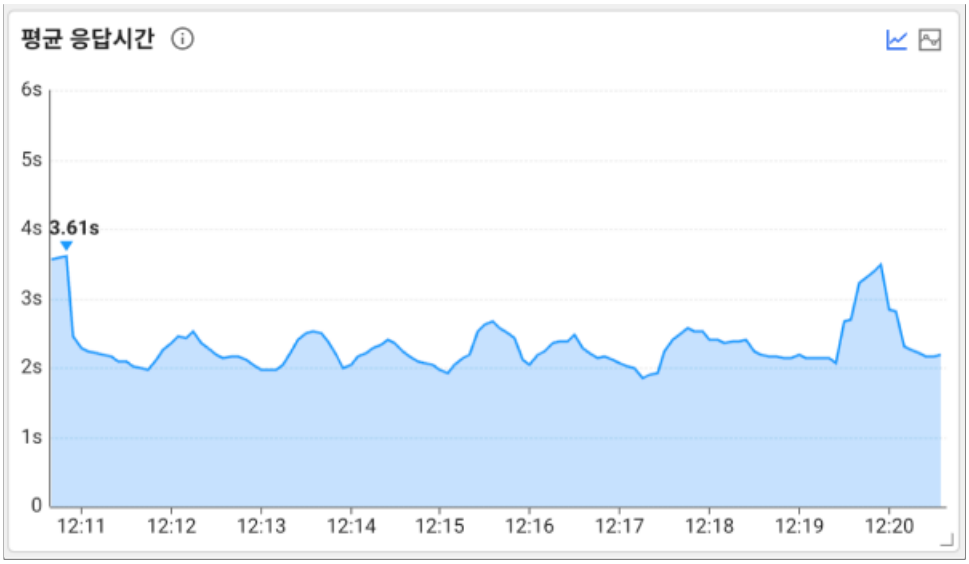
TPS



Tansaction Per Second(TPS)는 초당 처리한 트랜잭션 건수를 의미하며 서비스 성능 지표의 기준이 됩니다. 5초마다 처리한 트랜잭션의 수를 초당 건수로 환산해 차트로 표현합니다. 최근 10분간의 TPS를 보여줍니다.

> TPS, 응답시간, CPU 사용률 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

평균 응답시간

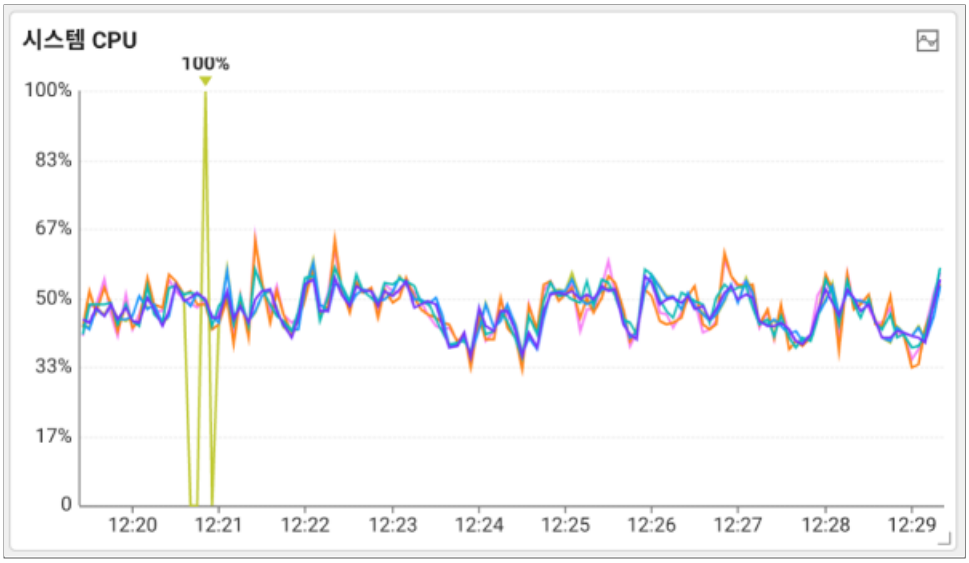


애플리케이션 서버가 사용자에게 요청 결과를 반환하는 데 걸리는 시간을 실시간으로 모니터링합니다. **평균 응답시간**은 튜닝 지표로서 의미를 가집니다. 와탭의 모니터링 서비스는 5초 간격으로 트랜잭션의 평균 응답 시간을 계산합니다.

> TPS, 응답시간, CPU 사용률 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

리소스 관련 위젯

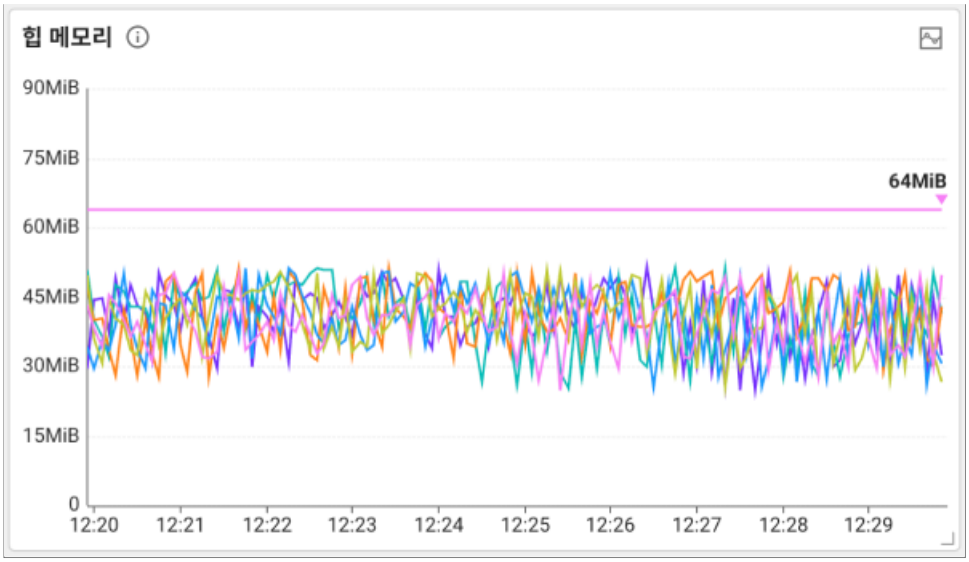
시스템 CPU



애플리케이션 서버의 CPU 사용량입니다. 실시간으로 CPU 사용량 변화 추이를 파악할 수 있습니다.

> TPS, 응답시간, CPU 사용률 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

힙 메모리



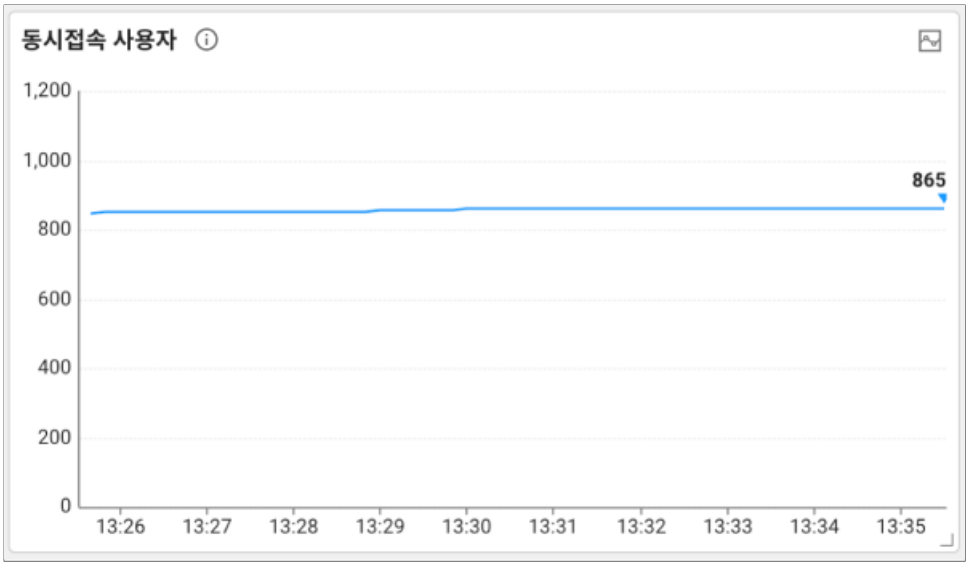
각 서버당 사용할 수 있는 최대 메모리와 현재 메모리를 표시해 위험 수치에 있는 서버를 확인할 수 있습니다. 시간에 따른 메모리 사용량 변화를 실시간으로 볼 수 있습니다.

메모리 라인 차트는 보통 계속해서 물결칩니다. 애플리케이션 서버가 요청을 처리하기 위해 메모리를 사용할 때 증가합니다. GC(Garbage Collection)를 통해서 메모리를 정리할 경우에는 감소합니다. **힙 메모리** 지표는 평균값을 사용해 표기합니다.

> 힙 메모리 차트 분석에 대한 자세한 내용은 다음 링크를 참조하세요.
>
> - Java 힙 메모리 차트 분석 : Ch.1 힙차트 관찰하기
> - JAVA 힙메모리 차트 분석 : Ch.2 메모리 릭, 그리고 힙덤프 분석

사용자 관련 위젯

동시접속 사용자



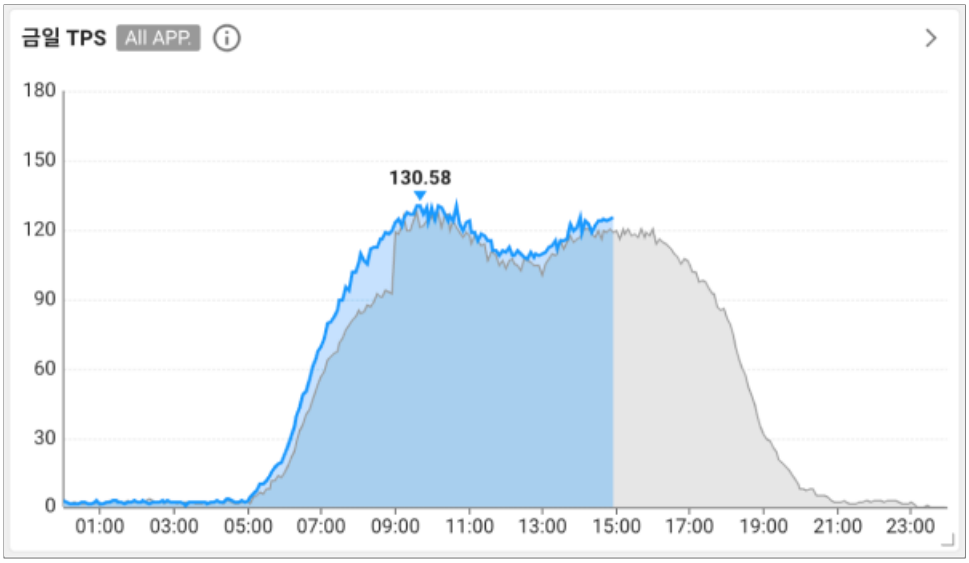
실시간 브라우저 사용자 수를 보여줍니다. 10초마다 최근 5분 이내에 트랜잭션을 일으킨 사용자를 중복없이 카운팅해 표시합니다. 사용자 브라우저의 IP를 기반으로 카운팅합니다.

- 에이전트 설정에서 사용자 구분을 위해 IP를 사용하거나 쿠키를 사용할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- 일반적으로 접속 사용자는 현재 네트워크로 접속해 연결된 사용자를 의미합니다. 하지만 웹 시스템은 비 연결 네트워크를 사용하기 때문에 다릅니다. 접속되어 있다는 의미보다는 최근 요청을 보낸 사용자가 측정 대상이 될 수 있습니다.
- 실시간(동시접속) 사용자는 같은 시간대에 시스템을 사용하고 있는 사용자를 의미합니다. 실시간(동시접속) 사용자를 산정하거나 측정하는 방식은 다양합니다.
- 사용자 수는 단순히 합산하는 것이 아니라 **HyperLogLog**로 산출합니다. **HyperLogLog**는 매우 적은 메모리로 집합의 원소 개수를 추정하는 확률적 자료 구조를 의미합니다.
- 사용자 수 집계를 위한 에이전트 옵션 설정 방법은 다음 문서를 참조하세요.

1일 기준 비교 위젯

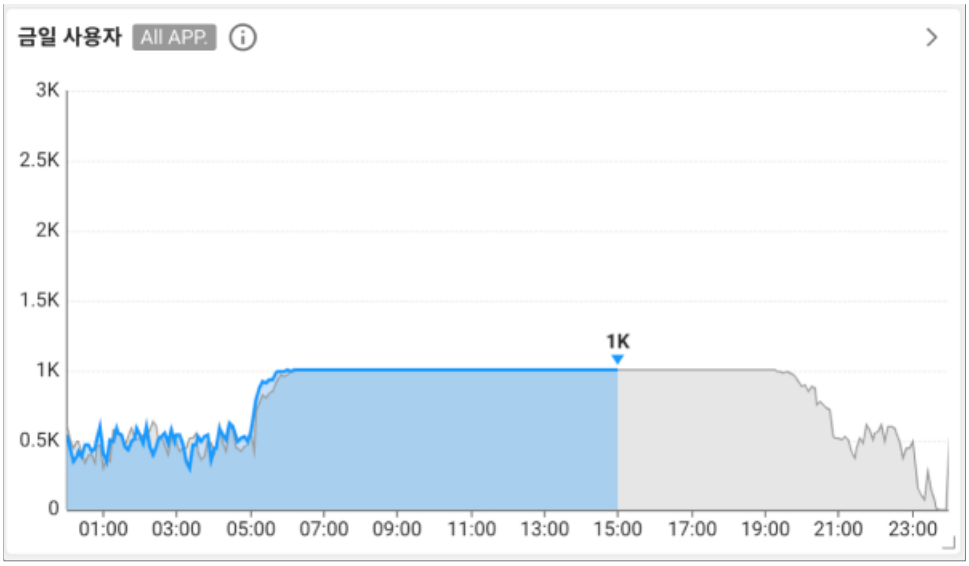
> All APPS. 태그가 표시된 위젯은 항상 모든 에이전트에 대한 데이터를 표시합니다.

금일 TPS



금일 하루 동안의 TPS를 시간대별로 표시합니다. 어제 동시간대의 TPS(회색)와 비교해 확인할 수 있습니다. > 버튼을 클릭하면 **통계/보고서** > **일자별 애플리케이션 현황** 메뉴로 이동합니다.

금일 사용자



하루 동안의 **동시접속 사용자** 수를 시간대별로 나타낸 지표입니다. 어제 동시간대의 사용자 수(회색)와 비교해 확인할 수 있습니다.
> 버튼을 클릭하면 **통계/보고서** > **일자별 애플리케이션 현황** 메뉴로 이동합니다.

실시간 알림

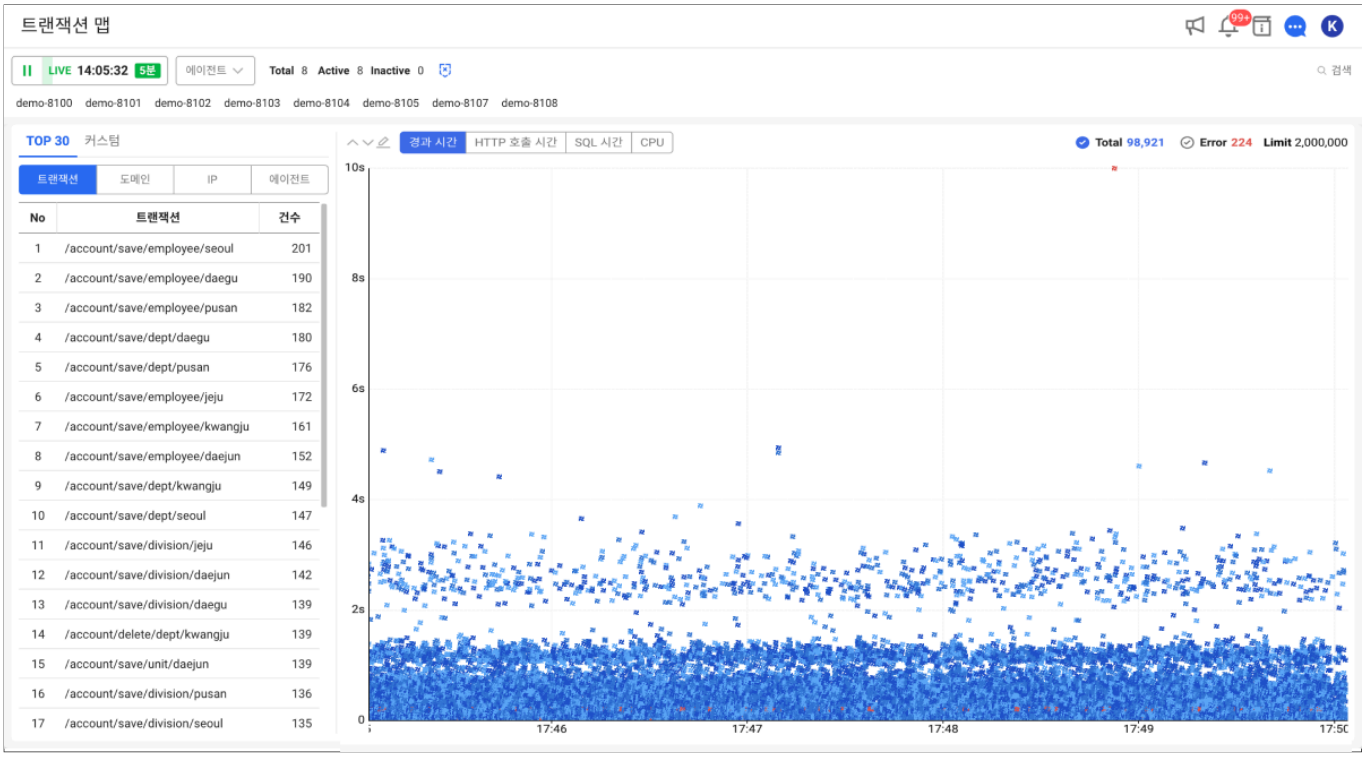
화면 오른쪽 위에 **실시간 알림** 버튼을 선택하면 최근 발생한 이벤트를 확인할 수 있습니다. 토글 메뉴를 클릭해 브라우저 알림을 켜거나 끌 수 있습니다.

> 화면 가장 위에 고정 메뉴 영역의 기본 요소들에 대한 자세한 내용은 다음 문서를 참조하세요.

트랜잭션 맵

홈 화면 > 프로젝트 선택 > **대시보드** > **트랜잭션 맵**

트랜잭션 맵 차트는 종료된 개별 트랜잭션의 응답시간을 분포도 형태로 표현한 차트입니다. 히트맵과 동일하게 분포 패턴에 따른 문제점을 발견하고 분석할 수 있습니다. **히트맵**은 5초 단위로 트랜잭션을 그룹화해서 보여주지만 **트랜잭션 맵**은 트랜잭션을 개별로 표시합니다.



히트맵 분석과 다른 점은?

히트맵은 합산된 데이터를 사용해 긴 시간 동안의 전체적인 추이를 확인할 수 있습니다. 반면, 트랜잭션 맵은 개별 트랜잭션 정보를 활용하여 상대적으로 짧은 시간 범위 내에서 트랜잭션 분포를 수행시간 제한 없이 확인할 수 있습니다. 또한 다양한 필터를 적용해 원하는 트랜잭션만 선택해 추이를 확인할 수 있다는 점이 다릅니다.

- 장기간, 대용량 트랜잭션 분포도를 분석하려면 히트맵 분석(**분석** > **히트맵**)을 이용하세요.
- 성능 테스트를 하거나 장애 상황에서 1~5분 이내의 트랜잭션 분포도 분석, 수행 시간 제한 없이 트랜잭션 분포도를 확인하려면 **트랜잭션 맵**을 확인하세요.

- **히트맵** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
- **히트맵 트랜잭션** 차트를 분석 방법에 대한 자세한 내용은 다음 문서를 참조하세요.
- 사용자의 PC 시간이 서버 시간보다 과거로 설정되어 있거나, 두 시간의 차이가 5분 이상일 경우 차트에 데이터가 출력되지 않을 수 있습니다.

트레이스 분석하기

트랜잭션 맵의 차트 영역을 드래그하면 세부 트랜잭션 정보를 확인할 수 있는 **트레이스 분석** 창이 나타납니다. 트랜잭션 목록과 각 트랜잭션 하위의 스텝 정보를 한 번에 확인할 수 있습니다.

차트 영역을 드래그한 다음 **트레이스 분석** 창에서 조회할 수 있는 데이터는 최대 1,000건입니다.

트레이스 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

조회 기준 변경하기

트랜잭션 맵의 차트 영역, 왼쪽 위에 버튼을 선택해 데이터 조회 기준을 변경할 수 있습니다.

- **경과 시간**: 트랜잭션이 수행된 시간을 기준으로 데이터를 표시
- **HTTP 호출 시간**: HTTP 호출에 대한 응답 시간 기준으로 데이터를 표시
- **SQL 시간**: SQL 수행 시간을 기준으로 데이터를 표시
- **CPU 사용 시간**: CPU 사용 시간을 기준으로 데이터를 표시

조회 결과 필터링

트랜잭션 맵의 차트 영역, 오른쪽 위에서는 Total, Error, Read 문구를 선택해 결과를 구분할 수 있습니다.

Total: 전체 트랜잭션 표시

Error: 에러가 발생한 트랜잭션만 표시

Read: 서버에서 읽은 트랜잭션 개수를 표시

예를 들어 에러를 선택한 후 차트의 트랜잭션을 드래그하면 에러 상태의 트랜잭션 목록이 표시된 **트레이스 분석** 창이 열립니다. 다시 전체 트랜잭션을 조회하려면 **Total**을 클릭하세요.

Y축 조정하기

- 차트의 왼쪽 위에 ∧ 또는 ∨ 버튼을 선택하면 Y축의 최댓값을 쉽게 변경할 수 있습니다. 또한 키보드의 위 또는 아래 방향 버튼을 눌러 같은 기능을 이용할 수 있습니다.

  Y축의 구간에 따라 ∧ 또는 ∨ 버튼, 키보드 버튼을 이용할 때 동작 방식은 다음을 참조하세요.

  - 10초에서 50초 구간은 10초 간격으로 변경
  - 50초에서 300초는 50초 간격으로 변경
  - 300초 이상은 100초 간격으로 변경
- Y축 최대값을 직접 입력해 변경할 수 있습니다. ✎ 버튼을 선택한 후 원하는 값을 입력하고 **적용** 버튼을 선택하세요.

트랜잭션 맵의 차트에 표시되는 트랜잭션의 최대 개수(**Limit**)는 2,000,000개입니다.

필터를 적용해서 조회할 때는 Total(화면에 표시되는 개수)보다 Read 값이 더 클 수 있습니다.

과거 데이터 조회하기

트랜잭션 맵은 실시간 모니터링을 기본 제공합니다. 기본값으로 5분 동안의 데이터를 실시간으로 조회할 수 있습니다. 시간 선택자의 녹색 버튼을 클릭해 원하는 조회 시간을 선택하세요.

과거 데이터를 조회하려면 시간 선택자에서 일시 정지 (⏸) 버튼을 클릭하세요. 기간 조회 모드로 변경되며 최대 1시간까지의 트랜잭션 데이터를 조회할 수 있습니다. 사용자가 설정한 날짜와 시간 기준으로 **트랜잭션 맵**의 차트 데이터를 갱신합니다.

과거 데이터를 조회하는 동안 언제든지 조회를 정지할 수 있습니다.

- 필터를 바꾸고 싶다면, 먼저 조회를 정지한 뒤 필터를 변경하고 다시 조회하세요.
- 필터를 바꾸지 않고 다시 조회 버튼을 누르면, 정지한 부분부터 이어서 조회를 계속합니다.

과거 데이터를 조회한 다음 실시간 모드로 전환하려면 ▶ 버튼을 클릭하세요.

> • 과거 조회 범위는 트랜잭션의 양에 따라 달라질 수 있습니다.

필터 적용하기

과거 데이터를 조회하는 경우 필터를 적용해 원하는 데이터만 조회할 수 있습니다.

1. 시간 선택자에서 일시 정지(⏸) 버튼을 클릭하면 필터 입력 상자가 나타납니다.
2. 필터 입력 상자를 클릭하면 아래와 같이 선택할 수 있는 필터 항목 목록이 나옵니다.
   - 일반 필터

항목	설명
로그인 ID	사용자의 로그인 식별자(아이디)로 조회
WClientID	클라이언트별 고유 식별자로 조회 (주의: 완전 일치로 조회)
트랜잭션 ID	개별 트랜잭션의 고유 식별자 (주의: 완전 일치로 조회)
Referer	트랜잭션 요청 시 전달된 Referer(참조 페이지 주소)로 조회
멀티 트랜잭션 ID	여러 트랜잭션을 묶는 고유 식별자 (주의: 완전 일치로 조회)
커스텀 TID	사용자가 직접 정의한 트랜잭션 식별자(Custom Transaction ID)로 조회
에러(클래스, 단계, 메시지)	트랜잭션에서 발생한 에러의 종류, 단계, 메시지로 조회(부분 일치 포함)
도시	트랜잭션이 발생한 사용자의 도시 정보로 조회
국가	트랜잭션이 발생한 사용자의 국가 정보로 조회
HTTP 메소드	HTTP 요청 방식(GET, POST 등) (주의: 완전 일치로 조회)
유저 에이전트	트랜잭션 요청에 사용된 브라우저 또는 클라이언트 정보(User Agent)로 조회

- 스텝(STEP) 필터
  STEP이란 트랜잭션을 수행할 때 수집되는 프로파일 데이터를 의미합니다. 히트맵 드래그 시 트랜잭션 LIST 에서 트랜잭션을 클릭해서 보는 Table View 나 Tree View 에 보이는 순차 데이터를 STEP이라고 합니다.

항목	설명
DBC	트랜잭션 수행 시 실행되는 메소드 정보가 저장된 STEP
HTTPC	HTTP 호출 정보가 저장된 STEP
MESSAGE	트랜잭션 수행 시 다양한 텍스트 정보가 저장된 STEP
METHOD	트랜잭션 수행 시 실행되는 메소드 정보가 저장된 STEP
SQL	수행된 쿼리 정보가 저장된 STEP
SQL_BIND	실제로는 존재하지 않고 SQL 스텝에 포함된 데이터. SQL 호출 시 바인드 변수와 치환 변수를 조회할 수 있는 STEP (주의: 민감 정보 조회 권한과 복호화키가 있어야 조회 가능)
ALL	위에 있는 모든 STEP 정보를 조회하는 스텝 (주의: 민감 정보 조회 권한과 복호화키가 있어야 조회 가능)

- 숫자 입력 필터

항목	설명
경과시간	트랜잭션이 수행되는 데 걸린 전체 시간
HTTP 호출 시간	HTTP 요청에 대한 응답 시간
HTTP 호출 건수	트랜잭션 동안 발생한 HTTP 요청의 횟수
SQL 시간	트랜잭션에서 실행된 SQL 쿼리의 총 실행 시간
SQL 건수	트랜잭션에서 실행된 SQL 쿼리의 개수

항목	설명
상태	트랜잭션의 상태(예: 정상, 에러 등)

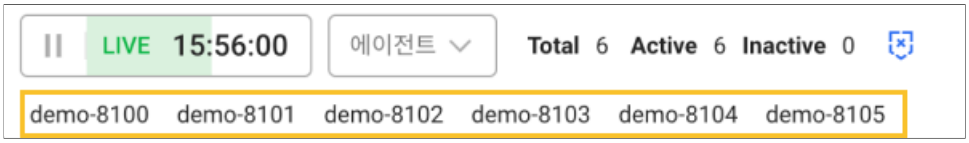
3. 필터링 항목에 원하는 값을 입력하세요. 필터링은 필터 키, 연산자, 값을 모두 입력해야 합니다. 입력이 완료되지 않은 필터가 남아 있는 경우, 검색이 불가능합니다.
4. 필터 입력 완료 후 버튼을 클릭하세요.

Top 30 목록과 **트랜잭션 맵** 차트에 필터링한 데이터를 갱신합니다.

> • 다중 조건을 설정해 필터링할 수 있습니다. 추가한 조건은 `&&` 기준으로 적용됩니다. 예를 들어, 트랜잭션이 `/account/save/employee/seoul` 이면서 경과 시간이 3초 이상인 경우 다음과 같이 적용할 수 있습니다.
>
> `트랜잭션` `=` `/account/save/employee/seoul` `경과 시간 (ms)` `>=` `3`

에이전트 확인

에이전트 연결 상태 확인하기



화면 왼쪽 위, 시간 선택자의 오른쪽에서는 해당 프로젝트와 연결된 에이전트의 상태를 확인할 수 있는 정보를 제공합니다. 이를 통해 모니터링 대상 서버의 동작 여부를 바로 확인할 수 있습니다.

- **Total**: 프로젝트와 연결된 모든 에이전트의 수
- **Active**: 활성화된 에이전트의 수
- **Inactive**: 비활성화된 에이전트의 수
- : 비활성화된 에이전트를 표시하거나 감출 수 있습니다.

에이전트별 모니터링

기본적으로 차트에는 모든 에이전트로부터 수집한 지표들을 차트에 표시하지만 에이전트별로 데이터를 조회할 수도 있습니다. 시간 선택자 아래에 위치한 에이전트를 하나 또는 둘 이상을 선택하세요. 선택한 에이전트의 트랜잭션 데이터로 차트를 갱신합니다.

> 에이전트를 하나 또는 둘 이상을 선택한 상태에서 다시 모든 에이전트를 선택하려면 선택을 해제하거나 **Total**을 클릭하세요.

> 프로젝트에 연결된 에이전트의 수가 많을 경우 에이전트의 이름을 짧게 설정하는 것이 효율적입니다. 에이전트 이름 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

분류별 에이전트 모니터링

에이전트 설정에서 분류한 단위로 여러 에이전트를 쉽게 모니터링할 수 있습니다.

- **에이전트**: 개별 에이전트
- **종류별**: 에이전트의 종류(예: 웹서버, 데이터베이스 등)로 분류된 그룹, `whatap.okind` 로 설정
- **서버별**: 서버(노드)별로 분류된 그룹, `whatap.onode` 로 설정
- **종류별 에이전트**: `whatap.okind` 로 분류된 에이전트 목록
- **서버별 에이전트**: `whatap.onode` 로 분류된 에이전트 목록

> 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

에이전트 검색하기

프로젝트에 연결된 에이전트의 목록이 많아 찾기 어렵다면 에이전트 검색 기능을 이용하세요. 화면 오른쪽 위에 **검색** 버튼을 클릭하면 문자열을 입력할 수 있는 상자가 나타납니다.

입력한 문자열과 일치하는 에이전트만 에이전트 목록에 표시됩니다. 검색한 에이전트를 기준으로 대시보드의 데이터를 필터링하려면 문자 입력 상자 오른쪽에 **선택** 버튼을 클릭하세요.

선택한 에이전트 항목을 초기화하려면 문자 입력 상자 오른쪽에 **닫기** 버튼을 클릭하세요.

상위 목록 확인하기

Top 30 섹션에서는 **트랜잭션** 및 **도메인**, **IP**, **에이전트** 기준으로 트랜잭션 수행 건수가 많은 상위 30개의 목록을 표시합니다.

- **트랜잭션**: 트랜잭션 URL 기준으로 집계된 상위 30개의 목록
- **도메인**: 클라이언트가 접속한 IP 주소에 지정된 인터넷 주소를 기준으로 집계된 상위 30개의 목록
- **IP**: 클라이언트의 IP 주소를 기준으로 집계된 상위 30개의 목록
- **에이전트**: 해당 프로젝트에 포함된 에이전트를 기준으로 집계된 상위 30개의 목록

목록에서 개별 항목을 선택하면 트랜잭션 맵 차트에 선택한 항목 기준으로 데이터를 반영합니다. 여러 개의 항목을 다중 선택하려면 `Ctrl` (Windows/Linux) 또는 `CMD` (Mac) 키를 누른 상태에서 원하는 항목을 클릭하세요.

> **Top 30** 섹션의 목록에서 키보드의 위 또는 아래 방향 버튼을 눌러 항목을 이동할 수 있습니다.

사용자 IP 주소 추가하기

TOP 30 섹션의 **IP** 기준으로 트랜잭션 수행 건수를 조회할 때 특정 IP 주소를 추가해 상시 확인할 수 있는 기능을 제공합니다.

1. **TOP 30** 섹션에서 **IP**를 선택하세요.
2. 목록의 가장 위에 입력란에 특정 IP 주소를 입력하세요.
3. ＋ 버튼을 클릭하세요.



목록의 가장 위에 입력한 IP 주소가 고정되어 표시됩니다.

> ⓘ 추가한 IP 주소를 삭제하려면 🗑 버튼을 클릭하세요.

사용자 맞춤 항목 모니터링

특정 트랜잭션, 도메인, IP를 **커스텀** 탭에 추가해 응답 시간 분포도 차트에서 선택한 항목만을 확인할 수 있는 기능을 제공합니다. 이를 통해 사용자는 문제가 발생하는 특정 항목을 효율적으로 감시하고 분석할 수 있습니다.

1. **커스텀** 탭을 클릭하세요.
2. **트랜잭션** 또는 **도메인**, **IP** 항목은 직접 입력, **Top 30** 목록 내 선택, 항목 검색이 가능합니다.
3. 추가한 항목을 클릭하세요.



선택한 항목들의 응답 시간 분포도가 오른쪽 차트에 차트에 실시간으로 필터링되는 것을 확인할 수 있습니다.

> ⓘ 추가한 커스텀 항목을 삭제하려면 🗑 버튼을 클릭하세요.

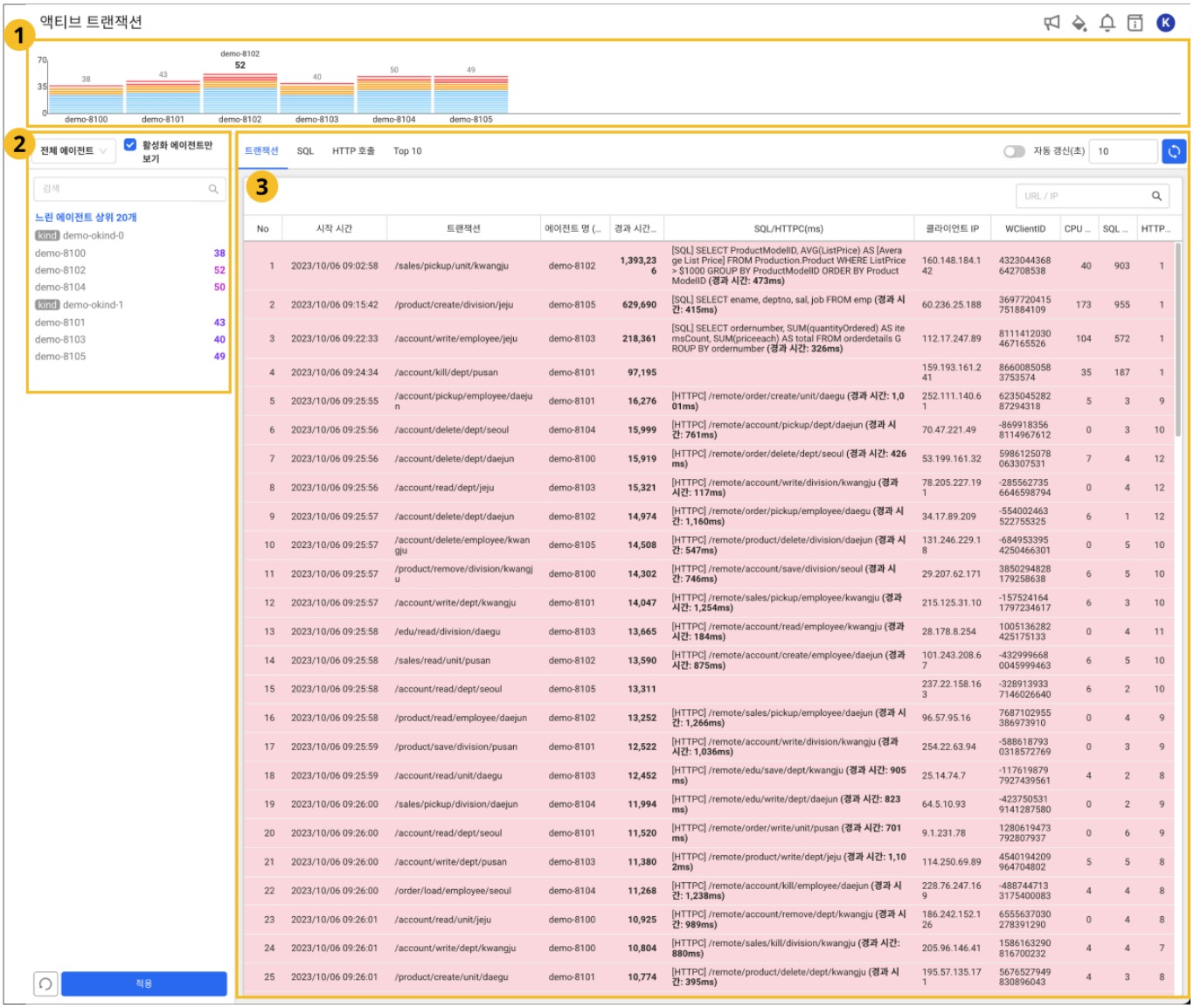
액티브 트랜잭션 대시보드

홈 화면 > 프로젝트 선택 > **대시보드** > **액티브 트랜잭션**

현재 진행 중인 트랜잭션의 URL, SQL, HTTP 호출 정보로 이상 트랜잭션을 신속하게 파악할 수 있으며, 상세 트레이스를 통해서 지연 구간을 상세하게 분석할 수 있습니다.

> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면 현재 진행 중인 트랜잭션과 연계된 데이터베이스의 SQL 성능을 조회할 수 있습니다. **연계 프로젝트 관리** 메뉴와 연계 프로젝트 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

화면 구성



1 에이전트 별 액티브 트랜잭션

전체 애플리케이션 서버의 목록과 진행 중인 트랜잭션의 총 개수를 표시합니다. 차트의 각 그래프에 마우스를 오버하면 트랜잭션 상태 속도를 표시합니다. 트랜잭션 상태의 속도는 Normal, Slow, Very Slow로 분류합니다.

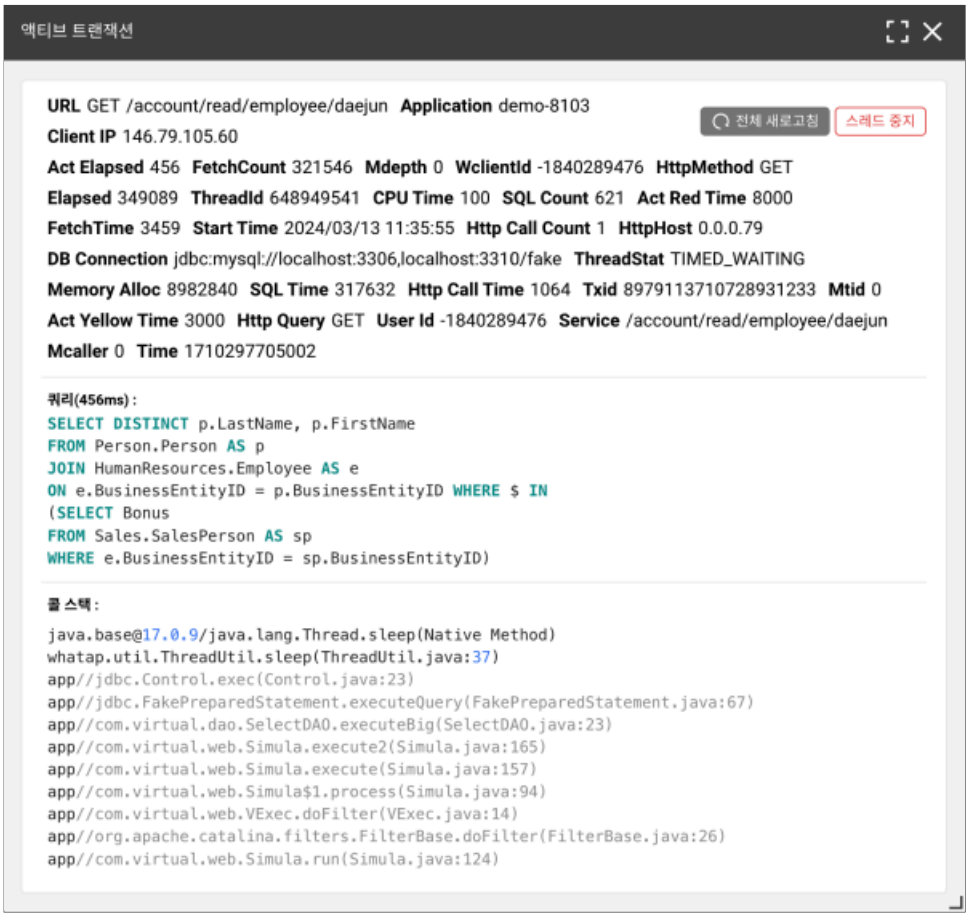
2 에이전트 목록

애플리케이션 서버의 목록과 진행 중인 트랜잭션의 총 개수를 표시합니다. 해당 애플리케이션 서버를 선택하면 3 트랜잭션 목록에 세부 정보가 나타납니다.

3 트랜잭션 목록

진행 중인 트랜잭션의 URL, SQL, HTTP 호출 정보의 상세 내용을 표시합니다. Top 10 항목 탭을 선택하면 각 항목별 경과 시간이 높은 순으로 정렬된 표를 확인할 수 있습니다. 마찬가지로 트랜잭션 상태의 속도는 Normal, Slow, Very Slow로 분류합니다.

트랜잭션 목록에서 항목을 선택하면 선택한 트랜잭션에 대한 상세 정보를 확인할 수 있는 **액티브 트랜잭션** 창이 나타납니다.



- 실행 중인 트랜잭션에 대한 상세 정보 및 외부 호출, 쿼리, 콜 스택 등을 확인할 수 있습니다.
  - **전체 새로고침**: 진행 중인 트랜잭션의 정보를 갱신할 수 있습니다.
  - **스레드 중지**: 진행 중인 트랜잭션을 중지할 수 있습니다. 스레드 중지를 위해서는 **시크릿키**가 필요합니다. **시크릿키**에 대한 자세한 내용은 다음 문서를 참조하세요.
  - **연계 DB 세션**: 진행 중인 트랜잭션과 연계된 데이터베이스의 SQL 성능을 조회할 수 있습니다. 선택한 트랜잭션에서 연계

프로젝트로 SQL을 실행한 트랜잭션일 때만 표시됩니다. 연계 프로젝트 추가 및 관리에 대한 자세한 내용은 다음 문서를 참조하세요.

> **스레드 중지** 기능은 애플리케이션에서 실행 중인 스레드를 강제로 종료하는 기능입니다. 그러나 이 기능은 모든 경우에 동작하지 않을 수 있습니다. 예를 들어, 스레드가 IO 작업을 수행 중이라면 스레드를 즉시 중지할 수 없습니다.
>
> **스레드 중지** 기능을 사용하면 스레드가 IO 작업을 수행 중이더라도 종료를 시도합니다. 하지만 스레드가 IO 작업을 완료하기 전까지는 중지되지 않습니다. 즉, 스레드는 중지 요청을 받아들이지만, 작업이 완료되고 다음 작업으로 넘어가기 전까지 중지되지 않습니다.
>
> 이러면 스레드를 강제로 종료하는 것은 비즈니스 로직에 문제가 발생할 수 있으며, 예기치 못한 동작을 유발할 수 있기 때문에 권장하지 않습니다. 그래도 스레드를 중지해야 한다면 애플리케이션 전체를 중지한 후 재시작해야 합니다.

- 이미 종료된 트랜잭션이라면 **트랜잭션 보기** 버튼을 표시됩니다. 버튼을 선택하면 **트랜잭션 정보** 창이 나타납니다. **트랜잭션 정보**에 대한 자세한 내용은 다음 문서를 참조하세요.

에이전트 목록 필터링

2 **에이전트 목록**에서 필터링 대상을 선택하거나 검색해 트랜잭션 목록에서 개별로 확인할 수 있습니다.

- 현재 활성화된 에이전트를 확인하려면 **활성화 에이전트만 보기** 체크박스를 선택하세요.
- **전체 에이전트** 목록 상자를 클릭하세요. 에이전트를 카테고리 별로 분류해 확인할 수 있습니다.
- 진행 중인 트랜잭션 정보를 에이전트 별로 확인하려면 에이전트를 선택한 다음 아래에 **적용** 버튼을 클릭하세요. 선택을 취소하려면 ↺ 버튼을 선택하세요.

자동 갱신 설정

진행 중인 트랜잭션을 설정한 간격마다 자동 갱신할 수 있습니다. **자동 갱신(초)** 토글 버튼을 선택해 옵션을 활성화한 다음 오른쪽 텍스트 입력 상자에 갱신 간격을 입력하세요.

자동 갱신 설정을 비활성화하려면 **자동 갱신(초)** 토글 버튼을 다시 선택하세요.

- 시간은 초 단위로 갱신합니다.
- 자동 갱신 설정을 비활성화한 상태에서 트랜잭션 목록을 갱신하려면 버튼을 선택하세요.

트랜잭션 목록 필터링

3 **트랜잭션 목록**에서 URL 또는 IP 값을 검색해 필터링할 수 있습니다. 오른쪽 위에 **URL / IP** 텍스트 입력 상자에 필터링할 대상을 입력한 다음 엔터를 입력하거나 버튼을 선택하세요. 입력한 텍스트 값을 포함하는 트랜잭션 목록으로 필터링합니다.

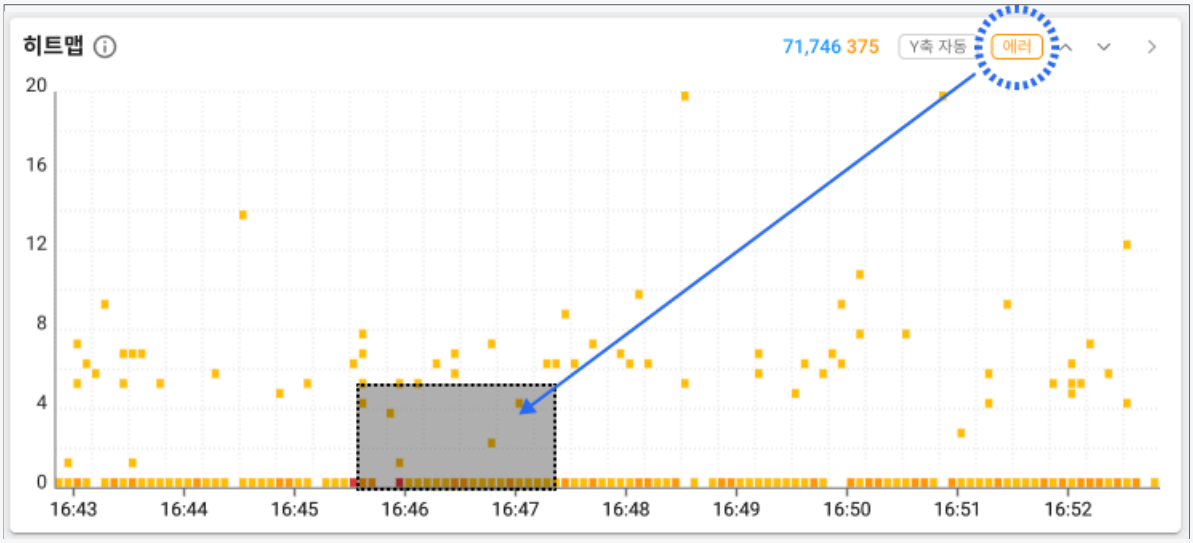
필터링한 목록을 초기화하려면 오른쪽 위에 **URL / IP** 텍스트 입력 상자의 텍스트를 삭제한 다음 엔터를 입력하거나 버튼을 선택하세요.

액티브 트랜잭션 설정에 관한 자세한 내용은 다음 문서를 참조하세요.

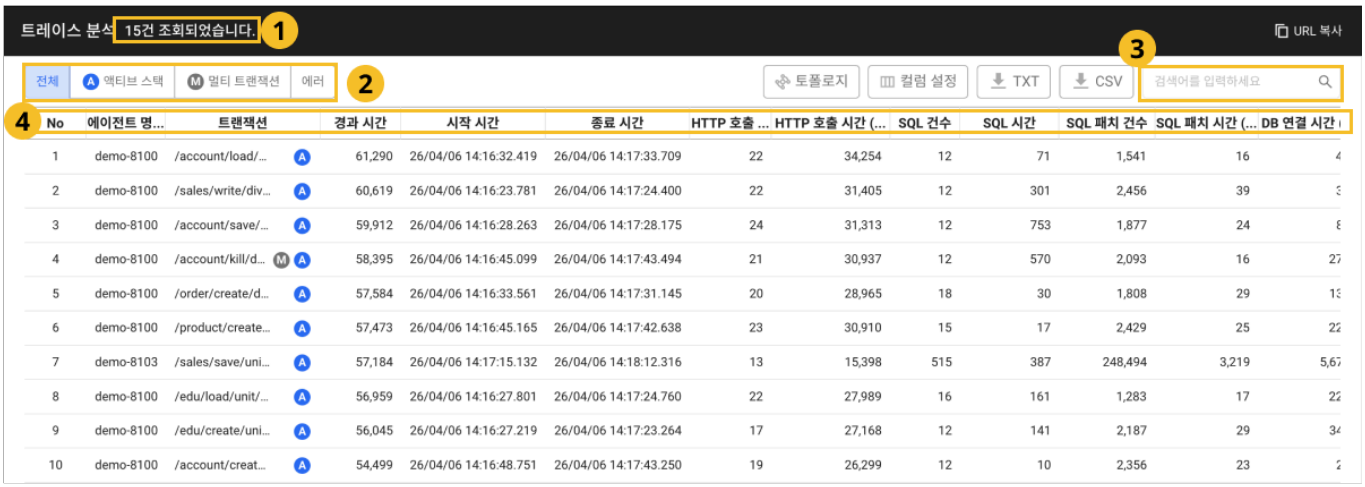
히트맵/트레이스 분석

애플리케이션 대시보드의 **히트맵** 위젯을 통해 **트레이스 분석** 창을 열어 트랜잭션 목록과 각 트랜잭션 하위의 스텝 정보를 한 번에 확인할 수 있습니다. 히트맵 위젯에서 원하는 영역을 드래그하세요. 선택한 영역의 트랜잭션 목록과 하위 스텝 정보를 포함하는 **트레이스 분석** 창이 나타납니다.

에러가 발생한 트랜잭션 정보만 분류해 확인하려면 **애플리케이션 대시보드**의 **히트맵** 위젯의 오른쪽 위에 **에러** 버튼을 선택한 다음 차트 영역을 드래그하세요.

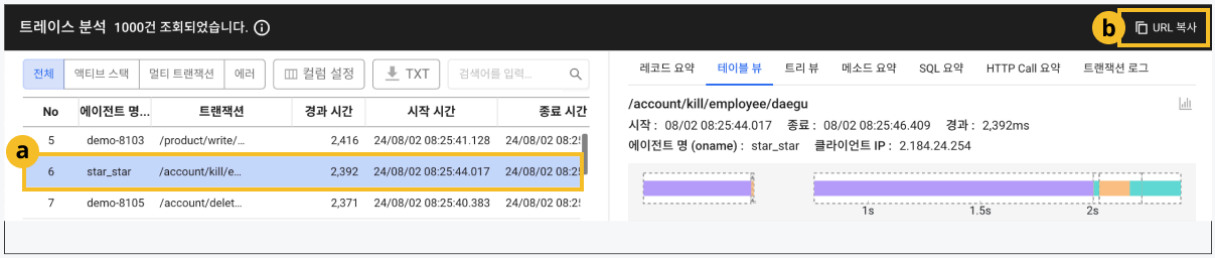


기본 화면 안내



- 1 **트랜잭션 조회 건수**: 히트맵 위젯에서 드래그한 영역의 트랜잭션 목록의 개수를 확인할 수 있습니다. 최대 1000건까지 조회할 수 있습니다.
- 2 **트랜잭션 분류**: A **액티브 스택**과 M **멀티 트랜잭션**, **에러** 항목을 구분하여 목록을 확인할 수 있습니다.
- 3 **트랜잭션 검색**: 트랜잭션 명과 에이전트 명, 에러 메시지를 기준으로 트랜잭션 목록을 검색할 수 있습니다.
- 4 **목록 정렬하기**: 테이블의 헤더 컬럼 항목을 클릭하면 선택한 항목을 기준으로 트랜잭션 목록을 정렬할 수 있습니다.
- 에러가 발생한 트랜잭션 항목은 목록에서 빨간색으로 표시됩니다.

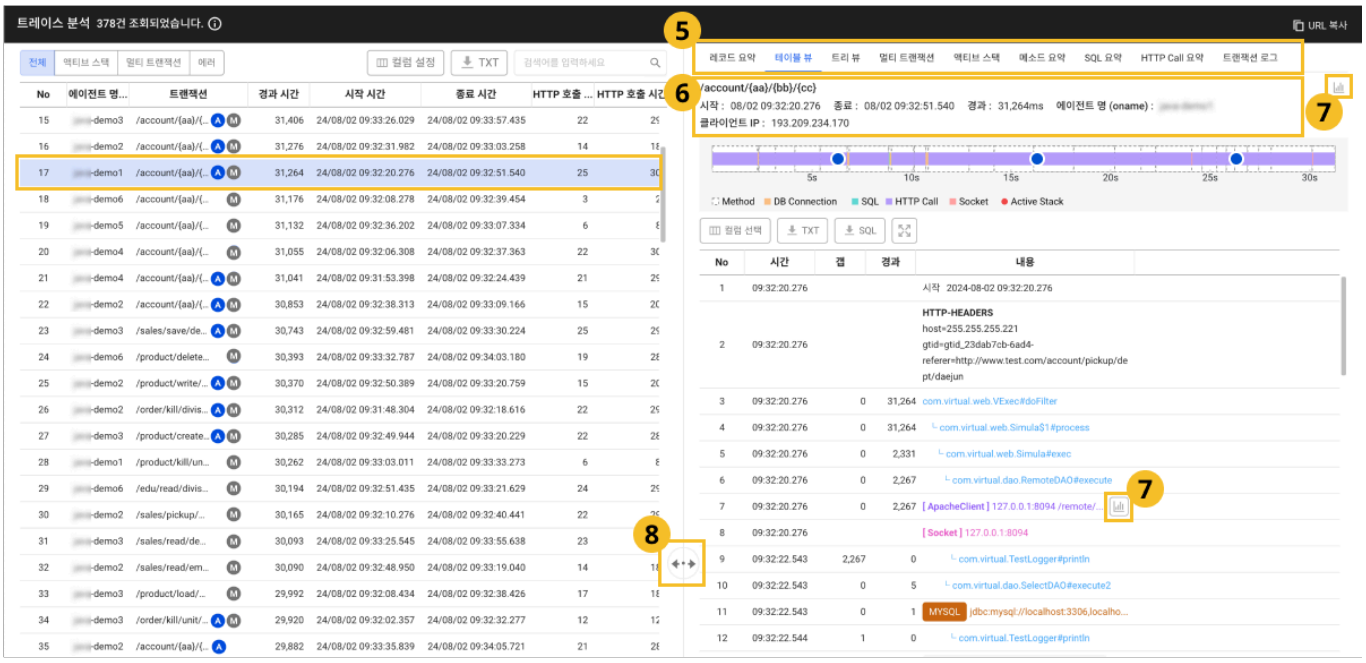
- 테이블에서 자주 보길 원하는 컬럼 항목을 앞으로 배치하거나 추가, 숨기기할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- **히트맵** 위젯에서 다른 영역을 드래그하면 새로운 트레이스 분석 창을 불러올 수 있습니다. 기존에 열린 트레이스 분석 창과 비교해 확인할 수 있습니다.
- 현재 사용자가 보고 있는 트랜잭션 항목을 다른 사용자에게 공유할 수 있습니다. 트랜잭션 항목을 선택한 상태(a)에서 트레이스 분석 창의 b **URL 복사** 버튼을 선택하세요. 복사한 URL 주소를 다른 사용자에게 전달하세요. URL 주소를 전달 받은 사용자는 브라우저 주소 표시줄에 URL 주소를 입력하면 같은 항목을 빠르게 확인할 수 있습니다.



트레이스 분석하기

트랜잭션 목록에서 상세 내용을 확인할 항목을 선택하세요. 화면 오른쪽에 선택한 트랜잭션에 대한 상세 분석 내용을 표시합니다. 이 상태에서 다른 트랜잭션 항목을 선택하면 선택한 트랜잭션 항목에 대한 상세 분석 내용으로 변경됩니다.

와탭은 트랜잭션의 성능을 분석하기 위해 이름과 클라이언트 정보 등의 속성, 트랜잭션의 처리 성능 그리고 각 구간별 상세 수행 이력을 수집하고 보여줍니다.



- 5 영역에서는 스텝 정보를 다양한 형식으로 확인할 수 있습니다. 원하는 형식의 탭을 선택하세요.
- 6 영역에서는 트랜잭션에 대한 기본 정보를 확인할 수 있습니다. 왼쪽 목록의 컬럼 항목과 일치하는 정보입니다.

- 7 : 해당 URL 또는 각 수행 구간의 통계 데이터 창이 나타납니다. 통계 데이터 창의 그래프 차트에서 원하는 시간을 클릭하면 **통계** 또는 **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간 기준으로 통계 데이터를 조회할 수 있습니다.
- 8 왼쪽 또는 오른쪽으로 드래그해서 화면의 크기를 조절할 수 있습니다. 더 많은 정보를 보길 원할 때 이용하면 유용합니다.

> - **통계** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
> - **트랜잭션 검색** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
> - ◀ 또는 ▶ 버튼을 클릭하면 트랜잭션 목록 또는 상세 분석 화면을 전체 크기로 변경할 수 있습니다.

수집하는 **스텝(Step)**의 종류는 다음과 같습니다. 다이어그램과 텍스트 색상을 구분해 표시합니다.

- **DB Connection** 스텝 START-END

  RDB에 대한 연결에 대한 성능을 포함합니다. 스텝 정보에는 이름, 응답시간, 에러를 포함합니다.

- **SQL** 스텝 START-END

  JDBC SQL에 대한 성능을 포함합니다. 스텝 정보에는 연결 정보, SQL문, 에러가 포함되어 있습니다.

- **HTTP Call** 스텝 START-END

  외부 http 서비스 호출에 대한 성능을 포함합니다. 스텝 정보에는 url, host, port, 응답시간, 에러가 포함됩니다.

- **Message** 스텝 ADD

  트레이스를 수집하는 과정에서 비정형적인 모든 구간에 대한 이력을 수집할 때 메시지 스텝을 사용합니다.

- **Socket** 스텝 ADD

  Socket 오픈을 표현하는 스텝입니다.

- **Method** 스텝 START-END

  메소드 응답시간을 추적합니다.

- **Active Stack** 스텝 START-END

  액티브 스택에 대한 정보를 포함합니다. 별도 스레드가 생성하여 트레이스에 추가하는 방식으로 수집합니다.

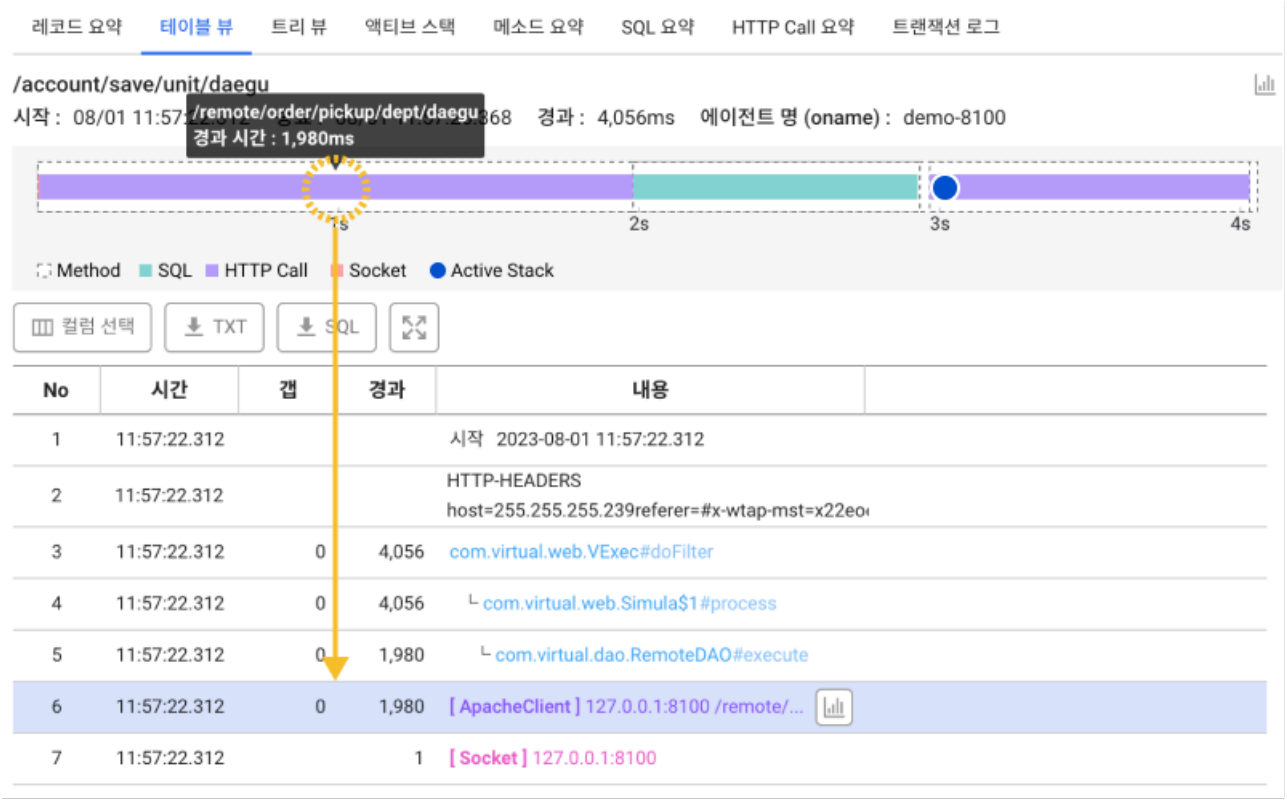
> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면 연계한 프로젝트의 성능 지표를 조회할 수 있습니다. 연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

> ⓘ • 와탭 모니터링 서비스에서 **스텝(Step)**은 **스팬(Span)**과 같은 뜻으로 사용됩니다.
> • 지난 2023년 7월 25일, 릴리스된 **트레이스 분석** 기능의 변경 사항을 확인하려면 다음 문서를 참조하세요.

테이블 뷰

테이블 뷰 탭에서는 트랜잭션의 수행 과정을 시간의 순서대로 확인할 수 있습니다.

- 트랜잭션을 수행 구간별로 분류해 다이어그램을 통해 확인할 수 있습니다. 전체 경과 시간 중 구간별 소요된 시간, 가장 오래 소요된 구간을 빠르게 파악할 수 있습니다.
- 다이어그램의 각 구간을 선택하면 해당 스텝이 위치한 테이블 목록으로 이동합니다.

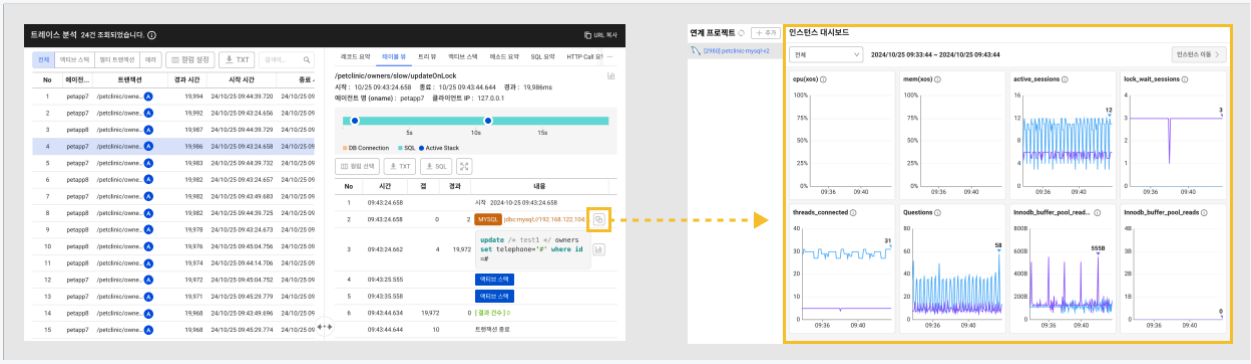


- 파란색 원(🔵)이 위치한 영역은 액티브 스택이 수집된 순간입니다. 파란색 원을 선택하면 **액티브 스택** 버튼이 위치한 테이블 목록으로 이동할 수 있습니다. **액티브 스택** 버튼을 선택하면 해당 구간 동안 수행된 스텝 정보를 확인할 수 있습니다.

> ◦ 트랜잭션 목록에서 A 아이콘이 표시된 항목에서 확인할 수 있습니다.
> ◦ 액티브 스택에 대한 자세한 내용은 다음 문서를 참조하세요.

- **컬럼 선택**: 테이블 목록에 컬럼으로 메모리 누적 정보 및 CPU 누적 정보를 추가하거나 감출 수 있습니다. 컬럼 정보는 다음과 같습니다.
  - **No**: 스텝의 발생 순서
  - **시간**: 각 스텝의 시작 시각
  - **갭**: 직전 스텝의 시작 시각부터 현재 스텝으로 넘어가기까지 대기 시간, 외부 요인으로 지연될 경우 경과 시간과 차이가 있을 수 있습니다.
  - **경과**: 각 메소드 시작부터 종료까지 총 소요 시간
  - **내용**: 해당 스텝의 세부 수행 내용
- **TXT**: 트랜잭션 기본 정보 및 구간별 수행 정보를 TXT 형식의 파일로 다운로드할 수 있습니다.
- **SQL**: 트랜잭션 기본 정보 및 SQL 수행 정보를 TXT 형식의 파일로 다운로드할 수 있습니다.
  - **치환값 포함 다운로드**: 바인드 변수값을 원래의 값으로 치환하여 SQL 통계 데이터를 다운로드합니다. 보안키(`paramkey`)가 설정되어 있다면 보안키 입력창이 나타납니다. 보안키를 입력를 입력해야 다운로드할 수 있습니다. 보안키와 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.
  - **치환값 미포함 다운로드**: 바인드 변수값을 치환하지 않은 상태로 SQL 통계 데이터를 다운로드합니다.
- : 트랜잭션 요약 정보와 다이어그램을 감추고 테이블 목록만 확인할 수 있습니다. 버튼을 선택하면 감춰진 정보를 다시 표시합니다. 테이블 목록이 긴 경우 이 기능을 이용하면 유용합니다.
- SQL 스텝을 선택하면 파라미터를 조회할 수 있는 **SQL** 창이 나타납니다. 자세한 내용은 다음 문서를 참조하세요.

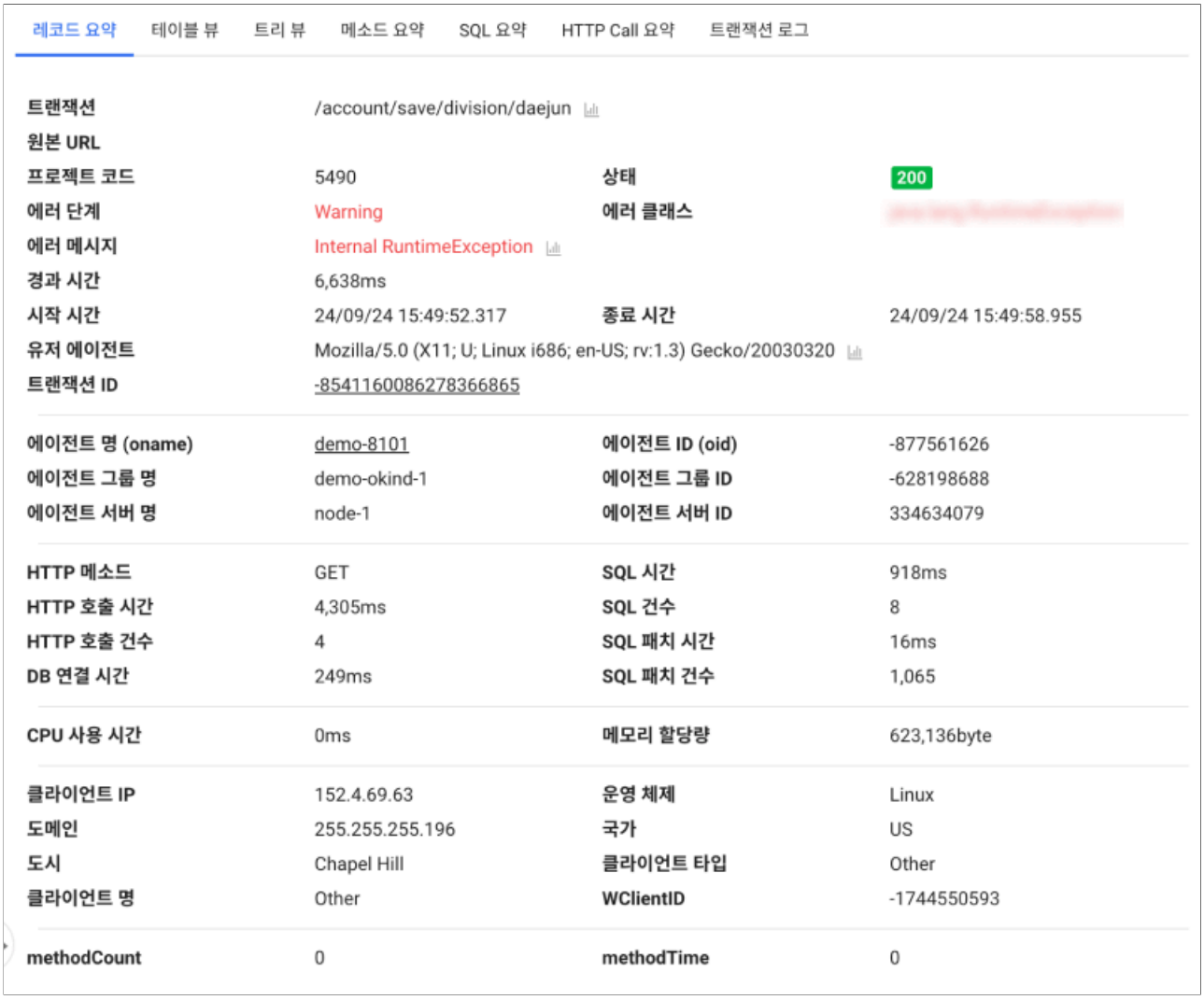
> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면, **DB Connection 스텝**에서 연계한 프로젝트의 성능 지표를 조회할 수 있습니다.



연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

레코드 요약

트랜잭션의 기본 정보를 확인하려면 **레코드 요약** 탭을 선택하세요. 트랜잭션의 기본 정보 및 해당 트랜잭션이 수행된 에이전트 관련 정보, 메모리 할당 사용량, 클라이언트 관련 정보를 확인할 수 있습니다.



분류	속성	설명
트랜잭션	트랜잭션	애플리케이션에서 실행된 API 또는 URL 호출, 해당 트랜잭션의 경로입니다. 정규 표현식을 사용해 일정한 형식으로 변환합니다.
	원본 URL	실제로 호출된 URL 경로입니다. 변수를 포함하지 않은 구체적인 요청 경로입니다. 원본 URL이 있을 경우 원본 URL 표시, 원본 URL이 없을 경우 표시하지 않습니다.
	프로젝트 코드	와탭 모니터링 서비스에 등록한 프로젝트의 식별 코드입니다.
	상태	해당 트랜잭션의 HTTP 응답 상태 코드입니다.

분류	속성	설명
	에러 단계	해당 트랜잭션에서 발생한 에러 레벨입니다. Warning 또는 Critical로 표시됩니다.
	에러 클래스	해당 트랜잭션에서 발생한 에러 관련 클래스입니다.
	에러 메시지	해당 트랜잭션에서 발생한 에러 메시지입니다.
	경과 시간	트랜잭션 수행 시간입니다.
	시작 시간	트랜잭션 시작 시각입니다.
	종료 시간	트랜잭션 종료 시각입니다.
	유저 에이전트	클라이언트 관련 정보를 추출하는데 이용하는 브라우저 정보입니다.
	Referer	클라이언트가 브라우저에서 이전에 이용한 페이지 주소 또는 유입 경로입니다.
	트랜잭션 ID	해당 트랜잭션의 고유 식별자(ID)입니다.
	멀티 트랜잭션 ID	멀티 트랜잭션의 고유 식별자(ID)입니다. 다른 애플리케이션의 트랜잭션과 병렬로 실행된 경우 표시됩니다.
에이전트	에이전트 명 (oname)	에이전트 이름입니다.
	에이전트 ID (oid)	에이전트의 고유 식별자(ID)입니다.
	에이전트 그룹 명	`okind` 옵션으로 설정된 에이전트 그룹 이름입니다.
	에이전트 그룹 ID	에이전트 그룹의 고유 식별자(ID)입니다.
	에이전트 서버 명	`ondoe` 옵션으로 설정한 에이전트 서버 이름입니다.

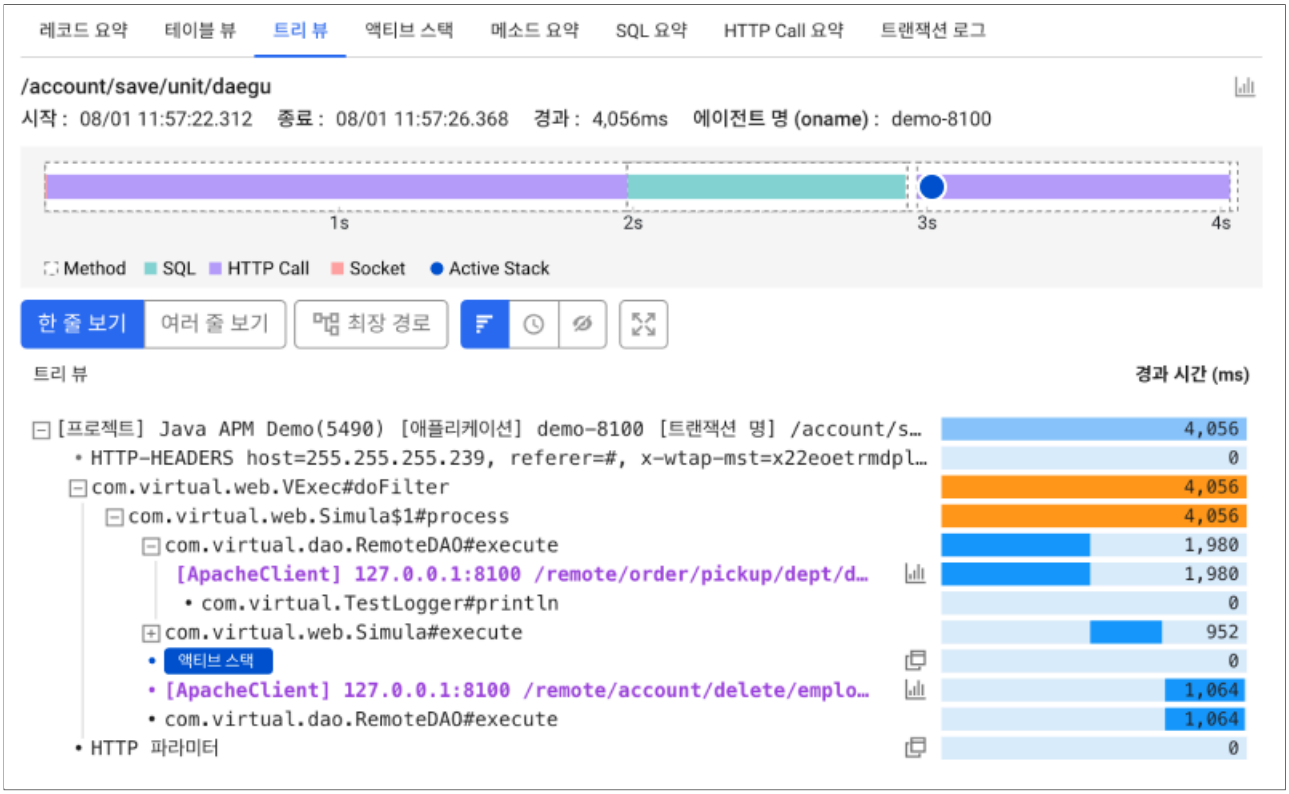
분류	속성	설명
	에이전트 서버 ID	에이전트 서버의 고유 식별자(ID)입니다.
스텝(Step)	HTTP 메소드	HTTP 메소드입니다. (GET, POST, PUT, HEAD 등)
	HTTP 호출 시간	외부 Http Call 시간입니다.
	HTTP 호출 건수	외부 HTTP Call 건수입니다.
	DB 연결 시간	데이터베이스에 연결되기까지 걸린 시간입니다.
	SQL 시간	SQL 수행 시간입니다.
	SQL 건수	SQL 수행 건수입니다.
	SQL 패치 시간	레코드를 조회하는 데 걸린 시간입니다. 중첩이 발생하거나 타 로직이 포함될 수 있습니다.
	SQL 패치 건수	SQL에서 데이터를 조회한 레코드 건수입니다.
자원(Resource)	CPU 사용 시간	트랜잭션 처리에 사용된 CPU 시간입니다.
	메모리 할당량	해당 트랜잭션에서 사용된 메모리 양입니다.
컨테이너	Pod 명	해당 트랜잭션을 처리한 Kubernetes Pod 이름입니다.
	컨테이너 ID	Pod 내에서 실행 중인 특정 컨테이너의 고유 식별자입니다.
클라이언트	클라이언트 IP	클라이언트의 IP 주소입니다.
	운영체제	브라우저가 실행되는 운영 체제 환경입니다.
	도메인	클라이언트가 접속한 IP 주소에 지정된 인터넷 주소입니다.
	국가	클라이언트의 국가 정보입니다.

분류	속성	설명
	도시	클라이언트의 국가 내 도시 정보입니다.
	클라이언트 타입	클라이언트가 이용한 브라우저 종류입니다.
	클라이언트 명	클라이언트가 이용한 기기 이름입니다.
	WClientID	클라이언트의 고유 식별자입니다.
Method	methodCount	트랜잭션에서 호출된 메소드의 수입니다.
	methodTime	메소드 실행에 소요된 시간입니다.

- 애플리케이션 종류나 설정, 스텝의 종류에 따라 수집하는 정보는 달라질 수 있습니다. **컨테이너** 항목은 애플리케이션이 컨테이너 환경에서 실행 중일 경우에만 표시됩니다.
- 에러 관련 항목은 해당 트랜잭션에서 에러가 발생할 경우 표시됩니다.
- 트랜잭션 속성 중 일부 항목은 통계 기능을 제공합니다. 버튼을 선택하면 시계열 차트를 통해 조회 시간 동안 발생한 트랜잭션 현황을 파악할 수 있는 창이 표시됩니다. 차트 내 막대 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동해 상세 조회할 수 있습니다.
- 와탭은 클라이언트와 관련한 정보를 기본 저장합니다. 사용자 데이터 수집과 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

트리 뷰

트랜잭션 수행 과정을 트리 형식으로 확인하려면 **트리 뷰** 탭을 선택하세요. 각 트랜잭션과 그에 속한 트레이스의 세부 정보, 트레이스의 시작 시간 및 소요 시간, 호출 관계를 확인할 수 있습니다. 다이어그램의 각 구간을 선택하면 해당 스텝이 위치한 트리뷰로 이동합니다.



- **한 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 한 줄로 표시해 트리 형식을 간격하게 정리할 수 있습니다.
- **여러 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 줄바꿈해 모두 표시합니다.
- **최장 경로**: 가장 긴 경로로 이동할 수 있습니다.
- **시간바 표시**: 경과 시간을 막대 형식의 차트로 표시합니다.
- **시간 표시**: 각 구간별 타임 스탬프, 갭, 경과 시간을 텍스트 형식으로 표시합니다.
  - 8초 이상: 초과 지연 상태로 빨간색으로 표현합니다.
  - 3초 이상 8초 미만: 지연 상태로 주황색으로 표현합니다.
  - 3초 미만: 정상 상태로 파란색으로 표현합니다.
- **시간 숨기기**: 시간 정보를 숨깁니다.
- : 트랜잭션 요약 정보와 다이어그램을 감추고 트리뷰만 확인할 수 있습니다. 버튼을 선택하면 감춰진 정보를 다시 표시합니다. 트리뷰 목록이 긴 경우 이 기능을 이용하면 유용합니다.
- : SQL 변수와 HTTP 쿼리를 조회할 수 있는 창이 나타납니다. 자세한 내용은 다음 문서를 참조하세요.

> 시작 및 소요 시간의 경우 트랜잭션 호출 환경에 따라 발생하는 시차를 상위 트랜잭션 내 트레이스와 매핑을 통해 보정하여 표현하기 때문에 실제 수집된 시간 데이터와 차이가 발생할 수 있습니다.

액티브 스택



액티브 스택 탭을 선택하면 해당 트랜잭션에서 수집한 액티브 스택 목록을 확인할 수 있습니다.

- 왼쪽의 **액티브 스택** 목록에서 수집 시점(No, 시간)을 선택하면 오른쪽에 해당 시점의 스택 정보를 표시합니다.
- **스택**: 액티브 스택 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

> - 트랜잭션 목록에서 A 아이콘이 표시된 항목에서 액티브 스택을 확인할 수 있습니다.
> - 액티브 스택에 대한 자세한 내용은 다음 문서를 참조하세요.

AI 액티브 스택 분석

Beta

액티브 스택 분석 버튼을 클릭하면 수집한 액티브 스택을 AI가 자동으로 분석합니다. 진행 중인 트랜잭션의 스택 덤프를 바탕으로

성능 문제를 찾고 개선 방법을 안내합니다.

분석 결과는 다음 4개 섹션으로 구성됩니다.

- **전체 요약**: 분석한 스택 수와 전체 실행 흐름을 요약합니다.
- **공통 패턴**: 반복적으로 나타나는 실행 패턴을 표시합니다. 패턴명, 발생 횟수, 설명을 포함합니다.
- **발견된 이슈**: 찾아낸 성능 문제를 심각도(**High**, **Medium**, **Low**)와 함께 표시하며, 판단의 바탕이 된 스택 정보도 제공합니다.

심각도	기준
High	데드락, 완전 블로킹, 동일 블로킹 4회 이상 반복
Medium	I/O 대기, 2~3회 락 경합, 느린 외부 호출
Low	단일 발생 대기, 가벼운 최적화 기회

- **권장 사항**: 우선순위에 따라 구체적인 개선 방법을 안내합니다.

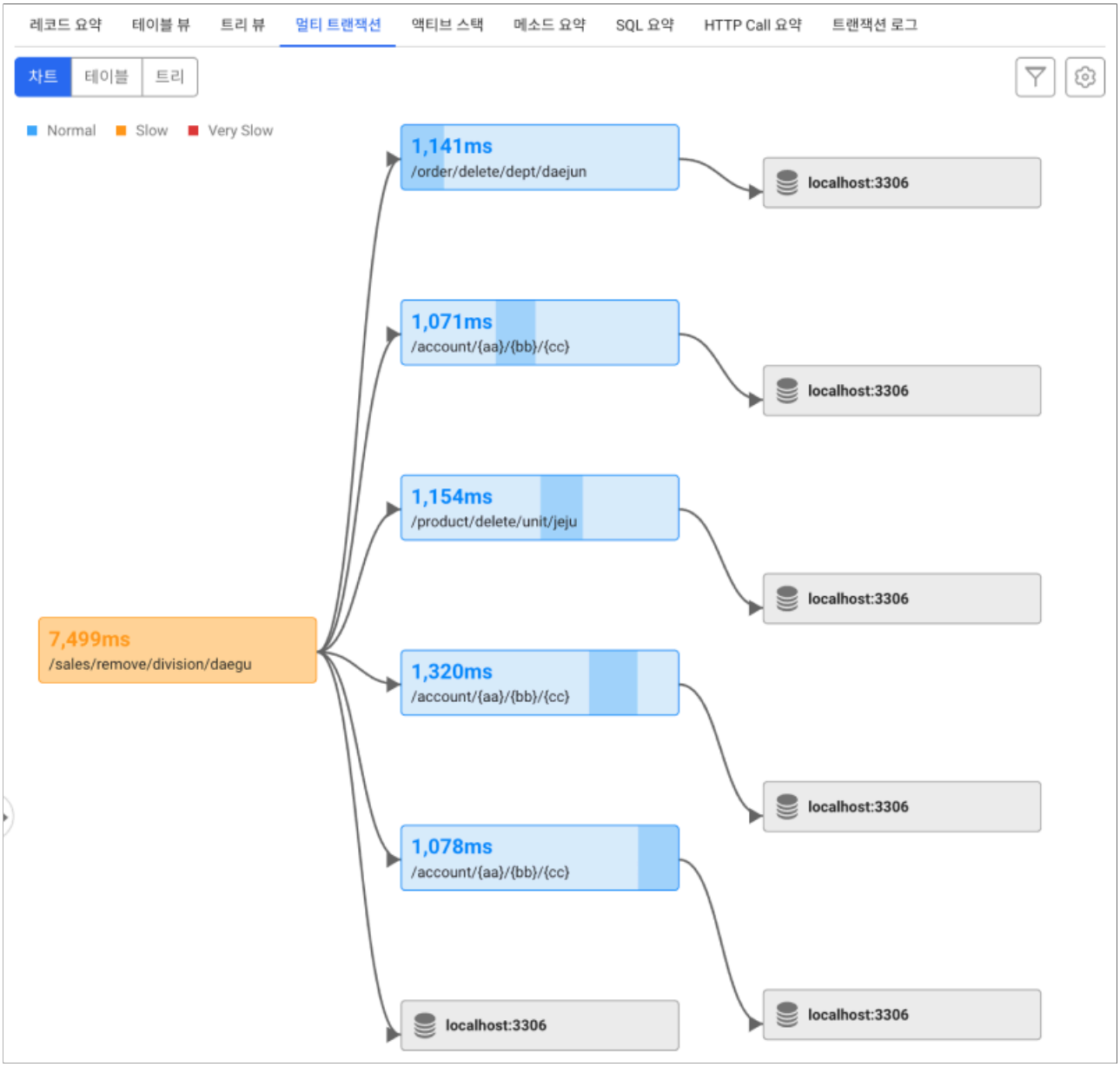
각 분석 항목에는 수집한 스택 번호(`No.N`)가 표시됩니다. 스택 번호를 클릭하면 해당 스택 원본을 확인할 수 있습니다.

> - AI가 만든 분석 결과로, 실제 상황과 다를 수 있습니다. 참고 자료로 활용하세요.
> - 프로젝트 **멤버** 권한 이상이 필요합니다.

멀티 트랜잭션

멀티 트랜잭션은 다른 에이전트나 프로젝트와의 연관된 트랜잭션을 의미합니다. **멀티 트랜잭션** 탭에서는 와탭 모니터링 서비스에 등록한 애플리케이션 간의 호출 관계를 확인할 수 있습니다.

> 멀티 트랜잭션을 추적하려면 **관리** > **에이전트 설정** 메뉴에서 `mtrace_enabled` 옵션을 `true` 로 설정하세요.
> 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.



- **차트**: 각 트랜잭션의 호출 관계를 플로우 차트 형식으로 제공합니다. 트랜잭션 노드를 선택하면 해당 트랜잭션 노드에 대한 트레이스 분석 정보를 확인할 수 있습니다. **차트** 모드에서는 마우스를 이용해 원하는 위치로 이동하거나 스크롤을 통해서 확대, 축소할 수 있습니다.
  - ⚙ **차트 뷰 설정**: 차트에 표시할 요소를 표시하거나 숨길 수 있습니다.
- **테이블**: 테이블 형식으로 멀티 트랜잭션 내에 포함된 각 트랜잭션 별 정보를 확인할 수 있습니다. ▥ 컬럼 선택 아이콘을 선택해 테이블 헤더 컬럼을 편집할 수 있습니다. 각 트랜잭션 항목을 선택하면 트레이스 분석 정보를 확인할 수 있습니다.

- **트리**: 트리 형식으로 트랜잭션 간의 호출 관계를 파악할 수 있습니다. 관련한 부가 기능은 **트리 뷰** 탭의 기능과 같습니다.
- **프로젝트 선택**: 차트에 표시될 프로젝트를 선택하거나 해제할 수 있습니다.

> 트랜잭션 목록에서 M 아이콘이 표시된 항목에서 확인할 수 있습니다.
>
> - 멀티 트랜잭션에 대한 자세한 내용은 다음 문서를 참조하세요.

메소드 요약



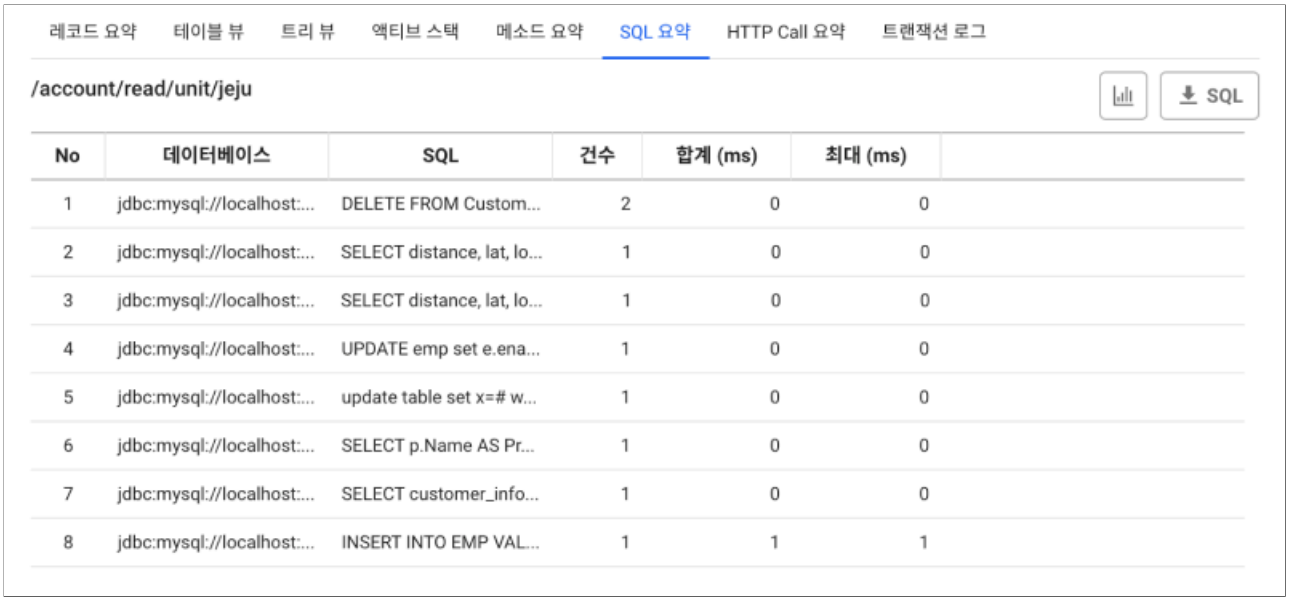
No	클래스	메소드	파라미터/리턴	건수	합계 (ms)	최대 (ms)
1	com.virtual.dao.Remot...	execute	()V	27	39,063	2,776
2	com.virtual.dao.PIITest	exec	(Ljava/sql/PreparedSta...	3	0	0
3	com.virtual.dao.InsertD...	execute2	(Ljava/lang/String;)V	2	4	3
4	com.virtual.dao.Update...	execute2	(Ljava/lang/String;)V	2	18	14
5	com.virtual.dao.Delete...	execute	()V	2	3	2
6	com.virtual.dao.Select...	execute2	(Ljava/lang/String;)V	1	5	5
7	com.virtual.dao.Update...	execute	()V	1	4	4
8	com.virtual.dao.Select...	execute2	()V	1	7	7
9	com.virtual.dao.Select...	execute	()V	1	9	9
10	com.virtual.dao.InsertD...	execute2	()V	1	3	3

메소드 정보만을 확인하려면 **메소드 요약** 탭을 선택하세요. 에이전트에 추적이 설정된 메소드 이름과 소요 시간을 표시합니다. 불필요한 로직이 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 메소드 로직 개선을 위한 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **메소드**: 메소드 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

> ⓘ 메소드(method)와 관련한 에이전트 설정 옵션에 대한 자세한 내용은 다음 문서를 참조하세요. 이용 중인 제품에 따라 메소드 추적 지원 여부는 다를 수 있습니다.

SQL 요약



No	데이터베이스	SQL	건수	합계 (ms)	최대 (ms)
1	jdbc:mysql://localhost:...	DELETE FROM Custom...	2	0	0
2	jdbc:mysql://localhost:...	SELECT distance, lat, lo...	1	0	0
3	jdbc:mysql://localhost:...	SELECT distance, lat, lo...	1	0	0
4	jdbc:mysql://localhost:...	UPDATE emp set e.ena...	1	0	0
5	jdbc:mysql://localhost:...	update table set x=# w...	1	0	0
6	jdbc:mysql://localhost:...	SELECT p.Name AS Pr...	1	0	0
7	jdbc:mysql://localhost:...	SELECT customer_info...	1	0	0
8	jdbc:mysql://localhost:...	INSERT INTO EMP VAL...	1	1	1

SQL 문에 대한 정보를 확인하려면 **SQL 요약** 탭을 선택하세요. 불필요하게 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 쿼리 성능 개선을 위한 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **SQL**: SQL 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

HTTP Call 요약



No	원격 서버	URL	건수	합계 (ms)	최대 (ms)
1	127.0.0.1:8105	/remote/account/save...	2	4,116	2,776
2	127.0.0.1:8105	/remote/order/save/di...	1	238	238
3	127.0.0.1:8105	/remote/sales/pickup/...	1	922	922
4	127.0.0.1:8105	/remote/sales/remove...	1	1,980	1,980
5	127.0.0.1:8105	/remote/order/save/de...	1	2,639	2,639
6	127.0.0.1:8105	/remote/edu/kill/unit/p...	1	1,519	1,519
7	127.0.0.1:8105	/remote/order/read/de...	1	1,467	1,467
8	127.0.0.1:8105	/remote/order/delete/...	1	485	485
9	127.0.0.1:8105	/remote/sales/kill/emp...	1	2,210	2,210

HTTP 호출의 호출 건수, 합계 시간, 평균 시간 등을 확인하려면 **HTTP Call 요약** 탭을 선택하세요. 불필요한 외부 호출이 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 트랜잭션 지연 요인이 외부인지 내부인지를 파악하는 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **HTTP**: HTTP Call 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

트랜잭션 로그

트랜잭션과 관련한 로그 정보를 확인하려면 **트랜잭션 로그** 탭을 선택하세요.



- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **로그 검색기**: 로그 검색으로 이동해 원하는 로그 데이터를 쉽게 조회할 수 있습니다.
- **키워드 검색**: 키워드 검색란에 검색하려는 텍스트를 입력한 다음 엔터 키를 입력하거나 버튼을 선택하세요. 입력한 텍스트와 일치하는 키워드를 하이라이트 표시합니다.
- **키워드 하이라이트**: 키워드와 색상을 설정하면 자동으로 로그 목록에서 키워드와 일치하는 텍스트를 하이라이트 표시합니다. 자세한 내용은 다음 문서를 참조하세요.
- **컬럼 설정**: 로그 목록 테이블에 표시되는 컬럼을 추가하거나 순서를 변경할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- **로그 표시 상세 설정**: 로그 목록에 표시되는 내용을 설정할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

- **트랜잭션 로그** 탭을 활성화하려면 로그와 관련한 에이전트 설정 옵션을 적용해야 합니다. 로그 설정에 대한 자세한 내용은 다음 문서를 참조하세요.
- **트랜잭션 로그** 탭은 **로그 조회** 권한을 가진 멤버만 진입할 수 있습니다. 멤버 권한에 대한 자세한 내용은 다음 문서를 참조하세요.

키워드 하이라이트 설정하기

1. 하이라이트 표시할 키워드를 추가하려면 버튼을 선택하세요.



2. 원하는 키워드를 입력하세요.
3. 색상 아이콘을 선택한 다음 원하는 색상을 선택하세요.
4. 엔터를 입력하세요. 추가한 키워드가 목록에 생성됩니다.

키워드 하이라이트 삭제하기

하이라이트 목록에서 이용하지 않는 항목을 삭제하려면 — 버튼을 선택하세요.

테이블 컬럼 표시하거나 숨기기

1. 테이블 오른쪽 위에 ▥ 버튼을 선택하세요.
2. **컬럼 설정** 창이 나타나면 왼쪽 목록에서 원하는 항목을 선택하거나 선택을 해제하세요.

   오른쪽 목록에서 × 버튼을 선택해 컬럼을 숨길 수도 있습니다.
3. 모든 설정을 완료한 다음 **확인** 버튼을 선택하세요. 설정한 내용이 테이블에 적용됩니다.

테이블 컬럼 순서 바꾸기

1. 테이블 오른쪽 위에 ▥ 버튼을 선택하세요.
2. **컬럼 설정** 창이 나타나면 오른쪽 목록에서 원하는 항목을 드래그해서 순서를 변경하세요.
3. 모든 설정을 완료한 다음 **확인** 버튼을 선택하세요. 설정한 내용이 테이블에 적용됩니다.

테이블 콘텐츠 표시 설정

1. 테이블 오른쪽 위에 ⚙ 버튼을 선택하세요.
2. **로그 표시 상세 설정** 창이 나타나면 원하는 옵션을 설정하세요.
   - **content** 또는 **태그** 옵션을 선택해 로그의 콘텐츠 표시 여부를 선택하세요.
   - **태그 관리**: 태그는 추가한 순서대로 로그에 표시됩니다. 태그의 순서는 드래그로 변경할 수 있습니다. 비활성화된 태그는 로그에서 보이지 않습니다.
3. 모든 설정을 완료한 다음 ✕ 버튼을 클릭하세요. 설정한 내용이 테이블에 적용됩니다.

> ⓘ **로그 표시 상세 설정**에 대한 자세한 내용은 다음 문서를 참조하세요.

부가 기능

HTTP 파라미터 조회

테이블 뷰 탭에서 해당 트랜잭션의 HTTP 파라미터를 조회할 수 있습니다.

1. 페이지 아래로 스크롤해 트랜잭션 수행의 가장 마지막 단계로 이동하세요.
2. 🔒 **HTTP 파라미터** 항목을 선택하세요.
3. **HTTP-PARAMETERS** 창이 나타나면 **비밀번호** 버튼을 선택하세요.
4. 설정한 Param Key를 입력하세요. 가려진 매개 변수를 확인할 수 있습니다.

> ⓘ
> - HTTP 파라미터와 관련한 에이전트 설정은 다음 문서를 참조하세요.
> - **비밀번호**: 복호화된 파라미터 값을 확인할 수 있습니다. 비밀번호는 `WHATAP_HOME` /security.conf (paramkey.txt) 파일 내 6자리 문자열입니다. 다른 문자열로 변경 가능합니다. SQL 변수 조회, HTTP 쿼리 조회, Thread 중지에 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

SQL 파라미터 조회

테이블 뷰 탭에서 SQL 스텝을 선택하거나 **트리 뷰** 탭에서 버튼을 선택하세요. 파라미터를 조회할 수 있는 **SQL** 창이 나타납니다.



- **비밀번호**: 복호화된 파라미터 값을 확인할 수 있습니다. 비밀번호는 `WHATAP_HOME` /security.conf 파일 내 6자리 문자열(`paramkey`)입니다. 다른 문자열로 변경 가능합니다.

  > **비밀번호**는 SQL 변수 조회, HTTP 쿼리 조회, Thread 중지에 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

- **SQL 포멧팅**: SQL 문장을 들여쓰기 및 포맷팅하여 가독성을 높일 수 있습니다.
- : SQL 문장을 클립보드에 복사할 수 있습니다.
- **통계**: **통계** > **SQL** 메뉴로 이동합니다.

> SQL 변수와 HTTP 쿼리를 조회하려면 다음 옵션을 에이전트 설정에 추가하세요.
>
> - SQL 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
> - HTTP 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
>
> whatap.conf

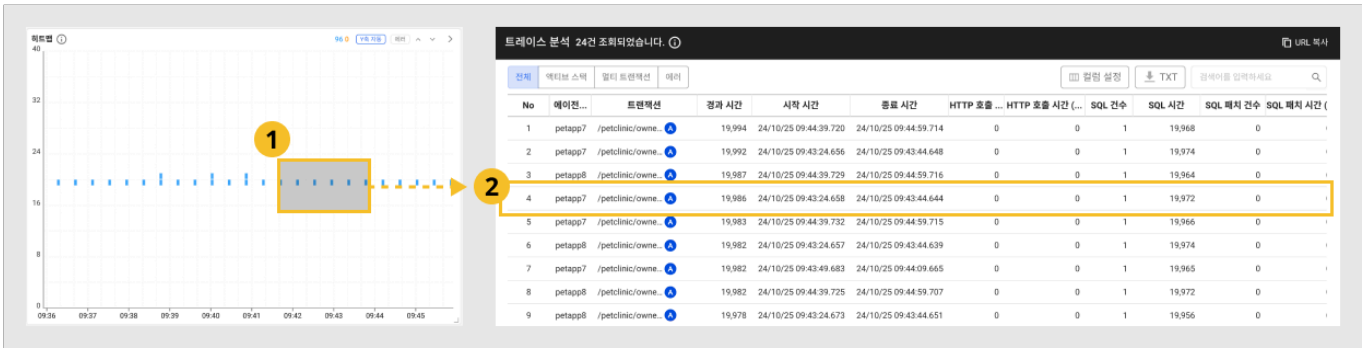
```
# SQL 파라미터 조회 옵션: 옵션이 적용되면 SQL 파라미터를 암호화하여 수집합니다.
profile_sql_param_enabled=true

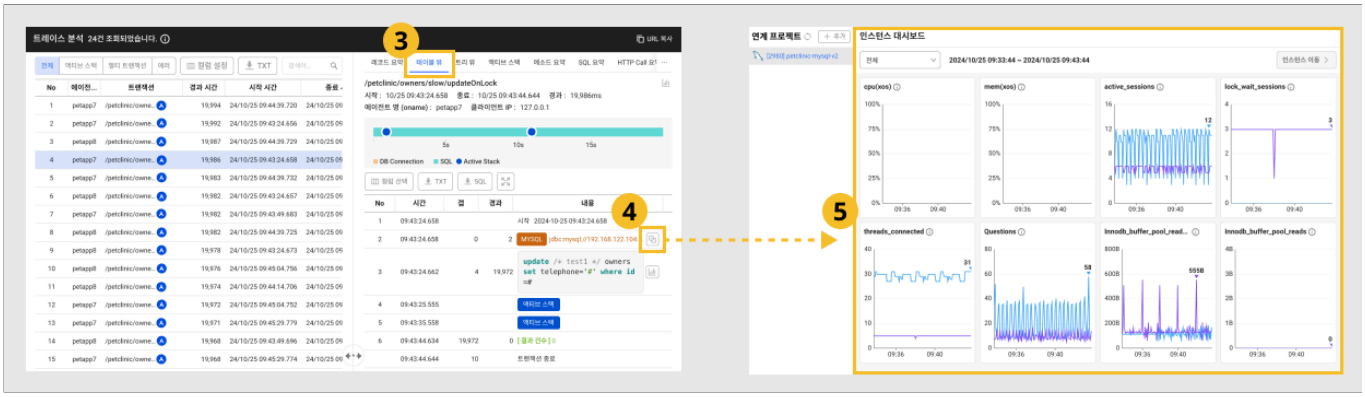
# HTTP 파라미터 조회 옵션: 옵션이 적용되면 HTTP 쿼리 파라미터를 암호화하여 수집합니다.
profile_http_parameter_enabled=true
```

연계 프로젝트 성능 지표 조회하기

연계 프로젝트 관리 메뉴를 통해 데이터베이스 프로젝트를 추가하면, **DB Connection 스텝** 및 **SQL 스텝**에서 연계한 프로젝트의 성능 지표를 조회할 수 있습니다.

DB Connection 스텝에서 확인하기



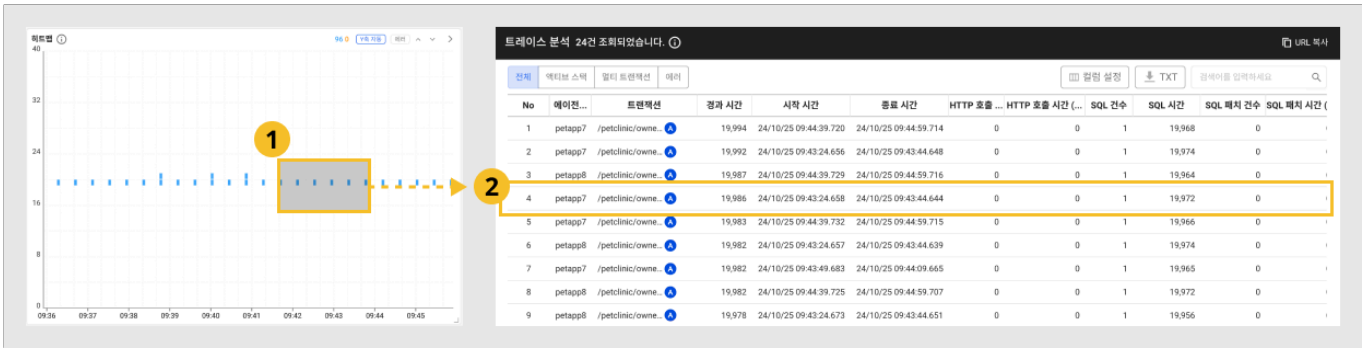


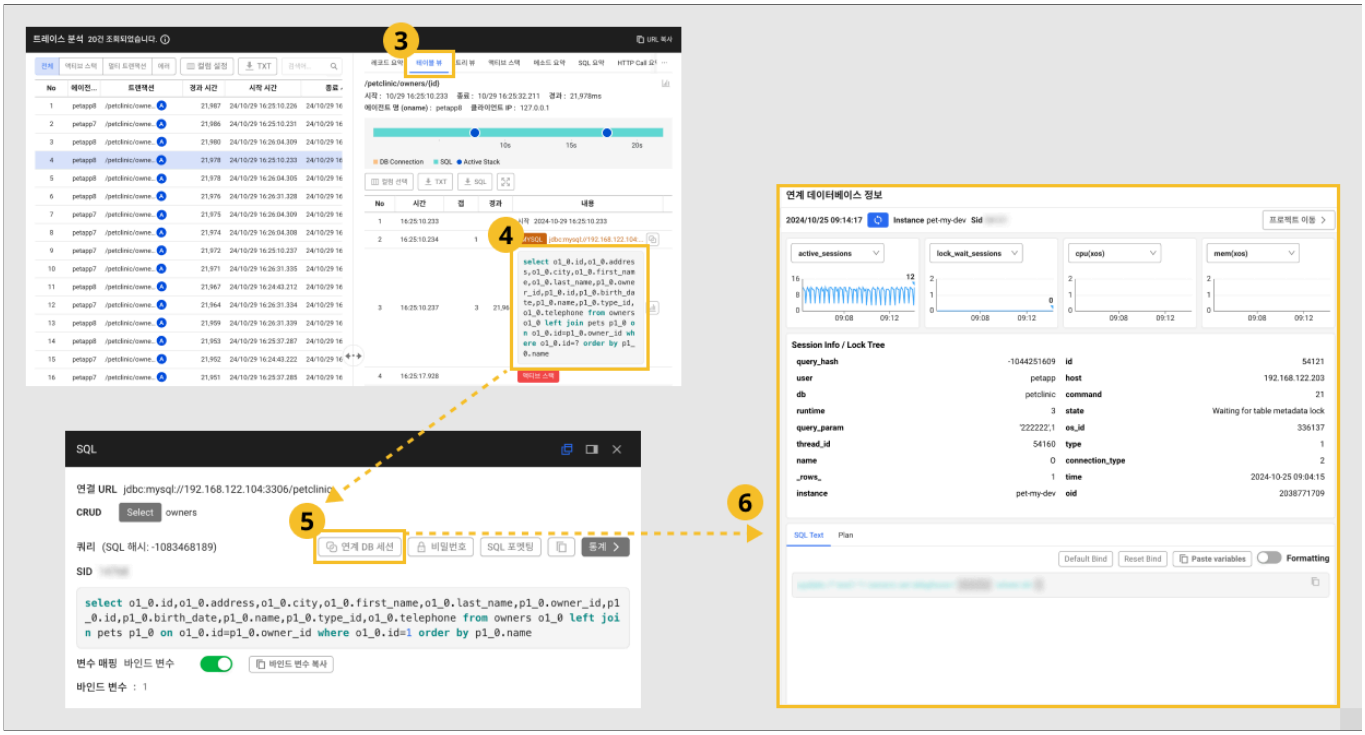
1. 대시보드에서 **히트맵** 위젯의 차트 영역을 드래그하세요.
2. **트레이스 분석** 창이 나타나면 조회하길 원하는 트랜잭션 항목을 선택하세요.

3. 트랜잭션 상세 정보가 오른쪽에 나타나면 **테이블 뷰** 탭을 선택하세요.
4. DB Connection 스텝에서 버튼을 선택하세요.
5. 연계 프로젝트의 **인스턴스 대시보드** 창이 나타나면 관련 지표를 확인하세요.

- 연계분석을 위해 등록한 데이터베이스 프로젝트의 DB가 현재 선택한 트랜잭션의 SQL을 실행한 경우에만, **테이블 뷰** 탭에서 버튼이 표시됩니다.
- 연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.
- **인스턴스 대시보드** 창에 대한 자세한 내용은 다음 문서를 참조하세요.

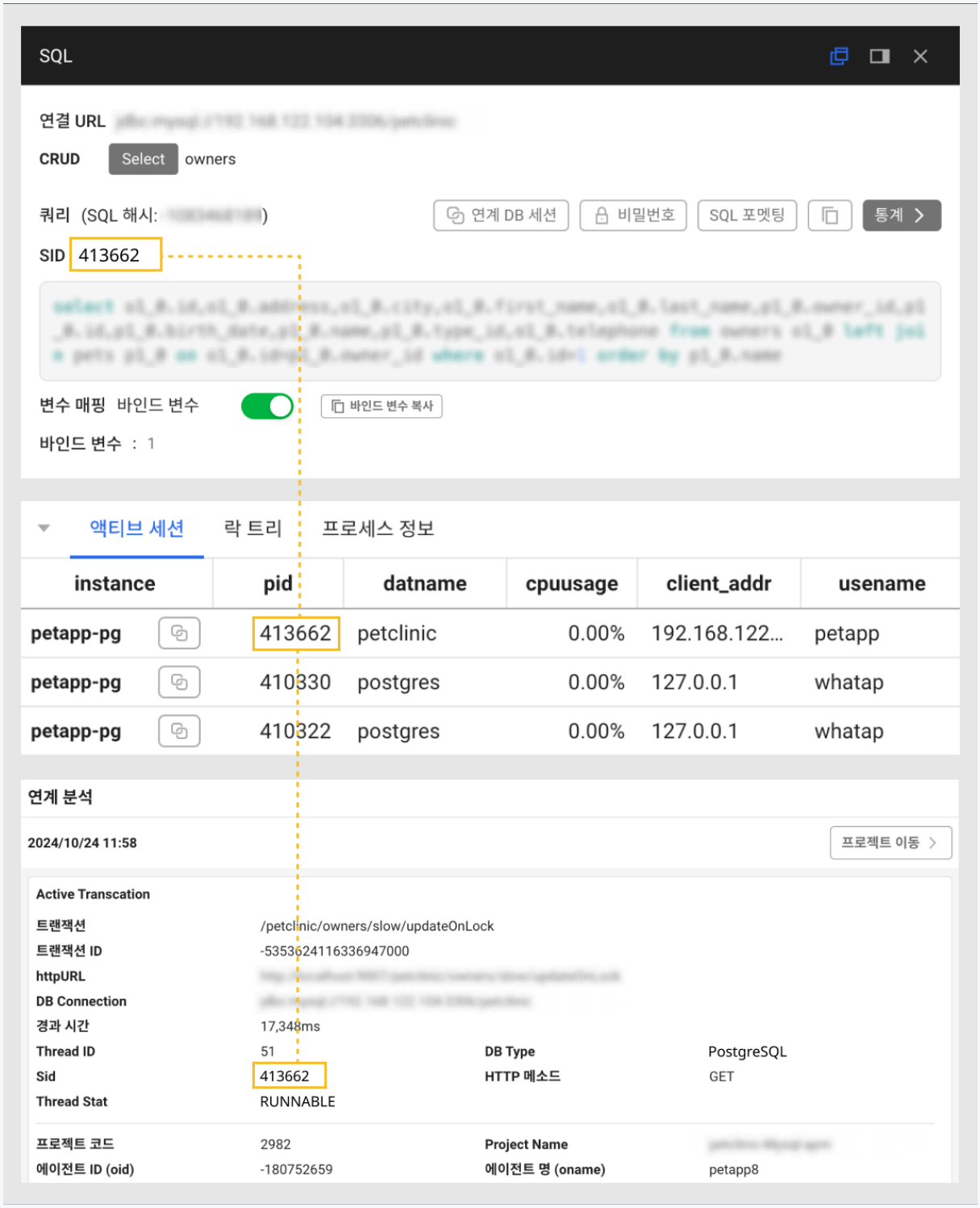
SQL 스텝에서 확인하기





1. 대시보드에서 **히트맵** 위젯의 차트 영역을 드래그하세요.
2. **트레이스 분석** 창이 나타나면 조회하길 원하는 트랜잭션 항목을 선택하세요.
3. 트랜잭션 상세 정보가 오른쪽에 나타나면 **테이블 뷰** 탭을 선택하세요.
4. SQL 스텝에서 SQL 문을 선택하세요.
5. **SQL** 창이 나타나면 **연계 DB 세션** 버튼을 선택하세요.
6. **연계 데이터베이스 정보** 창이 나타나면 관련 지표를 확인하세요.

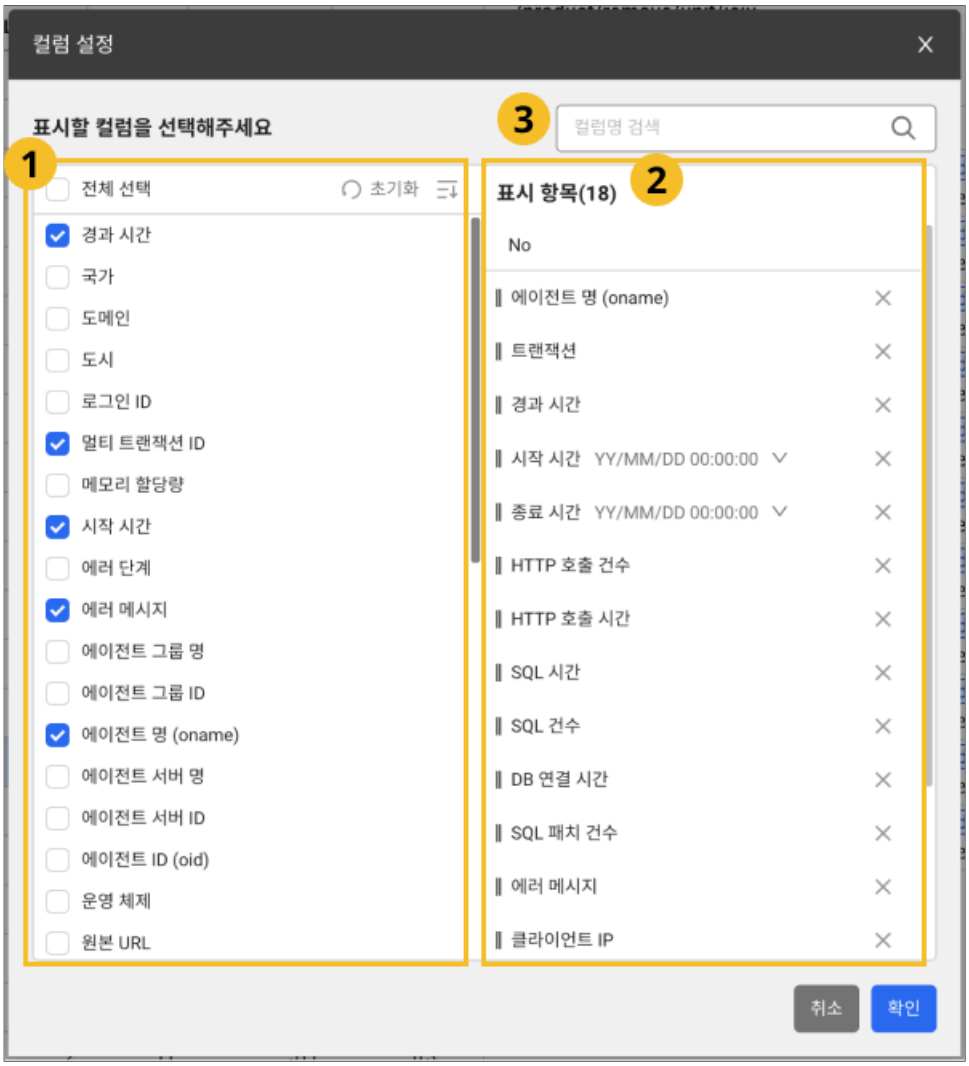
- 연계분석을 위해 등록한 데이터베이스 프로젝트의 DB가 현재 선택한 트랜잭션의 SQL을 실행한 경우에만, **SQL** 창에 **연계 DB 세션** 버튼이 표시됩니다.
- **바인드 변수 복사** 기능을 이용하면 변수를 복사해 **연계 데이터베이스 정보** 창의 **SQL Text** 탭에 변수를 붙여넣기할 수 있습니다. 이 기능은 **비밀번호** 버튼을 선택해 비밀번호를 입력 후 바인드 변수 값이 표시될 때 이용할 수 있습니다.
- **연계 데이터베이스 정보** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
- 이 기능을 지원하는 데이터베이스는 **PostgreSQL**, **Oracle**, **Oracle Pro**, **MySQL** 플랫폼입니다.
- **PostgreSQL** 플랫폼에서 **액티브 세션** 테이블의 **PID** 항목은 **SID** 항목과 같습니다.


SQL
연결 URL
CRUD
Select
owners
쿼리 (SQL 해시:
연계 DB 세션
비밀번호
SQL 포멧팅
통계
SID
413662
변수 매핑 바인드 변수
바인드 변수 복사
바인드 변수 : 1
액티브 세션
락 트리
프로세스 정보
instance
pid
datname
cpuusage
client_addr
usename
petapp-pg
413662
petclinic
0.00%
192.168.122...
petapp
petapp-pg
410330
postgres
0.00%
127.0.0.1
whatap
petapp-pg
410322
postgres
0.00%
127.0.0.1
whatap
연계 분석
2024/10/24 11:58
프로젝트 이동
Active Transcation
트랜잭션
/petclinic/owners/slow/updateOnLock
트랜잭션 ID
-5353624116336947000
httpURL
DB Connection
경과 시간
17,348ms
Thread ID
51
DB Type
PostgreSQL
Sid
413662
HTTP 메소드
GET
Thread Stat
RUNNABLE
프로젝트 코드
2982
Project Name
에이전트 ID (oid)
-180752659
에이전트 명 (oname)
petapp8

> **Java 에이전트 2.2.40** 이전 버전에서는 **액티브 트랜잭션** 창에서 **ThreadStat** 값이 숫자 형식으로 표시됩니다. 문자 형식으로 표시하려면 Java 에이전트 2.2.40 버전 이상을 설치하세요.

테이블 컬럼 설정하기

트랜잭션 목록에서 테이블 헤더 컬럼을 감추거나 원하는 항목을 추가할 수 있습니다. 컬럼 순서를 변경할 수도 있습니다. **컬럼 설정** 버튼을 선택하세요.



- 설정을 완료한 다음에는 **확인** 버튼을 선택해야 설정 사항이 테이블에 반영됩니다.
- 3 검색란에 텍스트를 입력해 원하는 컬럼 항목을 검색할 수 있습니다. 입력한 텍스트와 매칭되는 컬럼 항목만 표시됩니다.

컬럼 추가하기

1 목록에서 테이블 헤더 컬럼으로 추가할 항목을 선택하세요. 모든 항목을 추가하려면 **전체 선택**을 선택하세요.

컬럼 삭제하기

1 목록에서 삭제할 컬럼 항목의 체크 박스를 선택 해제하세요. 또는 2 목록에서 삭제할 항목의 오른쪽에 ✕ 버튼을 선택하세요.

컬럼 순서 변경하기

2 목록에 순서를 변경할 항목을 드래그해서 원하는 위치로 이동할 수 있습니다.

설정 사항 초기화하기

변경 사항은 모두 취소하고 초기화하려면 ↻ **초기화** 버튼을 선택하세요.

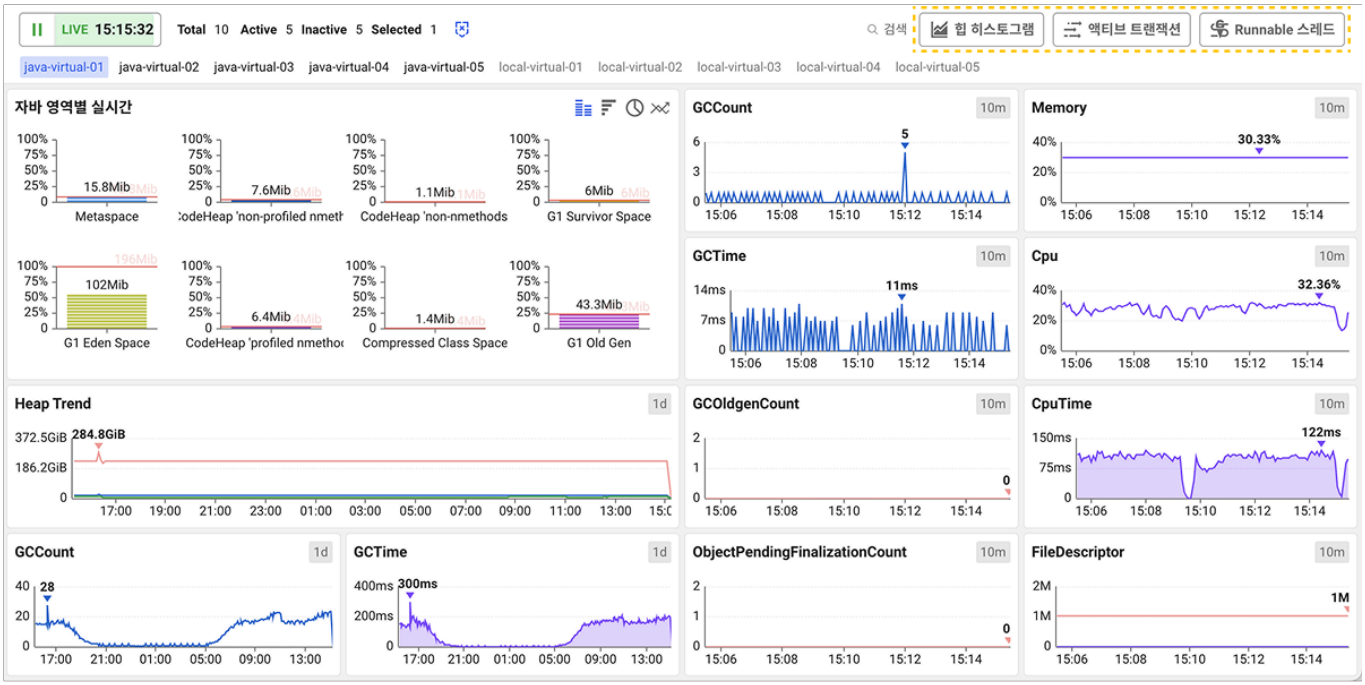
조회 데이터 다운로드

테이블에 출력된 데이터를 txt 파일 형식으로 다운로드하려면 ⬇ **TXT** 버튼을 선택하세요.

에이전트 자원 대시보드 설정

> **지원 에이전트 버전**
>
> 와탭 자바 에이전트 2.2.39 버전 이상이 필요합니다.

Agent별 리소스 대시보드 메뉴에서 개별 자바 에이전트의 메모리, 트랜잭션, 스레드 등 주요 지표를 실시간으로 모니터링하고, 최대 3시간 전 과거 시점까지 조회할 수 있습니다. 이를 통해 메모리 사용량 변화, 활성화된 트랜잭션, 실행 중인 스레드 정보를 파악하고, 리소스 문제를 유발하는 원인을 분석할 수 있습니다.



기본 화면

Agent별 리소스 대시보드를 통해 현재 실행 중인 에이전트의 메모리 풀 상태 및 관련 지표 변화 추이를 한눈에 확인할 수 있습니다. 대시보드에서 확인할 수 있는 실시간 자바 메모리 영역 및 주요 지표는 다음과 같습니다.

위젯 안내

실시간 자바 메모리 영역 위젯

자바 영역별 실시간 위젯을 통해 JVM 주요 메모리 풀 상태를 실시간으로 확인할 수 있습니다. 각 자바 메모리 영역 차트는 사용 중인 용량(MB)과 비율(%)을 표시합니다.

- **Metaspace**

  클래스 메타데이터를 저장하는 영역입니다.

- **CodeHeap non-profiled nmethod**, **CodeHeap profiled nmethod**, **CodeHeap non-nmethods**

  JIT(Just-In-Time) 컴파일러가 생성한 머신 코드를 저장하는 영역으로 **CodeHeap non-profiled nmethod**, **CodeHeap profiled nmethod**, **CodeHeap non-nmethods**는 각각 컴파일된 코드를 용도에 따라 구분해 보관합니다.

- **G1 Survivor Space**

  G1 GC에서 살아남은 객체가 일시적으로 저장되는 영역입니다.

- **G1 Eden Space**

  새로 생성된 객체가 먼저 할당되는 영역입니다.

- **G1 Old Gen**

  Young GC를 거쳐 살아남은 객체가 이동되는 영역입니다. 메모리가 증가할 경우 Old Gen을 중심으로 누수가 발생하는지 확인할 수 있습니다.

- **Compressed Class Space**

  클래스 관련 포인터를 압축하여 저장하는 영역으로 Metaspace와 별도로 관리됩니다.

> - 개별 차트: 각 지표를 별도의 막대 형식으로 표시하여 지표마다 최댓값을 명확히 비교할 수 있습니다.
> - 통합 차트: 전체 대비 각 지표의 상대적 크기를 한 눈에 파악할 수 있습니다.
> - 도넛 차트: 전체를 원형 비율로 표현하여 각 지표가 차지하는 비중을 확인할 수 있습니다.
> - 멀티 라인 차트: 전체 지표에 대한 추이를 확인할 수 있습니다.

주요 지표 위젯

- **Memory**

  전체 메모리 사용률(%)을 표시합니다. Eden, Old Gen, Metaspace 등 내부 구성을 종합한 총 사용량입니다.

- **CpuTime**

  JVM이 애플리케이션 코드를 실행하는 데 소요된 CPU 시간을 밀리초(ms) 단위로 나타냅니다.

- **GCCount**, **GCOldgenCount**

  GCCount는 일정 주기 내 발생한 GC 횟수, **GCOldgenCount**는 Old 영역을 대상으로 한 GC 횟수만 별도로 집계합니다.

- **GCTime**

  Garbage Collection에 소요된 시간을 ms 단위로 표시합니다.

- **Cpu**

  JVM 프로세스의 CPU 사용률(%)을 나타냅니다.

- **TPS**

  초당 트랜잭션 처리 횟수를 표시합니다.

- **FileDescriptor**

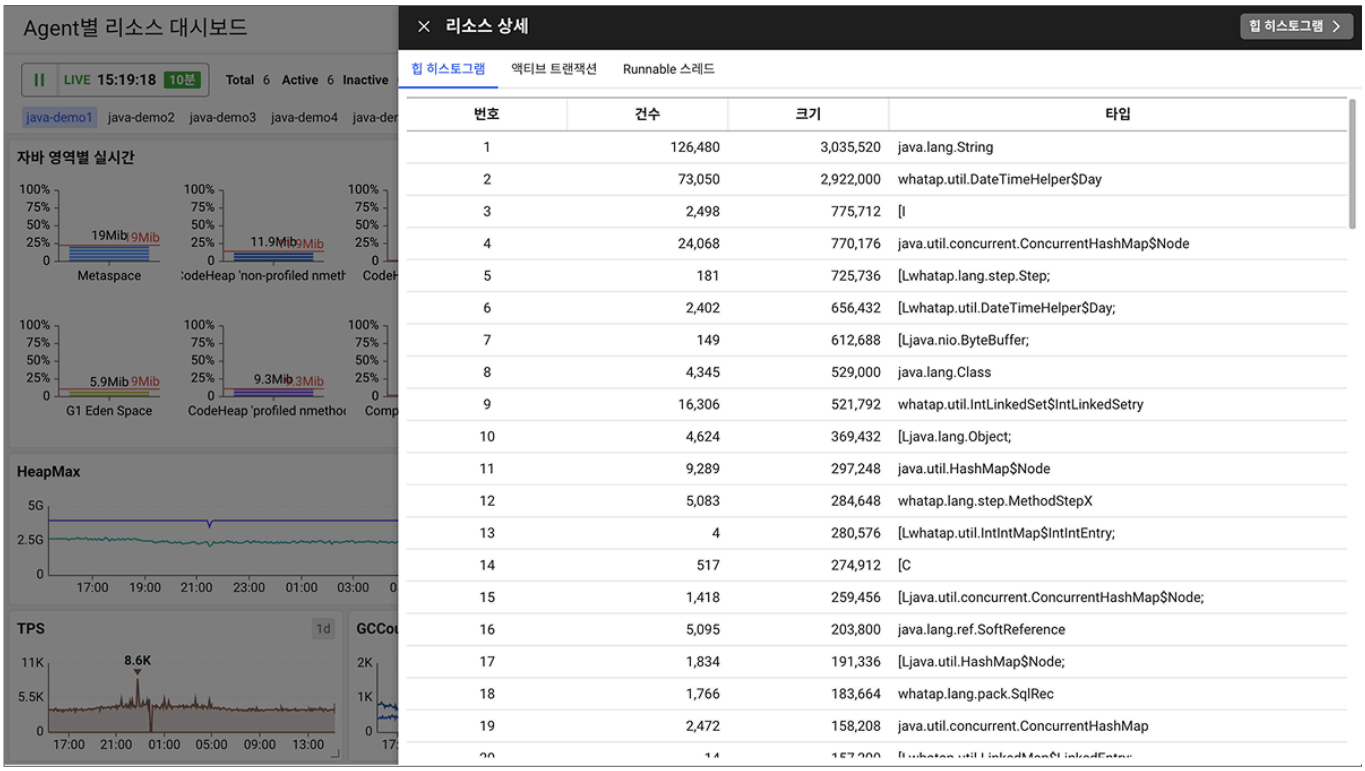
  열려 있는 파일이나 소켓 등 현재 사용 중인 파일 디스크립터 수를 나타냅니다.

- **ObjectPendingFinalizationCount**

  GC가 처리해야 할 객체가 `finalize()` 대기 상태로 남아 있는 개수를 표시합니다.

실시간 조회

실시간 조회 시 상단 오른쪽 **힙 히스토그램**, **액티브 트랜잭션**, **Runnable 스레드** 버튼을 통해 **리소스 상세** 패널을 확인할 수 있습니다.



힙 히스토그램

힙 히스토그램 탭에서 현재 힙 메모리에 상주하는 Java 클래스 정보를 분석하고, 메모리를 가장 많이 차지하는 클래스 TOP 100을 조회할 수 있습니다. 과도하게 메모리를 점유하는 클래스가 있는 경우 메모리 누수 가능성을 의심할 수 있습니다.

액티브 트랜잭션

액티브 트랜잭션 탭에서 실행 중인 트랜잭션 목록을 조회할 수 있습니다. 시작 시각, 트랜잭션 경과 시간, SQL/HTTP 호출 등의 세부 정보를 확인할 수 있습니다. 특정 트랜잭션이 오래 걸리거나 리소스를 많이 사용하는지 확인하여 병목 구간을 파악할 수 있습니다.

Runnable 스레드

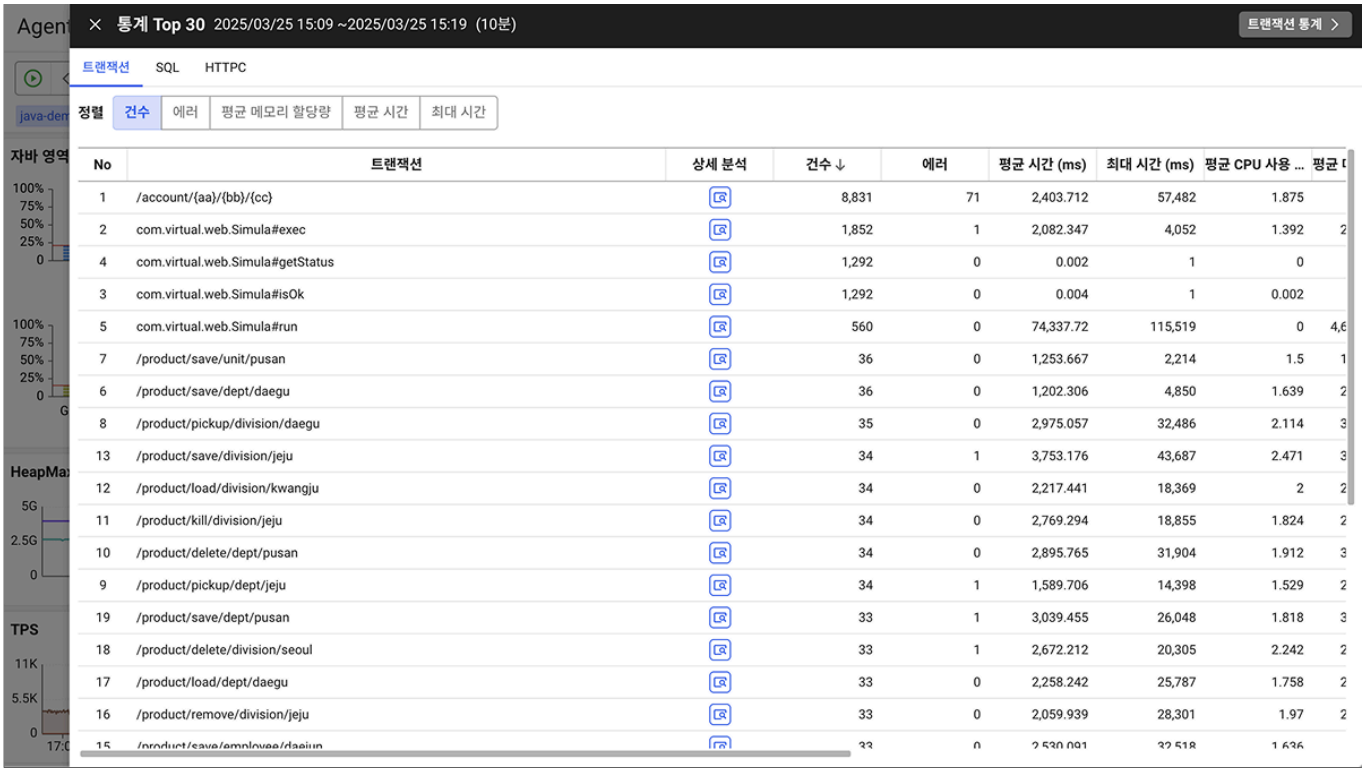
Runnable 스레드 탭에서 현재 JVM에서 Runnable 상태인 스레드 목록을 실시간으로 확인할 수 있습니다. CPU 사용량이 높아지거나 응답이 지연되는 경우 어떤 스레드가 문제를 일으키는지 파악하는 데 유용합니다.

과거 시점 조회

시간 선택자를 통해 과거 특정 시점을 선택 후 해당 시점의 리소스 상태를 확인할 수 있습니다. 최대 3시간 전의 리소스 상태를 조회하고, 해당 시점에 발생한 트랜잭션, SQL, HTTP 호출 정보를 분석할 수 있습니다. 이를 통해 CPU나 메모리 사용량이 급등하거나 TPS가 급감했던 시점의 원인을 효과적으로 추적할 수 있습니다.



과거 시점 조회 시 상단 오른쪽 **트랜잭션**, **SQL 통계**, **HTTPC 통계** 버튼 클릭 시 각 상위 30개 통계 정보를 확인할 수 있는 **통계 Top 30** 패널을 조회할 수 있습니다.



트랜잭션 통계

트랜잭션 탭에서 특정 시점에 수행된 트랜잭션들을 응답 시간, 실행 횟수 등의 조건으로 정렬해 상위 30개 목록을 조회할 수 있습니다. 응답 시간이 길거나 실행 횟수가 많은 항목을 우선 확인할 수 있습니다.

평균 메모리 할당량 조회 설정

과거 시점의 트랜잭션 통계에서 **평균 메모리 할당량** 항목이 표시되지 않는 경우, 다음 옵션의 활성화 여부를 확인해 주세요. 단 이 옵션은 평균 TPS가 높은 애플리케이션에서는 성능 영향이 발생할 수 있으므로 **평균 TPS 300 이하의 환경**에서만 적용을 권장합니다.

- **trace_malloc_enabled** Boolean

  기본값 `false`

  트랜잭션별 평균 메모리 할당량을 수집합니다.

SQL 통계

SQL 탭에서 특정 시점에 수행된 SQL 쿼리를 실행 횟수, 전체 시간 등의 정렬 조건에 따라 상위 30개 목록을 조회할 수 있습니다. DB 부하가 증가 시 동일 시점에 실행된 SQL 쿼리 중 높은 비용을 차지하는 쿼리를 파악할 수 있습니다.

HTTPC 통계

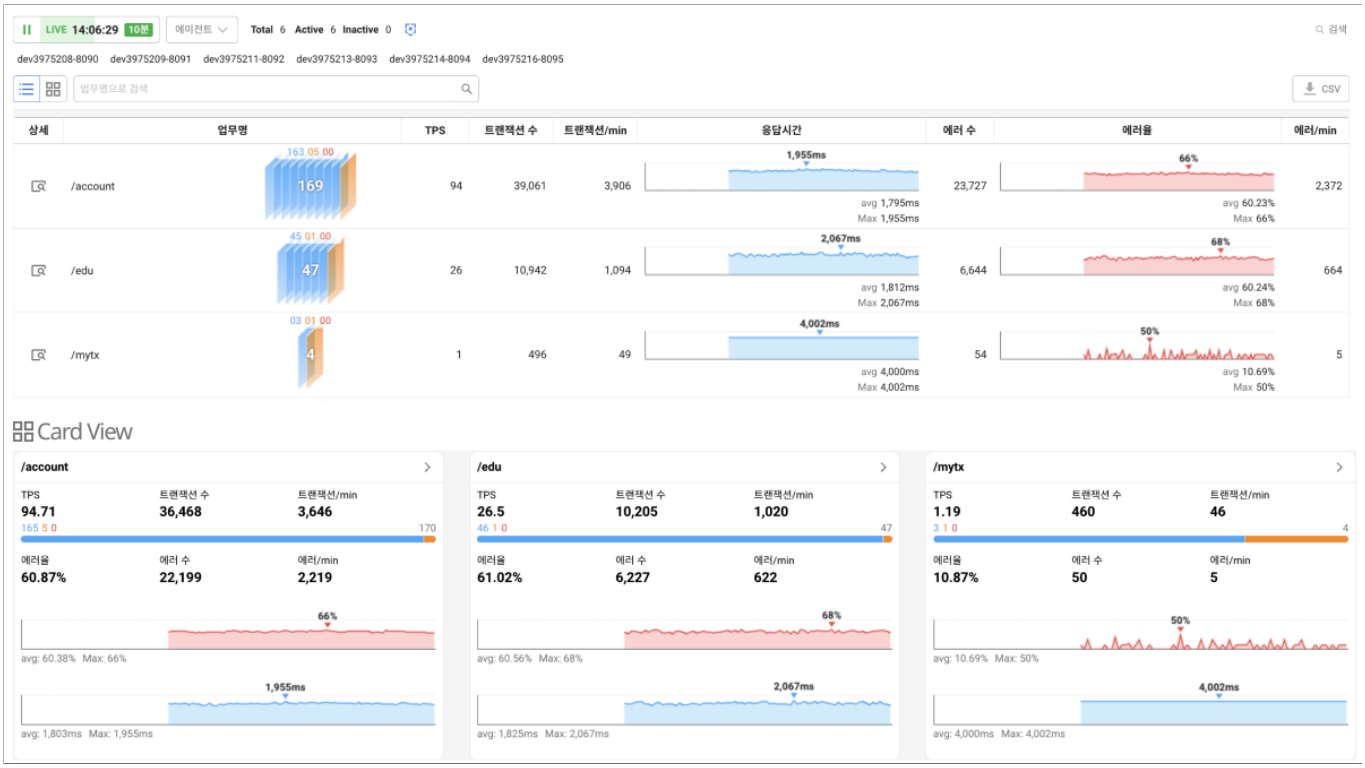
HTTPC 탭에서 외부 HTTP 호출 정보를 정렬하여 상위 30개 목록을 조회할 수 있습니다. 외부 API나 내부 HTTP 요청이 많은 시점에 병목 발생 여부를 확인할 수 있습니다.

비즈니스 대시보드

홈 화면 > 프로젝트 선택 > **대시보드** > **비즈니스 대시보드**

비즈니스 대시보드는 애플리케이션의 비즈니스 단위(App Context)별 성능을 모니터링할 수 있는 대시보드입니다. URL 패턴이나 경로를 기반으로 정의된 각 비즈니스 항목에 대해 실시간 및 과거의 성능 데이터를 조회하고 분석할 수 있습니다. TPS, 응답시간, 에러율 등의 주요 성능 지표를 비즈니스 단위로 확인함으로써, 특정 업무나 서비스의 성능 상태를 빠르게 파악할 수 있습니다.

화면 구성



시간 선택

화면 왼쪽 위, 시간 선택자를 통해 조회할 시간 범위를 설정할 수 있습니다. 화면 왼쪽 위, 시간 선택자의 오른쪽에서는 해당 프로젝트와 연결된 에이전트의 상태를 확인할 수 있는 정보를 제공합니다. 이를 통해 모니터링 대상 서버의 동작 여부를 바로 확인할 수 있습니다.

- **실시간 모드**: 5초 주기로 자동 갱신되며 최근 10분의 데이터를 표시합니다.
- **과거 조회 모드**: 최대 한달까지 과거 데이터를 조회할 수 있습니다.
- **데이터 집계 단위**:
  - 3시간 미만: Raw 데이터 사용
  - 3시간 ~ 3일: 5분 통계 데이터 사용
  - 3일 이상: 1시간 통계 데이터 사용

에이전트 선택

시간 선택자 아래의 프로젝트에 속한 에이전트 리스트를 확인할 수 있습니다. 에이전트 항목을 선택하면 파란색 태그가 표시됩니다. 선택한 특정 에이전트나 에이전트 그룹의 데이터만 필터링하여 볼 수 있습니다.

> ⓘ 에이전트 선택에 대한 자세한 내용은 다음 문서를 참조하세요.

뷰 모드 변경

비즈니스 대시보드는 사용자의 선호에 따라 두 가지 뷰 모드로 전환하여 데이터를 확인할 수 있습니다. 화면 왼쪽 위에 위치한 ☰(테이블뷰 아이콘)과 ⊞(카드뷰 아이콘)을 클릭하면 원하는 뷰로 전환할 수 있습니다.

- ☰ **테이블 뷰**

  전체 비즈니스 항목을 표 형식으로 나열하여 비교와 정렬이 용이합니다. 각 항목의 주요 성능 지표(TPS, 응답시간, 에러율 등)를 한눈에 확인할 수 있어, 전체적인 현황을 빠르게 파악할 때 유용합니다.

- ⊞ **카드 뷰**

  각 비즈니스 항목을 카드 형태로 시각적으로 표현합니다. 뷰의 열 수는 1열부터 최대 4열까지 선택할 수 있어, 모니터 크기나 사용자의 작업 환경에 따라 유연하게 조정할 수 있습니다. 카드 뷰는 특정 비즈니스의 상태를 개별적으로 집중해 살펴보거나, 대시보드 형태로 사용할 때 적합합니다.

정렬 선택

테이블 뷰와 카드 뷰에서 업무별 데이터를 정렬할 지표를 선택할 수 있습니다.

업무명 검색

검색 영역에서 업무명을 입력하여 특정 비즈니스를 빠르게 찾을 수 있습니다.

업무명 별칭 설정

버튼을 클릭하면 URL 접두사로 표시되는 업무명에 별칭을 지정할 수 있는 드로어 창이 열립니다.

- 왼쪽 컬럼에 별칭을 적용할 업무명(URL 접두사) 입력
- 오른쪽 컬럼에 해당 업무명에 적용할 별칭 입력
- **항목 추가** 버튼으로 항목 추가, **삭제** 버튼으로 항목 삭제
- 하단의 **적용** 버튼 클릭 시 설정 반영

CSV 다운로드

오른쪽 위 검색 영역 옆에 있는 CSV을 클릭하면, 현재 화면에 표시된 모든 비즈니스의 성능 데이터를 CSV 파일로 다운로드할 수 있습니다.

비즈니스 성능 현황

각 비즈니스 단위별로 다음과 같은 주요 성능 지표를 확인할 수 있습니다.
지표는 실시간 상태뿐 아니라, 설정된 조회 기간 동안의 누적 통계도 함께 제공합니다.
액티브 트랜잭션, **응답시간**, **에러율**은 차트를 통해 시각적으로 분석할 수 있습니다.

- **액티브 트랜잭션**: 현재 처리 중인 트랜잭션 수를 스피드미터 형태로 표시하여, 실시간 부하 상태를 한눈에 파악할 수 있습니다.
- **TPS (Transactions Per Second)**: 초당 처리되는 트랜잭션 수로, 시스템의 처리 성능을 평가할 수 있습니다.
- **트랜잭션 수**: 조회 기간 동안 수집된 총 트랜잭션 수로, 전체 요청 규모를 파악할 수 있습니다.
- **트랜잭션/min**: 분당 평균 트랜잭션 수를 나타내며, 시간 흐름에 따른 트래픽 변화를 분석하는 데 유용합니다.
- **응답시간**: 평균 응답시간과 최대 응답시간을 제공하며, 시간에 따른 응답 속도 변화를 차트로 확인할 수 있습니다.
- **에러 수**: 조회 기간 중 발생한 총 에러 수로, 시스템 오류 발생 빈도를 파악할 수 있습니다.
- **에러율(%)**: 전체 트랜잭션 중 오류가 발생한 비율을 백분율로 표시하며, 서비스 안정성을 평가하는 지표입니다.
- **에러/min**: 분당 평균 에러 수로, 에러 발생 추이를 시간 흐름에 따라 분석할 수 있습니다.

비즈니스 상세 분석

Top 트랜잭션 통계 조회

비즈니스별로 주요 트랜잭션의 성능 통계를 확인할 수 있습니다. 테이블 뷰 또는 카드 뷰에서 상세보기 아이콘을 클릭하면, 해당 비즈니스의 상위 30개 트랜잭션 통계가 오른쪽 드로어 창에 표시됩니다.

> **상세보기 버튼 안내**
>
> - 테이블 뷰: 테이블 가장 왼쪽 열에 있는 상세보기
> - 카드 뷰: 카드 오른쪽 위 화살표(>)

1. 테이블 또는 카드 뷰에서 상세보기 아이콘을 클릭합니다.
2. 오른쪽에 드로어 창이 열리며, 해당 비즈니스의 Top 30 트랜잭션 통계를 확인할 수 있습니다. 상단에서 정렬 기준을 선택할 수 있으며, 기본 정렬은 건수 기준입니다.
3. 트랜잭션 목록에서는 다음 항목들을 확인할 수 있습니다:
    - 트랜잭션: 트랜잭션 이름
    - 상세 분석: 클릭 시 상세 분석 페이지로 전환
    - 건수: 해당 트랜잭션의 호출 횟수
    - 에러: 에러 발생 횟수
    - 평균 시간(ms)
    - 최대 시간(ms)
    - 평균 CPU 사용 시간
    - 평균 메모리 할당량
    - SQL 건수
    - 평균 SQL 시간(ms)
    - SQL 패치 건수
    - 평균 SQL 패치 건수

- HTTP 호출 건수
- 평균 HTTP 호출 시간(ms)

4. 상단 검색창을 이용해 특정 트랜잭션을 검색할 수 있습니다.
5. 각 트랜잭션의 상세 분석 을 클릭하여 더 자세한 정보를 확인할 수 있습니다.
6. 조회 구간 내 URL 기준으로 설정된 트랜잭션 이벤트가 발생한 경우, 설정된 레벨이 표시됩니다.
   ⚠(경고) 또는 ⚠(위험) 레벨일 경우, Health 컬럼의 아이콘을 클릭하여 이벤트 히스토리를 확인할 수 있습니다.

이벤트 히스토리

조회 구간 내 집계 기준이 **업무별**로 설정된 트랜잭션 이벤트가 발생한 경우 설정된 레벨을 나타냅니다. 아이콘을 클릭하여 이벤트 히스토리를 확인할 수 있습니다.

- 이벤트 이름, 발생 시각, 메시지 등을 확인할 수 있습니다.
- 검색 기능을 통해 특정 이벤트를 찾을 수 있습니다.

에이전트 설정

비즈니스 대시보드를 사용하려면 에이전트에서 관련 설정이 필요합니다.

지원 에이전트 버전

- **Java**: 2.2.40 이상

필수 설정

```
# 기능 전체 활성화 여부
app_context_enabled=true
```

선택 설정

```
# 파서 선택 (default/prefix/match)
# default: 첫 번째 경로 세그먼트 추출
# prefix: 접두사 매칭 파서 (가장 유연함)
# match: 정확한 매칭 파서 (완전 일치만)
app_context_parser=default

# 파서의 내부 캐시와 설정을 강제로 재설정하는 카운터
app_context_parser_reset=0

# URL에서 추출할 경로 세그먼트의 깊이 지정 (default 파서에서 1 또는 2로 사용)
app_context_path_depth=1

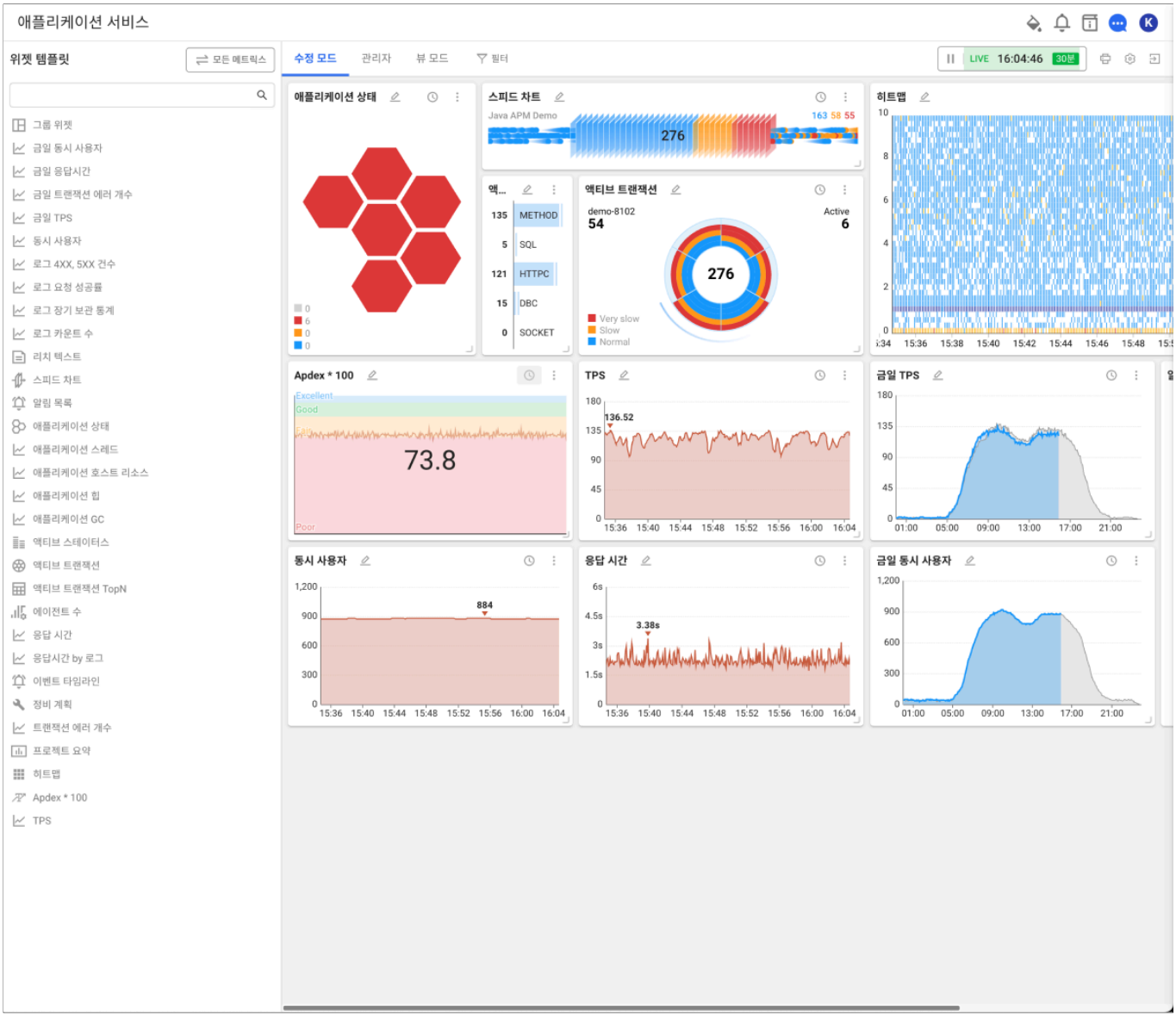
# 모니터링할 경로와 해당 이름을 정의하는 매핑 설정 (쉼표로 구분하여 사용)
app_context_path_set=name1@url1,name2@url2,url3,name4@url4
```

- 비즈니스 대시보드에 데이터가 표시되지 않는 경우, 에이전트 설정을 확인하세요.
- 에이전트 버전이 지원 버전보다 낮은 경우, 에이전트 업데이트가 필요합니다.
- 비즈니스 컨텍스트는 URL 패턴을 기준으로 자동 또는 수동으로 정의할 수 있습니다.

Flex 보드

Flex 보드는 사용자 정의 통합 대시보드입니다. 애플리케이션, 서버, 데이터베이스, 컨테이너 등 프로젝트의 데이터로 실시간 대시보드를 만들 수 있습니다.

사전 구성된 템플릿으로 초기 설정을 빠르게 마치고 원하는 형태의 대시보드를 만들 수 있습니다. 데이터 위젯을 추가하거나 속성을 수정해 필요한 형식으로 표시할 수 있으며, 필터링 기능으로 모니터링 대상을 간추릴 수도 있습니다. 시간 범위를 지정하면 특정 시점의 데이터를 확인할 수 있고, 보조 차트를 활용해 다양한 방식으로 분석할 수 있습니다. 대시보드는 즐겨찾기에 등록해 쉽게 접근할 수 있으며, 개인화한 대시보드는 다른 계정으로 복사해 재사용할 수 있습니다.



홈 화면 > 통합 Flex 보드

- 위젯 생성 시 조회 가능한 모든 프로젝트를 선택 옵션으로 제공합니다.
- 사용자 계정에 대시보드가 저장되며 다른 사용자에게 복사하기 기능을 이용해 공유할 수 있습니다.
- 개인 계정 대시보드로 권한에 따른 영향은 없으나 읽기 전용으로 공유된 대시보드의 경우 수정할 수 없습니다.

홈 화면 > 프로젝트 선택 > 대시보드 > Flex 보드

- 위젯 생성 시 해당 프로젝트 정보를 자동 입력합니다.

- 프로젝트 멤버들에게 생성한 Flex 보드가 자동 공유됩니다.
- 프로젝트 **수정** 권한 또는 **프로젝트 플렉스보드 편집** 권한, **Site Admin** 권한을 가진 사용자가 수정할 수 있습니다.

Flex 보드의 수정 권한이 있는 사용자는 다음 기능을 이용할 수 있습니다.

- 대시보드를 JSON 파일로 내보내기/가져오기
- 대시보드 내의 데이터 요청 및 응답 내용 확인
- 위젯 설정 옵션을 JSON 형식으로 조회 및 수정

프로젝트 내 **Flex 보드** 메뉴에서는 대시보드 수정 권한이 있는 사용자만 **수정 모드** 및 **관리자** 모드, **필터** 기능에 접근할 수 있습니다.

- **접근 가능 멤버 권한**
  - **프로젝트 수정** 권한
  - **프로젝트 플렉스보드 편집** 권한
  - **Site Admin** 권한
- **뷰 모드** 및 **필터** 기능에 대한 자세한 내용은 다음 문서를 참고하세요.
- 멤버 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

Flex 보드 사용 가이드

> 대시보드를 메트릭스 위젯을 포함해 다양한 위젯 템플릿을 활용해 구성할 수 있습니다.
> 각 템플릿의 특징과 사용법은 위젯 템플릿 문서에서 자세히 확인할 수 있습니다.

사전 템플릿 적용하기

사전 구성 템플릿으로 초기 설정을 빠르게 진행하고 필요에 따라 수정해 활용하세요.

1. **Flex 보드** 메뉴를 선택하세요.
2. 왼쪽 **템플릿** 목록에서 원하는 템플릿을 선택하세요.
3. **Flex 보드 관리** 창이 나타나면 **이름 변경**에 원하는 이름을 입력하세요.
    - **레이아웃**에서 위젯을 선택 후, 드래그하여 원하는 위치로 이동할 수 있습니다.
4. **저장** 버튼을 클릭세요. 선택한 템플릿 기준의 Flex 보드를 생성합니다.
    - **Flex 보드**의 보드 목록에서 생성한 보드를 확인할 수 있습니다.

Flex 보드 만들기

Flex 보드를 생성하고 위젯을 배치해 자신만의 대시보드를 만들 수 있습니다.

1. **Flex 보드**에서 오른쪽 위의 **대시보드 생성하기** 버튼을 클릭하세요.
2. **대시보드 생성하기** 창이 나타나면 입력 상자에 대시보드 **이름**을 입력하세요.
3. 위젯 배치 방식을 선택한 후, **대시보드 생성하기** 버튼을 클릭하세요.
    - **고정 레이아웃 플렉스보드**: 픽셀 기반으로 위젯 자유 배치
    - **반응형 플렉스보드**: 브라우저 크기 기준 그리드 반응형 레이아웃
      (그리드 설정한 뒤 **대시보드 생성하기** 버튼 클릭)
4. **위젯 템플릿** 선택 화면에서 대시보드에 배치할 위젯을 선택하세요.

- 메트릭스 위젯을 추가하려면 **위젯 템플릿**의 오른쪽 **모든 메트릭스** 버튼을 클릭하세요.

> ⓘ
> - **위젯 템플릿**: 일반적인 모니터링 상황에서 중요하게 다뤄지는 지표를 간추려 사전 정의된 위젯 목록을 선택할 수 있습니다.
> - **모든 메트릭스**: 사용자의 프로젝트에서 수집 중인 모든 메트릭스 데이터를 기준으로 위젯을 생성합니다. 메트릭스 위젯에 대한 자세한 설명은 다음 문서를 참고하세요.

5. 배치한 위젯의 위치를 이동하거나 크기를 조절하세요.
   - 위젯의 위쪽에 커서를 올려 십자 커서가 보이면, 위젯을 클릭한 채 드래그하여 위치를 이동합니다.
   - 위젯 오른쪽 아래에 커서를 올려 화살표 커서가 보이면, 위젯을 클릭한 채 드래그하여 크기를 조절합니다.
6. 추가할 위젯 배치를 완료하면 **뷰 모드**를 선택해 레이아웃을 확인하세요. 레이아웃을 다시 수정하려면 **수정 모드**를 선택해 위젯의 배치를 변경하세요.
7. 모든 과정을 완료했다면 오른쪽 위 (**목록으로 가기**) 아이콘을 클릭하세요.
   - **Flex 보드**의 **대시보드 목록**에서 생성한 보드를 확인할 수 있습니다. 생성한 보드를 선택해 새로 생성한 Flex 보드를 확인할 수 있습니다.

> ⓘ 고객의 의견을 반영해 대시보드 템플릿 또는 위젯을 추가하려고 합니다. 필요한 대시보드 템플릿 또는 위젯이 있다면 support@whatap.io로 문의해 주세요.

Flex 보드 편집하기

> ⓘ **Flex 보드 편집 권한**
> - **홈 화면** > **통합 Flex 보드**: 개인 계정 대시보드로 권한 영향은 없으나 읽기 전용으로 공유된 대시보드는 수정할 수 없습니다.
> - **홈 화면 > 프로젝트 선택** > **대시보드** > **Flex 보드**: 프로젝트 수정 권한 또는 프로젝트 플렉스보드 편집 권한, Site Admin 권한을 가진 사용자가 수정할 수 있습니다.
>
> 멤버 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

1. **Flex 보드**의 **대시보드 목록**에서 수정 아이콘을 클릭하세요. 화면 오른쪽에 **Flex 보드 관리** 창이 나타납니다.
2. **Flex 보드 관리** 창에서 이름, 프로젝트, 레이아웃, 위젯.json, 옵션.json을 설정합니다.
   - **이름 변경**: 대시보드의 이름을 수정합니다.
   - **프로젝트**: 대시보드 위젯에 데이터를 가져올 프로젝트를 선택합니다.

     > **프로젝트** 옵션은 **홈 화면** > **통합 Flex 보드** 메뉴에서만 설정할 수 있습니다.
     > **통합 Flex 보드** 메뉴로 진입한 경우, 위젯마다 데이터를 가져올 프로젝트를 선택할 수 있습니다.
     >
     > 1. 위젯 오른쪽 위 ＋ 버튼을 클릭하고 원하는 프로젝트를 검색해 하나 또는 다중 선택하세요.
     > 2. 프로젝트를 선택한 후 **적용** 버튼을 클릭하세요.

   - **레이아웃**
     - 위젯을 드래그하여 위치를 이동합니다.
     - 위젯 오른쪽 아래를 마우스로 클릭한 상태로 드래그해 위젯의 크기를 조절합니다.
     - 위젯을 삭제하려면 위젯 오른쪽 위에 아이콘을 클릭하세요.
   - **위젯.json**: 대시보드의 위젯 설정을 JSON 형식으로 불러올 수 있습니다. 아이콘을 클릭하면 JSON 내용을 복사합니다.
   - **옵션.json**: 대시보드에 설정된 옵션을 JSON 형식으로 불러올 수 있습니다. 아이콘을 클릭하면 JSON 내용을 복사합니다.

Flex 보드 삭제하기

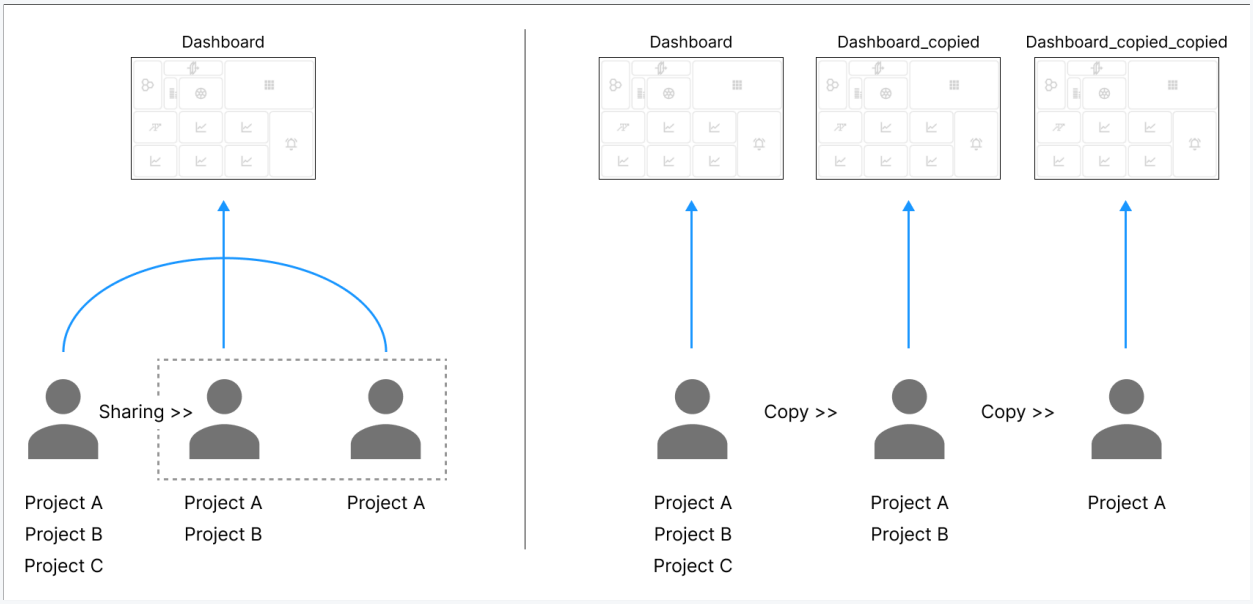
1. **Flex 보드** > **대시보드 목록**에서 삭제할 항목의 아이콘을 클릭하세요.
2. 확인 메시지에서 **삭제** 버튼을 클릭하세요.

> 대시보드 삭제는 소유자와 공유받은 사용자에 따라 다르게 적용됩니다.
> - 대시보드 소유자가 대시보드를 삭제하면, 공유된 모든 사용자 계정에서도 해당 대시보드를 삭제됩니다.
> - 대시보드를 공유받은 사용자가 대시보드를 삭제하면, 해당 사용자의 **대시보드 목록**에서만 삭제됩니다. 원본 대시보드는 유지됩니다.

Flex 보드 공유하기

Flex 보드 공유 기능은 **홈 화면 > 통합 Flex 보드** 메뉴에서만 사용할 수 있습니다.
프로젝트의 **Flex 보드** 메뉴에서는 제공되지 않습니다.

공유와 복사의 차이



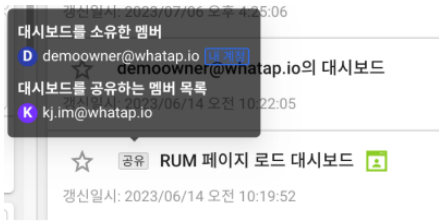
- 공유: 공유한 멤버는 모두 같은 대시보드를 볼 수 있습니다.
- 복사: 대시보드를 다른 멤버 또는 나에게 복사하면, 복사된 별도의 대시보드를 볼 수 있습니다.
- **읽기 전용**로 공유/복사받은 대시보드는 수정할 수 없습니다. 단, **수정 모드**로 공유/복사된 경우는 수정할 수 있습니다.
- **읽기 전용**로 공유받은 멤버도 해당 대시보드를 **읽기 전용으로 재공유**하거나 **복사**할 수 있습니다.
- 대시보드에 포함된 프로젝트 중 **최소 1개** 프로젝트에 초대된 멤버에게만 공유할 수 있습니다. 공유 또는 복사 받을 멤버의 프로젝트 소속 여부를 확인하세요.



통합 Flex 보드에서 생성한 Flex 보드를 다른 멤버와 공유할 수 있습니다. 소유자가 공유된 대시보드를 수정하면, 공유받은 모든

멤버에게 동일하게 반영됩니다.

1. **통합 Flex 보드**의 **대시보드 목록**에서 공유할 대시보드의 아이콘을 클릭하세요.
2. **대시보드 공유하기** 창이 나타나면 입력창에 공유할 멤버를 한 명 또는 여러명을 선택하거나 계정을 직접 입력하세요.
   - **읽기 전용**를 선택하면 공유받은 멤버는 대시보드를 수정할 수 없습니다.
   - 공유 대상은 대시보드에 포함된 프로젝트 중 최소 1개 프로젝트에 소속되어 있어야 합니다.
3. **공유** 버튼을 클릭하세요.
   - 공유된 항목은 **공유** 태그가 표시됩니다. **공유** 태그에 마우스를 올리면 공유한 멤버 정보를 확인할 수 있습니다.
   - 읽기 전용으로 공유한 항목은 **읽기 전용** 태그가 표시됩니다.

대시보드를 공유받은 경우	대시보드를 공유한 경우(대시보드 소유자)
	

대시보드 공유 조건

- 대시보드 공유 기능은 **홈 화면** > **통합 Flex 보드**에서만 제공합니다.
- 누구나 자신의 대시보드를 공유할 수 있으며, 공유받은 대시보드를 다시 공유하거나 복사할 수 있습니다.
- **읽기 전용**로 공유받은 대시보드는 수정할 수 없지만, **수정 모드**로 공유받은 대시보드는 수정할 수 있습니다.
- **소유자**가 대시보드를 삭제하면, 공유받은 모든 계정에서 해당 대시보드가 삭제됩니다.
- **공유받은 사용자**가 삭제하면, 해당 사용자의 **대시보드 목록**에서만 삭제되며 원본은 유지됩니다.
- 공유받은 대시보드의 프로젝트 중 **조회 분석** 권한이 없는 프로젝트는 데이터를 조회할 수 없습니다.

Flex 보드 복사하기

사용자가 Flex 보드를 복사해 다른 멤버에게 전달할 수 있습니다.

1. **통합 Flex 보드**의 **대시보드 목록**에서 아이콘을 클릭하세요.
2. 복사 옵션을 선택하세요.
   - **나에게 복사**: 자신의 계정으로 복사합니다.

     a. `_copied` 접미어가 붙은 항목이 **대시보드 목록**에 추가됩니다.

   - **다른 사람에게 복사**: 다른 멤버 계정으로 복사합니다.

     a. ＋ **계정**을 클릭하세요.
     b. 복사할 멤버의 이메일을 선택하고 **대시보드 복사하기** 버튼을 클릭하세요.
     c. 해당 멤버의 **대시보드 목록**에 `_copied` 접미어가 붙은 항목이 추가됩니다.

JSON 파일로 공유하기

Flex 보드 설정을 JSON 파일로 다른 멤버에게 전달하거나 다른 멤버의 설정을 가져올 수 있습니다.

내보내기

1. **통합 Flex 보드** 또는 프로젝트의 **Flex 보드** 메뉴로 이동하세요.
2. **대시보드 목록**에서 버튼을 클릭하세요.
3. 내보내기 옵션을 선택하세요.
   - **프로젝트 정보 포함**: 대시보드에 설정된 프로젝트 정보를 JSON 파일에 포함해 공유
   - **프로젝트 정보 미포함**: 대시보드에 설정된 프로젝트 정보를 JSON 파일에 포함하지 않고 공유
4. JSON 파일을 다운로드하고 공유할 멤버에게 전달하세요.

가져오기

1. **통합 Flex 보드** 또는 프로젝트의 **Flex 보드** 메뉴로 이동하세요.

2. 화면 오른쪽 위 **가져오기** 버튼을 클릭하세요.
3. 다운로드한 JSON 파일을 선택하세요.

> ⓘ 이 기능은 프로젝트의 **수정** 권한을 소유한 멤버만 이용할 수 있습니다.

Flex 보드 화면 모드

Flex 보드의 화면 모드에 따라 제공하는 기능을 설명합니다.

> ⓘ **접근 가능한 멤버 권한**
>
> 프로젝트 내 **Flex 보드** 메뉴에서 **수정 권한**이 있는 사용자만 **수정 모드**, **관리자**, **필터** 기능에 접근할 수 있습니다. 멤버 권한에 대한 자세한 내용은 다음 문서를 참조하세요.
>
> - **프로젝트 수정** 권한
> - **프로젝트 플렉스보드 편집** 권한
> - **Site Admin** 권한

모드 종류

Flex 보드는 화면 모드에 따라 설정할 수 있는 기능이 다릅니다.

- **수정 모드**: 위젯 템플릿 목록에서 Flex 보드의 레이아웃에 위젯을 배치하거나 위젯을 수정, 삭제, 위치 조정할 수 있습니다.
- **관리자**: 위젯의 세부 속성을 json 형태로 관리할 수 있습니다. 레이아웃에 배치한 위젯의 오른쪽 위에 위치한 🔧 버튼을 선택하세요.
- **뷰 모드**: 위젯 배치, 크기 조절, 세부 옵션 등의 설정을 완료한 다음 대시보드와 같이 각 위젯의 데이터를 실시간으로 모니터링할 수 있습니다.

공통 기능

- 데이터 필터링

- 실시간 데이터 조회 범위 설정
- 인쇄 모드
- Flex 보드 관리
- 데이터 병합 옵션

데이터 필터링

데이터 필터링을 통해 모니터링 대상을 간추려 효율적인 Flex 보드를 구성할 수 있습니다.

1. 화면 위 **필터** 탭을 클릭하세요.
2. 화면 오른쪽에 나타나는 **필터** 창에서 필터링 대상을 선택하세요. 필터링 대상을 모두 선택하려면 **전체 선택**을 선택하세요.
   - **에이전트** 옵션: 에이전트 종류 또는 애플리케이션 목록 선택, **검색** 기능 제공
   - **메트릭스** 옵션: 프로젝트 또는 에이전트 유형 선택, **+ And** 조건 추가하거나 **포함**, **제외**로 문자열 기반 자동 선택 가능
3. 필터링 대상을 선택했다면 **적용** 버튼을 클릭하세요.
4. 자주 사용하는 조건은 **필터 저장** 버튼으로 저장하세요.

   a. 필터 이름을 입력하고 **저장**을 선택하세요. **저장된 필터**는 **필터** 창 하단에 나타납니다.

필터링 조건

필터링 조건은 에이전트 타입과 메트릭스 타입으로 구분합니다.
에이전트는 기본 모니터링 대상만 선택 옵션으로 제공하며, 메트릭스는 프로젝트에서 수집한 모든 메트릭스 데이터의 태그를 선택 옵션으로 제공합니다. 메트릭스 타입은 에이전트가 아닌 형태로 데이터를 수집한 후 필터링을 적용할 때 유용합니다. 메트릭스에 대한 자세한 내용은 다음 문서를 참조하세요.

타입	에이전트	메트릭스
선택 옵션	와탭 프로젝트 모니터링 대상 (애플리케이션, 서버, 데이터베이스, 컨테이너, 쿠버네티스 POD와 NODE)	에이전트를 포함한 모든 메트릭스 태그
적용 범위	모든 위젯	메트릭스 데이터를 사용하는 모든 위젯

타입	에이전트	메트릭스
필터링 조건	체크박스	**일치**, **포함**, **제외** 방식

조회 시간 설정하기

Flex 보드의 화면 오른쪽 위 시간 선택자를 이용해 위젯에 데이터 조회 시간을 설정할 수 있습니다.

- ||: 실시간 모드 멈추고 특정 조회 시간으로 설정
- ▶: 다시 실시간 모드로 전환

인쇄하기

Flex 보드 화면을 PDF로 저장해 인쇄할 수 있습니다.

1. Flex 보드 화면의 오른쪽 위 🖨 아이콘을 클릭해 인쇄 모드로 전환합니다.
2. 옵션을 설정합니다.
    - **필터 옵션 보기**: 화면에 필터 옵션 켜거나 끄기
    - **{mode} 방향으로 변환 / {mode} 방향으로 변환**: 화면을 가로 또는 세로로 변환
3. 완료 후 **PDF다운로드**를 선택하세요.
4. 인쇄 모드를 종료하려면 오른쪽 위 ✕ **인쇄모드 종료**를 클릭하세요.

Flex 보드 관리

Flex 보드의 레이아웃을 변경하거나 위젯의 위치 변경, 크기 조절, 이름 변경 등의 관리 기능을 제공합니다. **Flex 보드 관리** 기능은 **대시보드** > **Flex 보드** 메뉴의 **대시보드 목록**에서 ✎ 아이콘을 클릭해 진입할 수도 있습니다.

1. Flex 보드 화면의 오른쪽 위 ⚙ 아이콘을 클릭하면 **Flex 보드 관리** 창이 열립니다.
    - **이름 변경**: Flext 보드의 이름 변경
    - **레이아웃**: 위젯의 위치 및 크기조절 및 삭제
    - **다른 사용자의 수정을 제한**: 수정 제한 옵션
        - 수정 권한이 있는 멤버는 수정, 삭제, 편집할 수 있으며 옵션을 활성화하면 **다른 사용자의 수정을 제한: On**으로

변경됨

- **위젯.json**: Flex 보드에 포함된 위젯을 JSON 형태로 편집 및 관리
- **옵션.json**: Flex 보드의 설정을 JSON 형태로 편집 및 관리(프로젝트 정보, Flex 보드 필터, 시간 선택 등 설정값 편집)

> **위젯.json**, **옵션.json** 기능은 대시보드 관리자 권한의 사용자에게만 제공합니다.

2. 모든 설정을 완료하고 **저장** 버튼을 클릭하세요.

대시보드 즐겨찾기로 등록

사용자가 생성한 Flex 보드는 즐겨찾기에 등록해 빠르게 접근할 수 있습니다. **대시보드** > **Flex 보드**의 **대시보드 목록**에서 즐겨찾기로 등록할 대시보드의 ☆ 버튼을 클릭하면 ★으로 변경됩니다.

Flex 보드 위젯 관리

Flex 보드에 배치한 위젯 관리를 통해 대시보드를 사용자가 원하는 대로 꾸밀 수 있습니다.

> 대시보드를 메트릭스 위젯을 포함해 다양한 위젯 템플릿을 활용해 구성할 수 있습니다.
> 각 템플릿의 특징과 사용법은 위젯 템플릿 문서에서 자세히 확인할 수 있습니다.

위젯 추가

이미 만들어진 Flex 보드에 위젯을 추가할 수 있습니다.

1. **Flex 보드** 메뉴의 **대시보드 목록**에서 위젯을 추가할 대시보드를 선택하세요.
2. 화면 왼쪽 위에 **수정 모드**를 선택하세요.
3. 왼쪽에 **위젯 템플릿** 목록에서 원하는 위젯을 선택하세요. 메트릭스 위젯을 추가하려면 **모든 메트릭스**를 선택한 다음 원하는 위젯을 클릭하세요.
   - 홈 화면의 **통합 Flex 보드**로 진입한 경우 위젯을 추가했다면 **프로젝트 선택** 버튼을 클릭해 수집 대상 프로젝트를 선택하세요.
   - 추가할 위젯이 있다면 같은 과정을 반복하세요.
4. 모든 과정을 완료했다면 화면 위에 **뷰 모드**를 선택해 레이아웃을 확인한 후, (**목록으로 가기**) 버튼을 클릭하세요.

위젯 속성 변경

Flex 보드 메뉴의 **대시보드 목록**에서 위젯 속성을 변경할 보드를 선택하세요. **수정 모드**로 진입합니다.

위젯 이름 변경

1. 이름을 변경할 위젯의 이름 오른쪽에 아이콘을 클릭하세요.
2. 위젯의 **제목**, **폰트 크기**, **색상**을 설정한 후, **저장** 버튼을 클릭하세요.

- 저장 시 **전체 위젯에 적용** 토글 버튼의 기능을 활성화하면 폰트 크기와 색상을 다른 위젯에도 적용합니다.
- 위젯의 제목을 잠금 설정하려면 버튼을 선택한 후 **저장** 버튼을 클릭하세요.

시간 설정

1. 시간을 설정하려는 위젯의 오른쪽 위에 버튼을 클릭하세요.
2. **시간 선택** 창에서 시간을 설정하세요.
   - 대시보드 화면에 설정한 시간을 적용하려면 **대시보드 설정을 따름**을 체크하세요. 대시보드의 시간 설정은 오른쪽 상단에 위치합니다.
   - 사용자가 원하는 시간 설정을 적용하려면 **대시보드 설정을 따름**을 체크 해제하세요. < 또는 > 버튼을 이용해 원하는 시간을 설정하세요.
3. 시간 설정을 완료했다면 버튼을 클릭하세요.

> 위젯의 데이터 유형에 따라서 시간 설정 기능을 지원하지 않을 수 있습니다.

출력 데이터 옵션 설정

차트형 위젯의 출력 데이터 옵션을 설정해 사용자가 원하는 데이터를 선택할 수 있습니다.

1. 위젯의 오른쪽 위에 ⋮ 아이콘을 클릭해, 출력 데이터 옵션을 설정하세요.
   - 프로젝트, 에이전트, 애플리케이션 등의 종류를 선택해 데이터를 필터링할 수 있습니다.
   - 개별, 병합 데이터를 선택해 차트에 표시되는 그래프의 유형을 변경할 수 있습니다.
   - ⋮ > 을 선택하세요. **차트 설정** 창을 통해 세부 옵션을 변경할 수 있습니다. 설정할 수 있는 옵션은 다음과 같습니다.
     - **차트 유형**: 집계하는 현재 수치를 차트에 표시할 수 있는 옵션입니다.
     - **y축 차트 최대값**: y축에 차트 최댓값을 **자동** 또는 **고정**으로 선택할 수 있습니다.
     - **수평선 설정**: **추가** 버튼을 선택해 임계치 또는 범위를 수평선으로 표시할 수 있습니다. 여러 개를 추가해 적용할 수 있습니다. 수평선을 추가한 다음에는 **적용** 버튼을 선택해야 차트에 반영됩니다.
     - **차트 최대값 표시** 차트에 표시되는 그래프의 최댓값을 표시합니다.

- **보조 차트**: 위젯의 주요 차트에 보조 차트를 추가 설정할 수 있습니다. 보조 차트의 종류를 선택한 다음 레이아웃, 크기, 데이터 병합 여부를 선택하세요. 다른 위젯에도 공통 적용하려면 **전체 위젯에 적용** 버튼을 선택하세요. 보조 차트는 데이터를 다양한 방법으로 보고 싶을 때 유용합니다.

2. 옵션 설정을 모두 완료했다면 **차트 설정** 창의 왼쪽 위에 ✕ 버튼을 클릭하거나 **차트 설정** 창 영역 밖을 클릭하세요.

> ⓘ 위젯의 데이터 유형에 따라서 선택할 수 있는 옵션은 다를 수 있습니다.

메트릭스 위젯 데이터 변경

메트릭스 위젯의 데이터 조회 조건을 변경할 수 있습니다.

1. **Flex 보드** > **대시보드 목록**에서 위젯 데이터를 변경할 대시보드를 선택하세요.
2. **수정 모드**에서 변경할 위젯의 오른쪽 위에 ⋮ 아이콘을 클릭하세요.
3. </> 아이콘을 클릭하면, 데이터 설정 편집 창이 나타납니다.
4. 다음을 참조해 설정을 변경한 다음 < **위젯에 적용** 버튼을 클릭해 적용하세요.
   - 위젯 생성 시 기본 생성되는 메트릭스 옵션이 탭에 표시됩니다.
   - 메트릭스 옵션에는 **카테고리**와 **필드**, **태그**가 있습니다.
   - **카테고리** 항목은 현재 변경 기능을 제한합니다.
   - **필드**, **태그** 항목은 복수 선택할 수 있습니다. 선택한 태그 옵션으로 차트 데이터를 그룹화합니다.
   - 🔍 버튼을 선택해 원본데이터를 조회할 수 있습니다.
   - **오브젝트 병합** 및 **시간 병합** 항목에 대한 자세한 내용은 데이터 병합 옵션을 참조하세요.

데이터 병합 옵션

Flex 보드의 **모든 메트릭스** 목록에서 배치한 메트릭스 위젯은 데이터 병합 옵션을 제공합니다. 데이터 병합은 **오브젝트 병합**과 **시간 병합** 방법을 제공합니다.

- **오브젝트 병합**

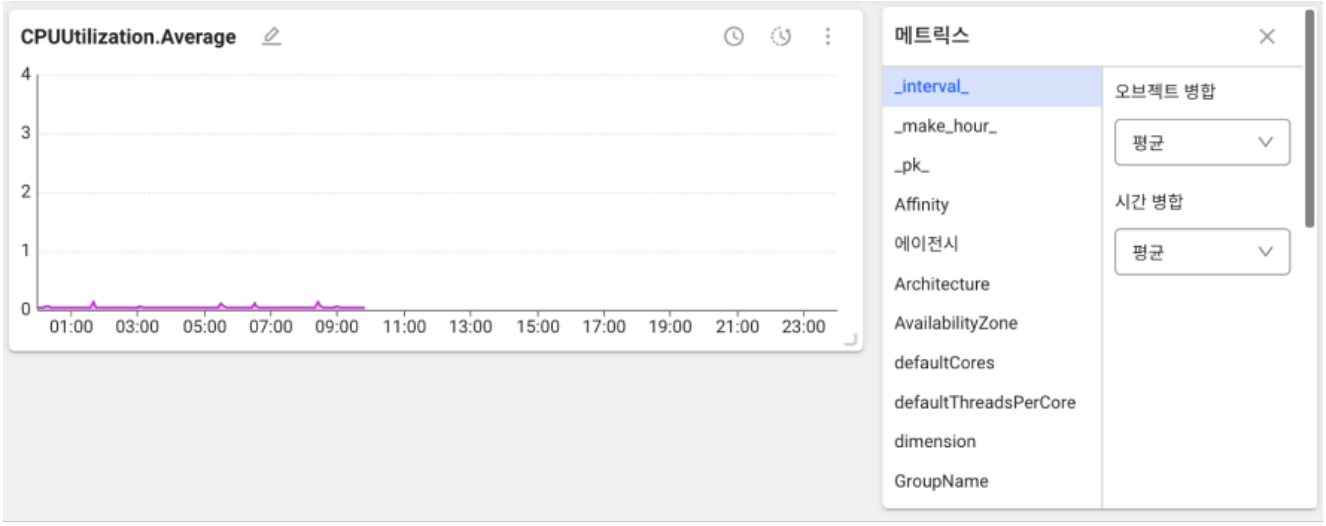
  좌측에서 선택한 태그(예: 에이전트 명)를 기준으로 차트 데이터를 그룹화할 때 사용합니다. 이때 서로 다른 필드(예: CPU)값을 가진 데이터들 중에서 태그가 일치하는 경우 해당 데이터를 병합하는 방법입니다.

- **시간 병합**

  조회 시간이 긴 경우 주로 사용합니다. 원본 데이터에서 필드 값이 같은 데이터끼리 시간 범위에 따라 일정한 시간 간격(5분 또는 1시간)으로 데이터를 병합합니다. 이 병합 방법을 시간 병합이라고 합니다.

- **AWS CloudWatch 메트릭스**

  AWS 모니터링 지표의 경우 통계 방식이 필드명 뒤에 suffix(접미사) 형식으로 추가됩니다. 데이터 병합 기본값은 이 suffix를 참조하고 있습니다.



1. **수정 모드**에서 **모든 메트릭스** 버튼을 클릭해 **위젯 템플릿** 목록을 **모든 메트릭스** 목록으로 변경하세요.
2. 메트릭스 위젯을 레이아웃에 배치하세요.
3. 메트릭스 위젯 오른쪽 위에 ⋮ 버튼을 선택하세요.
4. 태그(예: 에이전트 명) 옵션을 선택하면 데이터 병합 옵션이 나타납니다. 위젯의 크기에 따라 메트릭스 옵션의 위치가 다를 수 있습니다.

위젯 복사

1. 위젯의 오른쪽 위에 ⋮ 버튼을 클릭하세요.
2. 버튼을 클릭하세요.
3. **복사** 창이 나타나면 ＋ **위젯 추가** 버튼을 클릭하세요.
4. 목록에 위젯의 이름을 변경하세요.

5. **적용** 버튼을 클릭하세요.
6. **수정 모드** 화면에서 복사한 위젯을 확인할 수 있습니다.

위젯 삭제

1. 위젯의 오른쪽 위에 ⋮ 아이콘을 클릭하세요.
2. ✕ 버튼을 클릭하세요.
3. 확인 메시지 창이 나타나면 **삭제** 버튼을 클릭하세요. 삭제를 취소하려면 **취소** 버튼을 선택하세요.

분석 보고서

애플리케이션 환경을 다양한 관점에서 분석할 수 있는 메뉴를 제공합니다. 부하가 높거나 성능이 저하된 구간을 파악할 수 있으며, 다양한 성능 지표를 제공해 일별 현황을 확인할 수 있습니다. 또한 성능 측정을 위한 트랜잭션을 추적하고 분석할 수 있습니다.

와탭에서 제공하는 분석 기능을 통해 주요 성능을 분석하고, 문제를 파악해 발생할 수 있는 장애에 대처할 수 있습니다.

일자별 애플리케이션 현황

애플리케이션의 주요 성능 지표들의 하루 동안 추이를 시간 단위 차트를 통해 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

성능 추이

특정 기간 동안 애플리케이션 주요 성능 지표들의 추이를 차트를 통해서 부하가 높거나 성능이 저하된 구간을 빠르게 파악할 수 있고, 문제가 된 시점을 특정할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

메트릭스 차트

시각화한 차트를 통해 메트릭스 데이터를 조회할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

메트릭스 조회

카테고리화된 애플리케이션의 성능 지표를 태그와 필드 기반으로 구성된 데이터로 조회할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

스택

트랜잭션의 스택 정보를 수집해 실행 중인 메소드의 사용량 통계와 많이 사용되는 스택에 대한 통계를 제공합니다. 실행 중인 트랜잭션의 스택 정보를 통해 장시간 실행되거나 짧은 시간 실행되지만 자주 실행되는 메소드를 파악할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

큐브

5분 단위로 만든 성능 통계를 큐브(Cube)라고 부릅니다. 사후 분석을 위한 통계 분석 도구로 응답 시간 및 에러 건수, 처리량 등이 높은 시간대를 특정할 수 있기 대문에 동 시간대의 문제 요소를 파악하거나 서비스 현황을 모니터링할 수 있습니다. 트래픽의 지리적 분포, 리소스 사용량을 파악할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

히트맵

시간의 흐름에 따라 사용자 의 요청에 대한 응답시간을 분포도 형태로 표현한 차트를 제공합니다. 분포도 차트의 형태에 따라 어떤 장애가 발생했는지 특정하는데 도움을 받을 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

멀티 트랜잭션 추적

다른 에이전트나 프로젝트와 연관된 트랜잭션 간의 호출을 추적합니다. 시스템 내 또는 시스템 간에 발생하는 다양한 호출 관계를 한 눈에 파악하고 어느 부분에서 문제가 발생했는지 식별하여 개선할 수 있도록 트랜잭션과 트레이스 정보를 제공합니다. 자세한 내용은 다음 문서를 참조하세요.

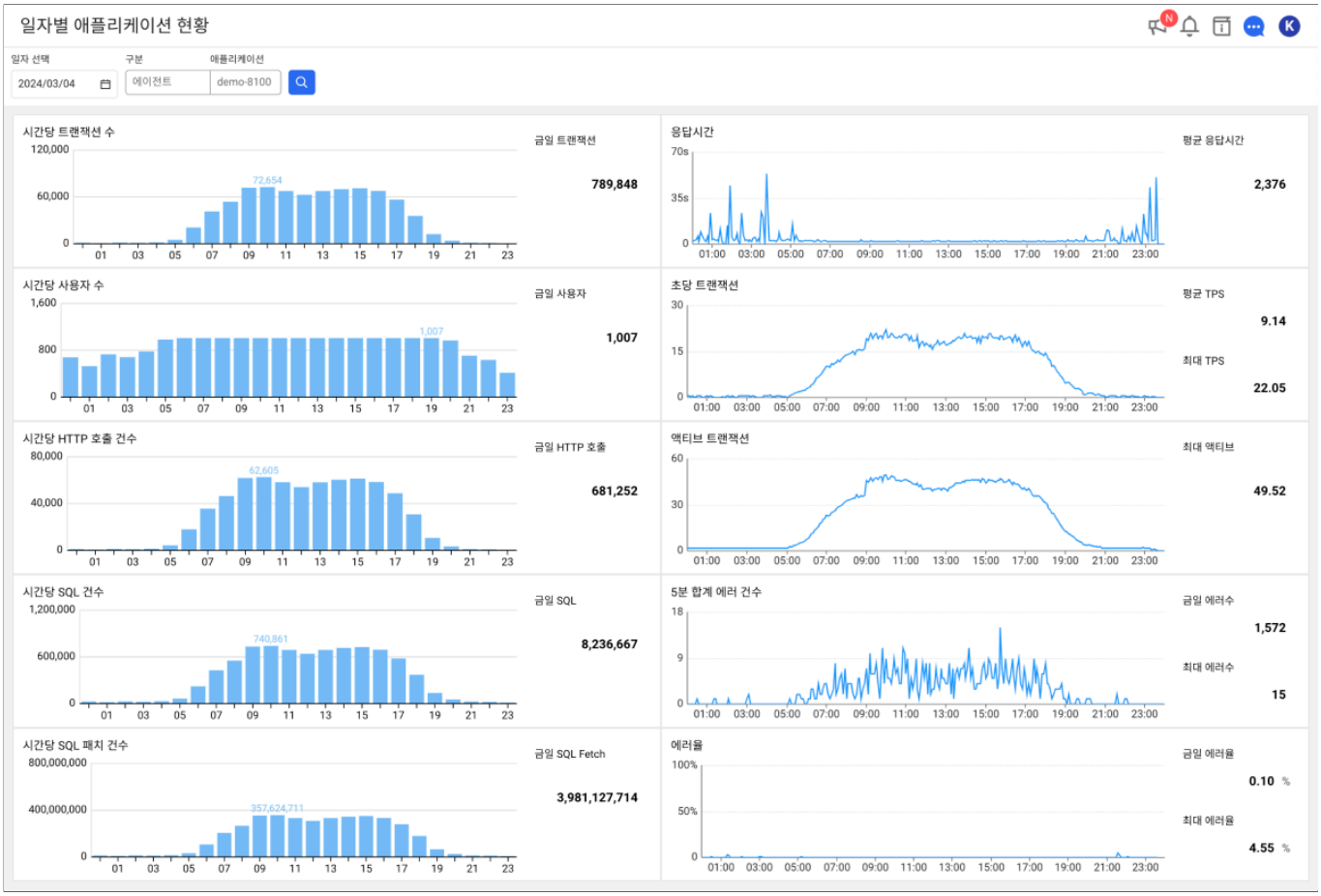
트랜잭션 검색

특정 기간 동안 각 트랜잭션의 성능 속성에 대한 통계와 실행 상태를 검색할 수 있습니다. 모든 트랜잭션을 확인하고 정상 및 에러 상태를 빠르게 확인할 수 있습니다. 제공되는 정보를 통해 어떤 트랜잭션의 사용량이 많고 리소스를 많이 소비하는지 알 수 있습니다. 트랜잭션의 속성에 따라 다양하게 필터링할 수 있어 특정 조건의 트랜잭션을 찾는데 유용합니다. 자세한 내용은 다음 문서를 참조하세요.

일일 애플리케이션 현황

홈 화면 > 프로젝트 선택 > **분석** > **일자별 애플리케이션 현황**

애플리케이션의 주요 성능 지표들의 하루 동안 추이를 시간 단위 차트를 통해 확인할 수 있습니다.



하루 동안의 애플리케이션 상태를 확인할 수 있습니다. 검색 조건(**일자 선택**, **구분**, **애플리케이션**)을 설정하고 버튼을 선택하세요. **구분** 옵션의 조건은 다음과 같습니다.

분류	설명
에이전트	프로젝트에 포함된 애플리케이션 에이전트 이름

분류	설명
에이전트 종류	에이전트 설정에서 `whatap.okind` 옵션으로 분류된 그룹 단위
에이전트 서버	에이전트 설정에서 `whatap.onode` 옵션으로 분류된 그룹 단위

제공하는 차트는 다음과 같습니다. 차트의 오른쪽에 위치한 수치는 하루 동안 누적된 값 또는 평균, 최댓값을 확인할 수 있습니다.

- **시간당 트랜잭션 수**
- **시간당 사용자 수**

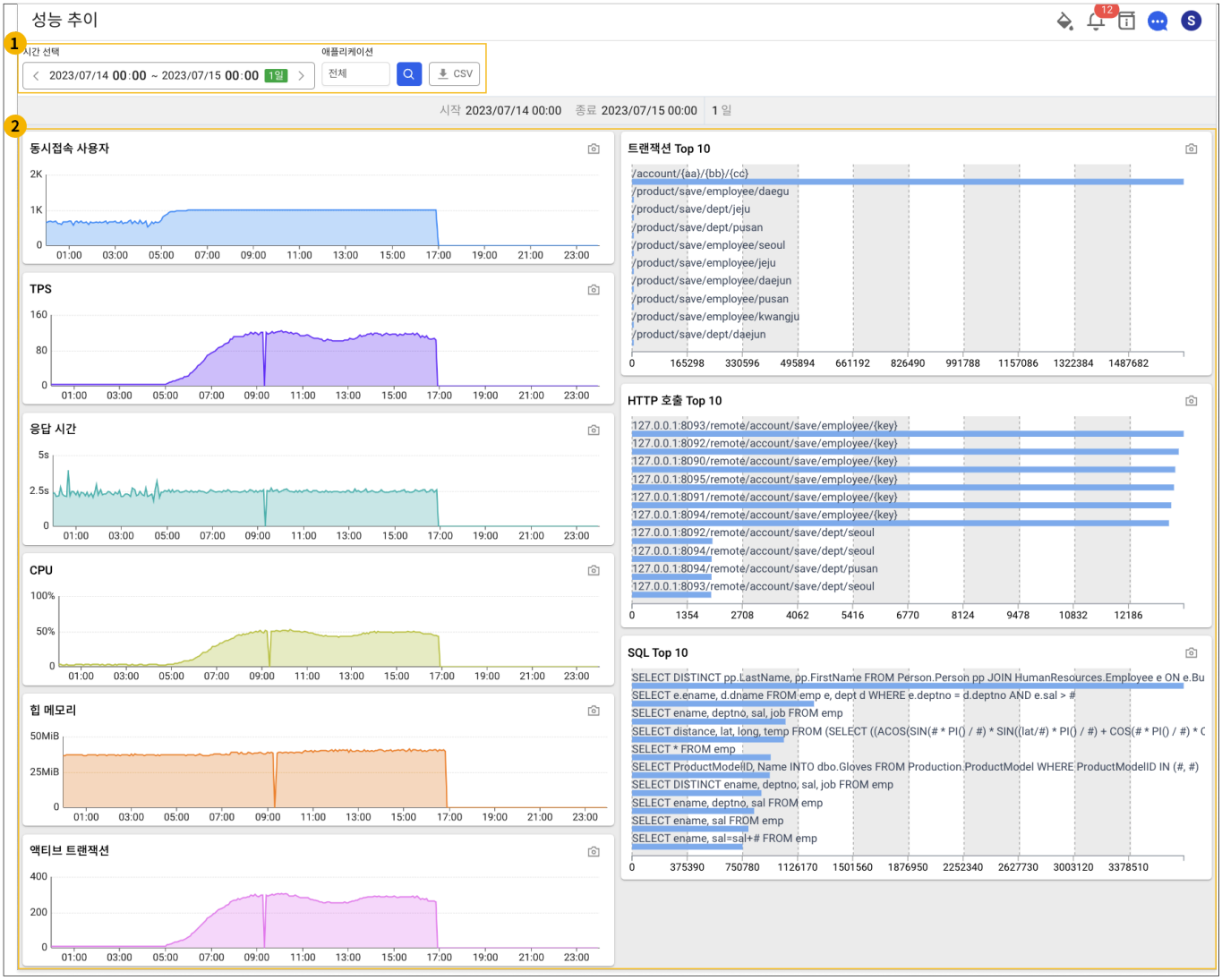
> ⓘ **금일 사용자**는 하루 동안 누적된 사용자를 중복없이 합산한 값입니다. 단순히 합산하는 것이 아니라 **HyperLogLog**로 산출합니다. **HyperLogLog**는 매우 적은 메모리로 집합의 원소 개수를 추정하는 확률적 자료 구조를 의미합니다.

- **시간당 HTTP 호출 건수**
- **시간당 SQL 건수**
- **시간당 SQL 패치 건수**
- **응답시간**
- **초당 트랜잭션**
- **액티브 트랜잭션**
- **5분 합계 에러 건수**
- **에러율**

성능 트렌드

홈 화면 > 프로젝트 선택 > **분석** > **성능 추이**

와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 다음 **분석** > **성능 추이** 메뉴를 선택하세요. 조회를 원하는 특정 시간 범위 내 수행된 성능에 대한 여러 지표를 확인할 수 있습니다.



1 성능 추이 조회 조건

- 시간 선택자를 통해 조회를 원하는 시간 범위를 선택할 수 있습니다. 기본 조회 기간은 **1일**입니다.
  - 조회 범위가 3시간 이하인 경우, Raw 데이터를 기반으로 5초 추이를 확인할 수 있습니다.
  - 조회 범위가 3일 이내인 경우, 5분 통계를 사용해 5분 추이를 확인할 수 있습니다.
  - 조회 범위가 3일 초과인 경우, 1시간 통계를 사용해 1시간 추이를 확인할 수 있습니다.
- 애플리케이션 선택자를 통해 전체 또는 조회를 원하는 특정 애플리케이션을 선택할 수 있습니다.
- CSV **CSV** 다운로드 아이콘을 선택해 화면에 표시된 목록 데이터를 CSV 파일로 다운로드할 수 있습니다.

2 성능 추이 차트

- 지정한 조회 시간 범위를 차트 상단에서 확인할 수 있습니다.
- 차트 영역에서 **동시접속 사용자** 및 **TPS**, **응답 시간**, **CPU**, **힙 메모리**, **액티브 트랜잭션**, **트랜잭션 Top 10**, **HTTP 호출 Top 10**, **SQL Top 10** 등의 정보를 확인할 수 있습니다.
- **스냅샷** 아이콘을 선택해 위젯의 옵션을 제외한 차트를 스냅샷 할 수 있습니다.

동시접속 사용자

브라우저 IP 기준의 실시간 사용자 수를 표시합니다.

- Raw 데이터를 사용하는 경우, 10초 동안 요청을 발생시킨 고유 브라우저 IP 수를 집계합니다.
- 5분 통계를 사용하는 경우, 최근 5분 동안 요청을 보낸 고유 브라우저 IP 수를 집계합니다.
- 1시간 통계를 사용하는 경우, 최근 1시간 동안 요청을 보낸 고유 브라우저 IP 수를 집계합니다.

TPS

TPS(Tansaction Per Second)는 초당 처리된 트랜잭션 건수를 의미합니다.

응답 시간

요청에 대한 응답 시간의 평균을 의미합니다. 평균이 높은 경우 특정 시점에 느린 트랜잭션이 많이 수행되었거나 또는 장애 발생 가능성이 높습니다.

CPU

애플리케이션 서버의 CPU 사용량을 의미합니다. 사용량 변화의 추이를 확인할 수 있습니다.

힙 메모리

모니터링 대상 프로세스의 **힙 메모리** 사용량을 의미합니다. 시간에 따른 메모리 사용량 변화를 확인할 수 있습니다.

액티브 트랜잭션

액티브 트랜잭션은 진행 중인 트랜잭션을 의미합니다. 액티브 트랜잭션에 대한 자세한 내용은 다음 문서를 참조하세요.

트랜잭션 Top 10

트랜잭션 호출 통계를 사용해 트랜잭션 호출이 많은 상위 10개 목록을 제공합니다.

HTTP 호출 Top 10

HTTPC 통계를 사용해 HTTP Call URL의 건수가 많은 상위 10개 목록을 제공합니다.

SQL Top 10

SQL 통계를 사용해 SQL 호출 건수가 많은 상위 10개 목록을 제공합니다.

큐브 분석

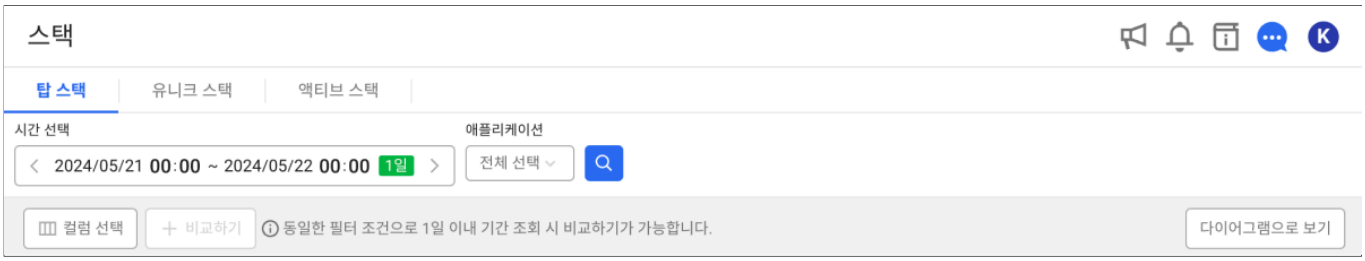
홈 화면 > 프로젝트 선택 > **분석** > **스택**

특정 시간대의 호출 스택의 빈도와 메소드, 트랜잭션에 대한 상세 정보를 확인할 수 있습니다. **탑 스택**, **유니크 스택**, **액티브 스택**을 조회할 수 있습니다. **탑 스택**은 호출 빈도가 높은 메소드의 사용량을 보여주며, **유니크 스택**은 동일한 호출 스택의 빈도를 분석합니다. **액티브 스택**은 실행 중인 트랜잭션의 스택 정보를 정기적으로 수집하여 실시간 성능 분석을 가능하게 합니다.

> ⓘ 스택 분석 기능은 **Java** 및 **Python**, **.NET** 제품 플랫폼에서 지원합니다.

주요 기능 안내

다음 주요 기능을 이용할 수 있습니다. 스택 조회를 통해 애플리케이션의 성능 문제를 신속하게 식별하고 해결할 수 있습니다. 특정 메소드 호출의 빈도, 호출 경로, 응답 시간 등을 확인하여 성능 향상을 위한 방안을 마련할 수 있습니다.



- **스택 유형 선택**

  탑 스택, **유니크 스택**, **액티브 스택** 중 원하는 스택을 선택하여 조회할 수 있습니다.

  - **탑 스택**: 호출 빈도가 가장 높은 스택 정보를 제공합니다. 자세한 내용은 다음 문서를 참조하세요.
  - **유니크 스택**: 실행된 메소드의 집합이 동일한 경우에 대한 통계 정보를 제공합니다. 자세한 내용은 다음 문서를 참조하세요.
  - **액티브 스택**: 실행 중인 트랜잭션의 스택 정보를 제공합니다. 자세한 내용은 다음 문서를 참조하세요.

- **시간 선택 / 시작일**: 조회할 시간을 설정하세요. 특정 시간대의 스택을 조회하거나 원하는 기간을 설정할 수 있습니다. 스택 유형에 따라 조회할 수 있는 기간은 제한될 수 있습니다.

> ◦ **탑 스택** 및 **유니크 스택**을 조회할 수 있는 최대 기간은 3일입니다.

- **애플리케이션**: 조회하려는 에이전트를 선택할 수 있습니다. 특정 에이전트를 선택할 수 있으며, 아무것도 선택하지 않으면 프로젝트에 포함된 전체 에이전트를 대상으로 조회합니다.
- **컬럼 선택**: 원하는 컬럼 항목을 표시하거나 숨길 수 있습니다.
- **비교하기**: 동일한 필터 조건으로 1일 이내로 조회 시 선택한 스택을 시간대비로 비교할 수 있습니다. 이를 통해 시간에 따른 호출 스택의 변화를 파악하고 성능 트렌드를 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- : 각 스택 유형의 테이블 목록에서 스택 또는 트랜잭션 항목을 확장하면 가장 오른쪽에 스냅샷 아이콘이 표시됩니다. 이 아이콘을 선택하면 해당 스택 또는 트랜잭션 정보를 이미지 파일로 저장할 수 있습니다.

기본 사용 안내

1. 화면 위에서 조회하길 원하는 스택 유형을 선택하세요.
2. **시간 선택** 또는 **시작일**에서 원하는 조회 시간을 설정하세요.
3. **애플리케이션**에서 조회할 에이전트를 선택하세요. 아무것도 선택하지 않으면 전체 에이전트를 대상으로 조회할 수 있습니다.
4. 버튼을 선택하세요.

조회 결과를 스택 테이블의 목록에서 확인할 수 있습니다.

탑 스택

탑 스택에서는 호출 빈도가 가장 높은 메소드의 정보를 제공합니다. 스택 최상단에 위치한 메소드의 호출 빈도를 통해 서비스에 가장 영향을 미치는 메소드를 신속하게 확인할 수 있습니다. 메소드의 호출 빈도를 파악하여 CPU 또는 메모리에 부하가 걸리는 원인을 분석할 수 있습니다. 이를 통해 성능 개선을 위한 방안을 도출할 수 있습니다.

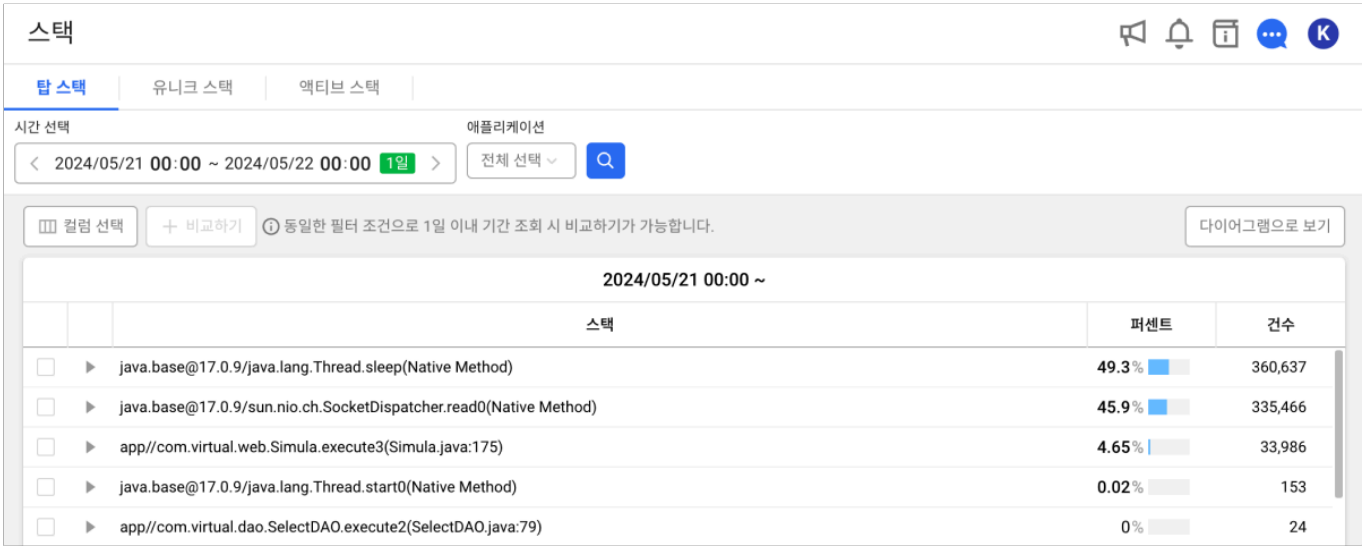
와탭은 10초(기본값) 간격으로 수집한 스레드 스택을 활용하여 메소드 레벨의 성능 지연 구간을 분석합니다.

st1

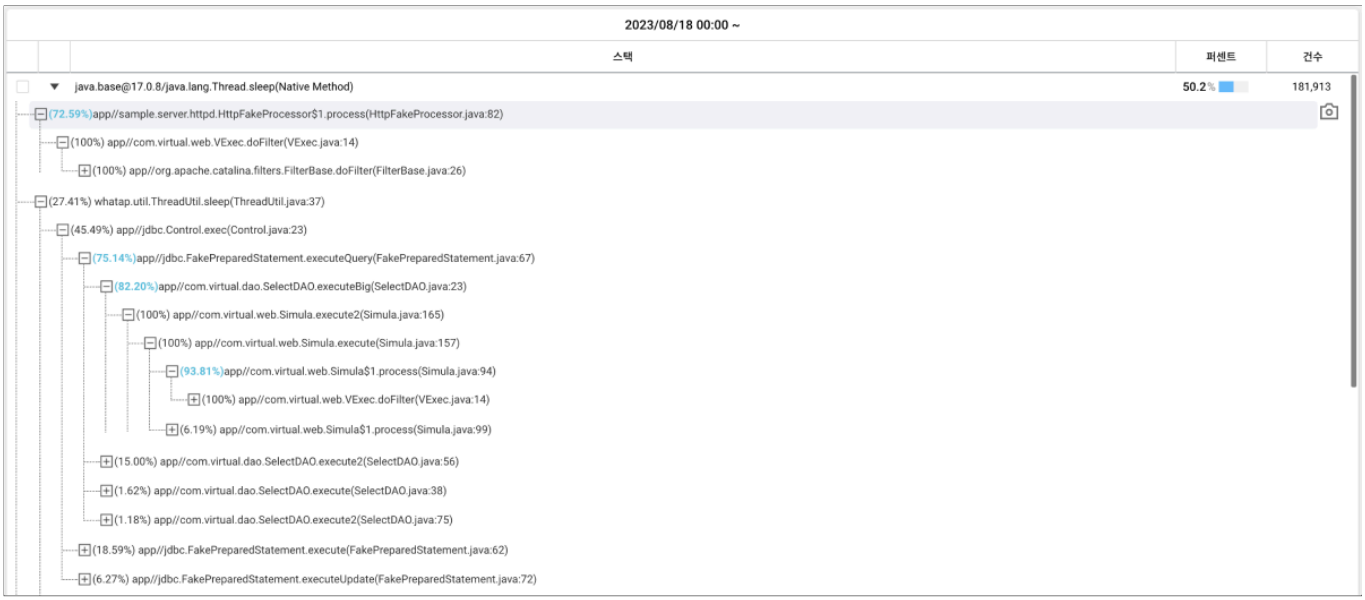
예시 스택에서 **탑 라인**은 `socketRead0` 입니다.

```
java.net.SocketInputStream.socketRead0(Native Method)
```

탑 라인은 덤프를 수행할 스레드가 해당 메소드를 수행 중이라는 것을 의미합니다. 순간적으로 잡혔을 가능성도 있지만 확률적으로 해당 모듈 처리 시간의 합의 비율만큼 스택에 나타납니다. 이 **탑 라인** 메소드의 빈도를 계산하여 메소드 레벨의 성능을 판단할 수 있습니다. 와탭은 **탑 라인 빈도 통계**를 **탑 스택**(Top Stack)이라고 합니다.



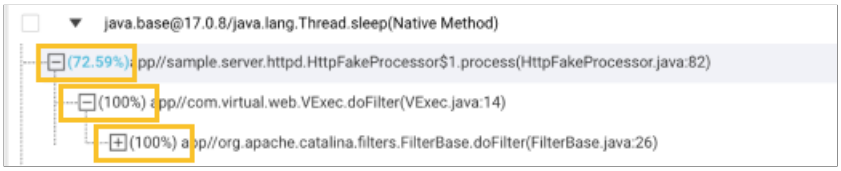
탑 스택 분석을 통해 도출된 메소드를 어떤 메소드가 호출했는지에 대한 빈도를 분석할 수 있습니다. Stack Trace 상의 각 스텝(스텝) 기준으로, 스텝과 스텝 간의 호출 비율을 백분율로 분석한 정보를 제공합니다. 최상위 스텝의 적체 빈도를 백분율로 산출하여 내림차순으로 정렬한 결과를 제시합니다.



각 스텝 항목의 가장 왼쪽에 ▸ 버튼을 선택하면 해당 스텝을 호출하는 상위 스텝의 호출 빈도 기준을 백분율로 산출하여 제공합니다.

탑 스택 통계는 충분히 많은 데이터를 가지고 판단하는 것이 좋습니다. 수집한 스택의 개수가 10개 미만의 소수인 경우 통계 의미를 갖기에 부족합니다.

탑 스택은 튜닝 시 인지하기 힘들었던 부분의 튜닝 포인트를 찾아내는 데 유용합니다. 호출 빈도가 가장 높은 스택은 현재 애플리케이션 서버에서 가장 많은 응답 지연을 발생하는 것으로 판단할 수 있습니다. 가장 왼쪽에 나타나는 비율은 애플리케이션 서버 성능에 영향을 미치는 정도입니다.



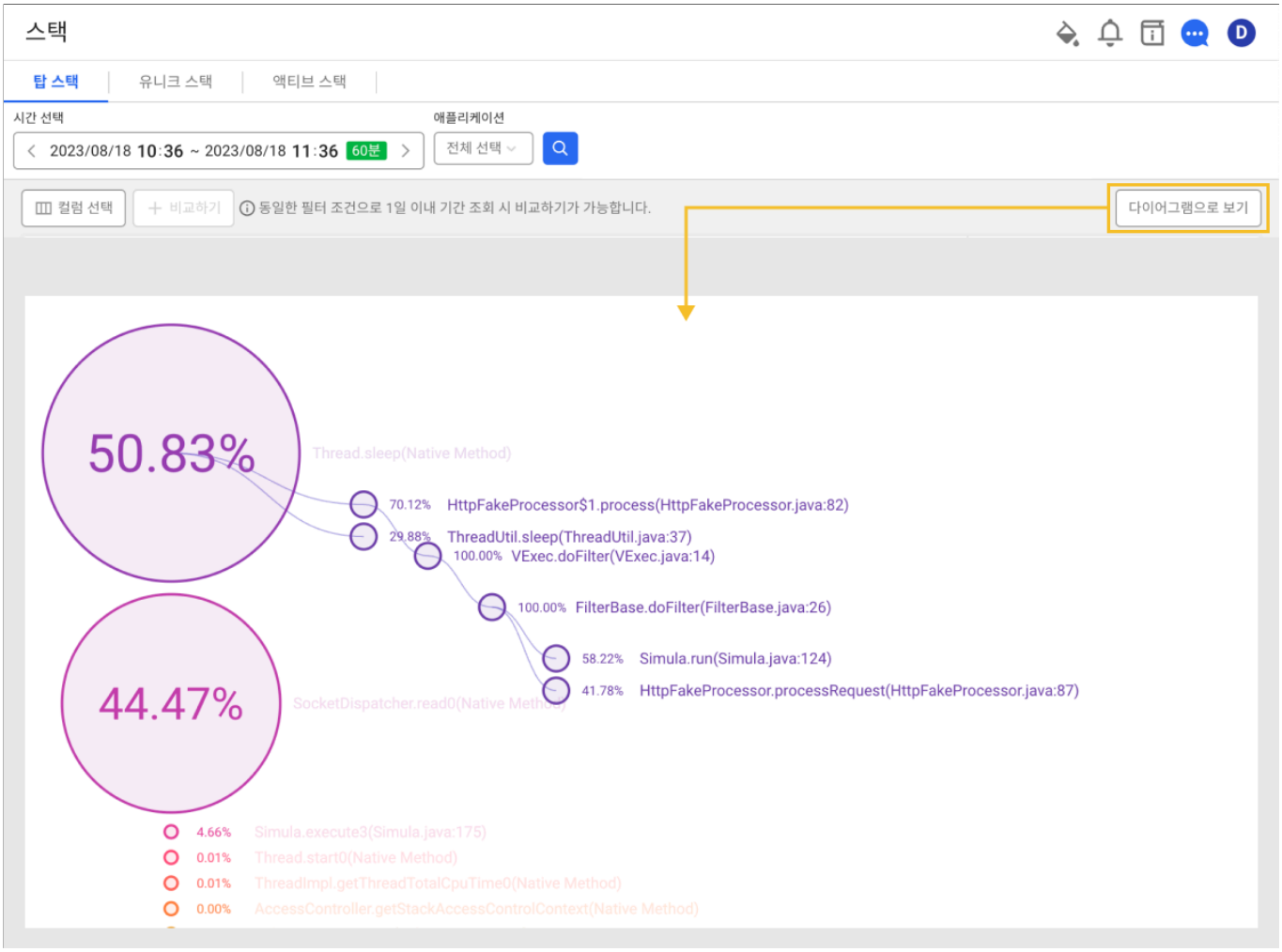
안정적인 애플리케이션 서버일지라도 호출 빈도가 높은 스택은 성능 저하를 일으킬 가능성이 있으므로 해당 클래스는 유심히 보는 것이 좋습니다.

탑 스택을 클릭하면 해당 최상위 스택에 대한 호출 빈도를 확인할 수 있습니다. **탑 스택**의 호출 관계는 1 대 1 관계이므로 **탑 스택**의 depth가 밑으로 내려갈수록 정보의 정확성이 떨어질 수 있습니다. 하위 depth에 대한 정보는 참고 용도로 사용하며 튜닝을 진행하세요.

애플리케이션 성능 개선을 위해 최상위 스텝의 적체 비율이 높은 모듈의 병목 가능성을 검토해야 합니다. 적체 비율이 높은 모듈의 경우 작은 성능 개선도 애플리케이션 전체에 상당한 개선 효과를 가져올 수 있습니다.

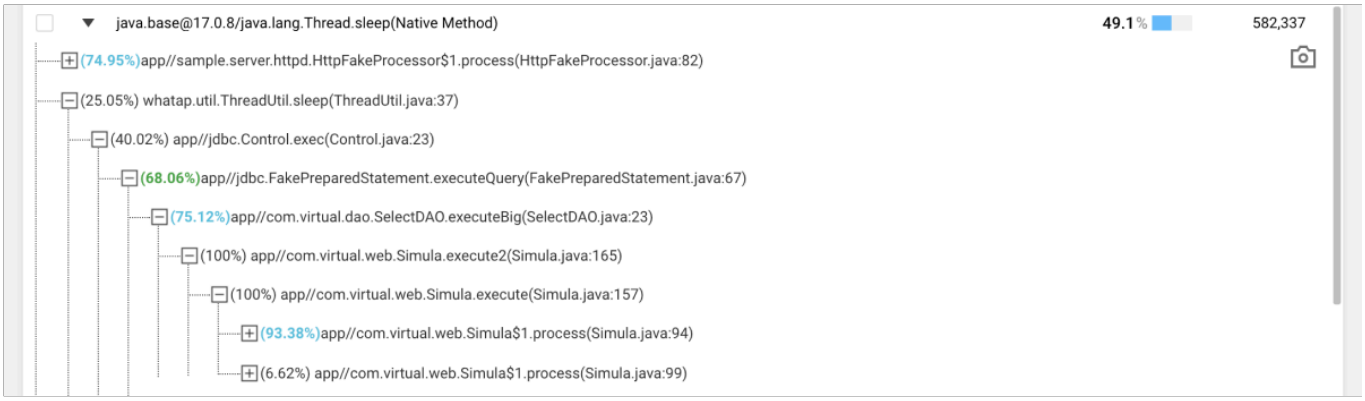
다이어그램으로 보기

복잡한 스택 호출 단계를 쉽게 이해할 수 있도록 다이어그램 차트를 제공합니다. 스택 테이블 오른쪽 위에 **다이어그램으로 보기** 버튼을 선택하세요. 이를 통해 각 단계의 호출 빈도와 실행 경로를 한눈에 파악할 수 있습니다.



- 각 노드를 선택하면 하위 depth의 스택이 표시됩니다.
- 마우스 스크롤을 통해 확대/축소하거나 드래그하여 움직일 수 있습니다.
- 스택 테이블 목록으로 돌아가려면 **테이블로 보기** 버튼을 선택하세요.

주의 사항



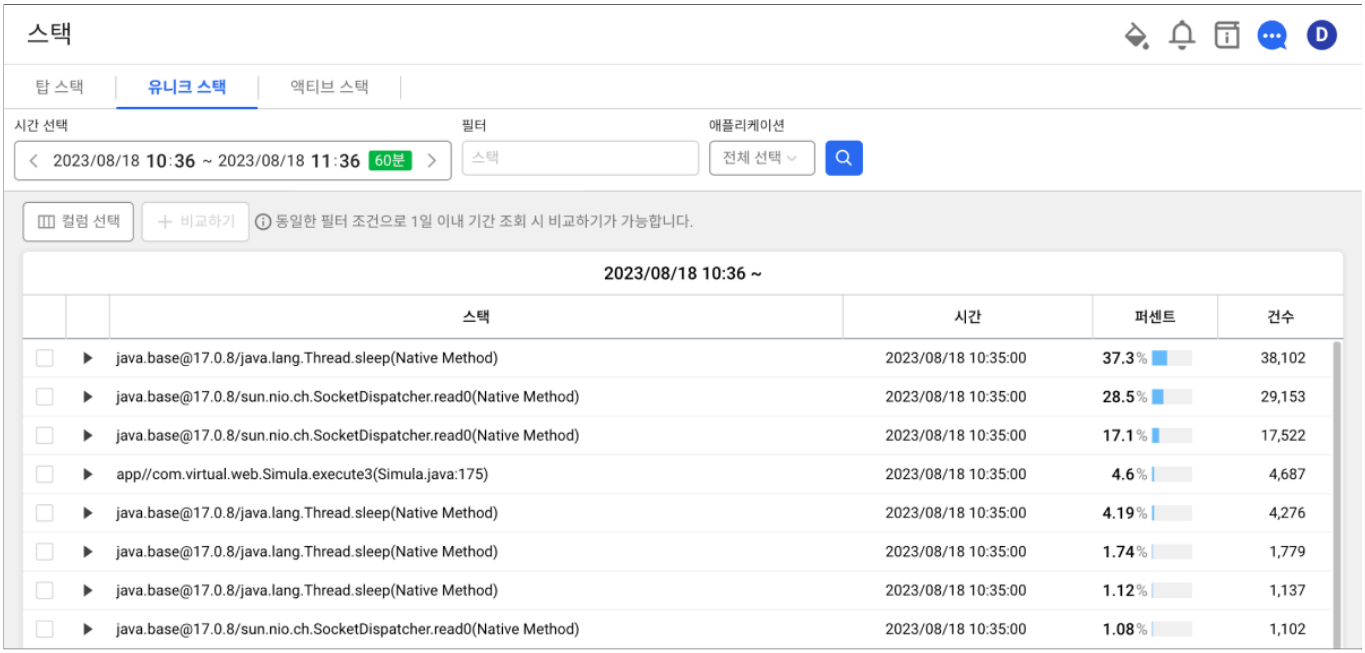

```
whatap.util.ThreadUtil.sleep
// jdbc.Control.exec의 호출 비율은 40.02%
jdbc.Control.exec
// jdbc.FakePreparedStatement.executeQuery의 호출 비율은 68.06%
```

`whatap.util.ThreadUtil.sleep` ← `jdbc.Control.exec` ← `jdbc.FakePreparedStatement.executeQuery`의 호출 비율이 40.02% * 68.06%를 의미하지는 않습니다. `jdbc.Control.exec`에서 타 모듈의 호출 가능성이 존재하기 때문입니다.

탑 스택을 통해 호출 비율을 판단할 경우, 각 스텝 간 호출 비율을 곱하여 전체 호출 관계 비율을 산출해서는 안 됩니다. **탑 스택**의 호출 비율은 Stack Trace 상에 노출된 정보의 스텝 간 호출 비율의 산출 결과이기 때문에, 스텝 간 호출 비율로 전체 호출 비율을 도출할 경우 왜곡된 결과를 도출하게 됩니다.

유니크 스택

유니크 스택은 Stack Trace 전체의 정확한 호출 정보를 기반으로 실행된 메소드의 집합이 동일한 경우에 대한 통계 정보를 제공합니다.

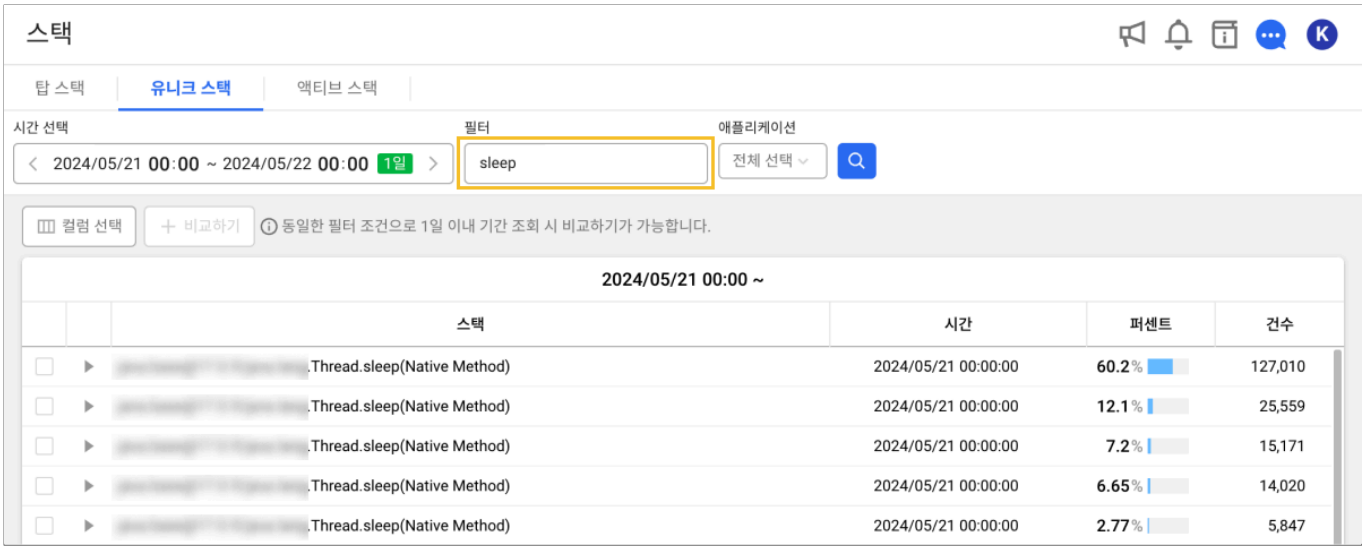


가장 많이 사용되는 스택의 정보를 파악할 수 있습니다. 동일한 스택이 여러 번 반복되더라도 중복을 제거하여 가장 높은 호출 빈도의 스택을 확인할 수 있습니다. 예를 들면 적체 비율이 높은 Stack Trace를 식별할 수 있습니다. 상세 호출 스텝 검토를 통한 호출 경로 상에 이상 모듈의 존재 여부를 파악할 수도 있습니다.



각 스택 항목에서는 메소드의 호출 경로(**스택**)와 해당 메소드가 차지하는 비율(**퍼센트**), 전체 호출 건수(**건수**)을 표시합니다. 각 스텝 항목의 가장 왼쪽에 ▶ 버튼을 선택하면 해당 호출 스택의 메소드 이름, 파일 이름, 라인 번호 등 문제 분석에 유용한 정보를 제공합니다.

필터링하기

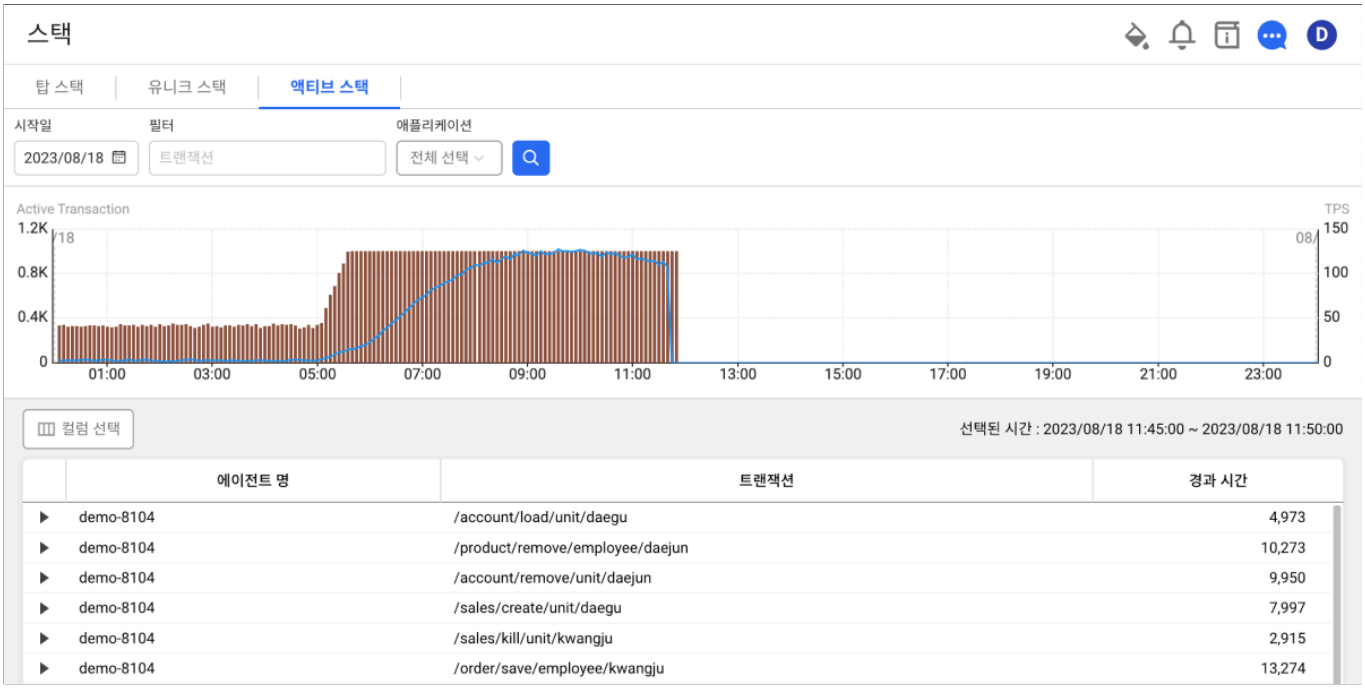


필터 항목에 메소드 명을 입력해 사용자가 원하는 조회 결과를 필터링할 수 있습니다. 문자열을 입력한 다음 버튼을 선택하세요.

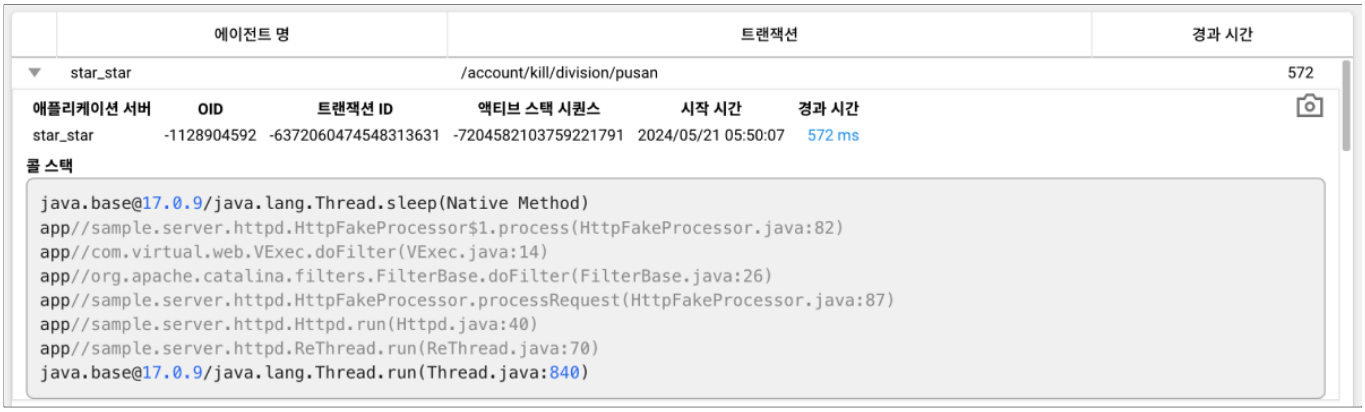
액티브 스택

진행 중인 트랜잭션을 액티브 트랜잭션이라고 합니다. **액티브 트랜잭션에서 정기적으로 덤프한 스택**을 **액티브 스택**이라 합니다.

액티브 스택에서는 수집된 **액티브 스택**을 시계열 차트로 확인할 수 있습니다. 장시간 실행되는 메소드와 단시간에 실행되지만 자주 실행되는 메소드의 비율을 통해 실행 중인 트랜잭션의 상태를 실시간으로 파악할 수 있습니다.



차트는 5분간의 단위 통계 데이터를 활용해 **Active Transaction**의 수를 막대 그래프로, **TPS**를 꺾은 선 그래프로 표시합니다. 특정 시간대의 막대 그래프를 클릭하면 해당 시간대의 **Active Transaction**의 목록을 확인할 수 있습니다.



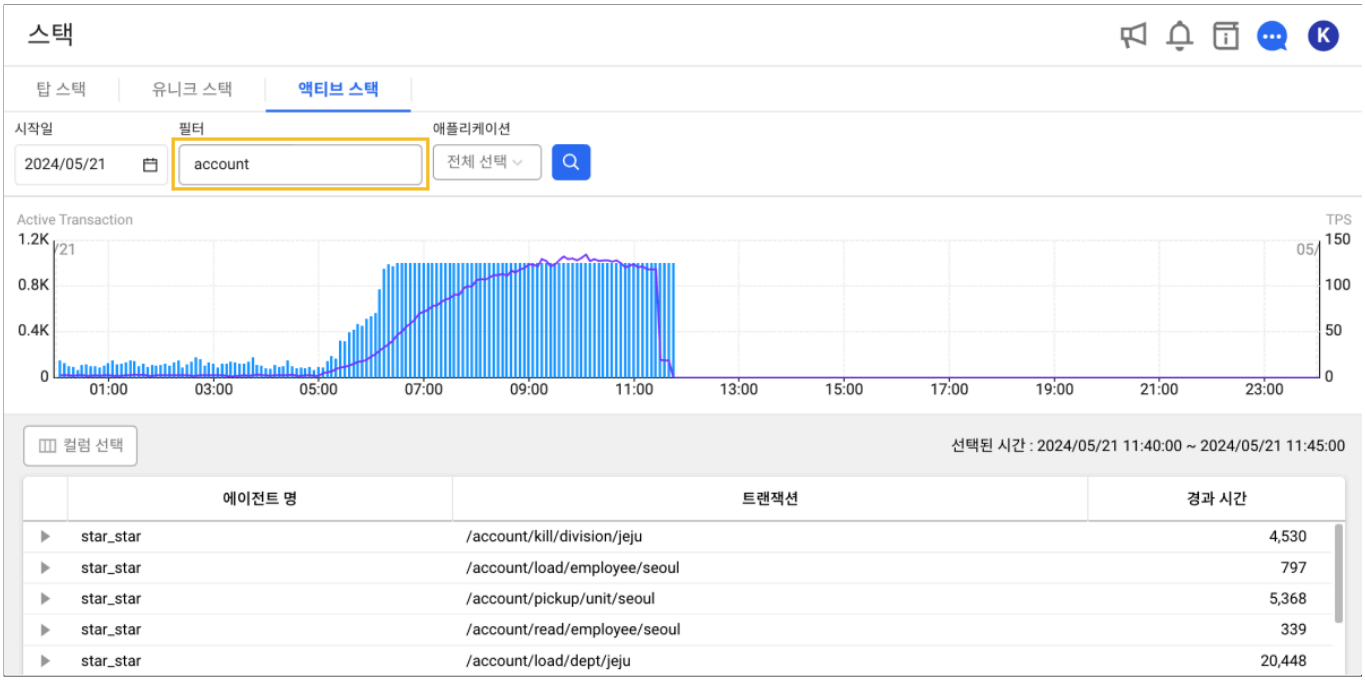
트랜잭션 목록에서 가장 왼쪽에 ▶ 버튼을 선택하면 해당 트랜잭션의 **액티브 스택** 정보를 확인할 수 있습니다.

- **액티브 트랜잭션** 수집에 대한 자세한 내용은 다음 문서를 참조하세요. 그 외 트랜잭션 수집과 관련한 에이전트 옵션은 다음 문서를 참조하세요.
- 와탭 에이전트는 **액티브 트랜잭션**에서 매 10초(옵션 가능)마다 덤프한 **액티브 스택**을 서버에 전송합니다. 수집 간격에 대한 에이전트 옵션은 다음 예시를 참조하세요.

whatap.conf

```
active_stack_second=10
```

필터링하기

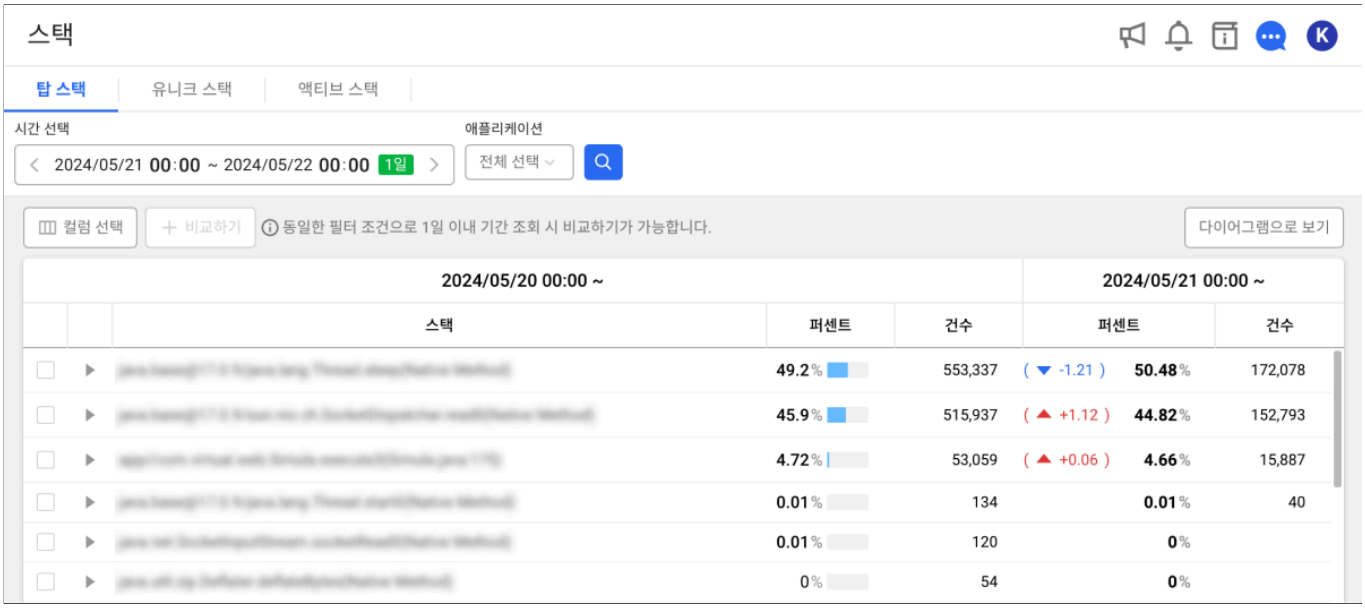


필터 항목에 트랜잭션 명을 입력해 사용자가 원하는 조회 결과를 필터링할 수 있습니다. 문자열을 입력한 다음 버튼을 선택하세요.

스택 비교하기

탑 스택과 **유니크 스택**에서는 일정 기간을 기준으로 시간에 따른 비율 변화와 수집 건수에 대한 히스토리를 제공합니다.

시간 선택에서 조회할 시간을 설정한 다음 **비교하기** 버튼을 선택하세요. 동일한 필터 조건에서 **1일 이내** 기간 조회 시 스택을 시간대비로 비교할 수 있습니다. 이를 통해 시간에 따른 호출 스택의 변화를 파악하고 성능 트렌드를 확인할 수 있습니다. 또한 장애 시점 현황 파악, 성능 개선 전/후 비교에 유용한 자료로 활용할 수 있습니다.



- **퍼센트**: 조회 기간 선택된 스택의 비율 변화를 확인할 수 있습니다.
- **건수**: 수집되는 스택의 수는 액티브 트랜잭션 수에 비례합니다. 예를 들어, 특정 구간에서 수집량이 증가했다면 서비스 지연이나 급격한 유입량 증가가 있었음을 알 수 있습니다.

> ⓘ **비교하기** 버튼이 활성화되지 않는다면 조회할 시간 범위를 초과했거나 비교할 시간의 데이터가 충분하지 않은 경우입니다. **시간 선택**에서 조회할 수 있는 시간 범위를 설정해야 **비교하기** 버튼이 활성화됩니다.

큐브

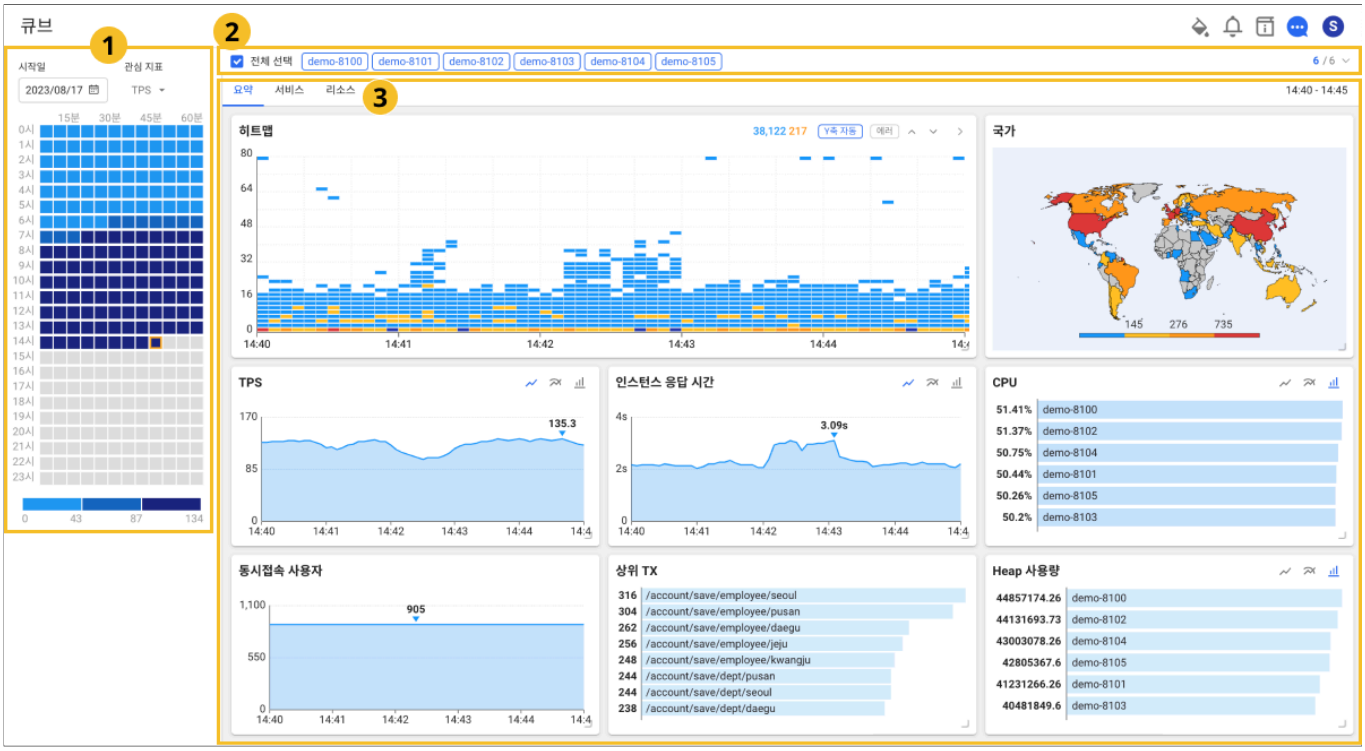
홈 화면 > 프로젝트 선택 > **분석** > **큐브**

국가별 접속, 히트맵, 트랜잭션, TOP 트랜잭션, TPS, 응답 시간, 자원 사용 정보를 제공합니다.

- 시간 경과에 따른 애플리케이션 성능 추세를 파악할 수 있습니다. 5분 간격으로 처리된 통계 데이터를 사용해 애플리케이션 성능의 변화를 빠르게 파악할 수 있습니다.
- 트래픽의 지리적 분포를 파악할 수 있습니다. 애플리케이션에 대한 트래픽 데이터를 통해 트래픽의 지리적 분포를 파악하고, 지역별 성능을 비교할 수 있습니다.
- 리소스 사용량을 파악할 수 있습니다. 애플리케이션이 사용하는 리소스 사용량을 파악하고 성능 저하의 원인을 분석할 수 있습니다.

큐브란?

와탭은 5분 단위로 만든 성능 통계를 **큐브**라고 부릅니다. 큐브 분석은 큐브에 저장된 5분 단위 성능 데이터를 활용한 분석 기능입니다.



1 큐브 셀렉트 패널

왼쪽 1 영역의 큐브 셀렉트 패널에서 특정 시간 기준으로 큐브를 선택하세요. 큐브는 5분 단위로 저장되어 있기 때문에 특정 시간을 선택해야 합니다.

- **패널 관심 지표**
  - **TPS**
  - **응답시간**
  - **TX Error 건수**
  - **액티브 TX 건수**
  - **동시접속 사용자**
- **큐브 표시색**

  큐브는 선택된 지표의 수치에 따라 다른 색으로 표현됩니다.

  - 남색: 높은 수치

- 하늘색: 낮은 수치

2 애플리케이션 셀렉트

상단 2 영역의 애플리케이션 셀렉트 옵션을 통해 특정 애플리케이션의 데이터를 선택할 수 있습니다. 오른쪽의 ⌄ 버튼을 선택하면 다음과 같이 선택 가능한 전체 애플리케이션을 확인할 수 있습니다.



- **전체 선택**: 모든 애플리케이션을 조회합니다. 기본은 **전체 선택**과 **활성 애플리케이션**이 선택되어 있습니다.
- **활성 애플리케이션**: 애플리케이션 중 활성화된 애플리케이션을 지정해 조회합니다.
- **선택된 것만 보기**: 애플리케이션 중 사용자가 지정한 해당 애플리케이션만 조회합니다.
- **검색**: 애플리케이션이 많은 경우 입력창을 통해 해당 애플리케이션을 검색할 수 있습니다.
- 숫자/숫자: 지정한 애플리케이션 개수/전체 애플리케이션 개수를 표시합니다.

3 큐브 데이터

3 영역에서 해당 큐브 데이터를 **히트맵**, **국가**, **TPS**, **인스턴스 응답 시간**, **CPU**, **동시접속 사용자**, **상위 TX**, **Heap 사용량** 등의 차트를 통해 확인할 수 있습니다.

- **히트맵**

  큐브 시간 동안 트랜잭션 분포를 보여줍니다. 트레이스 상세 분석이 가능합니다.

- **국가**

  클라이언트 아이피를 기준으로 국가를 매핑하여 어느 국가에서 트랜잭션이 들어오는지를 보여줍니다. 지도의 색은 트랜잭션의 양을 상대적으로 표시합니다.

> ⓘ 와탭은 클라이언트와 관련한 정보를 기본 저장합니다. 사용자 데이터 수집과 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

- **TPS**

  Tansaction Per Second(**TPS**)는 초당 처리한 트랜잭션 건수를 의미합니다.

- **인스턴스 응답 시간**

  트랜잭션 평균 응답시간입니다.

- **CPU**

  프로젝트 내 애플리케이션(에이전트)를 IP로 정렬하여 사용량이 많은 서버의 CPU를 바 차트로 보여줍니다.

- **동시접속 사용자**

  실시간 브라우저 사용자 수를 보여줍니다.

- **상위 TX**

  호출 건수가 많은 트랜잭션 5건을 보여줍니다. 트랜잭션 통계에서는 좀 더 상세한 데이터를 확인할 수 있습니다.

- **Heap 사용량**

  애플리케이션(인스턴스) 중 힙 메모리 사용량이 많은 순서대로 인스턴스를 정렬해 표시합니다.

> ⓘ
> - 차트 상단 오른쪽의 아이콘을 선택하면 병합 선형 차트를 제공합니다.
> - 차트 상단 오른쪽의 아이콘을 선택하면 개별 선형 차트를 제공합니다.
> - 차트 상단 오른쪽의 아이콘을 선택하면 바형 차트를 제공합니다.

> ⓘ WhaTap uses the IP2Location LITE database for IP geolocation.

히트맵 분석

트랜잭션 트레이싱이란?

트랜잭션 성능이 트랜잭션 시작과 종료 사이의 요약 지표들이나 속성들을 의미한다면 트랜잭션 트레이스는 트랜잭션이 수행되는 과정 중인 스텝들을 추적하는 것입니다. 트랜잭션이 느리거나 오류가 있다면 그 원인을 추적하기 위해서 수행 이력을 스텝별로 추적해야 합니다. 이것을 **트랜잭션 트레이싱**이라고 합니다.

트랜잭션 성능 추적을 위해 수집하는 스텝의 종류는 다음과 같습니다.

- **DB 연결 스텝** START-END

  RDB에 대한 연결에 대한 성능을 포함합니다. 스텝 정보에는 이름, 응답시간, 에러를 포함합니다.

- **SQL 스텝** START-END

  JDBC SQL에 대한 성능을 포함합니다. 스텝 정보에는 연결 정보, SQL문, 에러가 포함되어 있습니다.

- **HTTP Call 스텝** START-END

  외부 http 서비스 호출에 대한 성능을 포함합니다. 스텝 정보에는 url, host, port, 응답시간, 에러가 포함됩니다.

- **Message 스텝** ADD

  트레이스를 수집하는 과정에서 비정형적인 모든 구간에 대한 이력을 수집할 때 메시지 스텝을 사용합니다.

- **SOCKET 스텝** ADD

  Socket 오픈을 표현하는 스텝입니다.

- **METHOD 스텝** START-END

  메소드 응답시간을 추적합니다.

- **ACTIVE STACK 스텝** START-END

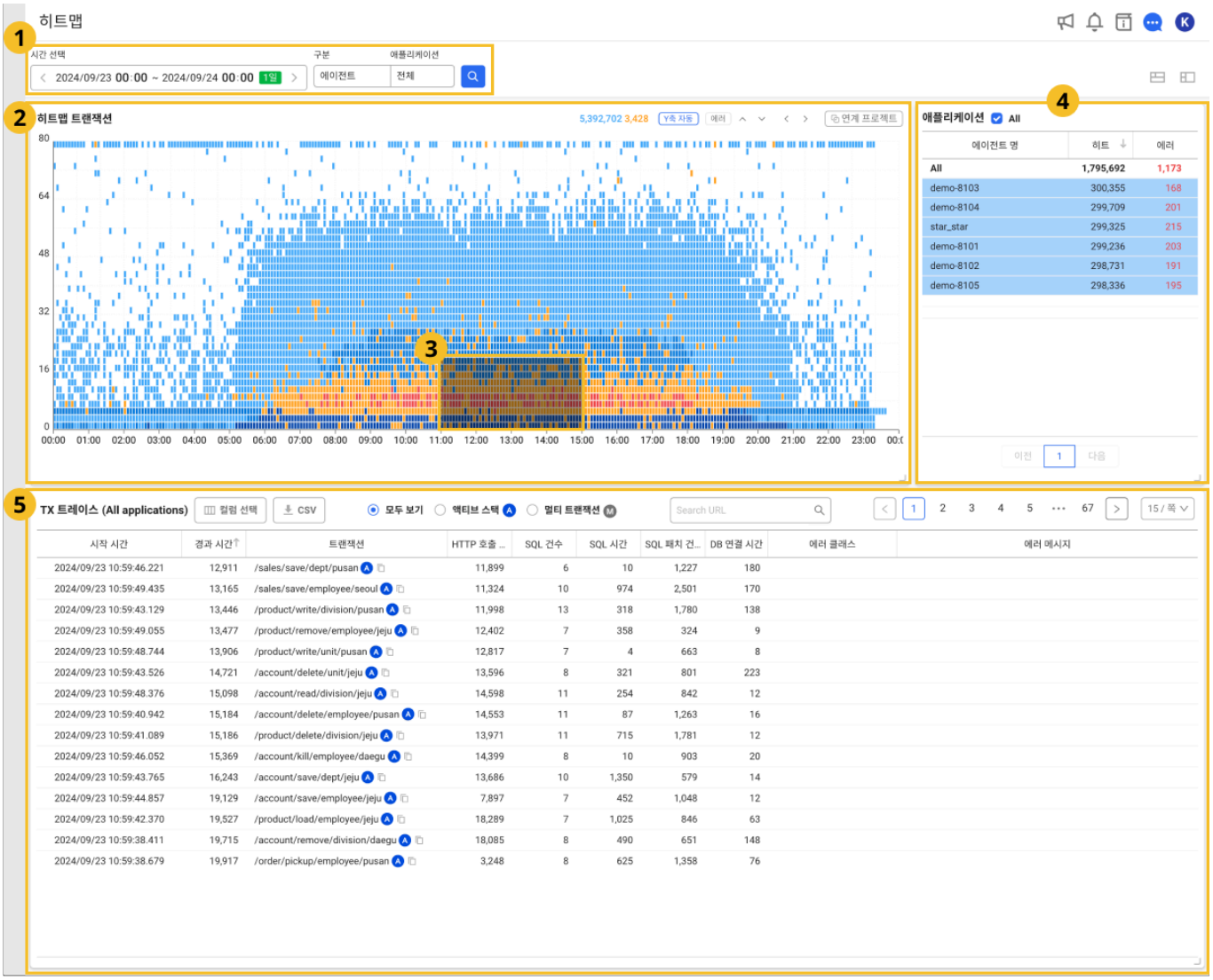
  액티브 스택에 대한 정보를 포함합니다. 별도 스레드가 생성하여 트레이스에 추가하는 방식으로 수집합니다.

> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면 연계한 프로젝트의 성능 지표를 조회할 수 있습니다. 연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

> ⓘ 와탭 모니터링 서비스에서 **스텝(Step)**은 **스팬(Span)**과 같은 뜻으로 사용됩니다.

상세 분석

홈 화면 > 프로젝트 선택 > **분석** > **히트맵**



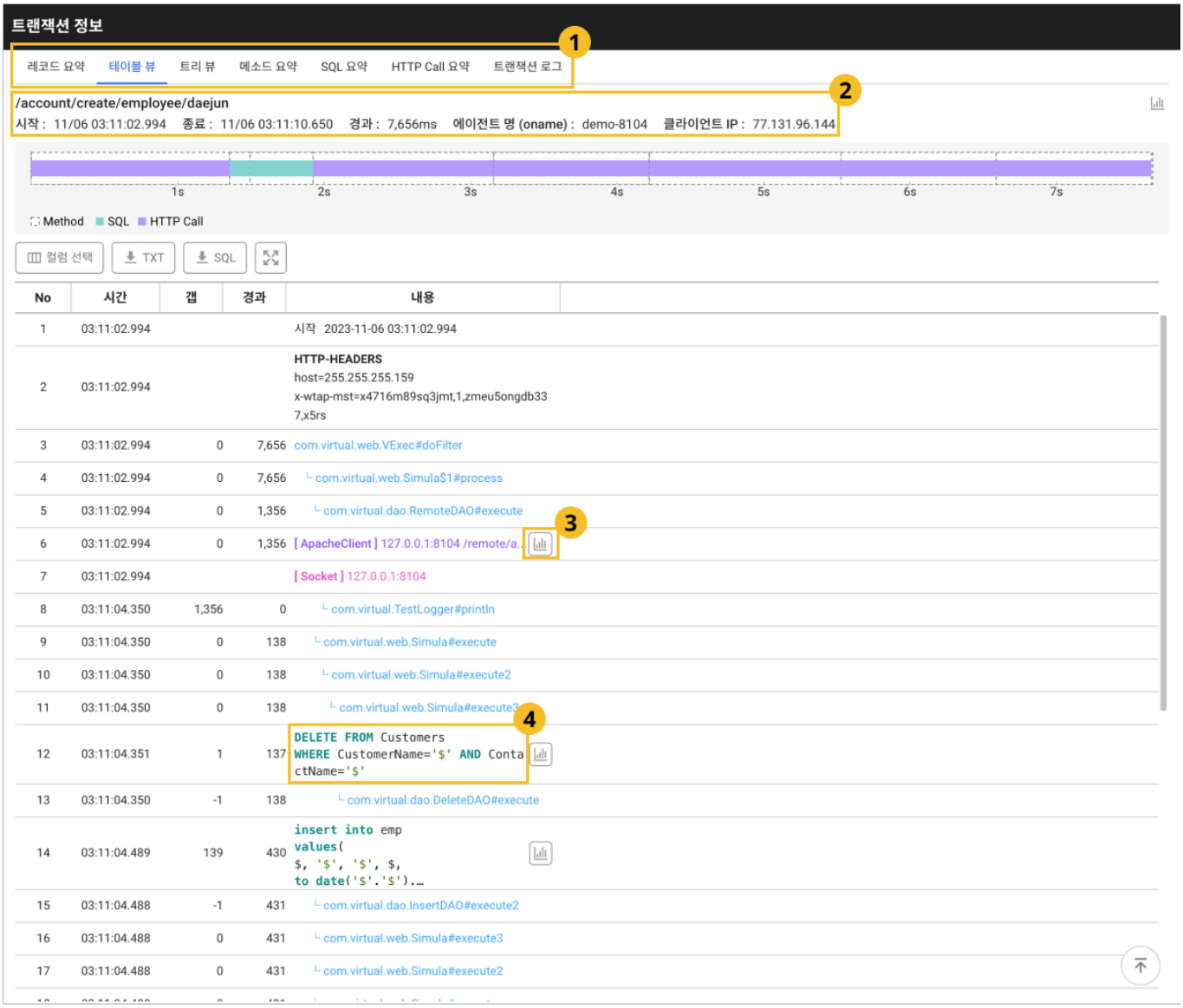
1. 1 **시간 선택**에서 히트맵을 조회할 기간을 선택하세요.
2. 1 **구분, 애플리케이션**에서 조회 대상을 선택하세요.

3. 버튼을 선택하세요.
4. 2 **히트맵 트랜잭션**에서 원하는 영역을 3과 같이 드래그하세요.
5. 5 **TX 트레이스**에 목록이 나타나면 분석하기 원하는 트랜잭션 항목을 선택하세요.

트랜잭션의 성능 분석을 위한 클라이언트 정보 등의 속성, 트랜잭션의 처리 성능, 각 구간별 상세 수행 이력 등을 확인할 수 있는 **트랜잭션 정보** 창이 나타납니다.

- **히트맵** 메뉴의 기능에 대한 자세한 내용은 다음 문서를 참조하세요.
- 애플리케이션 대시보드의 히트맵 위젯에 대한 자세한 내용은 다음 문서를 참조하세요.

트랜잭션 정보 창 안내



- 1 영역에서는 스텝 정보를 다양한 형식으로 확인할 수 있습니다. 원하는 형식의 탭을 선택하세요.
- 2 영역에서는 트랜잭션에 대한 기본 정보를 확인할 수 있습니다.
- 3 : 해당 URL 또는 각 수행 구간의 통계 데이터 창이 나타납니다. 통계 데이터 창의 그래프 차트에서 원하는 시간을 클릭하면 **통계** 또는 **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간 기준으로 통계 데이터를 조회할 수 있습니다.

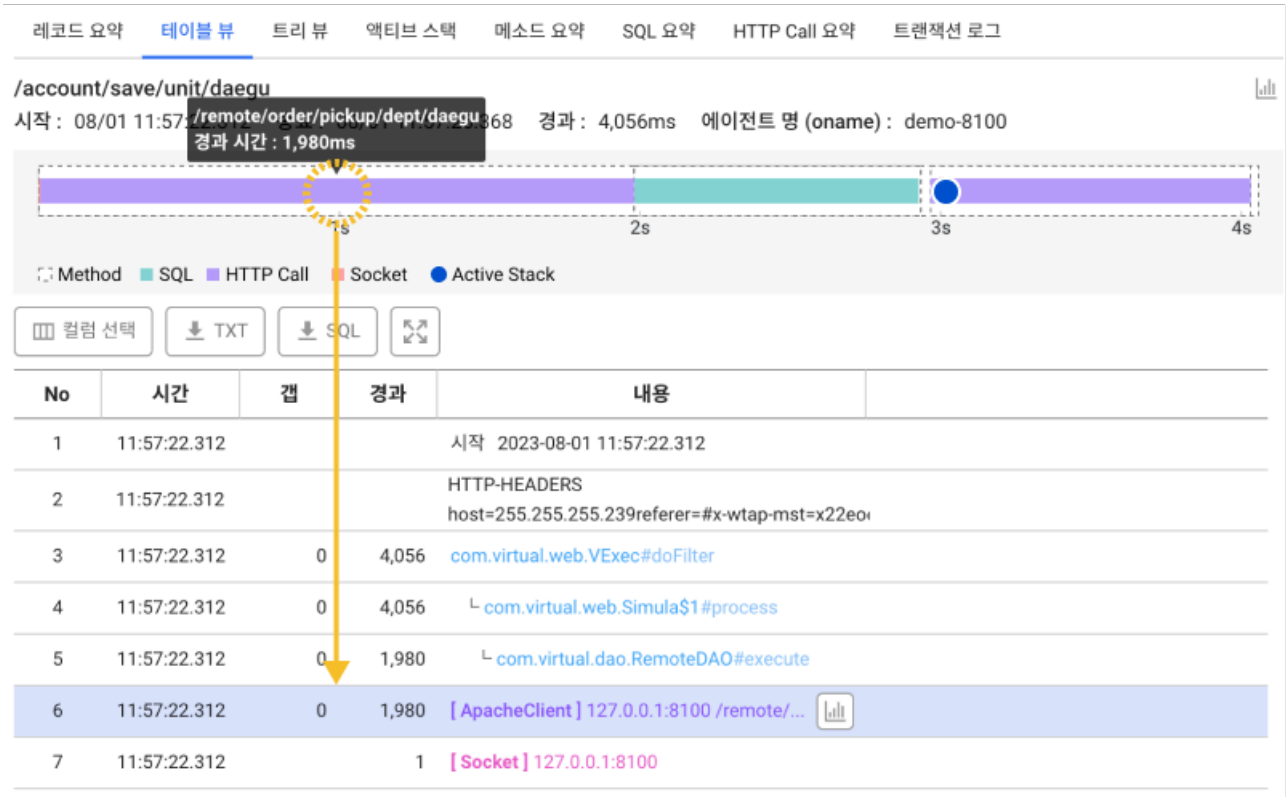
> ◦ **통계** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
> ◦ **트랜잭션 검색** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.

- 4 SQL 스텝을 선택하면 SQL 변수와 HTTP 쿼리를 조회할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

테이블 뷰

테이블 뷰 탭에서는 트랜잭션의 수행 과정을 시간의 순서대로 확인할 수 있습니다.

- 트랜잭션을 수행 구간별로 분류해 다이어그램을 통해 확인할 수 있습니다. 전체 경과 시간 중 구간별 소요된 시간, 가장 오래 소요된 구간을 빠르게 파악할 수 있습니다.
- 다이어그램의 각 구간을 선택하면 해당 스텝이 위치한 테이블 목록으로 이동합니다.

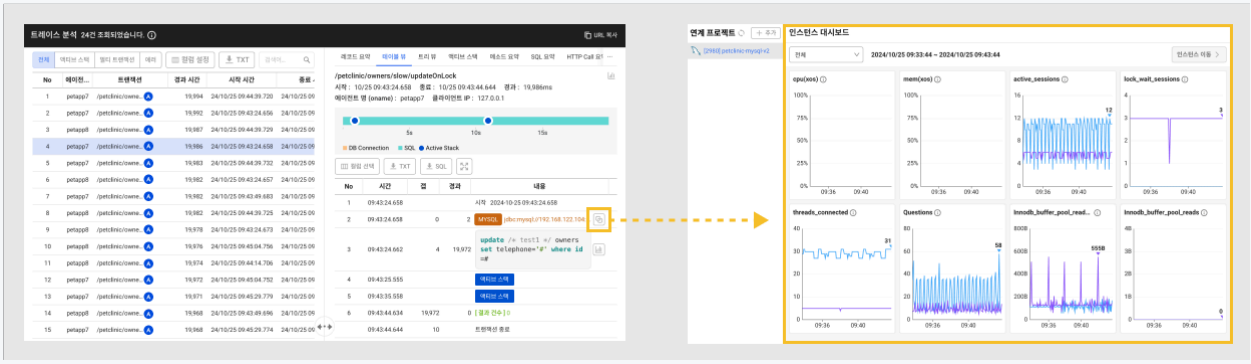


- 파란색 원(🔵)이 위치한 영역은 액티브 스택이 수집된 순간입니다. 파란색 원을 선택하면 **액티브 스택** 버튼이 위치한 테이블 목록으로 이동할 수 있습니다. **액티브 스택** 버튼을 선택하면 해당 구간 동안 수행된 스텝 정보를 확인할 수 있습니다.

> ◦ 트랜잭션 목록에서 A 아이콘이 표시된 항목에서 확인할 수 있습니다.
> ◦ 액티브 스택에 대한 자세한 내용은 다음 문서를 참조하세요.

- **컬럼 선택**: 테이블 목록에 컬럼으로 메모리 누적 정보 및 CPU 누적 정보를 추가하거나 감출 수 있습니다. 컬럼 정보는 다음과 같습니다.
  - **No**: 스텝의 발생 순서
  - **시간**: 각 스텝의 시작 시각
  - **갭**: 직전 스텝의 시작 시각부터 현재 스텝으로 넘어가기까지 대기 시간, 외부 요인으로 지연될 경우 경과 시간과 차이가 있을 수 있습니다.
  - **경과**: 각 메소드 시작부터 종료까지 총 소요 시간
  - **내용**: 해당 스텝의 세부 수행 내용
- **TXT**: 트랜잭션 기본 정보 및 구간별 수행 정보를 TXT 형식의 파일로 다운로드할 수 있습니다.
- **SQL**: 트랜잭션 기본 정보 및 SQL 수행 정보를 TXT 형식의 파일로 다운로드할 수 있습니다.
  - **치환값 포함 다운로드**: 바인드 변수값을 원래의 값으로 치환하여 SQL 통계 데이터를 다운로드합니다. 보안키(`paramkey`)가 설정되어 있다면 보안키 입력창이 나타납니다. 보안키를 입력를 입력해야 다운로드할 수 있습니다. 보안키와 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.
  - **치환값 미포함 다운로드**: 바인드 변수값을 치환하지 않은 상태로 SQL 통계 데이터를 다운로드합니다.
- : 트랜잭션 요약 정보와 다이어그램을 감추고 테이블 목록만 확인할 수 있습니다. 버튼을 선택하면 감춰진 정보를 다시 표시합니다. 테이블 목록이 긴 경우 이 기능을 이용하면 유용합니다.
- SQL 스텝을 선택하면 파라미터를 조회할 수 있는 **SQL** 창이 나타납니다. 자세한 내용은 다음 문서를 참조하세요.

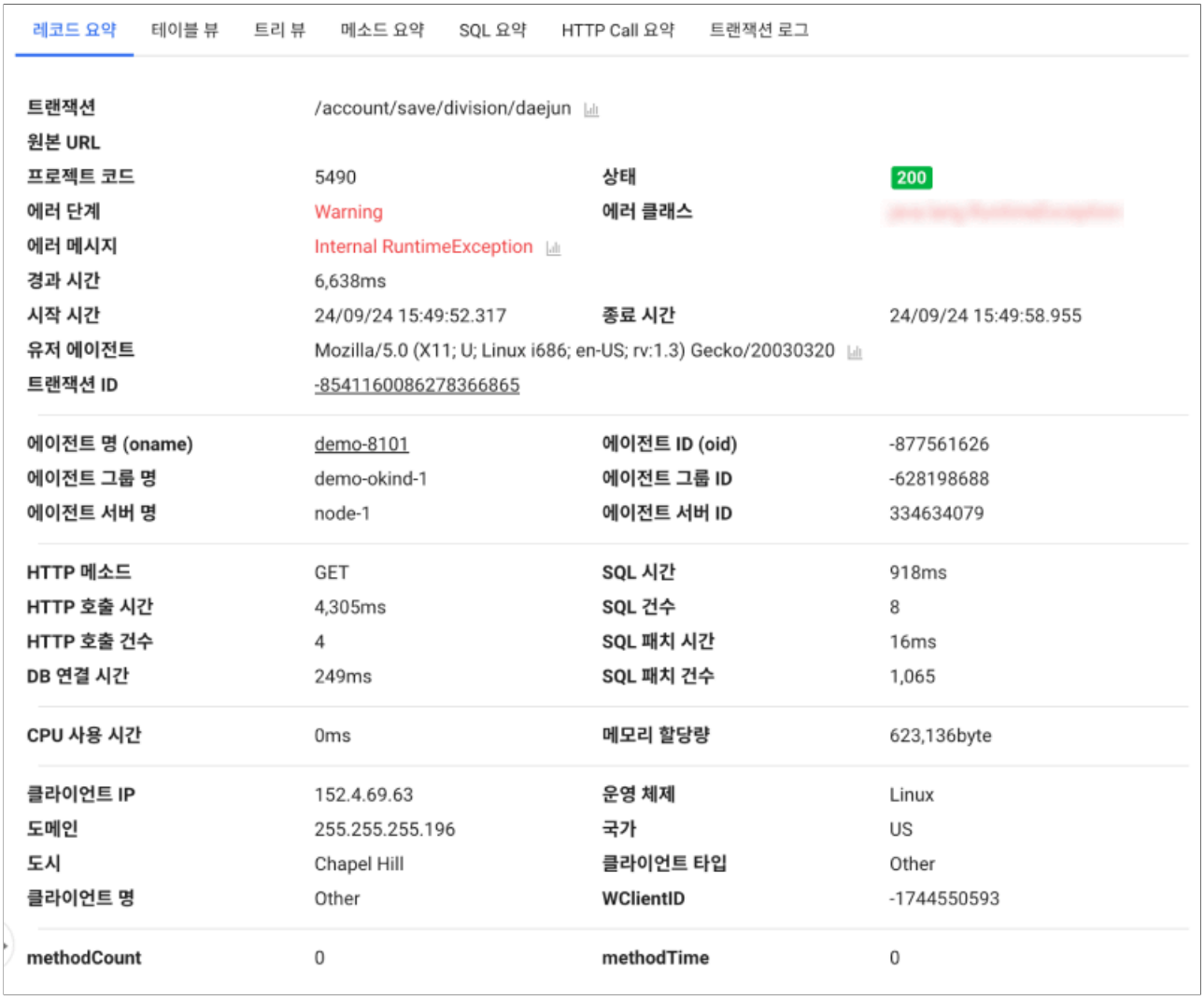
> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면, **DB Connection 스텝**에서 연계한 프로젝트의 성능 지표를 조회할 수 있습니다.



연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

레코드 요약

트랜잭션의 기본 정보를 확인하려면 **레코드 요약** 탭을 선택하세요. 트랜잭션의 기본 정보 및 해당 트랜잭션이 수행된 에이전트 관련 정보, 메모리 할당 사용량, 클라이언트 관련 정보를 확인할 수 있습니다.



분류	속성	설명
트랜잭션	트랜잭션	애플리케이션에서 실행된 API 또는 URL 호출, 해당 트랜잭션의 경로입니다. 정규 표현식을 사용해 일정한 형식으로 변환합니다.
	원본 URL	실제로 호출된 URL 경로입니다. 변수를 포함하지 않은 구체적인 요청 경로입니다. 원본 URL이 있을 경우 원본 URL 표시, 원본 URL이 없을 경우 표시하지 않습니다.
	프로젝트 코드	와탭 모니터링 서비스에 등록한 프로젝트의 식별 코드입니다.
	상태	해당 트랜잭션의 HTTP 응답 상태 코드입니다.

분류	속성	설명
	에러 단계	해당 트랜잭션에서 발생한 에러 레벨입니다. Warning 또는 Critical로 표시됩니다.
	에러 클래스	해당 트랜잭션에서 발생한 에러 관련 클래스입니다.
	에러 메시지	해당 트랜잭션에서 발생한 에러 메시지입니다.
	경과 시간	트랜잭션 수행 시간입니다.
	시작 시간	트랜잭션 시작 시각입니다.
	종료 시간	트랜잭션 종료 시각입니다.
	유저 에이전트	클라이언트 관련 정보를 추출하는데 이용하는 브라우저 정보입니다.
	Referer	클라이언트가 브라우저에서 이전에 이용한 페이지 주소 또는 유입 경로입니다.
	트랜잭션 ID	해당 트랜잭션의 고유 식별자(ID)입니다.
	멀티 트랜잭션 ID	멀티 트랜잭션의 고유 식별자(ID)입니다. 다른 애플리케이션의 트랜잭션과 병렬로 실행된 경우 표시됩니다.
에이전트	에이전트 명 (oname)	에이전트 이름입니다.
	에이전트 ID (oid)	에이전트의 고유 식별자(ID)입니다.
	에이전트 그룹 명	`okind` 옵션으로 설정된 에이전트 그룹 이름입니다.
	에이전트 그룹 ID	에이전트 그룹의 고유 식별자(ID)입니다.
	에이전트 서버 명	`ondoe` 옵션으로 설정한 에이전트 서버 이름입니다.

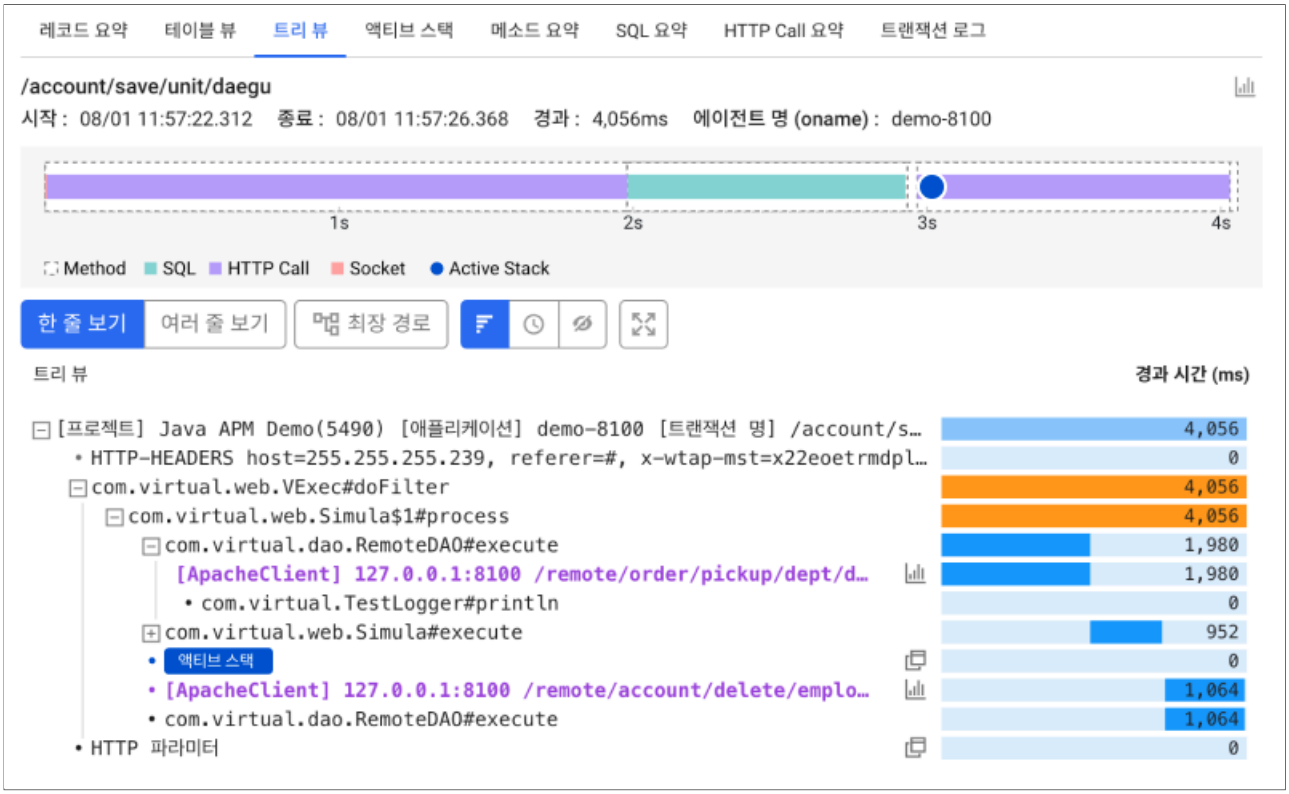
분류	속성	설명
	에이전트 서버 ID	에이전트 서버의 고유 식별자(ID)입니다.
스텝(Step)	HTTP 메소드	HTTP 메소드입니다. (GET, POST, PUT, HEAD 등)
	HTTP 호출 시간	외부 Http Call 시간입니다.
	HTTP 호출 건수	외부 HTTP Call 건수입니다.
	DB 연결 시간	데이터베이스에 연결되기까지 걸린 시간입니다.
	SQL 시간	SQL 수행 시간입니다.
	SQL 건수	SQL 수행 건수입니다.
	SQL 패치 시간	레코드를 조회하는 데 걸린 시간입니다. 중첩이 발생하거나 타 로직이 포함될 수 있습니다.
	SQL 패치 건수	SQL에서 데이터를 조회한 레코드 건수입니다.
자원(Resource)	CPU 사용 시간	트랜잭션 처리에 사용된 CPU 시간입니다.
	메모리 할당량	해당 트랜잭션에서 사용된 메모리 양입니다.
컨테이너	Pod 명	해당 트랜잭션을 처리한 Kubernetes Pod 이름입니다.
	컨테이너 ID	Pod 내에서 실행 중인 특정 컨테이너의 고유 식별자입니다.
클라이언트	클라이언트 IP	클라이언트의 IP 주소입니다.
	운영체제	브라우저가 실행되는 운영 체제 환경입니다.
	도메인	클라이언트가 접속한 IP 주소에 지정된 인터넷 주소입니다.
	국가	클라이언트의 국가 정보입니다.

분류	속성	설명
	도시	클라이언트의 국가 내 도시 정보입니다.
	클라이언트 타입	클라이언트가 이용한 브라우저 종류입니다.
	클라이언트 명	클라이언트가 이용한 기기 이름입니다.
	WClientID	클라이언트의 고유 식별자입니다.
Method	methodCount	트랜잭션에서 호출된 메소드의 수입니다.
	methodTime	메소드 실행에 소요된 시간입니다.

- 애플리케이션 종류나 설정, 스텝의 종류에 따라 수집하는 정보는 달라질 수 있습니다. **컨테이너** 항목은 애플리케이션이 컨테이너 환경에서 실행 중일 경우에만 표시됩니다.
- 에러 관련 항목은 해당 트랜잭션에서 에러가 발생할 경우 표시됩니다.
- 트랜잭션 속성 중 일부 항목은 통계 기능을 제공합니다. 버튼을 선택하면 시계열 차트를 통해 조회 시간 동안 발생한 트랜잭션 현황을 파악할 수 있는 창이 표시됩니다. 차트 내 막대 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동해 상세 조회할 수 있습니다.
- 와탭은 클라이언트와 관련한 정보를 기본 저장합니다. 사용자 데이터 수집과 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

트리 뷰

트랜잭션 수행 과정을 트리 형식으로 확인하려면 **트리 뷰** 탭을 선택하세요. 각 트랜잭션과 그에 속한 트레이스의 세부 정보, 트레이스의 시작 시간 및 소요 시간, 호출 관계를 확인할 수 있습니다. 다이어그램의 각 구간을 선택하면 해당 스텝이 위치한 트리뷰로 이동합니다.



- **한 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 한 줄로 표시해 트리 형식을 간격하게 정리할 수 있습니다.
- **여러 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 줄바꿈해 모두 표시합니다.
- **최장 경로**: 가장 긴 경로로 이동할 수 있습니다.
- **시간바 표시**: 경과 시간을 막대 형식의 차트로 표시합니다.
- **시간 표시**: 각 구간별 타임 스탬프, 갭, 경과 시간을 텍스트 형식으로 표시합니다.
  - 8초 이상: 초과 지연 상태로 빨간색으로 표현합니다.
  - 3초 이상 8초 미만: 지연 상태로 주황색으로 표현합니다.
  - 3초 미만: 정상 상태로 파란색으로 표현합니다.
- **시간 숨기기**: 시간 정보를 숨깁니다.
- : 트랜잭션 요약 정보와 다이어그램을 감추고 트리뷰만 확인할 수 있습니다. 버튼을 선택하면 감춰진 정보를 다시 표시합니다. 트리뷰 목록이 긴 경우 이 기능을 이용하면 유용합니다.
- : SQL 변수와 HTTP 쿼리를 조회할 수 있는 창이 나타납니다. 자세한 내용은 다음 문서를 참조하세요.

> 시작 및 소요 시간의 경우 트랜잭션 호출 환경에 따라 발생하는 시차를 상위 트랜잭션 내 트레이스와 매핑을 통해 보정하여 표현하기 때문에 실제 수집된 시간 데이터와 차이가 발생할 수 있습니다.

액티브 스택



액티브 스택 탭을 선택하면 해당 트랜잭션에서 수집한 액티브 스택 목록을 확인할 수 있습니다.

- 왼쪽의 **액티브 스택** 목록에서 수집 시점(No, 시간)을 선택하면 오른쪽에 해당 시점의 스택 정보를 표시합니다.
- **스택**: 액티브 스택 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

> - 트랜잭션 목록에서 A 아이콘이 표시된 항목에서 액티브 스택을 확인할 수 있습니다.
> - 액티브 스택에 대한 자세한 내용은 다음 문서를 참조하세요.

AI 액티브 스택 분석

Beta

액티브 스택 분석 버튼을 클릭하면 수집한 액티브 스택을 AI가 자동으로 분석합니다. 진행 중인 트랜잭션의 스택 덤프를 바탕으로

성능 문제를 찾고 개선 방법을 안내합니다.

분석 결과는 다음 4개 섹션으로 구성됩니다.

- **전체 요약**: 분석한 스택 수와 전체 실행 흐름을 요약합니다.
- **공통 패턴**: 반복적으로 나타나는 실행 패턴을 표시합니다. 패턴명, 발생 횟수, 설명을 포함합니다.
- **발견된 이슈**: 찾아낸 성능 문제를 심각도(**High**, **Medium**, **Low**)와 함께 표시하며, 판단의 바탕이 된 스택 정보도 제공합니다.

심각도	기준
High	데드락, 완전 블로킹, 동일 블로킹 4회 이상 반복
Medium	I/O 대기, 2~3회 락 경합, 느린 외부 호출
Low	단일 발생 대기, 가벼운 최적화 기회

- **권장 사항**: 우선순위에 따라 구체적인 개선 방법을 안내합니다.

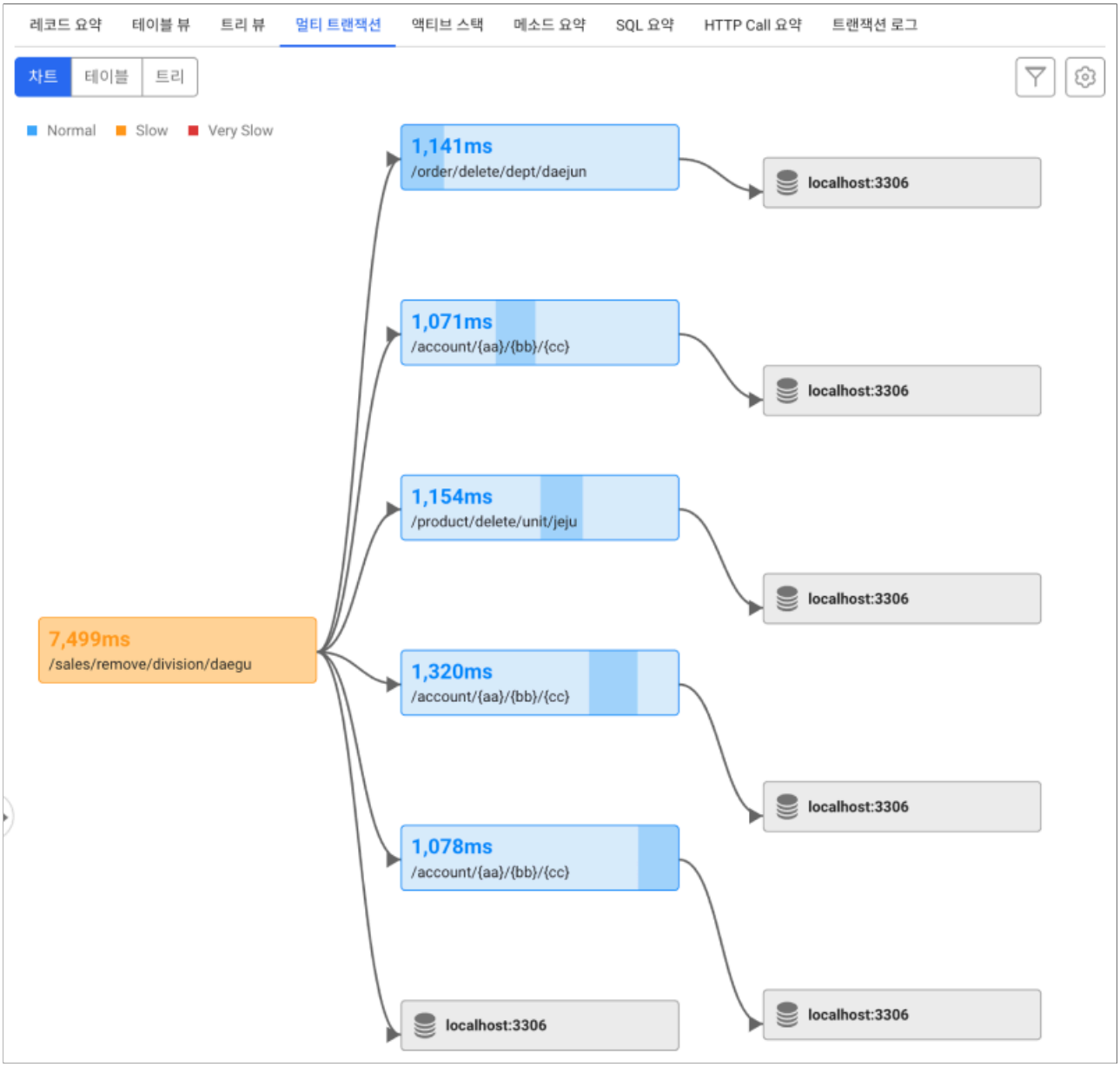
각 분석 항목에는 수집한 스택 번호(`No.N`)가 표시됩니다. 스택 번호를 클릭하면 해당 스택 원본을 확인할 수 있습니다.

> - AI가 만든 분석 결과로, 실제 상황과 다를 수 있습니다. 참고 자료로 활용하세요.
> - 프로젝트 **멤버** 권한 이상이 필요합니다.

멀티 트랜잭션

멀티 트랜잭션은 다른 에이전트나 프로젝트와의 연관된 트랜잭션을 의미합니다. **멀티 트랜잭션** 탭에서는 와탭 모니터링 서비스에 등록한 애플리케이션 간의 호출 관계를 확인할 수 있습니다.

> 멀티 트랜잭션을 추적하려면 **관리** > **에이전트 설정** 메뉴에서 `mtrace_enabled` 옵션을 `true` 로 설정하세요.
> 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.



- **차트**: 각 트랜잭션의 호출 관계를 플로우 차트 형식으로 제공합니다. 트랜잭션 노드를 선택하면 해당 트랜잭션 노드에 대한 트레이스 분석 정보를 확인할 수 있습니다. **차트** 모드에서는 마우스를 이용해 원하는 위치로 이동하거나 스크롤을 통해서 확대, 축소할 수 있습니다.
  - ⚙ **차트 뷰 설정**: 차트에 표시할 요소를 표시하거나 숨길 수 있습니다.
- **테이블**: 테이블 형식으로 멀티 트랜잭션 내에 포함된 각 트랜잭션 별 정보를 확인할 수 있습니다. ▥ 컬럼 선택 아이콘을 선택해 테이블 헤더 컬럼을 편집할 수 있습니다. 각 트랜잭션 항목을 선택하면 트레이스 분석 정보를 확인할 수 있습니다.

- **트리**: 트리 형식으로 트랜잭션 간의 호출 관계를 파악할 수 있습니다. 관련한 부가 기능은 **트리 뷰** 탭의 기능과 같습니다.
- **프로젝트 선택**: 차트에 표시될 프로젝트를 선택하거나 해제할 수 있습니다.

> 트랜잭션 목록에서 M 아이콘이 표시된 항목에서 확인할 수 있습니다.
>
> - 멀티 트랜잭션에 대한 자세한 내용은 다음 문서를 참조하세요.

메소드 요약



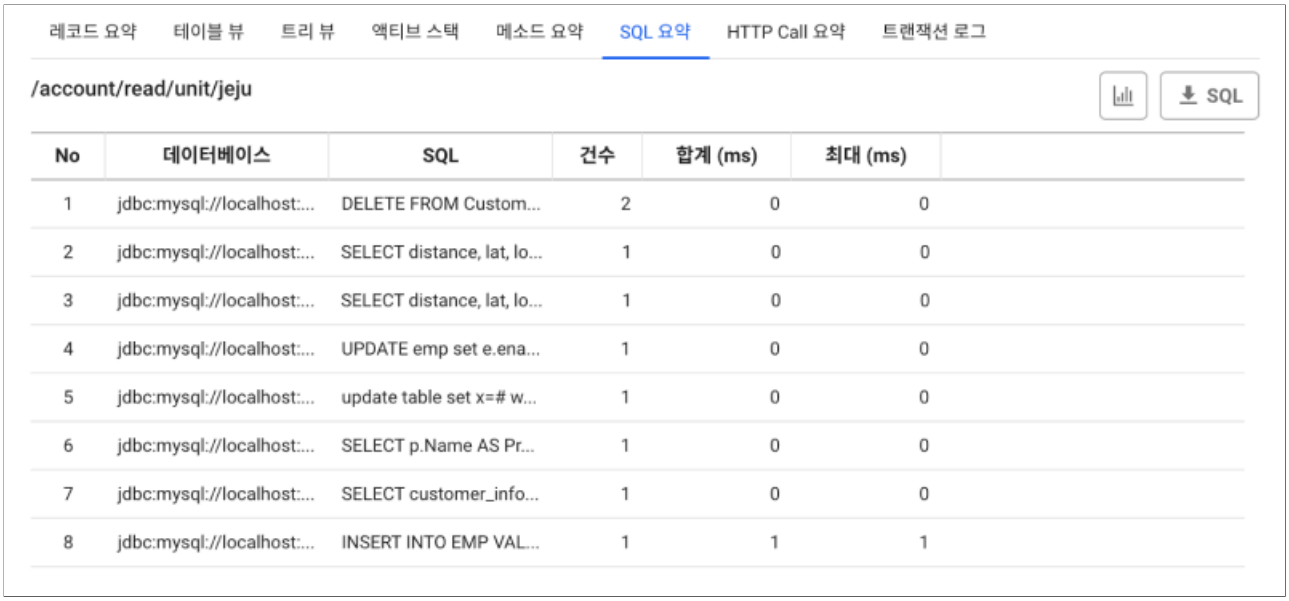
No	클래스	메소드	파라미터/리턴	건수	합계 (ms)	최대 (ms)
1	com.virtual.dao.Remot...	execute	()V	27	39,063	2,776
2	com.virtual.dao.PIITest	exec	(Ljava/sql/PreparedSta...	3	0	0
3	com.virtual.dao.InsertD...	execute2	(Ljava/lang/String;)V	2	4	3
4	com.virtual.dao.Update...	execute2	(Ljava/lang/String;)V	2	18	14
5	com.virtual.dao.Delete...	execute	()V	2	3	2
6	com.virtual.dao.Select...	execute2	(Ljava/lang/String;)V	1	5	5
7	com.virtual.dao.Update...	execute	()V	1	4	4
8	com.virtual.dao.Select...	execute2	()V	1	7	7
9	com.virtual.dao.Select...	execute	()V	1	9	9
10	com.virtual.dao.InsertD...	execute2	()V	1	3	3

메소드 정보만을 확인하려면 **메소드 요약** 탭을 선택하세요. 에이전트에 추적이 설정된 메소드 이름과 소요 시간을 표시합니다. 불필요한 로직이 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 메소드 로직 개선을 위한 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **메소드**: 메소드 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

> ⓘ 메소드(method)와 관련한 에이전트 설정 옵션에 대한 자세한 내용은 다음 문서를 참조하세요. 이용 중인 제품에 따라 메소드 추적 지원 여부는 다를 수 있습니다.

SQL 요약



No	데이터베이스	SQL	건수	합계 (ms)	최대 (ms)
1	jdbc:mysql://localhost:...	DELETE FROM Custom...	2	0	0
2	jdbc:mysql://localhost:...	SELECT distance, lat, lo...	1	0	0
3	jdbc:mysql://localhost:...	SELECT distance, lat, lo...	1	0	0
4	jdbc:mysql://localhost:...	UPDATE emp set e.ena...	1	0	0
5	jdbc:mysql://localhost:...	update table set x=# w...	1	0	0
6	jdbc:mysql://localhost:...	SELECT p.Name AS Pr...	1	0	0
7	jdbc:mysql://localhost:...	SELECT customer_info...	1	0	0
8	jdbc:mysql://localhost:...	INSERT INTO EMP VAL...	1	1	1

SQL 문에 대한 정보를 확인하려면 **SQL 요약** 탭을 선택하세요. 불필요하게 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 쿼리 성능 개선을 위한 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **SQL**: SQL 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

HTTP Call 요약



No	원격 서버	URL	건수	합계 (ms)	최대 (ms)
1	127.0.0.1:8105	/remote/account/save...	2	4,116	2,776
2	127.0.0.1:8105	/remote/order/save/di...	1	238	238
3	127.0.0.1:8105	/remote/sales/pickup/...	1	922	922
4	127.0.0.1:8105	/remote/sales/remove...	1	1,980	1,980
5	127.0.0.1:8105	/remote/order/save/de...	1	2,639	2,639
6	127.0.0.1:8105	/remote/edu/kill/unit/p...	1	1,519	1,519
7	127.0.0.1:8105	/remote/order/read/de...	1	1,467	1,467
8	127.0.0.1:8105	/remote/order/delete/...	1	485	485
9	127.0.0.1:8105	/remote/sales/kill/emp...	1	2,210	2,210

HTTP 호출의 호출 건수, 합계 시간, 평균 시간 등을 확인하려면 **HTTP Call 요약** 탭을 선택하세요. 불필요한 외부 호출이 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 트랜잭션 지연 요인이 외부인지 내부인지를 파악하는 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **HTTP**: HTTP Call 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

트랜잭션 로그

트랜잭션과 관련한 로그 정보를 확인하려면 **트랜잭션 로그** 탭을 선택하세요.



- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **로그 검색기**: 로그 검색으로 이동해 원하는 로그 데이터를 쉽게 조회할 수 있습니다.
- **키워드 검색**: 키워드 검색란에 검색하려는 텍스트를 입력한 다음 엔터 키를 입력하거나 버튼을 선택하세요. 입력한 텍스트와 일치하는 키워드를 하이라이트 표시합니다.
- **키워드 하이라이트**: 키워드와 색상을 설정하면 자동으로 로그 목록에서 키워드와 일치하는 텍스트를 하이라이트 표시합니다. 자세한 내용은 다음 문서를 참조하세요.
- **컬럼 설정**: 로그 목록 테이블에 표시되는 컬럼을 추가하거나 순서를 변경할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- **로그 표시 상세 설정**: 로그 목록에 표시되는 내용을 설정할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

- **트랜잭션 로그** 탭을 활성화하려면 로그와 관련한 에이전트 설정 옵션을 적용해야 합니다. 로그 설정에 대한 자세한 내용은 다음 문서를 참조하세요.
- **트랜잭션 로그** 탭은 **로그 조회** 권한을 가진 멤버만 진입할 수 있습니다. 멤버 권한에 대한 자세한 내용은 다음 문서를 참조하세요.

키워드 하이라이트 설정하기

1. 하이라이트 표시할 키워드를 추가하려면 버튼을 선택하세요.



2. 원하는 키워드를 입력하세요.
3. 색상 아이콘을 선택한 다음 원하는 색상을 선택하세요.
4. 엔터를 입력하세요. 추가한 키워드가 목록에 생성됩니다.

키워드 하이라이트 삭제하기

하이라이트 목록에서 이용하지 않는 항목을 삭제하려면 — 버튼을 선택하세요.

테이블 컬럼 표시하거나 숨기기

1. 테이블 오른쪽 위에 ▥ 버튼을 선택하세요.
2. **컬럼 설정** 창이 나타나면 왼쪽 목록에서 원하는 항목을 선택하거나 선택을 해제하세요.

   오른쪽 목록에서 × 버튼을 선택해 컬럼을 숨길 수도 있습니다.
3. 모든 설정을 완료한 다음 **확인** 버튼을 선택하세요. 설정한 내용이 테이블에 적용됩니다.

테이블 컬럼 순서 바꾸기

1. 테이블 오른쪽 위에 ▥ 버튼을 선택하세요.
2. **컬럼 설정** 창이 나타나면 오른쪽 목록에서 원하는 항목을 드래그해서 순서를 변경하세요.
3. 모든 설정을 완료한 다음 **확인** 버튼을 선택하세요. 설정한 내용이 테이블에 적용됩니다.

테이블 콘텐츠 표시 설정

1. 테이블 오른쪽 위에 ⚙ 버튼을 선택하세요.
2. **로그 표시 상세 설정** 창이 나타나면 원하는 옵션을 설정하세요.
   - **content** 또는 **태그** 옵션을 선택해 로그의 콘텐츠 표시 여부를 선택하세요.
   - **태그 관리**: 태그는 추가한 순서대로 로그에 표시됩니다. 태그의 순서는 드래그로 변경할 수 있습니다. 비활성화된 태그는 로그에서 보이지 않습니다.
3. 모든 설정을 완료한 다음 ✕ 버튼을 클릭하세요. 설정한 내용이 테이블에 적용됩니다.

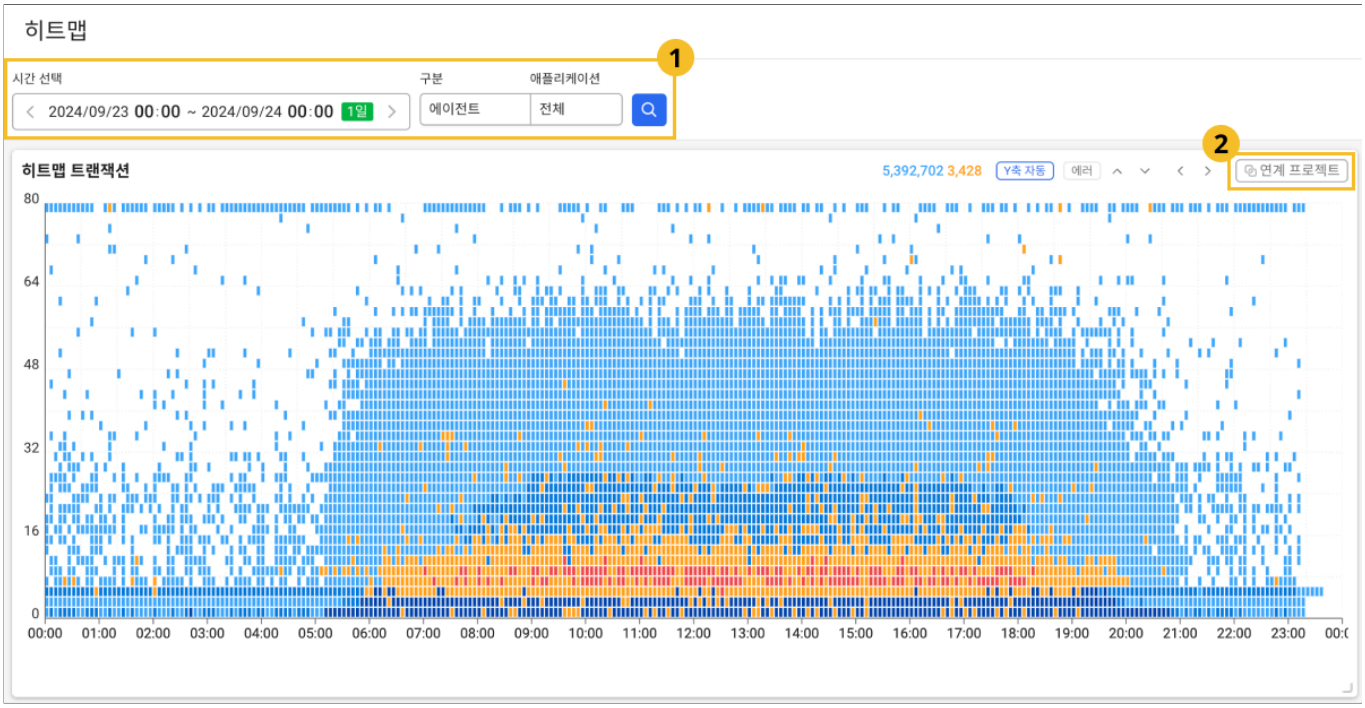
> ⓘ **로그 표시 상세 설정**에 대한 자세한 내용은 다음 문서를 참조하세요.

부가 기능

연계 프로젝트 지표 확인하기

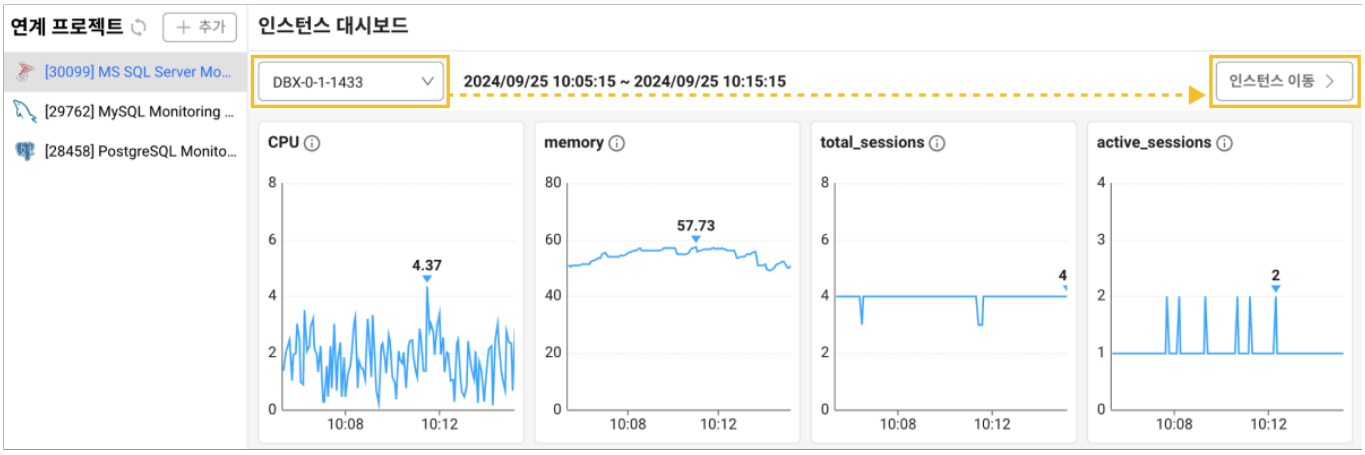
데이터베이스 모니터링(DPM) 프로젝트와 애플리케이션 모니터링(APM) 프로젝트를 생성한 경우, 두 프로젝트를 연계하여 DPM에서 수집한 모니터링 데이터를 APM 프로젝트에서 확인할 수 있습니다. **히트맵 트랜잭션** 섹션의 오른쪽 상단의 **연계 프로젝트** 기능을 통해 트랜잭션을 조회하는 동안 연계된 프로젝트의 성능 지표를 함께 확인할 수 있습니다.

연계 프로젝트 기능은 APM 프로젝트에서 트랜잭션을 분석할 때, 느린 SQL을 발견했을 경우 비슷한 여러 DB를 사용하는 환경에서 정확히 어느 DB에서 문제가 발생했는지 확인하기 어려운 상황에 유용합니다.



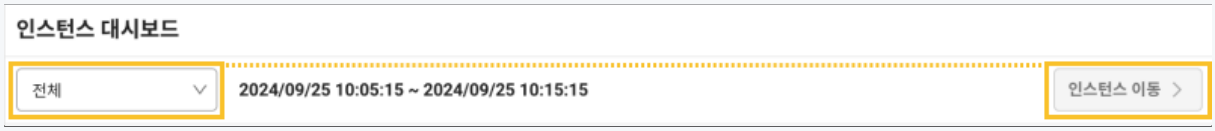
1. 트랜잭션을 조회할 시간과 **애플리케이션**을 설정한 후 버튼을 선택하세요.
2. **히트맵 트랜잭션** 섹션에서 오른쪽 상단의 **연계 프로젝트** 버튼을 선택하세요.
3. 연계 프로젝트의 지표를 확인할 수 있는 새창이 나타납니다.

왼쪽 목록에서 연계 프로젝트를 선택하고 **인스턴스 대시보드** 섹션의 지표를 확인하세요.



선택한 프로젝트의 대시보드 메뉴로 이동하려면 인스턴스를 선택한 후, 오른쪽 상단의 **인스턴스 이동** 버튼을 클릭하세요. **대시보드 > 인스턴스 모니터링** 메뉴로 이동합니다.

- **히트맵 트랜잭션** 섹션의 **연계 프로젝트** 버튼은 연계된 프로젝트가 있을 때만 표시됩니다. 연계 프로젝트 추가에 대한 자세한 내용은 다음 문서를 참조하세요.
- 왼쪽 **연계 프로젝트** 목록 상단의 **추가** 버튼을 선택해 프로젝트를 추가할 수 있습니다.
- **인스턴스 이동** 버튼은 **전체**를 선택한 상태에서는 비활성화됩니다.



- 데이터베이스 제품 종류에 따라 **분석** > **카운트 추이** 메뉴로 이동할 수 있습니다.

HTTP 파라미터 조회

테이블 뷰 탭에서 해당 트랜잭션의 HTTP 파라미터를 조회할 수 있습니다.

1. 페이지 아래로 스크롤해 트랜잭션 수행의 가장 마지막 단계로 이동하세요.
2. **HTTP 파라미터** 항목을 선택하세요.
3. **HTTP-PARAMETERS** 창이 나타나면 **비밀번호** 버튼을 선택하세요.
4. 설정한 Param Key를 입력하세요. 가려진 매개 변수를 확인할 수 있습니다.

- HTTP 파라미터와 관련한 에이전트 설정은 다음 문서를 참조하세요.
- **비밀번호**: 복호화된 파라미터 값을 확인할 수 있습니다. 비밀번호는 `WHATAP_HOME` /security.conf (paramkey.txt) 파일 내 6자리 문자열입니다. 다른 문자열로 변경 가능합니다. SQL 변수 조회, HTTP 쿼리 조회, Thread 중지에 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

SQL 파라미터 조회

테이블 뷰 탭에서 SQL 스텝을 선택하거나 **트리 뷰** 탭에서 버튼을 선택하세요. 파라미터를 조회할 수 있는 **SQL** 창이 나타납니다.



- **비밀번호**: 복호화된 파라미터 값을 확인할 수 있습니다. 비밀번호는 `WHATAP_HOME` /security.conf 파일 내 6자리 문자열(`paramkey`)입니다. 다른 문자열로 변경 가능합니다.

  > ⓘ **비밀번호**는 SQL 변수 조회, HTTP 쿼리 조회, Thread 중지에 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

- **SQL 포멧팅**: SQL 문장을 들여쓰기 및 포맷팅하여 가독성을 높일 수 있습니다.
- : SQL 문장을 클립보드에 복사할 수 있습니다.
- **통계**: **통계** > **SQL** 메뉴로 이동합니다.

> ✓ SQL 변수와 HTTP 쿼리를 조회하려면 다음 옵션을 에이전트 설정에 추가하세요.
>
> - SQL 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
> - HTTP 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
>
> whatap.conf
>
> ```
> # SQL 파라미터 조회 옵션: 옵션이 적용되면 SQL 파라미터를 암호화하여 수집합니다.
> profile_sql_param_enabled=true
>
> # HTTP 파라미터 조회 옵션: 옵션이 적용되면 HTTP 쿼리 파라미터를 암호화하여 수집합니다.
> profile_http_parameter_enabled=true
> ```

트레이스 목록 컬럼 설정하기

TX 트레이스 섹션의 컬럼 설정을 통해 조회된 결과에 대한 세부 항목을 추가 확인할 수 있습니다.

1. **TX 트레이스** 영역의 **컬럼 선택** 버튼을 클릭하세요.
2. 원하는 컬럼 항목을 선택하거나 해제하세요. 설정한 상태가 테이블에 바로 반영됩니다.
   - **전체 선택**: 모든 컬럼 항목을 선택
   - **전체 해제**: 모든 컬럼 항목을 해제
   - **초기화(기본값)**: 컬럼 항목을 기본값으로 설정
3. 설정을 모두 완료했다면 오른쪽 상단의 × 아이콘을 클릭하세요.

트레이스 목록 필터링하기

TX 트레이스 섹션에서 조회된 결과를 다음 옵션을 통해 필터링할 수 있습니다.



- **모두 보기**: 조회된 결과에서 모든 트랜잭션을 확인할 수 있습니다.
- **액티브 스택**: 조회된 결과에서 액티브 스택을 포함한 트랜잭션만 확인할 수 있습니다.
- **멀티 트랜잭션**: 조회된 결과에서 멀티 트랜잭션을 포함한 트랜잭션만 확인할 수 있습니다.
- **Search**: 입력한 문자열과 일치하는 항목을 포함한 트랜잭션만 표시합니다. 원하는 문자열을 입력한 후 키보드의 `Enter` 키를 누르세요.
  - **트랜잭션**
  - **클라이언트 IP**
  - **도메인**
  - **에러 클래스**
  - **에러 메시지**

조회 목록 다운로드하기

TX 트레이스 섹션에서 조회된 결과를 CSV 형식의 파일로 다운로드할 수 있습니다. 다운로드한 CSV 파일 이름은 profile_

`{project_code}_YYYYMMDD.csv` 형식입니다.

1. 트랜잭션을 조회한 후에 CSV 버튼을 선택하세요.
2. **최대 CSV 라인 수** 옵션에 원하는 값을 입력하세요.
3. **다운로드** 버튼을 선택하세요.

멀티 트랜잭션 분석

멀티 트랜잭션은 다른 에이전트나 프로젝트와 연관된 트랜잭션을 의미합니다. 와탭 프로젝트에 등록된 애플리케이션 서비스 간의 호출을 추적하는 것이 **멀티 트랜잭션 추적**입니다.

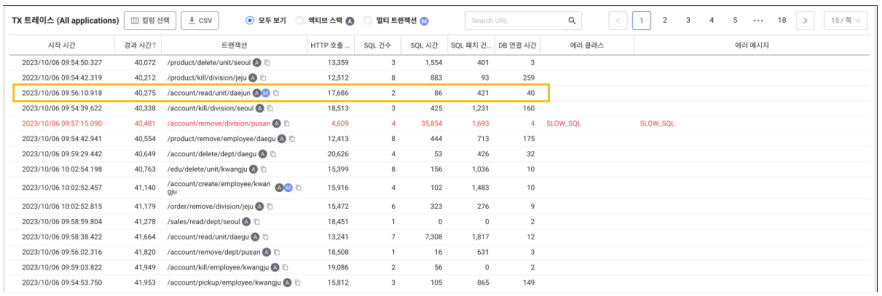
> ⓘ Java 에이전트는 세 개의 HTTP 헤더 키값(`x-wtap-po` , `x-wtap-mst` , `x-wtap-sp1`)으로 멀티 트랜잭션을 추적합니다. 게이트웨이를 통과하는 HTTP 트랜잭션이 연계 추적이 안 된다면 HTTP 헤더 조건을 확인하세요.

> ⓘ **멀티 트랜잭션 활성화**
>
> 멀티 트랜잭션을 추적하려면 **관리** > **에이전트 설정** 메뉴에서 `mtrace_enabled` 옵션을 `true` 로 설정하세요. 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

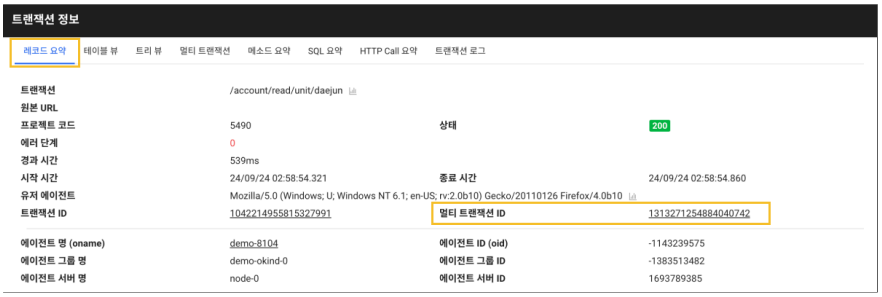
멀티 트랜잭션 ID 확인하기

멀티 트랜잭션 추적 메뉴를 이용하려면 **MTID**(Multi Transaction ID)가 필요합니다. 다음 과정을 통해 **MTID** 값을 확인할 수 있습니다.

1. **분석** > **히트맵** 메뉴에서 차트 영역을 드래그하세요.
2. 드래그한 차트 영역의 트랜잭션 정보가 다음과 같이 하단 **TX 트레이스** 목록에 나타납니다.



3. Ⓜ 아이콘이 표시된 트레이스를 선택하면 **트랜잭션 정보** 창이 나타납니다.
4. **레코드 요약** 탭에서 **멀티 트랜잭션 ID** 값을 확인할 수 있습니다.



- 트랜잭션에서 외부 호출을 하는 경우에도 동일한 **멀티 트랜잭션 ID**가 생성됩니다. 서비스별로 프로젝트가 분리되어 있더라도 처음 발급한 **멀티 트랜잭션 ID**를 통해 애플리케이션 간의 모든 트랜잭션을 확인할 수 있습니다. **트랜잭션 정보** 창을 활용한 트랜잭션 트레이스 상세 분석에 관한 자세한 내용은 다음 문서를 참조하세요.
- **트랜잭션 정보** 창에서 **멀티 트랜잭션 ID**를 선택하면 **멀티 트랜잭션** 탭으로 이동합니다. 시스템 내 또는 시스템 간에 발생하는 다양한 호출 관계를 한 눈에 파악하고 어느 부분에서 문제가 발생했는지 식별하여 개선할 수 있도록 트랜잭션과 트레이스 정보를 제공합니다.

멀티 트랜잭션 추적 기능 이용하기

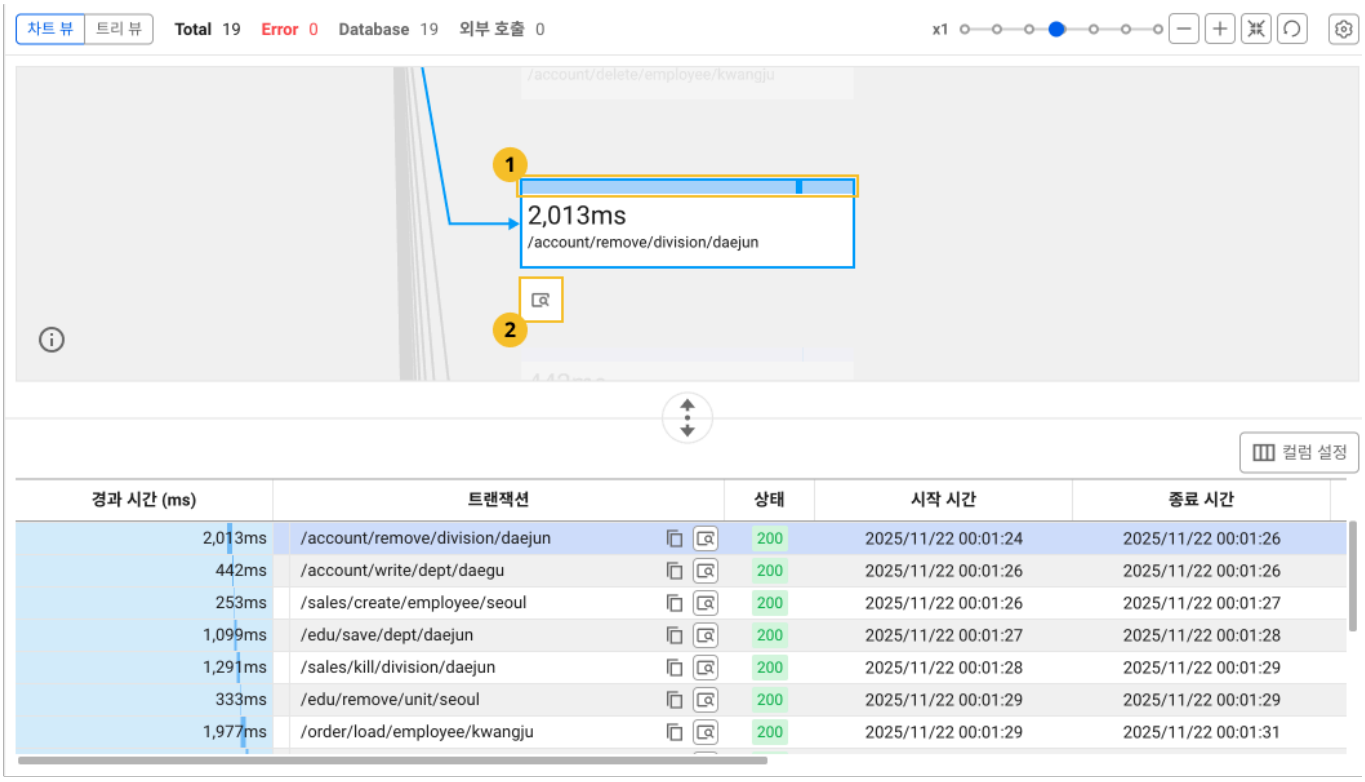
1. **분석** > **멀티 트랜잭션 추적**로 이동하세요.
2. **트랜잭션 정보** 창에서 확인한 **MTID** 값을 **MTID / CUSTID 조회** 항목에 입력하세요.
3. 조회할 날짜와 프로젝트를 선택하세요.
4. 화면 아래에 **적용** 버튼을 클릭하세요.
   - 다음 조건에서 화면 하단의 **적용** 버튼이 비활성화됩니다.
     - 이전 검색과 동일한 필터값(MTID/CUSTID, 날짜, 선택 프로젝트)을 설정한 경우
     - 필터값(MTID/CUSTID, 날짜, 선택 프로젝트)을 하나도 입력하지 않은 경우
5. 오른쪽 **차트** 탭에 각 트랜잭션의 호출 관계를 파악할 수 있는 다이어그램과 테이블이 표시됩니다.

   다이어그램과 테이블 영역은 상하로 구분되어 그 비율을 직접 조절할 수 있습니다.

- 위쪽 화살표를 클릭하면 테이블이 전체 영역을 차지합니다. 단, 다이어그램 영역이 전체 영역을 차지하고 있었다면 다이어그램과 테이블이 각각 절반의 영역으로 나뉩니다.
- 아래쪽 화살표를 클릭하면 차트 영역이 전체 영역을 차지합니다. 단, 테이블 영역이 전체 영역을 차지하고 있었다면 다이어그램과 테이블이 각각 절반의 영역으로 나뉩니다.

차트

차트는 각 트랜잭션의 호출 관계를 빠르고 명확하게 사용자에게 제공합니다. 동일한 **멀티 트랜잭션 ID**를 갖는 트랜잭션 서비스들의 개별 수행 시간을 확인할 수 있습니다. 트랜잭션 노드의 상단에 표현되어 있는 소요시간(1 **타임바**)를 통해 트랜잭션 간 호출 관계를 확인할 수 있습니다.



차트는 마우스로 자유롭게 이동하거나 확대, 축소할 수 있습니다.

- 차트 영역을 드래그하면 원하는 위치로 이동할 수 있습니다.

- 차트 영역을 스크롤하면 원하는 위치로 이동할 수 있습니다.
  - 상하 스크롤: 상하 이동
  - 좌우 스크롤: 좌우 이동
- Ctrl + 상하 스크롤로 차트를 확대하거나 축소할 수 있습니다.
  - 터치패드의 확대, 축소 동작으로도 가능합니다.

트랜잭션을 선택하면 연결된 노드만 활성화되고, 연결 관계없는 모든 노드는 흐리게 표시됩니다. 선택된 트랜잭션의 하단의 2 **상세보기** 버튼이 활성화되며, 클릭 시 **트랜잭션 정보** 창이 열립니다. 해당 트랜잭션의 상세 내역을 확인할 수 있습니다. **트랜잭션 정보** 창을 활용한 트랜잭션 트레이스 상세 분석에 관한 자세한 내용은 다음 문서를 참고하세요.

- **확대**, **축소**: 현재 확대 배율과 확대 축소 동작이 가능함
- **화면에 맞춤**: 모든 노드를 화면에 맞게 표시
- **확대/축소 초기화**: 차트의 표시 위치와 확대 정도를 최초 상태로 초기화
- **차트 뷰 설정**: 차트에 표시할 요소를 표시하거나 숨김
  - **상세 정보**: 애플리케이션 명, 프로젝트 정보, IP 정보, 상태 코드를 표시하거나 숨김
  - **데이터베이스 / 외부 호출**: 해당 트랜잭션에서 발생한 다른 데이터베이스 커넥션 요청이나 HTTP Call의 정보 또한 차트의 노드로 확인 가능함

테이블

테이블 탭에서 멀티 트랜잭션 내에 포함된 각 트랜잭션 별 정보를 테이블 형식으로 확인할 수 있습니다.

각 트랜잭션 항목을 선택하면, 해당 트랜잭션 **차트**의 **노드**가 활성화되고, 연결된 노드가 하이라이트 됩니다. 트랜잭션 컬럼의 상세보기 버튼을 클릭하면, **트랜잭션 정보** 창이 나타납니다. 트랜잭션 트레이스에서 트랜잭션의 상세 내역을 확인할 수 있습니다. **트랜잭션 정보** 창을 활용한 트랜잭션 트레이스 상세 분석에 관한 자세한 내용은 다음 문서를 참조하세요.

- **컬럼 설정**: 테이블 내 컬럼 편집

트리

트리 탭에서 각 트랜잭션과 그에 속해 있는 트레이스의 세부 정보를 확인할 수 있습니다. 전체 트랜잭션 소요 시간 내의 각 하위 트랜잭션이나 트레이스의 시작 및 소요 시간을 시각화해 트랜잭션 호출 관계를 트리 형식으로 제공합니다.

- **한 줄 보기**: 구간별 수행 정보에 표시된 텍스트를 한 줄로 표시해 트리 형식을 간격하게 정리함
- **여러 줄 보기**: 구간별 수행 정보에 표시된 텍스트를 줄바꿈해 모두 표시함
- **최장 경로**: 가장 긴 경로로 이동함
- **시간바 표시**: 경과 시간을 막대 형식의 차트로 표시
- **시간 표시**: 구간별 타임 스탬프, 갭, 경과 시간을 텍스트 형식으로 표시
  - 8초 이상: 초과 지연 상태로 빨간색으로 표현
  - 3초 이상 8초 미만: 지연 상태로 주황색으로 표현
  - 3초 미만: 정상 상태로 파란색으로 표현

> 시작 및 소요 시간의 경우 트랜잭션 호출 환경에 따라 발생하는 시차를 상위 트랜잭션 내 트레이스와 매핑을 통해 보정하여 표현하기 때문에 실제 수집된 시간 데이터와 차이가 발생할 수 있습니다.

- **시간 숨기기**: 시간 정보를 숨김

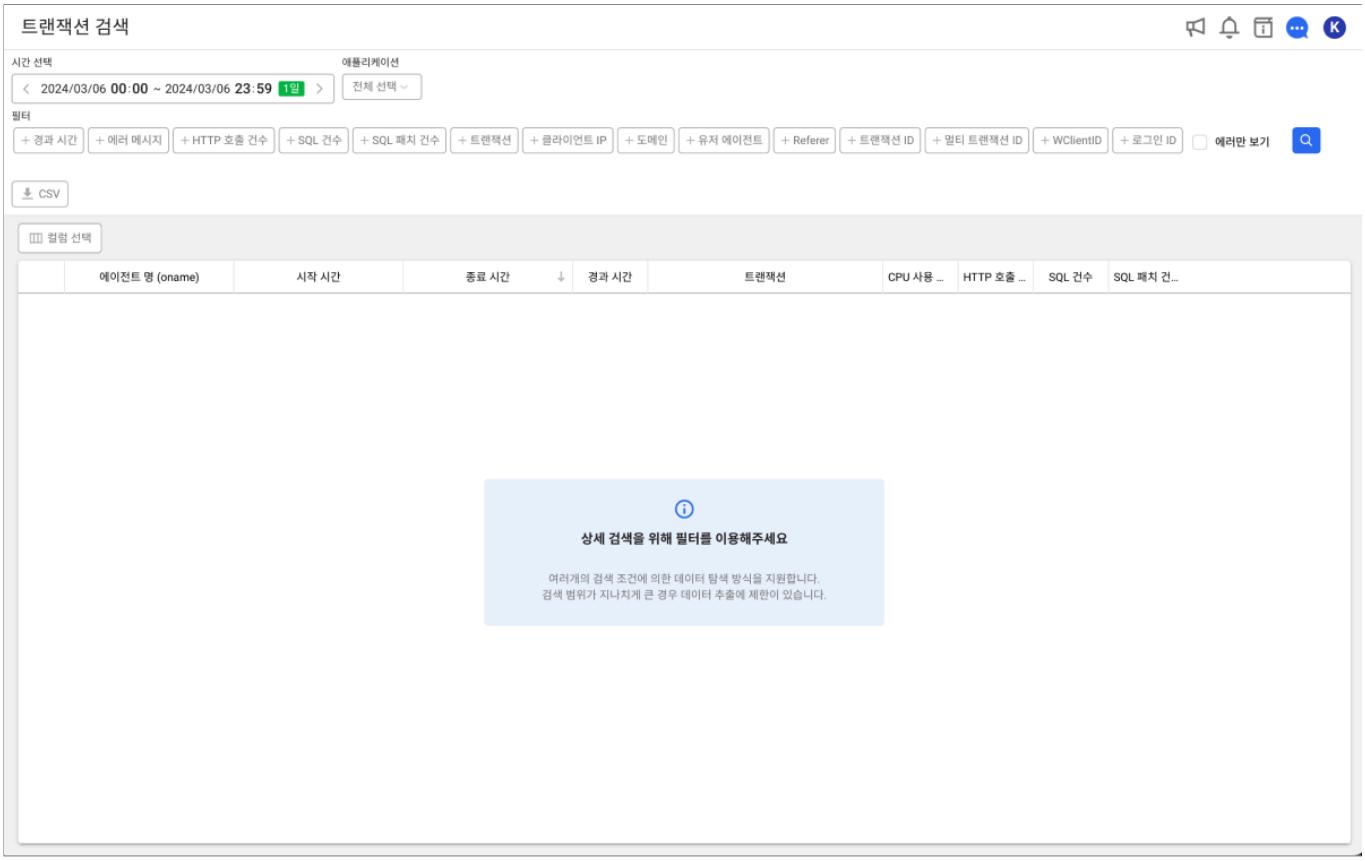
> 해당 트레이스의 버튼 또는 버튼을 선택하면 **HTTP 호출 통계**, **액티브 스택** 등의 요약 창을 확인할 수 있습니다.

> SQL 변수와 HTTP 쿼리를 조회하려면 다음 옵션을 에이전트 설정에 추가하세요.
>
> - SQL 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
> - HTTP 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
>
> whatap.conf
>
> ```
> # SQL 파라미터 조회 옵션: 옵션이 적용되면 SQL 파라미터를 암호화하여 수집합니다.
> profile_sql_param_enabled=true
>
> # HTTP 파라미터 조회 옵션: 옵션이 적용되면 HTTP 쿼리 파라미터를 암호화하여 수집합니다.
> profile_http_parameter_enabled=true
> ```

TX 프로파일

홈 화면 > 프로젝트 선택 > **분석** > **트랜잭션 검색**

특정 기간 동안 각 트랜잭션의 성능 속성에 대한 통계와 실행 상태를 검색할 수 있습니다. 모든 트랜잭션을 확인하고 정상 및 에러 상태를 빠르게 확인할 수 있습니다. 제공되는 정보를 통해 어떤 트랜잭션의 사용량이 많고 리소스를 많이 소비하는지 알 수 있습니다. 트랜잭션의 속성에 따라 다양하게 필터링할 수 있어 특정 조건의 트랜잭션을 찾는데 유용합니다.



조회 시간을 설정한 다음 **애플리케이션** 옵션에서 조회 대상을 선택하세요. 버튼을 선택하면 검색 결과가 테이블에 표시됩니다. 결과 목록에서 원하는 항목을 선택하면 **트랜잭션 정보** 창이 나타납니다. **트랜잭션 정보** 창을 활용한 트랜잭션 트레이스 상세 분석에 관한 자세한 내용은 다음 문서를 참조하세요.

- 에러 상태의 트랜잭션만 검색하려면 **에러만 보기**를 클릭한 다음 버튼을 선택하세요.

- 검색 조건에 따라 응답 지연이 발생할 수 있기 때문에 검색 시간을 가급적 짧게 설정할 것을 권고합니다.

컬럼 편집하기



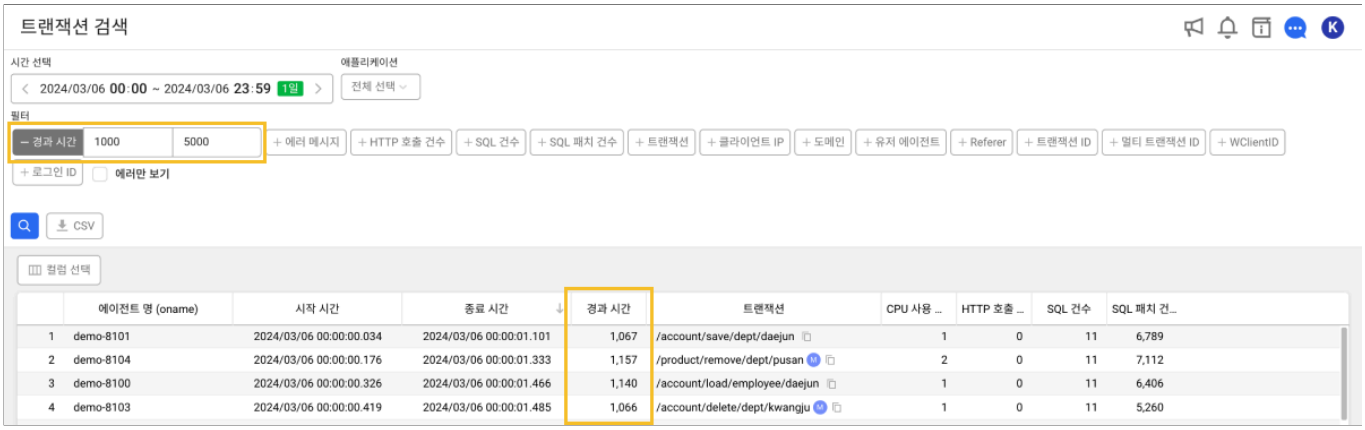
결과 조회 시 **컬럼 선택** 버튼을 클릭하면 확인이 필요한 컬럼을 적절하게 선택할 수 있습니다.

필터 적용하기

상세 검색을 위해 필터를 이용하세요. 여러 개의 검색 조건에 의한 데이터 탐색 방식을 지원합니다. 필터를 설정한 다음 버튼을 선택하세요.

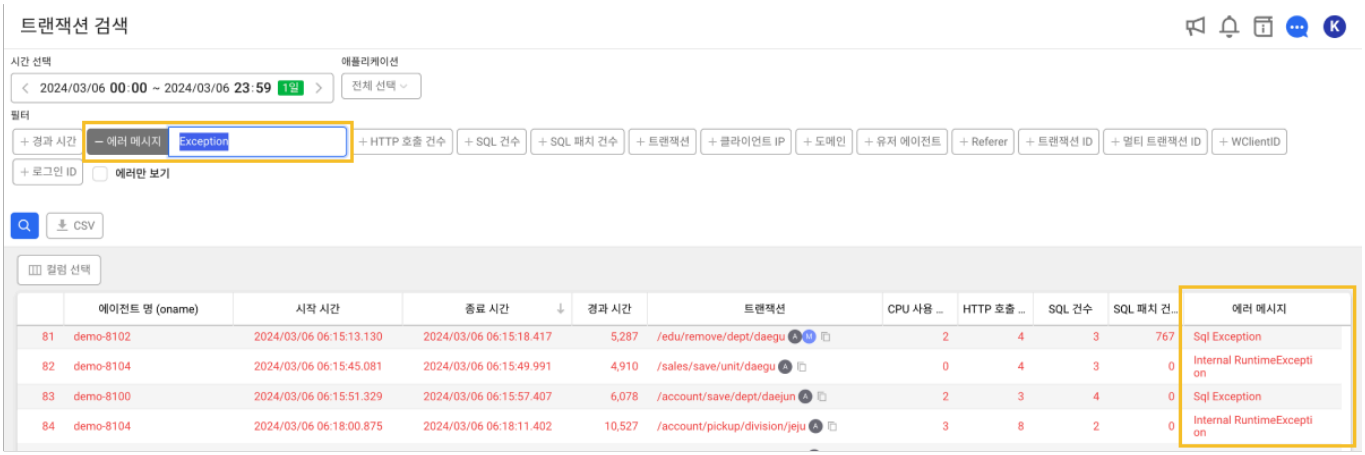
일부 필터 항목은 제품에 따라 지원하지 않을 수 있습니다.

경과 시간



트랜잭션 응답시간 기준으로 검색합니다. 기준 이상과 미만의 2개 값을 입력해 검색합니다.

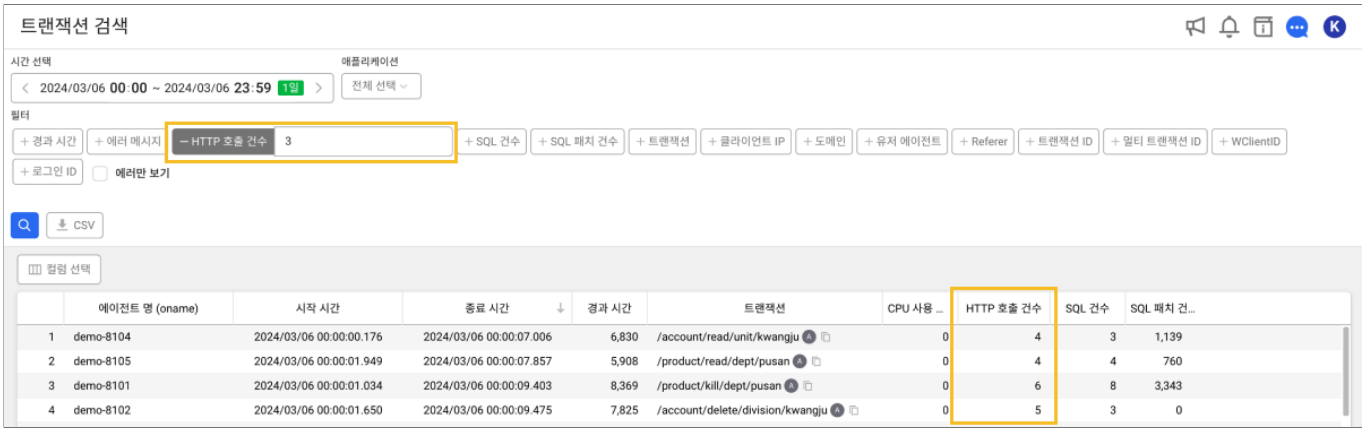
에러 메시지



트랜잭션 에러 메시지를 기반으로 검색합니다. 디폴트 상태에서 에러 관련 컬럼이 보이지 않을 수 있습니다. 컬럼을 선택하세요.

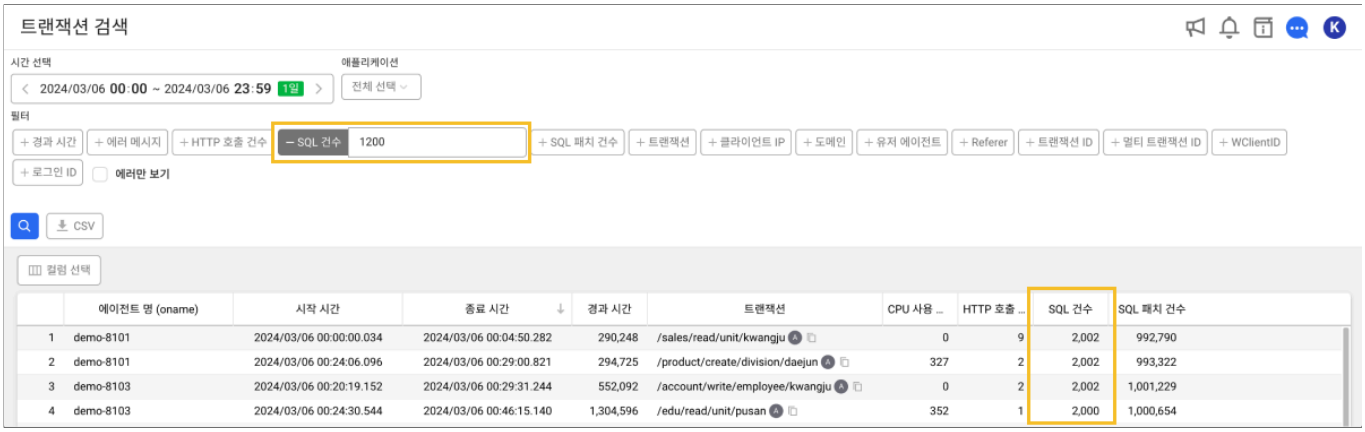
> ⓘ 에러 상태의 트랜잭션은 빨간색으로 표시됩니다.

HTTP 호출 건수



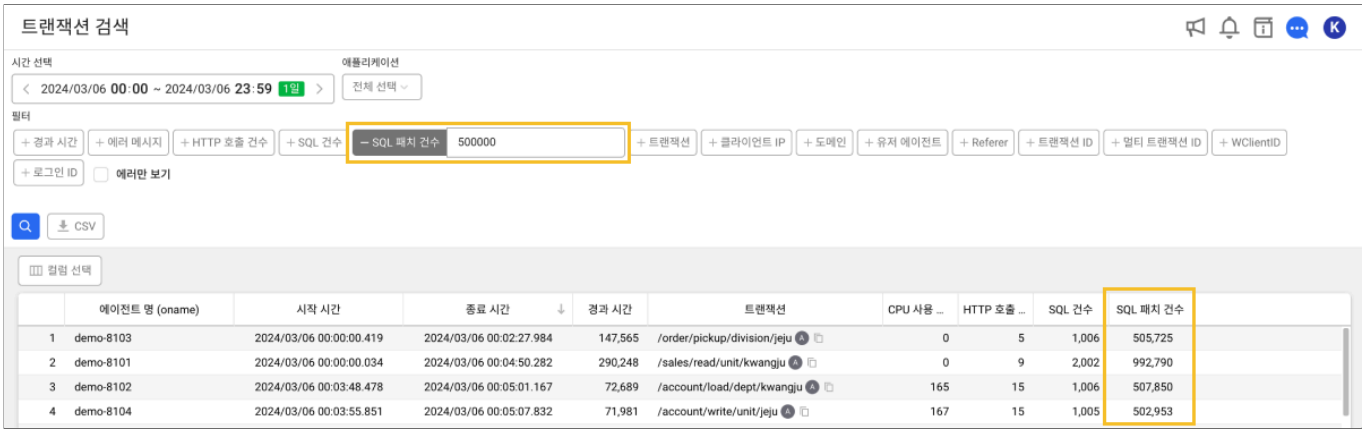
트랜잭션 중 외부 HTTP 서비스를 호출한 건수를 기준으로 입력한 자연수 이상에 해당하는 트랜잭션을 검색합니다.

SQL 건수



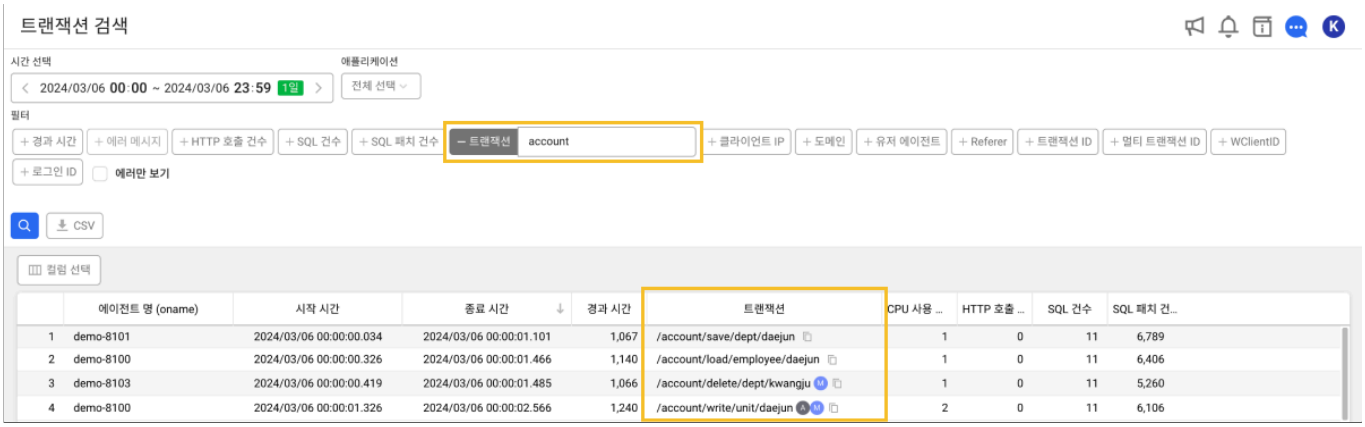
트랜잭션별 SQL 수행 건수가 기준으로 입력한 자연수 이상에 해당하는 트랜잭션을 검색합니다. 예외적으로 과도한 수의 SQL을 호출하는 트랜잭션을 검색할 때 이용합니다.

SQL 패치 건수



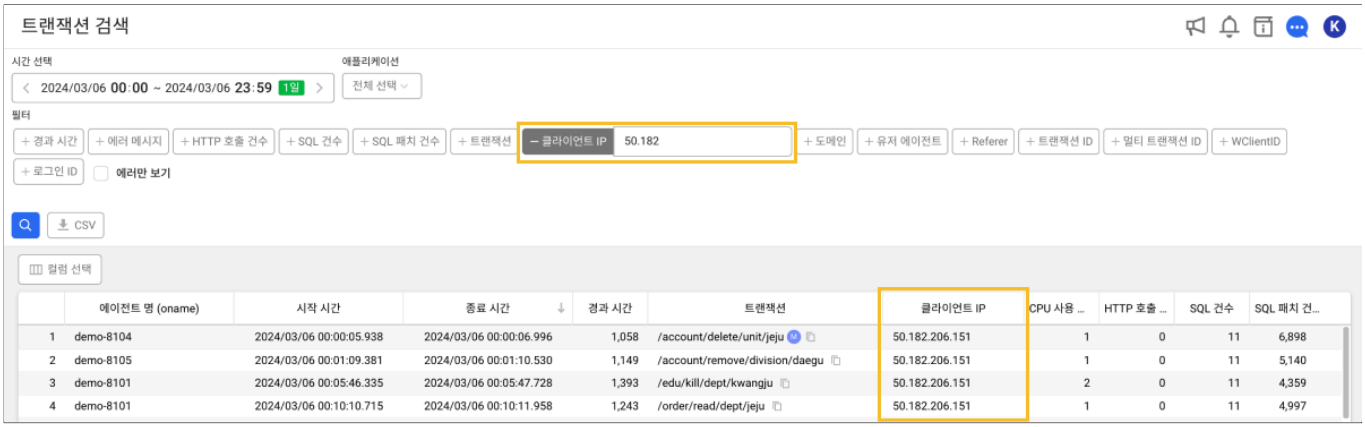
SQL 수행 후 DB 패치 건수를 기준으로 입력한 자연수 이상에 해당하는 트랜잭션을 검색합니다. 과도한 데이터 패치를 수행하는 트랜잭션을 검색할 때 이용합니다.

트랜잭션



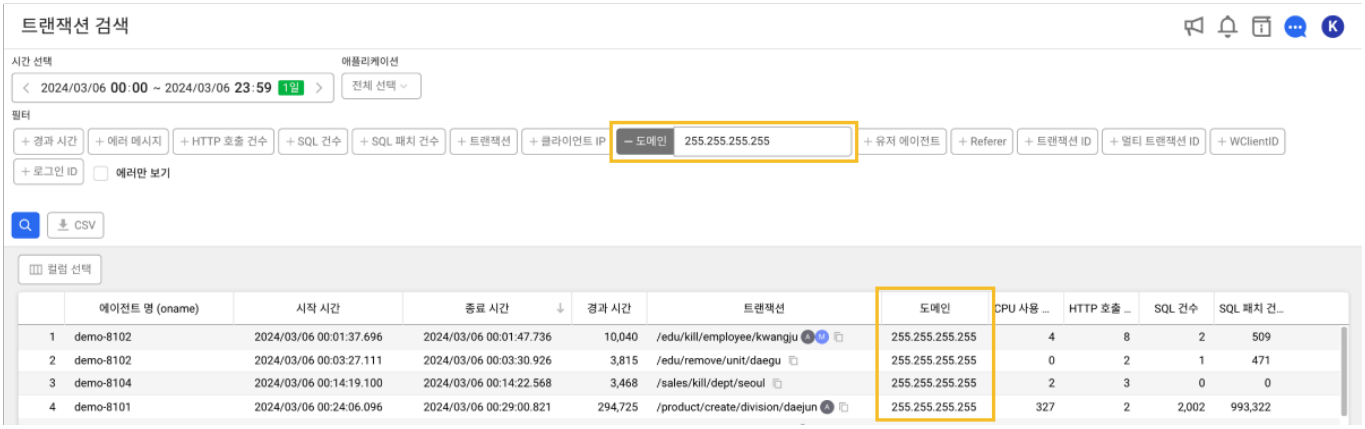
트랜잭션을 기준으로 검색합니다. 입력한 검색 문자열은 포함 여부(부분 일치)로 비교 검색합니다.

클라이언트 IP



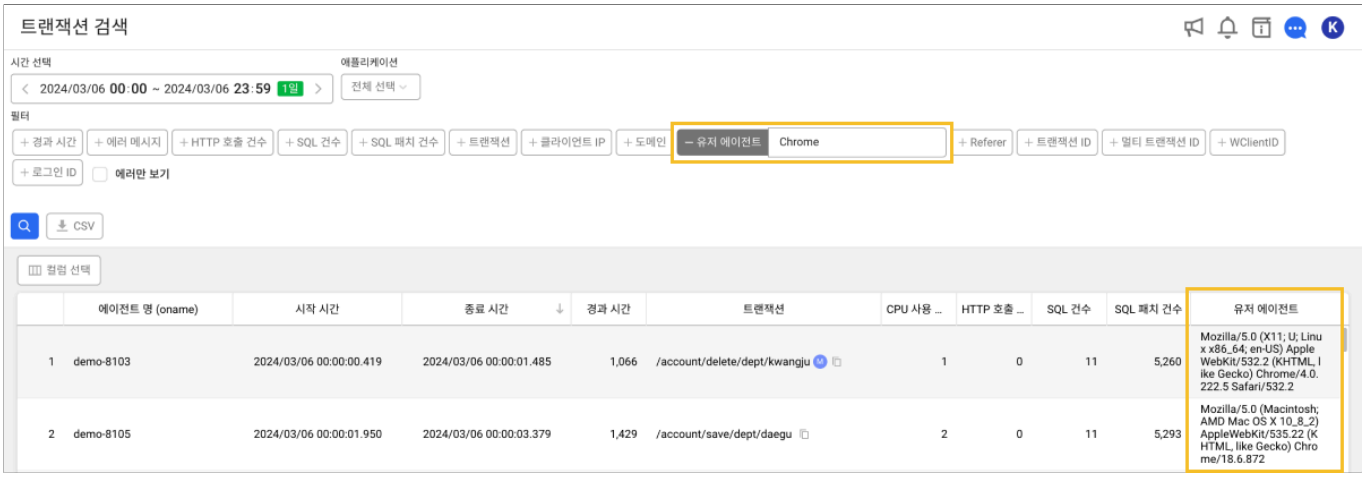
클라이언트 IP 기반으로 트랜잭션을 검색합니다. 단, IP 문자열은 PREFIX 비교(전방 일치)로 검색합니다.

도메인



HTTP 서비스 도메인(host:port) 기반으로 트랜잭션을 검색합니다. 검색 조건은 도메인이 완전히 일치(전체 일치)해야 합니다.

유저 에이전트



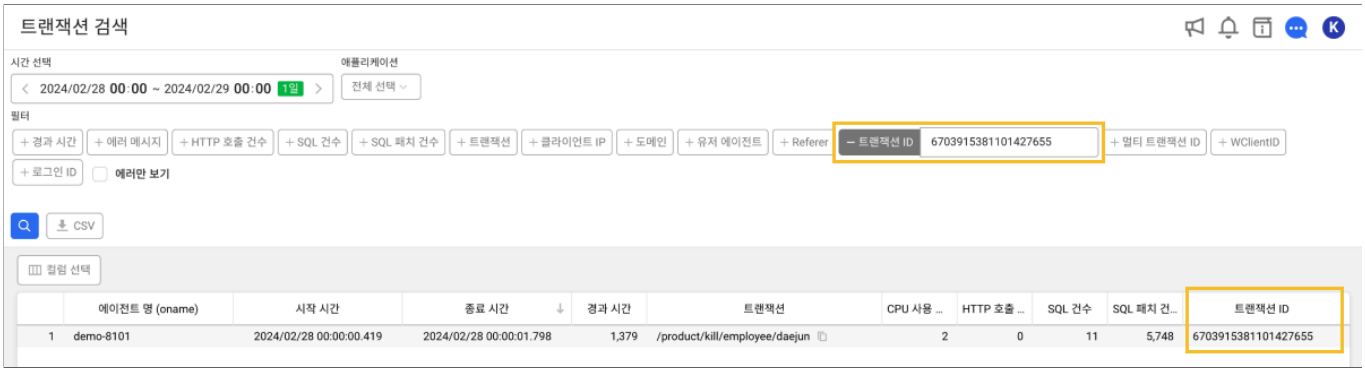
유저 에이전트 기반으로 트랜잭션을 검색(부분 일치)합니다.

Referer

Referer 문자열 기반으로 트랜잭션을 검색합니다.

> ⓘ 기본 설정 상태의 에이전트는 Referer 정보를 수집하지 않습니다. Referer 설정이 필요한 경우 다음 문서를 참조해 관련 설정을 추가하세요.

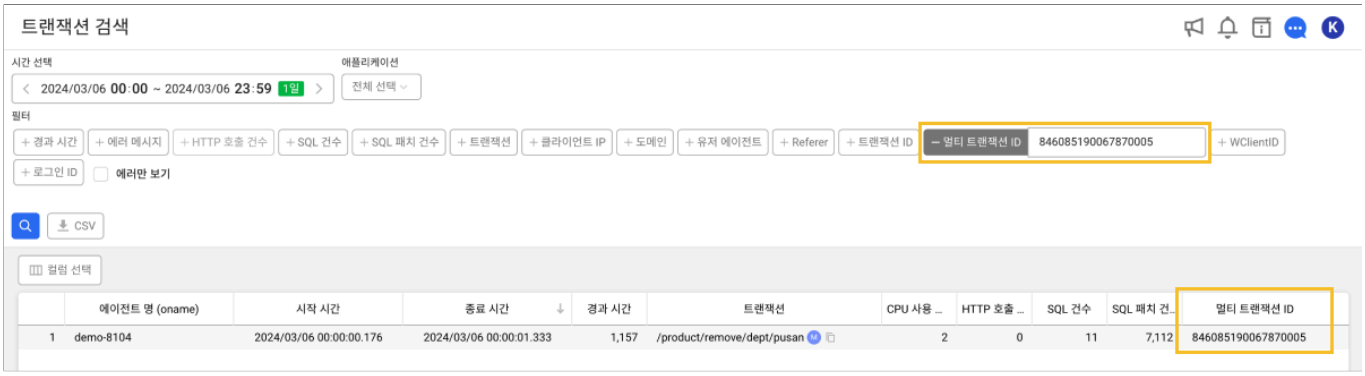
트랜잭션 ID



트랜잭션 ID 기반으로 트랜잭션을 검색(전체 일치)합니다. 트랜잭션 ID의 경우 검색 인덱스가 생성되어 있기 때문에 빠른 속도로 검색할 수 있습니다.

- 트랜잭션 ID는 **트랜잭션 정보** 창의 **레코드 요약** 탭에서 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- 최신 버전의 에이전트가 아닌 경우 인덱스 사용이 아닌 풀스캔이 발생할 수 있습니다.

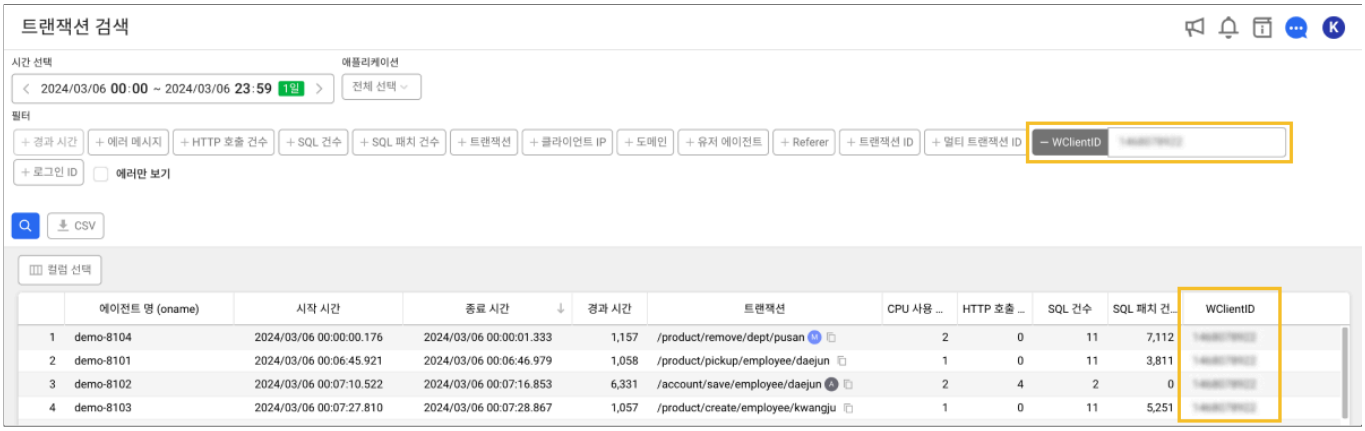
멀티 트랜잭션 ID



멀티 트랜잭션 ID 기반으로 트랜잭션을 검색(전체 일치)합니다.

- 멀티 트랜잭션 ID는 **트랜잭션 정보** 창의 **레코드 요약** 탭에서 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- 최신 버전의 에이전트가 아닌 경우 인덱스 사용이 아닌 풀스캔이 발생할 수 있습니다.
- **멀티 트랜잭션 ID** 필터를 적용하여 검색할 경우 **트랜잭션 ID** 필터만 추가 적용할 수 있습니다.

WClientID



클라이언트 ID(`wClientId`)는 와탭 에이전트가 사용자를 구분하기 위해 부여하는 ID입니다. 이 ID 값 기반으로 트랜잭션을 검색(전체 일치)합니다. 검색을 위한 별도 인덱스는 없습니다.

> ⓘ 와탭은 클라이언트와 관련한 정보를 기본 저장합니다.

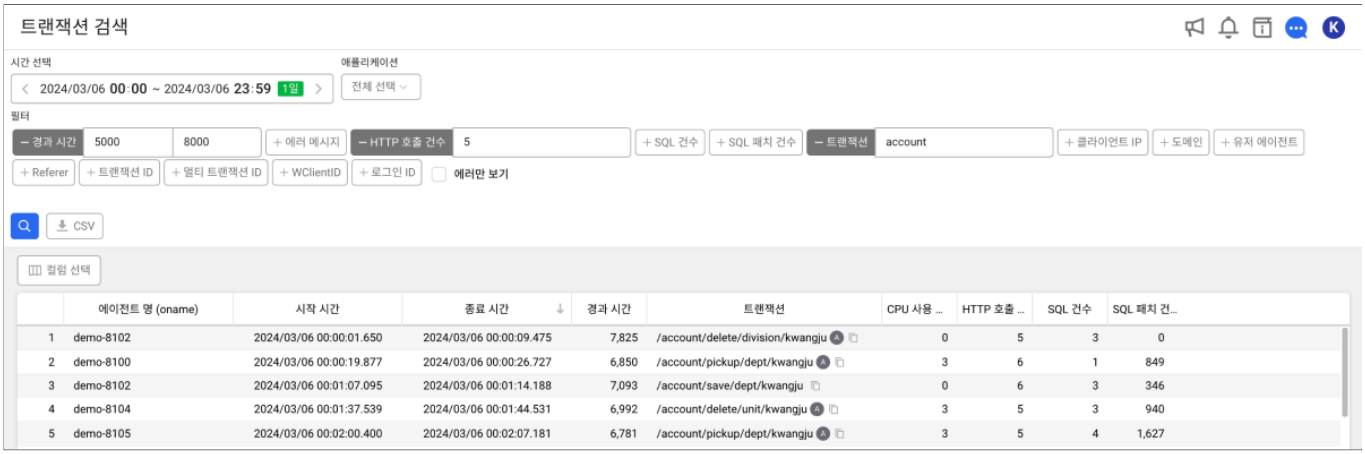
로그인 ID

로그인 아이디 기반으로 트랜잭션을 검색합니다.

> ⓘ 기본 설정 상태의 에이전트는 로그인 ID 정보를 수집하지 않습니다. 로그인 ID 설정이 필요한 경우 다음 문서를 참조해 설정을 추가하세요.

다중 필터 적용하기

여러 개의 필터 항목을 적용해 원하는 트랜잭션 목록을 확인할 수 있습니다. 다음 사례는 경과 시간이 5 ~ 8초 사이이면서, HTTP 호출 건수가 5건 이상, 트랜잭션 경로에 'account'가 포함된 경우입니다.



ⓘ **멀티 트랜잭션 ID** 필터를 적용하여 검색할 경우 **트랜잭션 ID** 필터만 추가 적용할 수 있습니다.

검색 결과 다운로드하기

사용자가 설정한 필터 조건에 따라 검색된 결과를 CSV 파일로 다운로드할 수 있습니다.

1. 버튼을 클릭해 검색 결과를 조회하세요.
2. CSV 버튼을 선택하세요.
3. 최대 CSV 라인 수를 입력하세요. 입력한 라인 수만큼 조회된 결과를 저장할 수 있습니다.
4. **다운로드** 버튼을 선택하면 CSV 파일로 다운로드합니다.

ⓘ CSV 파일 이름은 profile_ {pcode} _ YYMMDD .csv 형식입니다.

MSA 분석

MSA 분석이란?

MSA 분석 기능은 와탭의 **멀티 트랜잭션 추적** 기능을 통해서 수집된 트랜잭션들 사이의 호출 비중이 어떠한지를 분석합니다. 이를 기반으로 마이크로 서비스간 의존성을 인스턴스 중심이 아닌 URL 레벨에서 확인할 수 있습니다.


Microservice Architecture Pattern

예를 들어 A,B,C,D,E 라는 5개 마이크로 서비스가 있는데, 사용자가 A를 호출하면 로직에 따라서 다음과 같은 3가지의 호출 패턴이 발생한다고 가정하겠습니다.

1. A→B,C 호출
2. A→B,C 호출 + 다시 B→D 호출
3. A→B,C 호출 + 다시 B→D,E 호출

이 세가지 패턴이 한번씩 수행되었다면 MSA 분석에서는 기준 URL에 따라 연관도 분석 데이터를 확인할 수 있습니다.

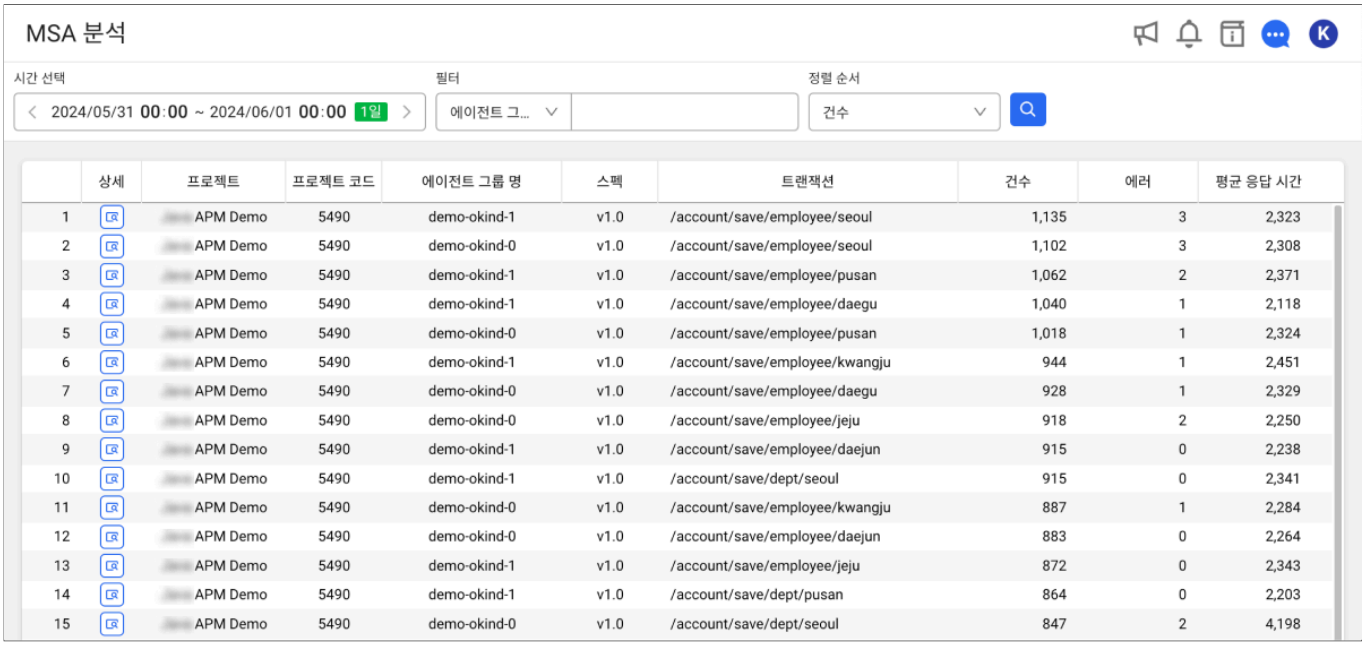

Microservice Architecture Caller & Callee

> **멀티 트랜잭션 추적**과 관련한 자세한 내용은 다음 문서를 참조하세요.

MSA 트랜잭션 통계

홈 화면 > 프로젝트 선택 > **사이트맵** > **분석** > **MSA 분석**

MSA 분석 메뉴는 마이크로 서비스 환경에서 각 서비스의 성능과 안정성을 효과적으로 분석할 수 있는 도구입니다. 트랜잭션의 상세 호출 관계를 분석하고, 이를 기반으로 시스템 성능을 최적화하며 잠재적인 문제를 사전에 식별할 수 있습니다.



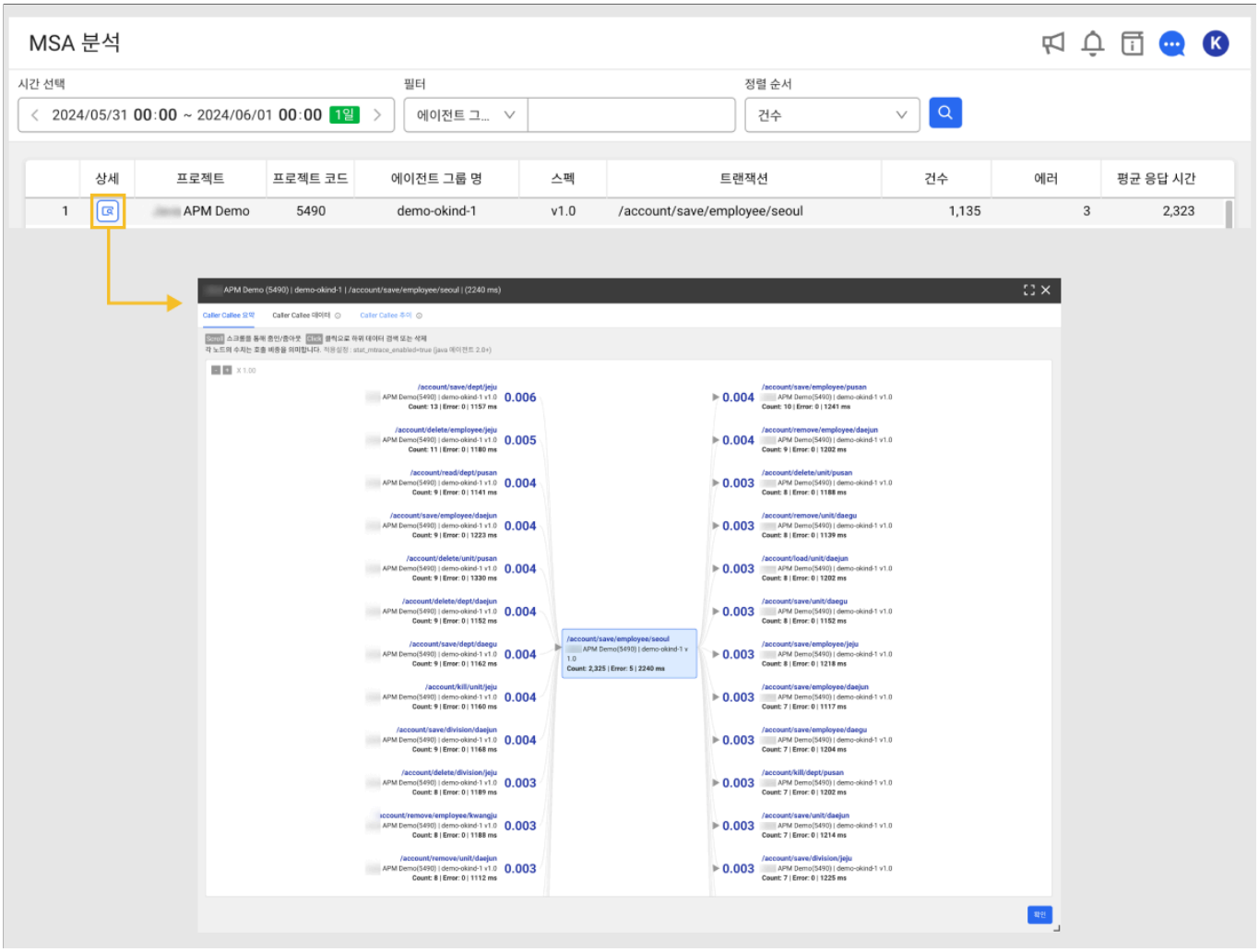
	상세	프로젝트	프로젝트 코드	에이전트 그룹 명	스펙	트랜잭션	건수	에러	평균 응답 시간
1		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/seoul	1,135	3	2,323
2		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/seoul	1,102	3	2,308
3		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/pusan	1,062	2	2,371
4		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/daegu	1,040	1	2,118
5		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/pusan	1,018	1	2,324
6		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/kwangju	944	1	2,451
7		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/daegu	928	1	2,329
8		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/jeju	918	2	2,250
9		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/daejun	915	0	2,238
10		APM Demo	5490	demo-okind-1	v1.0	/account/save/dept/seoul	915	0	2,341
11		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/kwangju	887	1	2,284
12		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/daejun	883	0	2,264
13		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/jeju	872	0	2,343
14		APM Demo	5490	demo-okind-1	v1.0	/account/save/dept/pusan	864	0	2,203
15		APM Demo	5490	demo-okind-0	v1.0	/account/save/dept/seoul	847	2	4,198

- **시간 선택**: 원하는 조회 기간을 선택하세요. 기본 1일 기준으로 Caller Callee를 포함한 트랜잭션 목록을 조회할 수 있습니다.
- **필터**: **에이전트 그룹 명**, **스펙**, **트랜잭션** 기준으로 필터링하여 검색할 수 있습니다.
- **정렬 순서**: **건수**, **에러**, **평균 시간**(평균 응답 시간) 기준으로 목록을 정렬할 수 있습니다.

> - **에이전트 그룹 명**은 에이전트 설정 옵션의 `whatap.okind` 로 분류된 그룹 이름입니다.
> - 조건을 설정한 다음 트랜잭션 목록을 갱신하려면 버튼을 선택하세요.

상세 분석하기

트랜잭션 목록에서 조회를 원하는 항목의 버튼을 선택하면 상세 보기 창이 나타납니다. 상세 보기 창에서 해당 트랜잭션에 대한 호출자(Caller), 피호출자(Callee) 정보를 조회할 수 있습니다.



각 탭에서는 다음의 분석 기능을 제공합니다.

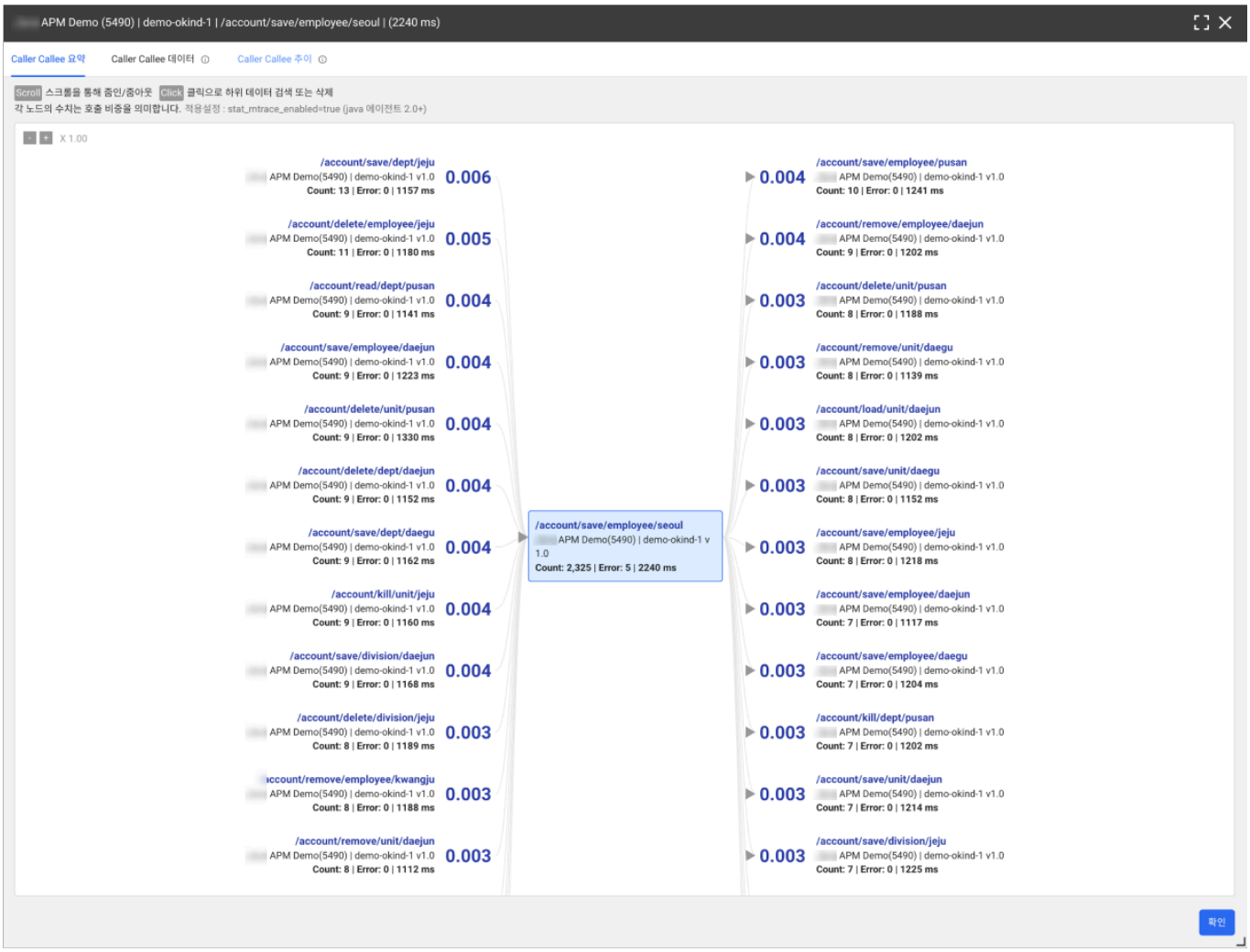
- **Caller Callee 요약**: 트랜잭션의 호출자와 피호출자 간의 관계를 시각화된 다이어그램으로 제공합니다.
- **Caller Callee 데이터**: 트랜잭션의 호출 횟수, 오류 수, 응답 시간을 상세히 제공합니다.
- **Caller Callee 추이**: 시간에 따른 트랜잭션의 호출 횟수 및 비율의 성능 변화를 시계열 차트로 제공합니다.

> ⓘ
> - 상세 보기 창에서 오른쪽 위에 ⛶ 버튼을 선택하면 전체 화면으로 볼 수 있습니다.
> - 상세 보기 창을 닫으려면 오른쪽 아래에 **확인** 버튼을 선택하거나 ESC 키를 누르세요.

Caller Callee 요약

Caller Callee 요약 탭에서는 트랜잭션 목록에서 선택한 트랜잭션의 Caller, Callee 사이의 의존성 정보를 시각화된 다이어그램으로 표시합니다. 서비스 간의 호출 관계를 직관적으로 이해할 수 있습니다. 호출 경로와 종속성을 시각적으로 파악할 수 있어 문제 원인 분석에 유리합니다.

호출 관계 분석하기



Caller와 Callee 노드에는 최대 0.004, 0.003 등 최대 1인 숫자 정보를 표시하는데, 이것은 전체 Caller 가운데 차지하는 비중을 의미합니다. 마찬가지로 각 Callee의 비중은 오른쪽에 함께 표시합니다.

위 화면을 통해 /account/save/employee/seoul 이라는 트랜잭션을 기준으로 Caller와 Callee들 사이의 의존성을 분석할 수 있습니다. 트랜잭션이 어떤 서비스에서 호출되고 어떤 서비스로 호출되는지 시각적으로 확인할 수 있습니다.

성능 지표 확인하기

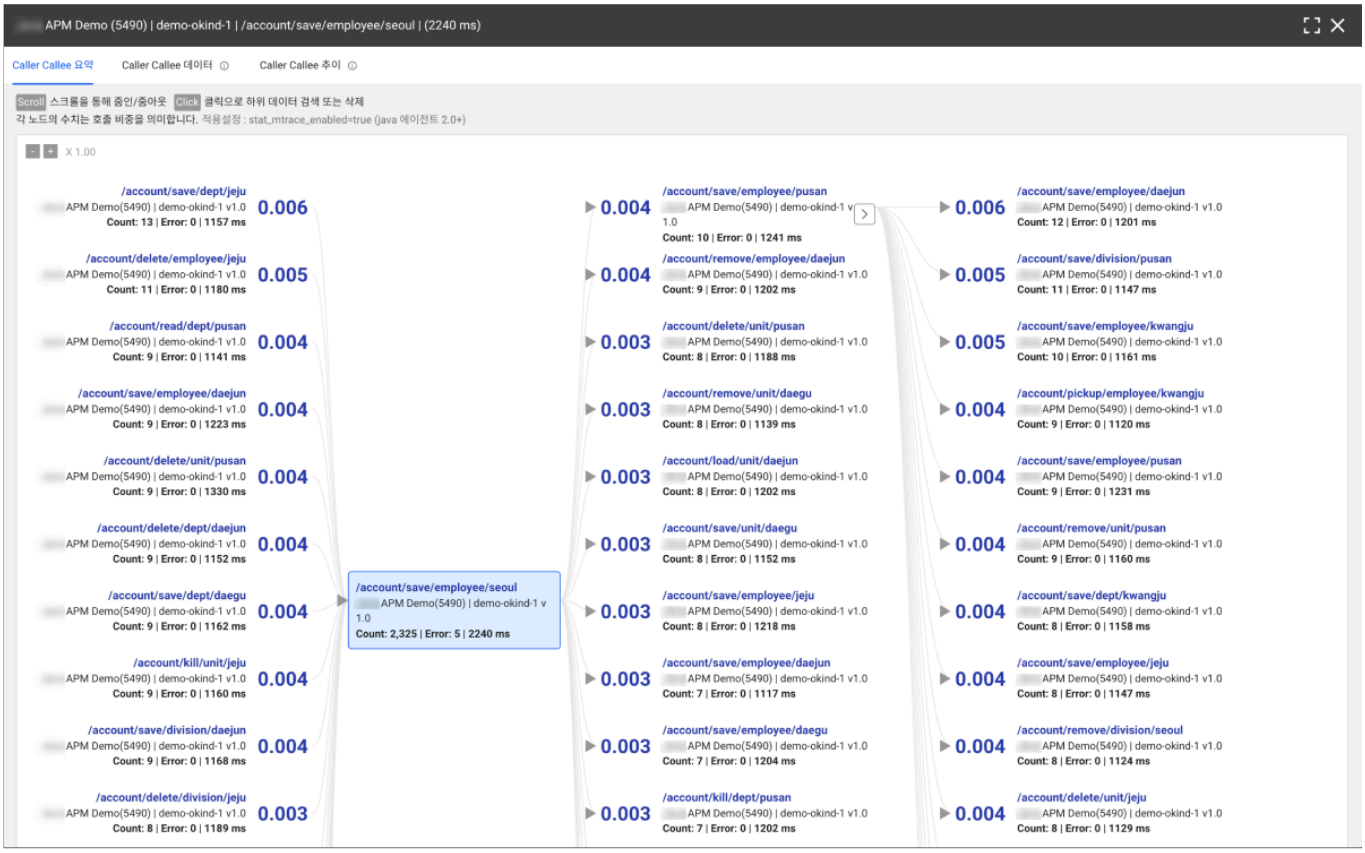


각 노드는 트랜잭션 URL, 프로젝트 이름, 에이전트 그룹 이름 그리고 다음 성능 정보로 구성되어 있습니다.

- **Count**: 해당 트랜잭션을 호출한 횟수
- **Error**: 호출 중 발생한 에러 수
- 평균 응답 시간: 해당 트랜잭션의 평균 응답 시간, 밀리초(ms) 단위

트랜잭션 흐름 파악하기

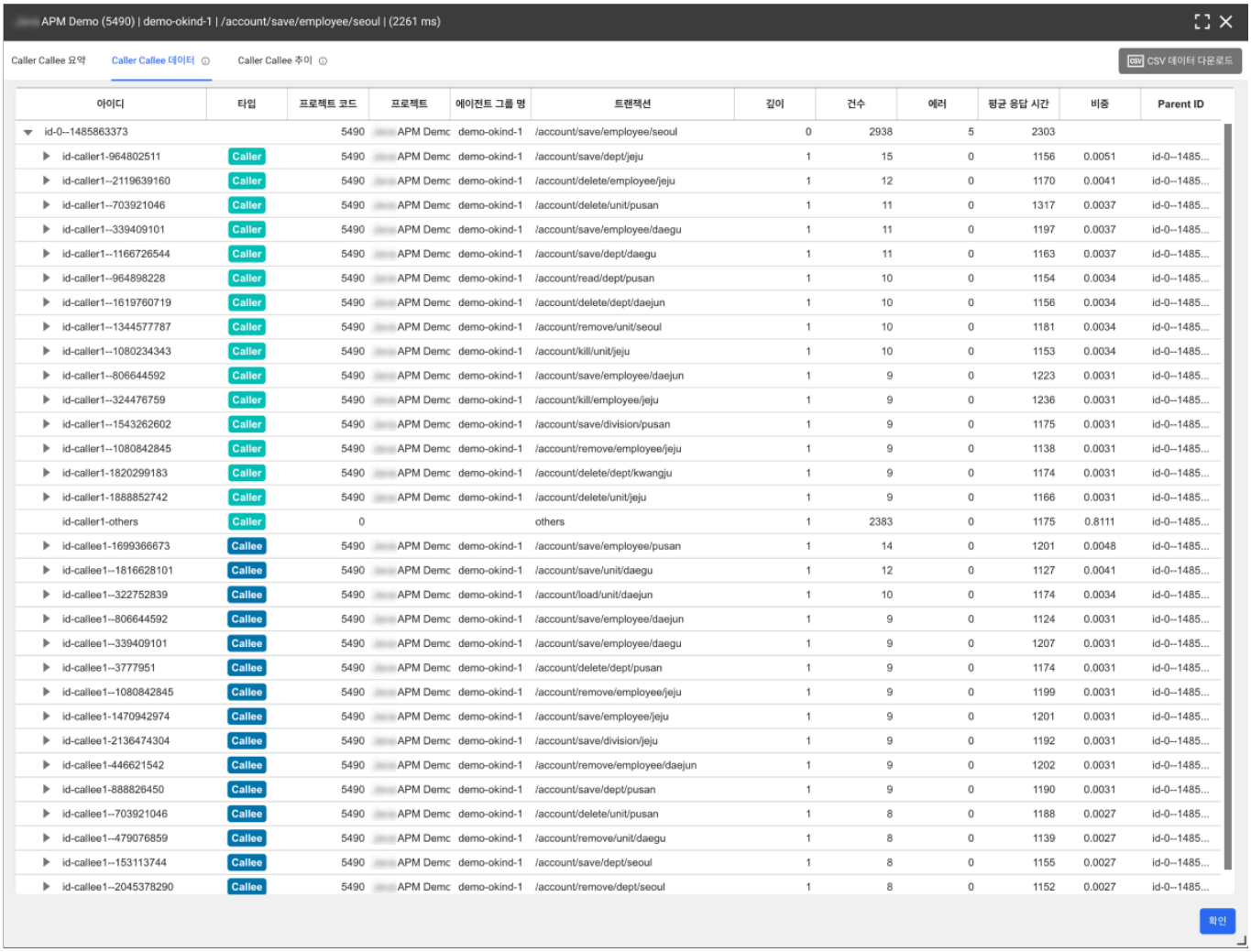
Caller나 Callee 노드에 마우스를 오버하면 > 버튼이 표시됩니다. 이 버튼을 선택하면 하위 노드로 상세 전개할 수 있습니다. 트랜잭션의 전체 흐름을 파악하여 문제 발생 지점을 찾는데 도움을 받을 수 있습니다.



- 하위 노드를 닫으려면 다시 > 버튼을 선택하세요.
- 마우스의 스크롤 기능으로 화면을 확대하거나 축소할 수 있습니다. 또는 화면 왼쪽 위에 + / - 버튼을 클릭하세요.

Caller & Callee 데이터

Caller Callee 데이터 탭에서는 기준 URL(예: /account/save/employee/seoul)에 대한 호출자(Caller)와 피호출자(Callee)의 상관 관계를 표 형식으로 확인할 수 있습니다. 각 트랜잭션의 세부 정보(호출 횟수, 오류 수, 평균 응답 시간 등)를 구체적으로 확인할 수 있어 데이터 기반의 정밀한 분석이 가능합니다.

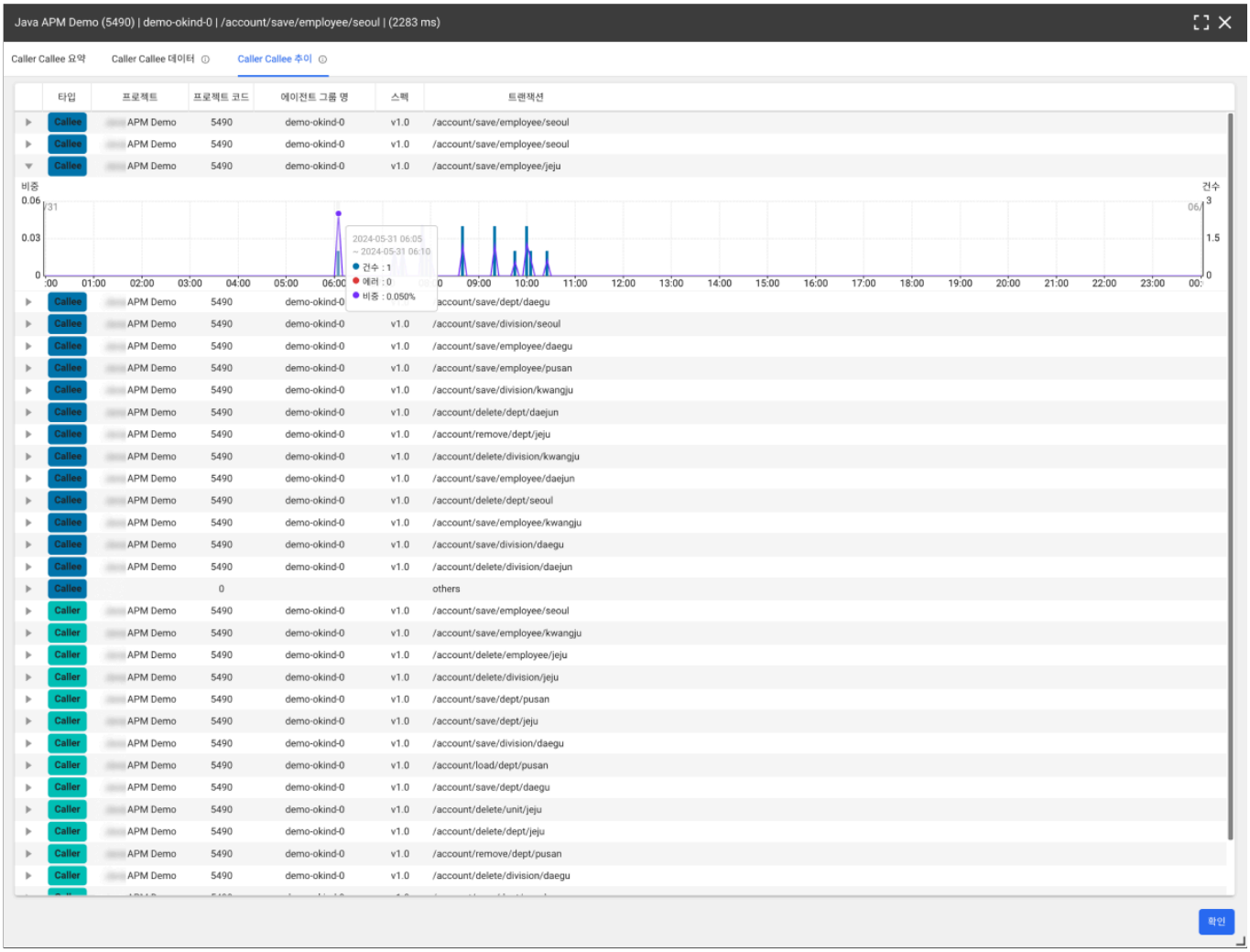


아이디	타입	프로젝트 코드	프로젝트	에이전트 그룹 명	트랜잭션	깊이	건수	에러	평균 응답 시간	비중	Parent ID
id-0--1485863373		5490	APM Demc	demo-okind-1	/account/save/employee/seoul	0	2938	5	2303		
id-caller1-964802511	Caller	5490	APM Demc	demo-okind-1	/account/save/dept/jeju	1	15	0	1156	0.0051	id-0--1485...
id-caller1--2119639160	Caller	5490	APM Demc	demo-okind-1	/account/delete/employee/jeju	1	12	0	1170	0.0041	id-0--1485...
id-caller1--703921046	Caller	5490	APM Demc	demo-okind-1	/account/delete/unit/pusan	1	11	0	1317	0.0037	id-0--1485...
id-caller1--339409101	Caller	5490	APM Demc	demo-okind-1	/account/save/employee/daegu	1	11	0	1197	0.0037	id-0--1485...
id-caller1--1166726544	Caller	5490	APM Demc	demo-okind-1	/account/save/dept/daegu	1	11	0	1163	0.0037	id-0--1485...
id-caller1--964898228	Caller	5490	APM Demc	demo-okind-1	/account/read/dept/pusan	1	10	0	1154	0.0034	id-0--1485...
id-caller1--1619760719	Caller	5490	APM Demc	demo-okind-1	/account/delete/dept/daejun	1	10	0	1156	0.0034	id-0--1485...
id-caller1--1344577787	Caller	5490	APM Demc	demo-okind-1	/account/remove/unit/seoul	1	10	0	1181	0.0034	id-0--1485...
id-caller1--1080234343	Caller	5490	APM Demc	demo-okind-1	/account/kill/unit/jeju	1	10	0	1153	0.0034	id-0--1485...
id-caller1--806644592	Caller	5490	APM Demc	demo-okind-1	/account/save/employee/daejun	1	9	0	1223	0.0031	id-0--1485...
id-caller1--324476759	Caller	5490	APM Demc	demo-okind-1	/account/kill/employee/jeju	1	9	0	1236	0.0031	id-0--1485...
id-caller1--1543262602	Caller	5490	APM Demc	demo-okind-1	/account/save/division/pusan	1	9	0	1175	0.0031	id-0--1485...
id-caller1--1080842845	Caller	5490	APM Demc	demo-okind-1	/account/remove/employee/jeju	1	9	0	1138	0.0031	id-0--1485...
id-caller1-1820299183	Caller	5490	APM Demc	demo-okind-1	/account/delete/dept/kwangju	1	9	0	1174	0.0031	id-0--1485...
id-caller1-1888852742	Caller	5490	APM Demc	demo-okind-1	/account/delete/unit/jeju	1	9	0	1166	0.0031	id-0--1485...
id-caller1-others	Caller	0			others	1	2383	0	1175	0.8111	id-0--1485...
id-callee1-1699366673	Callee	5490	APM Demc	demo-okind-1	/account/save/employee/pusan	1	14	0	1201	0.0048	id-0--1485...
id-callee1--1816628101	Callee	5490	APM Demc	demo-okind-1	/account/save/unit/daegu	1	12	0	1127	0.0041	id-0--1485...
id-callee1--322752839	Callee	5490	APM Demc	demo-okind-1	/account/load/unit/daejun	1	10	0	1174	0.0034	id-0--1485...
id-callee1--806644592	Callee	5490	APM Demc	demo-okind-1	/account/save/employee/daejun	1	9	0	1124	0.0031	id-0--1485...
id-callee1--339409101	Callee	5490	APM Demc	demo-okind-1	/account/save/employee/daegu	1	9	0	1207	0.0031	id-0--1485...
id-callee1--3777951	Callee	5490	APM Demc	demo-okind-1	/account/delete/dept/pusan	1	9	0	1174	0.0031	id-0--1485...
id-callee1--1080842845	Callee	5490	APM Demc	demo-okind-1	/account/remove/employee/jeju	1	9	0	1199	0.0031	id-0--1485...
id-callee1-1470942974	Callee	5490	APM Demc	demo-okind-1	/account/save/employee/jeju	1	9	0	1201	0.0031	id-0--1485...
id-callee1-2136474304	Callee	5490	APM Demc	demo-okind-1	/account/save/division/jeju	1	9	0	1192	0.0031	id-0--1485...
id-callee1-446621542	Callee	5490	APM Demc	demo-okind-1	/account/remove/employee/daejun	1	9	0	1202	0.0031	id-0--1485...
id-callee1-888826450	Callee	5490	APM Demc	demo-okind-1	/account/save/dept/pusan	1	9	0	1190	0.0031	id-0--1485...
id-callee1--703921046	Callee	5490	APM Demc	demo-okind-1	/account/delete/unit/pusan	1	8	0	1188	0.0027	id-0--1485...
id-callee1--479076859	Callee	5490	APM Demc	demo-okind-1	/account/remove/unit/daegu	1	8	0	1139	0.0027	id-0--1485...
id-callee1--153113744	Callee	5490	APM Demc	demo-okind-1	/account/save/dept/seoul	1	8	0	1155	0.0027	id-0--1485...
id-callee1--2045378290	Callee	5490	APM Demc	demo-okind-1	/account/remove/dept/seoul	1	8	0	1152	0.0027	id-0--1485...

- 각 트랜잭션의 호출 횟수와 비율, 오류 수, 평균 응답 시간 등을 확인하여 성능 문제를 분석하는데 활용할 수 있습니다.
- **타입** 컬럼을 통해서 Caller와 Callee를 쉽게 구분할 수 있으며 호출 관계를 파악할 수 있습니다.
- 각 트랜잭션의 ▶ 버튼을 선택하면 해당 트랜잭션과 관계된 트랜잭션을 추가로 확인할 수 있습니다.
- 데이터 목록을 CSV 형식으로 저장하려면 오른쪽 위에 **CSV 데이터 다운로드** 버튼을 선택하세요.

Caller Callee 추이

Caller Callee 추이 탭에서는 기준 URL(/account/save/employee/seoul)의 호출 관계와 성능 데이터를 시간의 흐름에 따라 시각적으로 분석할 수 있습니다. 이를 통해 트랜잭션의 성능 변동 패턴을 파악할 수 있습니다.



- 특정 트랜잭션의 호출 횟수와 비율을 시간대별 그래프로 시각화하여 성능 변화를 한눈에 파악할 수 있습니다.
- **타입** 컬럼을 통해서 Caller와 Callee를 쉽게 구분할 수 있으며 호출 관계를 파악할 수 있습니다.
- 시계열 차트에 마우스를 오버하면 툴팁을 통해서 수치 정보를 확인할 수 있습니다.
- 차트의 특정 시간대를 클릭하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 조회할 수 있습니다.

Java 메트릭스

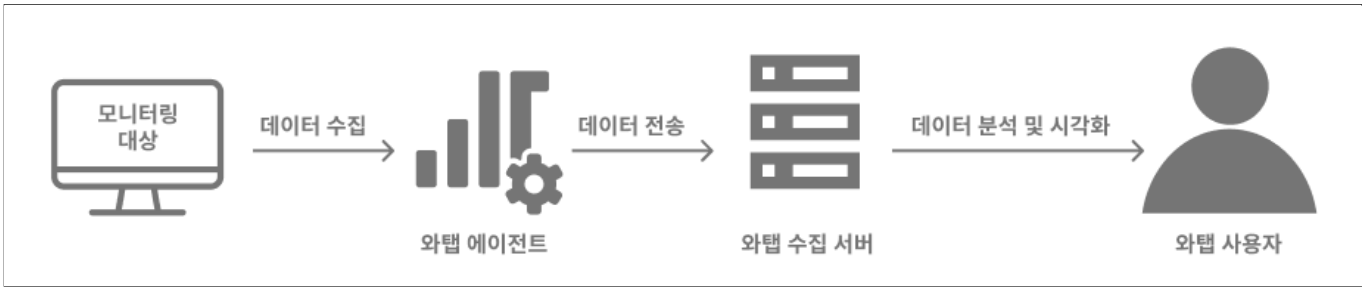
메트릭이란?

메트릭은 모니터링 대상의 상태를 수치로 표현한 시계열 데이터입니다. 와탭 에이전트가 모니터링 대상에서 수집한 성능 데이터를 가공한 결과입니다.

메트릭을 통해 서버별 메모리 사용률, DB 연결 시간 등 주요 성능 지표를 한눈에 파악할 수 있습니다. 문제를 발견한 후에는 로그와 트레이스를 통해 상세 분석할 수 있습니다.

자원 사용량 통계를 기반으로 필요 자원량을 산정할 수 있어, 비용 효율화에도 활용할 수 있습니다.

와탭의 메트릭 수집 방식



메트릭을 수집하려면 모니터링 대상에 와탭 에이전트를 설치해야 합니다. 에이전트가 전송한 메트릭은 와탭 수집 서버에 저장됩니다.

예를 들어 **Java 애플리케이션**을 모니터링하려면 먼저 에이전트를 설치하세요. 수집 가능한 메트릭은 애플리케이션 메트릭에서 확인할 수 있습니다.

와탭의 메트릭 구성 요소

와탭의 **메트릭**은 다음의 요소로 구성되어 있습니다.

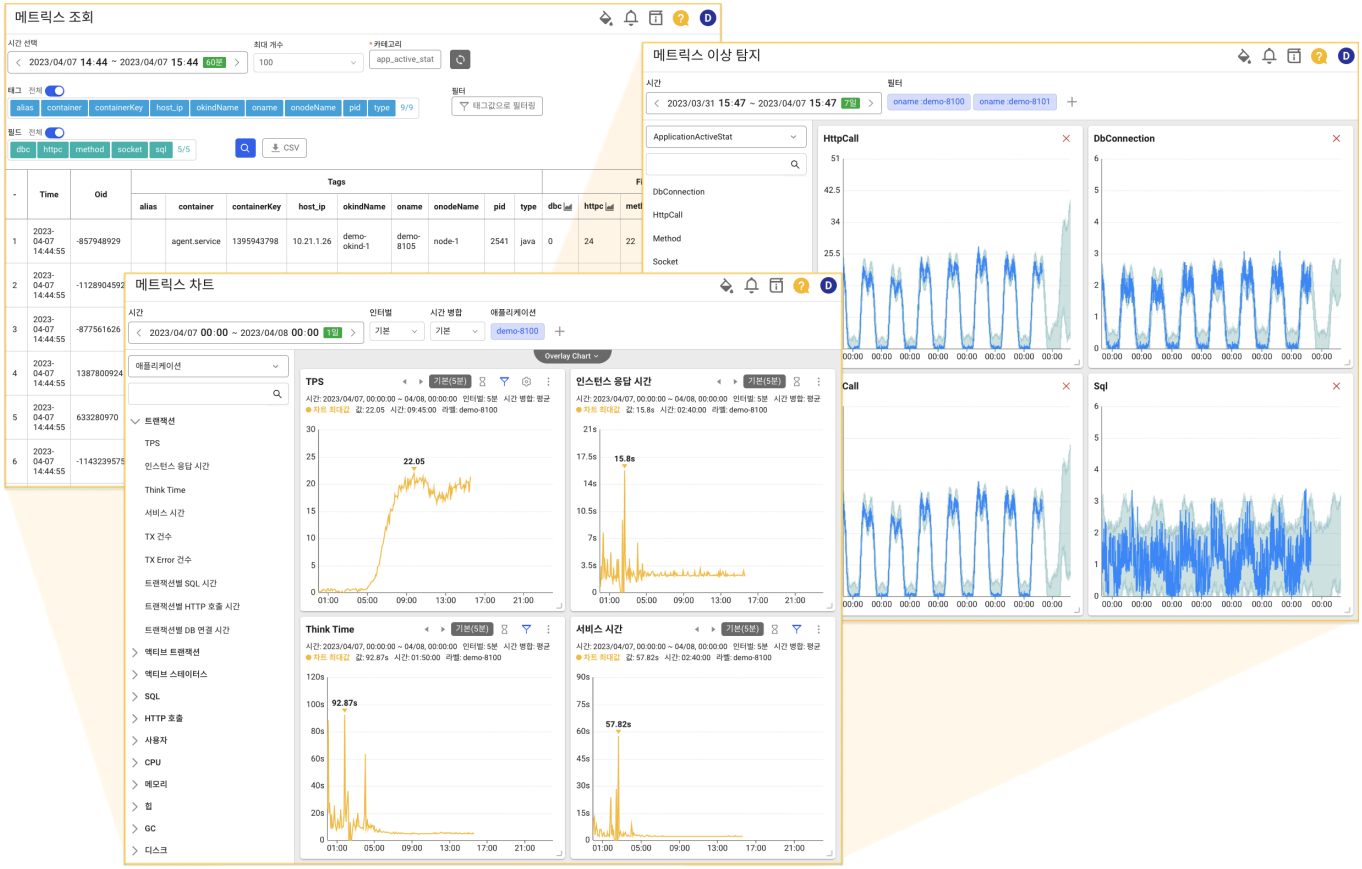
- **Category**: 관련 메트릭을 하나의 그룹으로 묶는 이름입니다.

- **Tags**: 수집 대상을 식별하는 라벨입니다. IP, Oname, Host 등 변경이 드문 고유 정보를 저장합니다. 하나의 메트릭에 여러 태그가 포함될 수 있습니다.
- **Fields**: 실제 측정값을 저장합니다. 하나의 메트릭에 여러 필드가 포함될 수 있습니다.
- **Time**: 메트릭을 수집한 시간입니다.
- **Oid**: 메트릭을 수집한 에이전트의 고유 번호입니다.
- **Oname**: 메트릭을 수집한 에이전트의 이름입니다.

메트릭 데이터 조회 및 시각화

와탭은 수집한 메트릭을 다양한 방식으로 조회할 수 있는 메뉴를 제공합니다.

- **메트릭스 조회**: 메트릭의 원본 데이터를 조회합니다.
- **메트릭스 차트**: 시각화한 차트를 통해 메트릭 데이터를 조회합니다.
- **메트릭스 이상 탐지**: AI가 학습한 메트릭 패턴과 비교해 예상 패턴을 벗어난 이상을 탐지합니다.


메트릭스 조회
메트릭스 이상 탐지
메트릭스 차트
TPS
인스턴스 응답 시간
Think Time
서비스 시간
HttpCall
DbConnection
Sql

애플리케이션 메트릭

홈 화면 > 프로젝트 선택 > **분석** > **메트릭스 조회**

메트릭스 조회 메뉴에서 검색할 수 있는 메트릭입니다. 메트릭은 아래와 같은 정보를 포함합니다. 일부 메트릭에는 태그 정보가 없을 수 있습니다.

- **Tags**: 수집 대상을 식별하는 정보 (에이전트 이름, IP 등)
- **Fields**: 실제 측정값 (CPU 사용률, 응답 시간 등)

---

agent_status_summary

- 에이전트 상태 관련 메트릭을 수집
- 수집 간격: 10초

Fields

Field	Type	Unit	Description
inActTime	-	밀리초(ms)	에이전트가 비활성화된 상태로 유지된 시간
isActive	Boolean	-	현재 에이전트의 활성 상태 여부
isRestart	Boolean	-	에이전트가 최근에 재시작되었는지 여부 (`true` / `false`)
lastActTime	-	밀리초(ms)	마지막으로 에이전트가 활성화된 상태의 시각 (`0`: 비활성화된 경우)
oid	-	-	프로젝트에 포함된 각 에이전트의 고유 식별자
startTime	-	밀리초(ms)	에이전트가 시작된 시점의 타임스탬프

app_active_stat

- 액티브 트랜잭션 구간(Step) 관련 메트릭을 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
container	-	-	컨테이너 이름 (고유값)
containerKey	-	-	컨테이너 키값 (고유값)
host_ip	-	-	호스트 IP 주소 (고유값)
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	-	에이전트 이름 (고유값)
onodeName	-	-	whatap.onode로 분류된 그룹 이름
pid	-	-	프로세스의 고유 식별자(ID)
type	-	-	플랫폼 유형 (애플리케이션 개발 언어)

Fields

Field	Type	Unit	Description
dbc	-	건수	Database Connection을 수행 중인 트랜잭션 건수
httpc	-	건수	HTTP Call을 수행 중인 트랜잭션 건수

Field	Type	Unit	Description
method	-	건수	Method 로직 수행 중인 트랜잭션 건수
socket	-	건수	Socket 연결 수행 중인 트랜잭션 건수
sql	-	건수	SQL 쿼리를 수행 중인 트랜잭션 건수

---

app_actx_meter

- 액티브 트랜잭션의 HTTP 요청, SQL 쿼리, 데이터 전송 등 다양한 활동 메트릭을 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
alias	-	-	에이전트 별칭
container	-	-	컨테이너 이름
containerKey	-	-	컨테이너 키값
host_ip	-	-	호스트 IP 주소
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	-	에이전트 이름
onodeName	-	-	whatap.onode로 분류된 그룹 이름

Fields

Field	Type	Unit	Description
active_httpc	-	건수	활성 HTTP 클라이언트 요청 수
active_sql	-	건수	활성 SQL 쿼리 수
data	-	byte	데이터 전송량
txcaller_group	-	Int	트랜잭션 호출자의 그룹을 식별하기 위한 상수
txcaller_oid	-	Int	트랜잭션 호출자의 에이전트 ID를 식별하기 위한 상수
txcaller_unknown	-	Int	알 수 없는 트랜잭션 호출자를 식별하기 위한 상수

---

app_counter

- 애플리케이션의 다양한 성능 메트릭 및 통계를 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
alias	-	-	애플리케이션 별칭 (고유값)
container	-	-	컨테이너 이름 (고유값)
containerKey	-	-	컨테이너 키 값 (고유값)
host_ip	-	-	호스트 IP 주소 (고유값)

Tag	Type	Unit	Description
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	-	에이전트 이름 (고유값)
onodeName	-	-	`whatap.onode`로 분류된 그룹 이름
pid	-	-	프로세스의 고유 식별자 (ID)
type	-	-	플랫폼 유형 (애플리케이션 개발 언어)

Fields

Field	Type	Unit	Description
active_tx_0	-	건수	3초 이하 구간 트랜잭션 수
active_tx_3	-	건수	3초 초과 8초 이하 구간 트랜잭션 수
active_tx_8	-	건수	8초 초과 구간 트랜잭션 수
active_tx_count	-	건수	수행 중인 전체 트랜잭션 수
apdex_satisfied	-	건수	APDEX 만족 수
apdex_tolerated	-	건수	APDEX 허용 수
apdex_total	-	건수	APDEX 트랜잭션 총 수
arrival_rate	-	퍼센트	트랜잭션 요청률
httpc_count	-	건수	HTTP 외부 호출 수
httpc_error	-	건수	HTTP 외부 호출 에러 수
httpc_time	-	밀리초(ms)	HTTP 외부 호출 평균 시간

Field	Type	Unit	Description
metering	-	코어	애플리케이션이 작동 중인 호스트의 코어 수 (컨테이너의 경우 limit cpu)
resp_time	-	밀리초(ms)	평균 응답 시간
sql_count	-	건수	실행 완료된 SQL 건수
sql_error	-	건수	SQL 에러 건수
sql_fetch_count	-	건수	SQL Fetch 건수
sql_fetch_time	-	밀리초(ms)	SQL Fetch 수행 시간
sql_time	-	밀리초(ms)	SQL 평균 수행 시간
tps	-	건수	초당 트랜잭션 처리 수
tx_count	-	건수	트랜잭션 처리 건 수
tx_dbc_time	-	밀리초(ms)	DB 평균 연결 시간
tx_error	-	건수	트랜잭션 에러 건수
tx_httpc_time	-	밀리초(ms)	HTTP 호출 평균 시간
tx_sql_time	-	밀리초(ms)	트랜잭션별 SQL 수행시간의 합에 대한 평균
tx_time	-	밀리초(ms)	트랜잭션 수행 시간
tx_time_sqr_sum	-	건수	트랜잭션 수행 시간의 제곱합, 트랜잭션 수행 시간의 분산이나 표준 편차와 같은 통계적 메트릭을 구할 때 사용

app_counter_okind

- `app_counter` 메트릭을 `whatap.okind` 그룹 기준으로 집계
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
okindName	-	-	whatap.okind로 분류된 그룹 이름

Fields

Field	Type	Unit	Description
active_tx_0	-	건수	3초 이하 구간 트랜잭션 수
active_tx_3	-	건수	3초 초과 8초 이하 구간 트랜잭션 수
active_tx_8	-	건수	8초 초과 구간 트랜잭션 수
active_tx_count	-	건수	수행 중인 전체 트랜잭션 수
apdex	-	number	APDEX(애플리케이션 고객 만족도)
apdex100	-	퍼센트	apdex 필드의 백분율
apdex_satisfied	-	건수	APDEX 만족 수
apdex_tolerated	-	건수	APDEX 허용 수
apdex_total	-	건수	APDEX 트랜잭션 총 수

Field	Type	Unit	Description
arrival_rate	-	퍼센트	트랜잭션 요청률
num_of_agent	-	개수	그룹 내 활성화된 에이전트 수
realtime_user	-	건수	동시 접속 사용자 수
sql_count	-	건수	실행 완료된 SQL 건수
sql_error	-	건수	SQL 에러 건수
sql_fetch_count	-	건수	SQL Fetch 건수
sql_fetch_time	-	밀리초(ms)	SQL Fetch 수행 시간
sql_time	-	밀리초(ms)	SQL 평균 수행 시간
tps	-	건수	초당 트랜잭션 처리 수
tx_count	-	건수	트랜잭션 처리 건 수
tx_dbc_time	-	밀리초(ms)	DB 평균 연결 시간
tx_error	-	건수	트랜잭션 에러 건수
tx_httpc_time	-	밀리초(ms)	HTTP 호출 평균 시간
tx_sql_time	-	밀리초(ms)	트랜잭션별 SQL 수행시간의 합에 대한 평균
tx_time	-	밀리초(ms)	트랜잭션 수행 시간

---

app_counter_project

- `app_counter` 메트릭을 프로젝트 기준으로 집계

- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Fields

Field	Type	Unit	Description
active_tx_0	-	건수	3초 이하 구간 트랜잭션 수
active_tx_3	-	건수	3초 초과 8초 이하 구간 트랜잭션 수
active_tx_8	-	건수	8초 초과 구간 트랜잭션 수
active_tx_count	-	건수	수행 중인 전체 트랜잭션 수
apdex	-	number	APDEX(애플리케이션 고객 만족도)
apdex100	-	퍼센트	apdex 필드의 백분율
apdex_satisfied	-	건수	APDEX 만족 수
apdex_tolerated	-	건수	APDEX 허용 수
apdex_total	-	건수	APDEX 트랜잭션 총 수
arrival_rate	-	퍼센트	트랜잭션 요청률
num_of_agent	-	개수	프로젝트 내 활성화된 에이전트 수
realtime_user	-	건수	동시 접속 사용자 수
sql_count	-	건수	실행 완료된 SQL 건수
sql_error	-	건수	SQL 에러 건수
sql_fetch_count	-	건수	SQL Fetch 건수

Field	Type	Unit	Description
sql_fetch_time	-	밀리초(ms)	SQL Fetch 수행 시간
sql_time	-	밀리초(ms)	SQL 평균 수행 시간
tps	-	건수	초당 트랜잭션 처리 수
tx_count	-	건수	트랜잭션 처리 건 수
tx_dbc_time	-	밀리초(ms)	DB 평균 연결 시간
tx_error	-	건수	트랜잭션 에러 건수
tx_httpc_time	-	밀리초(ms)	HTTP 호출 평균 시간
tx_sql_time	-	밀리초(ms)	트랜잭션별 SQL 수행시간의 합에 대한 평균
tx_time	-	밀리초(ms)	트랜잭션 수행 시간

---

app_host_resource

- 애플리케이션 서버의 리소스 메트릭을 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
alias	-	-	애플리케이션 별칭
container	-	-	컨테이너 이름

Tag	Type	Unit	Description
containerKey	-	-	컨테이너 키값
host_ip	-	-	호스트 IP 주소
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	-	에이전트 이름
onodeName	-	-	whatap.onode로 분류된 그룹 이름
pid	-	-	특정프로세스의 고유 식별자(ID)
type	-	-	플랫폼 유형(애플리케이션 개발 언어)

Fields

Field	Type	Unit	Description
cpu	-	percent	호스트 CPU 사용률
cpu_cores	-	정수	호스트 CPU 코어 수
cpu_irq	-	percent	호스트 CPU IRQ 사용률
cpu_proc	-	percent	자바 프로세스 CPU 사용률
cpu_steal	-	percent	호스트 CPU Steal 사용률
cpu_sys	-	percent	호스트 CPU SYS 사용률
cpu_usr	-	percent	호스트 CPU USER 사용률
cpu_wait	-	percent	호스트 CPU WAIT 사용률
disk	-	percent	루트 파일시스템 디스크 사용률 (JAVA는 설정으로 지정한 파일시스템 경로의 디스크 사용률)

Field	Type	Unit	Description
mem	-	percent	호스트 메모리 사용률
swap	-	percent	호스트 SWAP 사용률

---

app_remote_active_stat

- DB 연결과 HTTP Call의 응답 시간 범위별 트랜잭션 수를 측정
- 수집 간격: 5초

Tags

Tag	Type	Unit	Description
alias	-	고유값	애플리케이션 별칭
container	-	고유값	컨테이너 이름
host_ip	-	고유값	호스트 IP 주소
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	고유값	에이전트 이름
onodeName	-	-	whatap.onode로 분류된 그룹 이름

Fields

Field	Type	Unit	Description
act0	-	count	응답 시간이 3초 이내인 액티브 트랜잭션의 개수

Field	Type	Unit	Description
act3	-	count	응답 시간이 3초 이상 8초 미만인 액티브 트랜잭션의 개수
act8	-	count	응답 시간이 8초 이상인 액티브 트랜잭션의 개수
actx	-	count	act0, act3, act8의 합계
count	-	count	모니터링 기간 동안 수집한 액티브 트랜잭션의 총 수
error	-	count	에러가 발생한 트랜잭션의 수
id	Int32	-	액티브 트랜잭션을 식별하기 위한 고유 ID
timeSum	number	-	액티브 트랜잭션의 응답 시간 합계
type	string	-	db(DB 연결) 또는 httpc(HTTP Call)

---

app_user

- 애플리케이션 사용자 관련 메트릭을 수집
- 수집 간격: 10초
- 통계 데이터: 5분

Tags

Tag	Type	Unit	Description
host_ip	-	-	호스트 IP 주소 (고유값)
oname	-	-	에이전트 이름 (고유값)
okindName	-	-	whatap.okind로 분류된 그룹 이름

Tag	Type	Unit	Description
onodeName	-	-	whatap.onode로 분류된 그룹 이름

Fields

Field	Type	Unit	Description
logbits	-	byte	사용자 로그 비트 정보(사용자 활동 이벤트로 발생한 데이터 단위)
realtime_user	-	count	동시 접속 사용자 수

---

db_pool_detail

- 데이터베이스 연결 풀의 상세 상태를 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
$okind	-	-	whatap.okind로 분류된 그룹 이름
$onode	-	-	whatap.onode로 분류된 그룹 이름
oname	-	-	에이전트 이름
pool	-	-	데이터베이스 풀(Pool)의 유형
class	-	-	데이터베이스 풀링(Pooling) 클래스

Fields

Field	Type	Unit	Description
act	-	count	현재 활성 상태의 데이터베이스 연결 수
idle	-	count	현재 유휴 상태의 데이터베이스 연결 수

---

event_5sec

- 5초마다 발생하는 이벤트를 수집
- 수집 간격: 5초

Tags

Tag	Type	Unit	Description
type	-	-	이벤트 유형
no_cache	-	-	이벤트 정보의 캐시 유무

Fields

Field	Type	Unit	Description
curtime	-	밀리초(ms)	이벤트 발생 시간(UNIX epoch time, millisecond)

event_cube_done

- 큐브 작업 완료 시 발생하는 이벤트 정보를 수집
- 수집 간격: 5초

Tags

Tag	Type	Unit	Description
type	-	-	이벤트 유형
no_cache	-	-	이벤트 정보의 캐시 유무

Fields

Field	Type	Unit	Description
cubetime	-	밀리초(ms)	큐브 작업이 완료된 시간(UNIX epoch time, millisecond)
date	-	date	cubetime을 yyyymmdd 형식으로 변환
date9	-	date	date 값을 GMT+9 시간대로 변환
hhmmss	-	time	cubetime을 hhmmss 형식으로 변환
hhmmss9	-	time	hhmmss 값을 GMT+9 시간대로 변환

visitor

- 애플리케이션 사용자 통계를 수집
- 수집 간격: 1시간

- 통계 데이터: 1시간

Tags

Tag	Type	Unit	Description
host_ip	-	고유값	호스트 IP 주소
oname	-	고유값	에이전트 이름
okindName	-	-	whatap.okind로 분류된 그룹 이름

Fields

Field	Type	Unit	Description
logbits	-	byte	사용자 로그 비트 정보(사용자 활동 이벤트로 발생한 데이터 단위)
visit_user	-	count	조회한 기간 동안의 방문자 수

---

app_proc_counter

- 애플리케이션 프로세스 메트릭을 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
alias	-	-	애플리케이션 별칭 (고유값)

Tag	Type	Unit	Description
container	-	-	컨테이너 이름 (고유값)
containerKey	-	-	컨테이너 키값 (고유값)
host_ip	-	-	호스트 IP 주소 (고유값)
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	-	에이전트 이름 (고유값)
onodeName	-	-	whatap.onode로 분류된 그룹 이름
pid	-	-	특정프로세스의 고유 식별자(ID)
type	-	-	플랫폼 유형(애플리케이션 개발 언어)

Fields

Field	Type	Unit	Description
cputime	-	밀리초(ms)	트랜잭션 cpu 시간
db_num_active	-	count	DB Connection Pool Active 수
db_num_idle	-	count	DB Connection Pool Idle 수
gc_count	-	count	GC 발생 횟수
gc_oldgen_count	-	count	Old Generation GC 건수
gc_time	-	밀리초(ms)	GC 수행 시간
heap_max	-	byte	힙 메모리의 최대 크기(-Xmx 값)
heap_perm	-	byte	Perm 영역의 메모리 사용량

Field	Type	Unit	Description
heap_tot	-	byte	현재 할당된 힙 메모리의 총량
heap_use	-	byte	현재 사용 중인 힙 메모리의 양
pending_finalization	-	count	GC 중 finalize 수행을 위해 대기 중인 객체 수
proc_fd	-	count	프로세스가 열고 있는 파일 디스크립터(file descriptors)의 수
proc_fd_max	-	count	프로세스가 열 수 있는 최대 파일 디스크립터 수
thread_count	-	count	JVM에서 실행 중인 스레드의 총 수
thread_daemon	-	count	JVM에서 실행 중인 데몬 스레드의 수
thread_peak_count	-	count	JVM이 실행되는 동안 관찰된 스레드 수의 최댓값
thread_total_started	-	count	JVM이 시작된 이후 생성된 총 스레드 수
threadpool_activeCount	-	count	현재 실행 중인 스레드 풀의 스레드 수

---

java_memory

- JVM 메모리 사용 상태 메트릭을 수집
- 수집 간격: 5초
- 통계 데이터: 5분, 1시간

Tags

Tag	Type	Unit	Description
host_ip	-	-	호스트 IP 주소 (고유값)

Tag	Type	Unit	Description
okindName	-	-	whatap.okind로 분류된 그룹 이름
oname	-	-	에이전트 이름 (고유값)
onodeName	-	-	whatap.onode로 분류된 그룹 이름
pid	-	-	특정프로세스의 고유 식별자(ID)

Fields

Field	Type	Unit	Description
nonheap_committed	-	byte	JVM에서 사용하기 위해 보장된 비 힙 메모리 크기
nonheap_init	-	byte	JVM이 시작 시 OS로부터 요청하는 초기 비 힙 메모리 크기
nonheap_max	-	byte	메모리 관리에 사용할 수 있는 최대 비 힙 메모리 크기
nonheap_used	-	byte	현재 사용 중인 비 힙 메모리의 양
objectPendingFinalizationCount	-	count	GC 중 finalize 수행을 위해 대기 중인 객체의 수

성능 카운터

와탭 에이전트는 애플리케이션 성능과 관련된 다양한 정보를 수집합니다. 크게 3가지로 분류할 수 있습니다.

- **User** : 실시간 사용자 혹은 방문 사용자
- **Service** : 트랜잭션, SQL, 외부 호출 건수 및 응답, 에러율 등
- **Resource** : 시스템, 프로세스 자원 사용량

User Counter

사용자는 모니터링 대상 시스템을 사용하는 클라이언트를 말합니다. 클라이언트에서는 일반적으로는 브라우저를 기준으로 사용자 수를 계산합니다.

웹 시스템 성능에서 사용자는 부하를 발생시키는 시작이기 때문에 중요합니다. 사용자 추적을 위해서는 사용자는 어떤 기준으로 구분하고, 어떻게 카운팅 할지에 대한 고려가 필요합니다.

사용자 구분

와탭 에이전트 사용자를 구분하기 위해 다양한 옵션을 제공합니다.

- **Remote IP**

  기본값은 remote ip를 사용하여 사용자를 구분합니다. remote ip는 실제 사용자를 구분하는 데 한계가 있습니다.

- **Cookie**

  쿠키를 사용하여 사용자를 구분합니다. 모든 접속 클라이언트를 대상으로 **WHATAP**이라는 쿠키에 UUID를 저장합니다.

  whatap.conf

  ```
  trace_user_using_ip=false

  # Java agent v2.2.0 or later
  wclient_using_ip=false
  ```

- **Header Key**

HTTP 헤더에 전달되는 값으로 사용자를 구분할 수 있습니다.

whatap.conf

```
user_header_ticket=USER

# Java agent v2.2.0 or later
wclient_header_ticket=USER
```

사용자 카운팅

사용자를 카운팅 하는 방법에 따라서 다르게 사용합니다. 실시간 사용자는 현재 시스템을 사용하는 사용자의 수를 알기 위해서 측정합니다. 일일 방문 사용자는 하루 동안 해당 서비스에 관심을 갖는 사용자가 몇 명인지에 대한 비즈니스적인 관리를 위해 측정합니다.

- **실시간 사용자**

  최근 5분 동안 사용자 수를 카운팅 합니다. 5초마다 shifting 하면 사용자를 카운팅 합니다. 각 서버에서 카운팅 된 숫자는 HyperLogLog 알고리즘을 통해서 머지 됩니다.

- **일일 방문자(DAU**, Daily Active User)

  하루 동안 시스템에 접속한 사용자를 카운팅 합니다. 24시간 동안 접속한 사용자를 HyperLogLog를 통해서 계산합니다.

> 와탭에서는 장기간 사용자를 카운팅 하기 위해 사용자 데이터에 대한 byte block을 서버로 수집합니다. 이 데이터를 Hyperloglog로 머지하면 이론적으로 한 달 이상의 방문 사용자를 계산할 수 있습니다.

Service Counter

트랜잭션과 트랜잭션이 사용하는 SQL 혹은 외부 호출 등에 대한 건수, 응답시간 에러 건수 등에 대한 성능지표가 포함됩니다.

- **Transaction Counter**

  트랜잭션을 수행하면 측정하는 카운터입니다.

  - **건수**

- **응답 시간**
- **에러 건수**

- **Active Transaction Counter**

  진행 중인 트랜잭션의 수를 카운팅 합니다.

  - **건수**
  - **Active Status**

    진행 상태는 METHOD, SQL, HTTPC, DBC, SOCKET 5가지 상태로 고정되어 있습니다.

    - METHOD - 일반 함수를 호출하는 상태
    - SQL - db sql을 수행 중인 상태
    - HTTPC - 외부 Http Api(서비스)를 호출 중인 상태
    - DBC - DB 연결을 요청한 상태, 일반적으로 Pool에서 가져옴
    - SOCKET - TCP 세션을 Connecting 중인 상태

- **SQL**

  SQL 수행 현황을 카운팅 합니다.

  - **건수**
  - **응답 시간**
  - **에러 건수**
  - **패치 건수**

- **HTTP Call**

  외부 Http 호출에 대한 현황을 카운팅 합니다.

  - **건수**
  - **응답 시간**
  - **에러 건수**

Resource Counter

서버 자원 혹은 node 프로세스 내부의 자원 사용량을 카운팅 합니다.

- **CPU** (sys, usr, wait, steal, irq, cores)

  CPU 사용량 %입니다. 각 종류별로 수집됩니다. 가상환경에서만 Steal이 의미가 있습니다. Cpu Core 개수를 같이 수집하고 있습니다.

- **Process CPU**

  자바 프로세스가 사용하는 CPU%입니다.

- **Memory**

  시스템 메모리 사용률(%)입니다.

- **Swap**

  Swap 메모리 사용률(%)입니다.

- **Disk**

  Disk는 Java Process의 Current 디렉터리의 사용률(%)입니다.

- **Heap** (Total, Used, Perm)

  Java Heap 메모리의 Total, Used, Perm 양입니다. 데이터 단위는 KBytes 입니다.

- **GC** (Count, Time)

  5초 동안 발생한 GC 건수와 시간의 합계입니다.

- **ObjectPendingFinalizationCount**

  GC 호출되는 도중에 finalize() 수행하기 위해 대기 중인 객체 숫자입니다. 이 값이 커지면 GC time이 지연될 수 있습니다.

- **Thread** (start Count, Count, Daemon, Peak Count)

  JVM이 실행된 이후부터 시작된 스레드 수, 현재 돌고 있는 스레드 수, 데몬, 최대 스레드 수를 수집합니다. 스레드 풀이 필요한지에 대한 판단 등을 할 수 있습니다.

- **DB Connection Count** (active,idle)

  Connection Pool의 idle와 active 카운트를 수집합니다.

- **Tomcat Thread Pool** (active, queueSize)

  톰캣의 Executor 스레드 풀이 설정되었을 때 성능 정보를 수집합니다.

  XML

```
<Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
maxThreads="50" minSpareThreads="2" maxQueueSize="78"/>
```

DB Connection Pool 카운터

DB Connection Pool은 DataSource라고 합니다. WAS가 제공하는 경우가 일반적이지만 일부에서는 오픈소스 Pool을 사용하기도 합니다. 사용량 정보가 JMX로 노출되기도 하지만 그렇지 않은 경우도 많습니다.

와탭은 2가지 방식을 제공하고 있습니다. 하나의 BCI에 의한 직접 조회 방식과 JMX를 이용한 방식입니다. 기본은 BCI 방식을 사용하고 보조로 JMX를 사용할 수 있도록 제공하고 있습니다.

다음은 개별 옵션별 디폴트 값과 연관되어 ByteCode Injection되는 클래스입니다.

hikari_pool_enabled=false

```
com.zaxxer.hikari.pool.HikariPool
```

dbcp_pool_enabled=true

```
org.apache.commons.dbcp.BasicDataSource
org.apache.commons.dbcp2.BasicDataSource
org.apache.tomcat.dbcp.dbcp.BasicDataSource
org.apache.tomcat.dbcp.dbcp2.BasicDataSource
```

tomcat_pool_enabled=true

```
org.apache.tomcat.jdbc.pool.ConnectionPool
```

weblogic_pool_enabled=true

```
weblogic.jdbc.common.internal.ConnectionPool
```

jeus_pool_enabled=true

```
jeus.jdbc.connectionpool.ConnectionPoolImpl
```

jboss_pool_enabled=true

```
org.jboss.jca.core.connectionmanager.pool.PoolStatisticsImpl
```

> 옵션을 변경한다면 BCI 관련 옵션이기 때문에 **다시 시작**해야 합니다. 클래스가 instrument된 후에도 추적 테이블에 등록되는 과정이 필요합니다. 따라서 실행 중에 redefine을 해도 동작하지 않습니다.

DataSource 현황 JMX로 가져오기

```
dbcp_pool_enabled==true 이면 tomcat_ds_enabled=false
weblogic_pool_enabled== true 이면 weblogic_ds_enabled=false
```

1. **옵션 자동 결정** 내용을 확인하세요.
2. 관련된 pool 쪽 옵션 값을 `false` 로 선언하세요.
3. DataSource 관련 옵션 값을 `true` 로 변경하세요. 기본값은 `false` 입니다.

whatap.conf

```
tomcat_ds_enabled=false
weblogic_ds_enabled=false
```

ThreadPool 카운터

ThreadPool 카운터는 JMX Mbean 정보를 통해 수집됩니다. Tomcat과 같이 사용 사례가 많은 경우라면 문제없지만 특정 환경에서는 이 지표가 수집되지 않을 수 있습니다.

수집을 위해서 추가 설정이 필요합니다. 관련 설정은 다음과 같습니다.

whatap.conf

```
jmx_threadpool_objectname=Catalina:type=ThreadPool,name="http-bio-8080"
// JMX ThreadPool MBean 이름을 설정합니다.
jmx_threadpool_atter_activecount=connectionCount
```

```
// activecount 수치를 나타내는 attribute를 설정합니다.
jmx_threadpool_atter_queuesize=maxThreads
// maxThreads 수치를 나타내는 attribute를 설정합니다.
```

JMX ThreadPool ObjectName 및 AttributeName 과 같은 Mbean 명은 JMX 모니터링 도구를 통해 확인하거나 각 WAS 별 Mbean 문서를 참고합니다.

> ⓘ 사용 중인 WAS, 프레임워크 관련 문서를 참고해 Mbean 정보를 확인하세요.

Jconsole 사용 시

- **ObjectName** : Catalina

  type : ThreadPool

  name : "http-bio-8080"

- **ActiveName** : connectionCount

  QueueName : maxThreads

결과 확인

설정이 정상적으로 되면 카운터에서 다음과 같이 데이터를 조회할 수 있습니다.

성능 카운터 - 정상 상태

Apdex

Apdex(Appliccation Performance Index)는 개방형 표준을 따르는 애플리케이션 성능지표입니다. Apdex는 응답시간에 기반하며 전체 요청 중 만족과 허용 건 비율로 수치화합니다. 대시보드에 Apdex 그래프가 추가되었습니다.

Apdex는 사용자 만족도에 대한 지표로 활용할 수 있으며, 0 ~ 1 사이의 값을 갖습니다.

(만족 횟수 + (허용 횟수 * 0.5)) / 전체 요청 수

상태	설명
만족 (Satisfied, **S**)	업무처리에 전혀 문제가 없음 ≤ 1.2초 (만족 **S** 기본값)
허용 (Tolerating, **T**)	사용자가 지연을 느끼나 업무처리는 가능함 ≤ 4.8초 (만족 **S** * 4)
불만 (Frustrated, **F**)	업무처리가 불가능함 > 4.8초 (허용 **T** 초과 및 오류)

- **whatap.apdex_time** millisecond

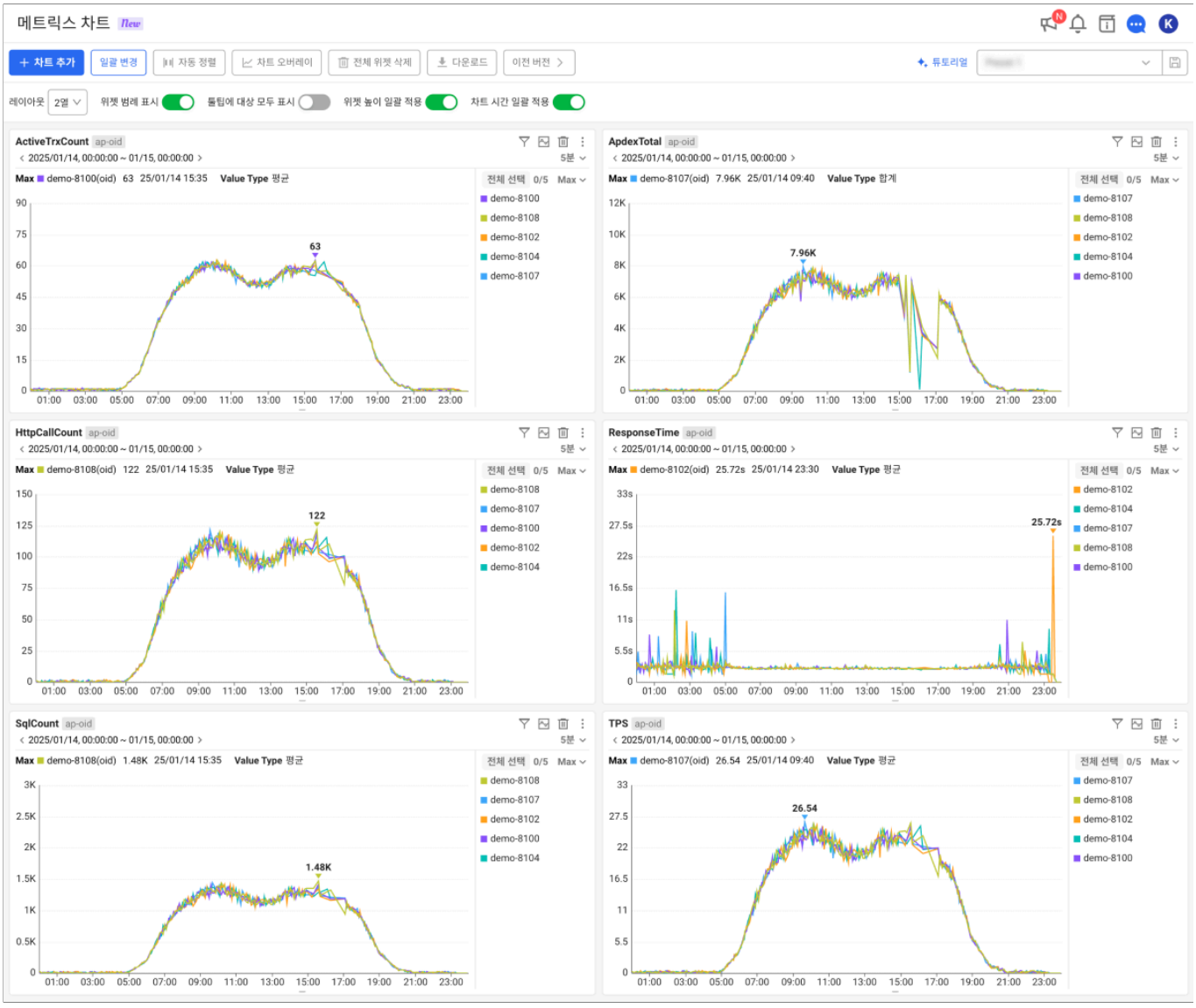
  기본값 `1200`

  만족 **S** 기본값은 에이전트 설정 메뉴에서 변경할 수 있습니다.

메트릭 차트 v2

홈 화면 > 프로젝트 선택 > **분석** > **메트릭스 차트** *New*

메트릭스 차트 *New* 메뉴는 와탭 모니터링 솔루션의 핵심 기능으로, 다양한 모니터링 대상으로부터 수집한 데이터를 시계열 차트로 시각화하여 제공합니다. 이를 통해 사용자는 시스템 성능을 효과적으로 모니터링하고 이상 징후를 빠르게 식별하여 문제 해결에 필요한 인사이트를 얻을 수 있습니다.



- 다양한 지표를 시계열 차트를 통해서 시각적으로 확인할 수 있으며, 장애 발생 시점의 여러 지표를 한번에 분석할 수 있습니다.
- 특정 시간 범위를 선택하여 과거 데이터를 조회하고, 해당 기간의 성능을 분석할 수 있습니다.
- 특정 시간대나 조건에 맞는 데이터를 필터링하여 보다 정확한 분석을 수행할 수 있습니다.
- 사용자는 원하는 차트를 화면에 배치하고 레이아웃을 자유롭게 조정할 수 있습니다.
- 사용자가 원하는 지표들을 배치하고 프리셋으로 저장하여 필요시 불러올 수 있습니다.

> ⓘ **기존 메트릭스 차트 메뉴와의 차이점**
>
> Service 2.8.0 업데이트 이후 변경 사항은 다음과 같습니다. 업데이트 이후 변경된 사항을 화면에서 확인하려면 오른쪽 상단의 **튜토리얼** 버튼을 선택하세요.
>
> - 사용자가 시간, 대상, 인터벌 옵션을 설정한 다음 모니터링하려는 대상의 지표를 특정하고 차트를 추가할 수 있도록 사용자 경험(UX)을 개선했습니다.
> - 대시보드에 배치한 위젯을 삭제하지 않고 차트 조회 시간 범위를 변경할 수 있습니다.
> - 대시보드에 배치한 모든 위젯의 옵션을 일괄로 적용할 수 있는 **일괄 변경** 기능을 제공합니다.
> - 프리셋 기능을 통해 대시보드에 배치한 위젯의 정보를 저장하고 불러올 수 있습니다.
> - 제공하는 다양한 옵션을 통해 대시보드의 레이아웃 및 위젯의 기능을 조정할 수 있습니다.

차트 위젯 추가하기

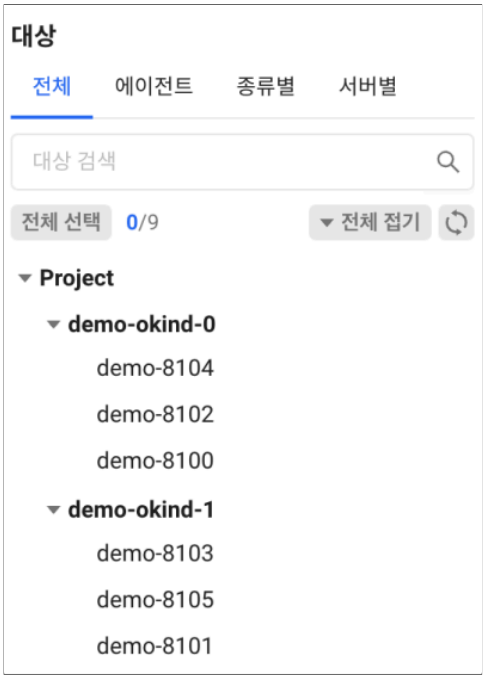
대시보드에 지표를 위젯으로 추가하려면 화면 왼쪽 상단의 **차트 추가** 버튼을 선택하세요.

조회 시간 설정하기

시간에서 지표를 조회할 날짜와 시간을 설정하세요.

대상 선택하기

프로젝트에 포함된 에이전트 중 원하는 대상을 선택하세요. 개별 에이전트를 선택하거나 에이전트 설정을 통해 분류한 그룹 단위로 선택할 수 있습니다.

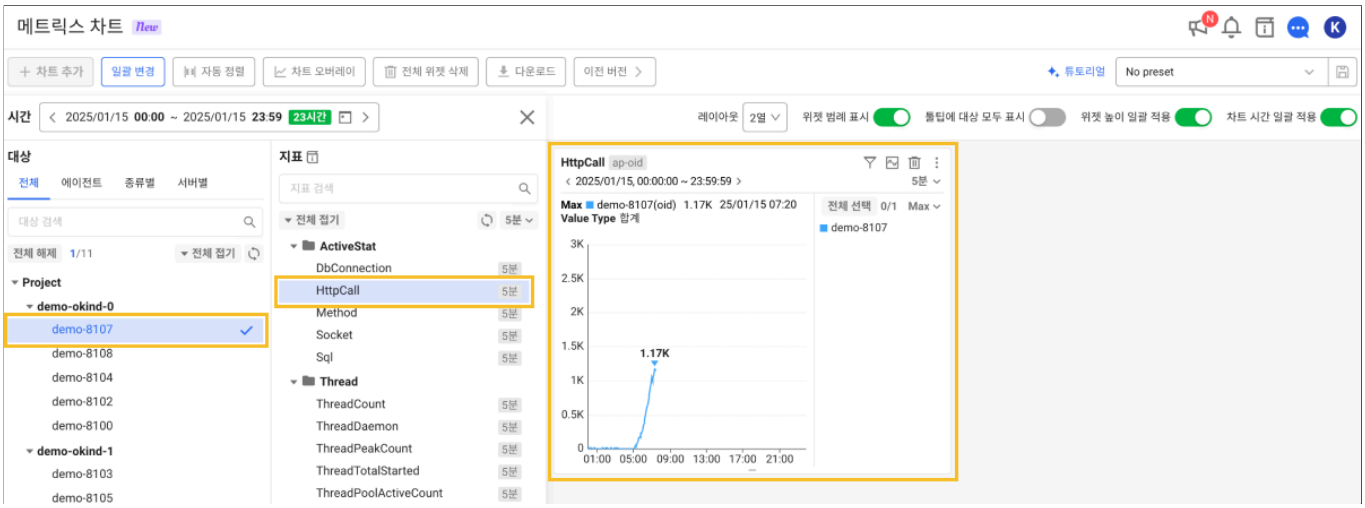


- **에이전트**: 개별 에이전트를 선택하거나 모두 선택할 수 있습니다.
- **종류별**: 에이전트 설정에서 `whatap.okind` 로 분류된 그룹 단위로 에이전트를 모니터링합니다.
- **서버별**: 에이전트 설정에서 `whatap.onode` 로 분류된 그룹 단위로 에이전트를 모니터링합니다.

> ⓘ
> - 설정한 시간과 날짜에 따라 선택할 수 있는 대상이 변경될 수 있습니다.
> - 에이전트 목록에서 마우스 클릭해 대상을 선택하거나 해제할 수 있습니다. `Shift` 키를 이용해 여러 개의 에이전트를 한번에 선택할 수도 있습니다.
> - **전체 선택** 또는 **전체 해제** 버튼을 선택하면 관련 하위 대상들을 한번에 선택 또는 해제할 수 있습니다.
> - 원하는 에이전트가 목록에 표시되지 않는다면 **새로고침** 버튼을 선택하세요.
> - **전체 접기** 또는 **전체 펼치기** 버튼을 선택해 그룹 단위로 목록을 감추거나 펼칠 수 있습니다.

지표 선택하기

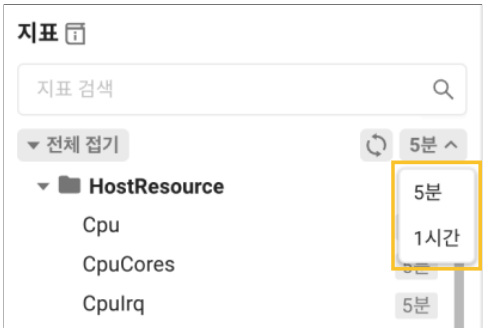
모니터링할 대상 에이전트를 선택한 후 **지표** 패널에 선택할 수 있는 목록이 표시됩니다. 목록에서 원하는 지표를 선택하세요.



선택한 지표에 해당하는 차트 위젯이 오른쪽의 대시보드에 배치됩니다. 다른 지표를 추가로 선택해 차트 위젯을 추가할 수 있습니다. 차트 영역을 확대해 넓게 보려면 **시간** 옵션 오른쪽의 ✕ 버튼을 선택하세요. **대상**과 **지표** 패널이 사라지고 차트 영역만 화면에 표시됩니다.

- **지표 검색** 입력란에 문자열을 입력하면 원하는 지표를 빠르게 검색할 수 있습니다.
- 지표의 카테고리 및 필드 이름을 기준으로도 검색할 수 있습니다. 카테고리 및 필드에 대한 자세한 내용은 다음 문서를 참조하세요.
- **전체 접기** 또는 **전체 펼치기** 버튼을 선택해 카테고리 단위로 지표 목록을 감추거나 펼칠 수 있습니다.
- 원하는 지표 항목이 목록에 표시되지 않는다면 **새로고침** 버튼을 선택하세요.

인터벌 설정하기

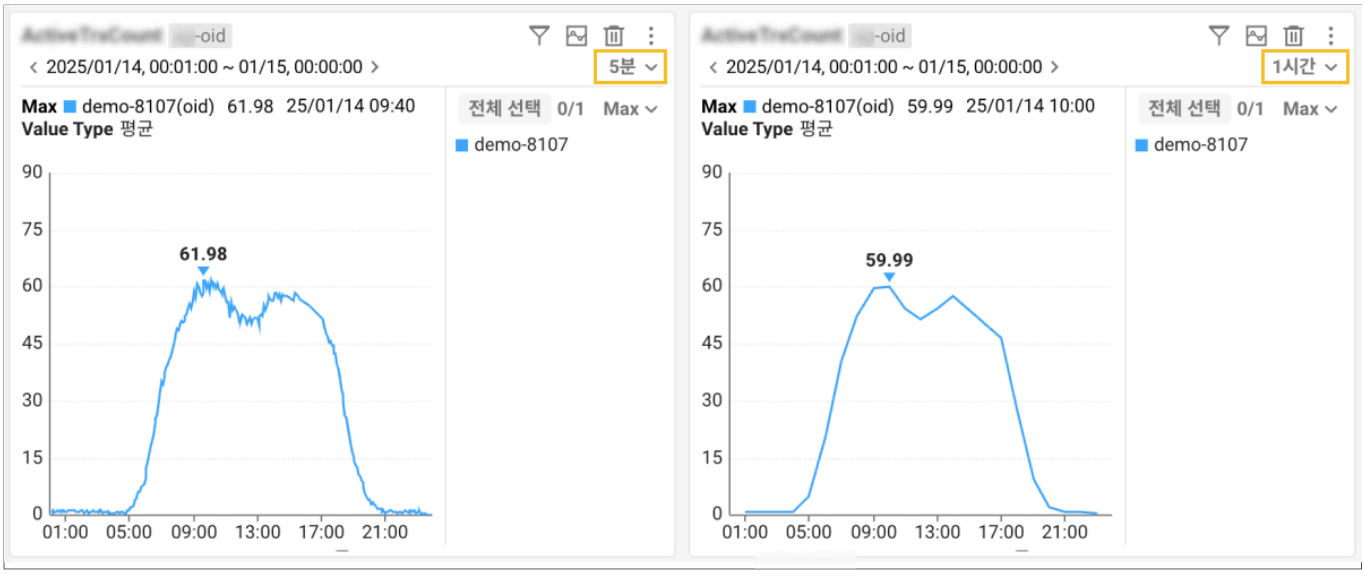


차트마다 인터벌을 설정할 수 있어 데이터의 집계 단위를 조정하여 다양한 시간 간격의 데이터를 시각화할 수 있습니다. **지표**

패널의 오른쪽 상단에 버튼을 선택하면 인터벌 옵션을 선택할 수 있습니다.

지표마다 기본 설정된 인터벌 시간이 다르며 조회 시간 범위에 따라 인터벌 시간이 변경됩니다.

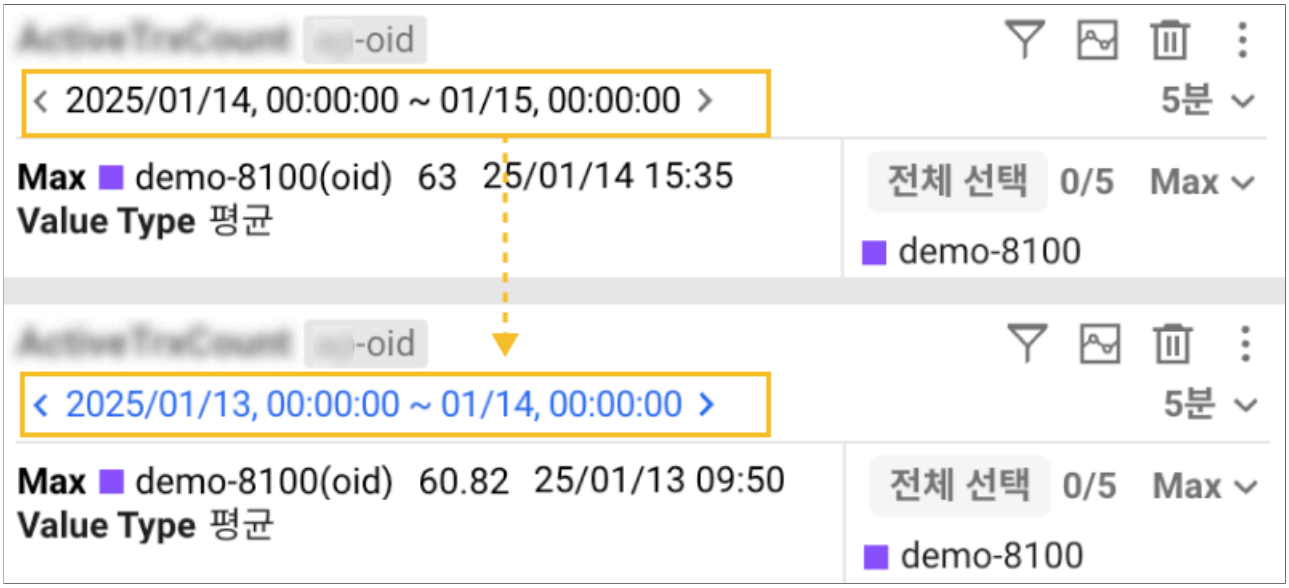
예를 들어, **5분** 인터벌은 5분 단위의 평균값을 표시하고, **1시간** 인터벌은 1시간 단위의 평균값을 표시합니다.



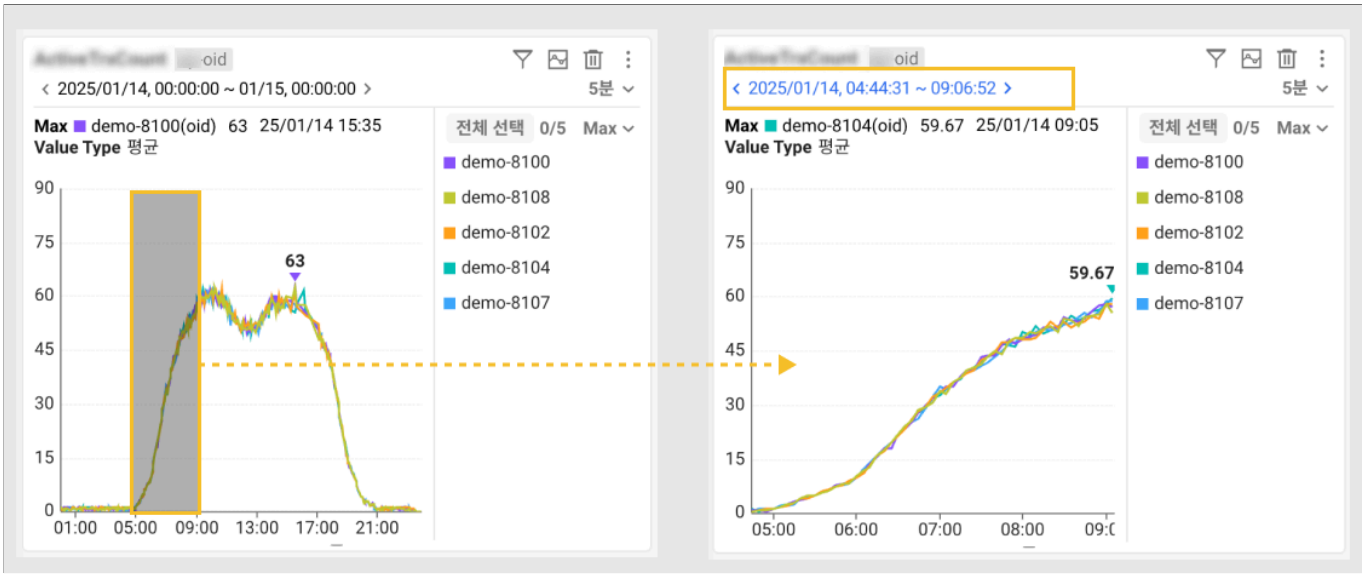
인터벌 옵션을 조정하여 짧은 시간 간격의 세부 데이터부터 긴 시간 간격의 전체 추세까지 다양한 관점에서 데이터를 분석할 수 있습니다.

조회 시간 변경하기

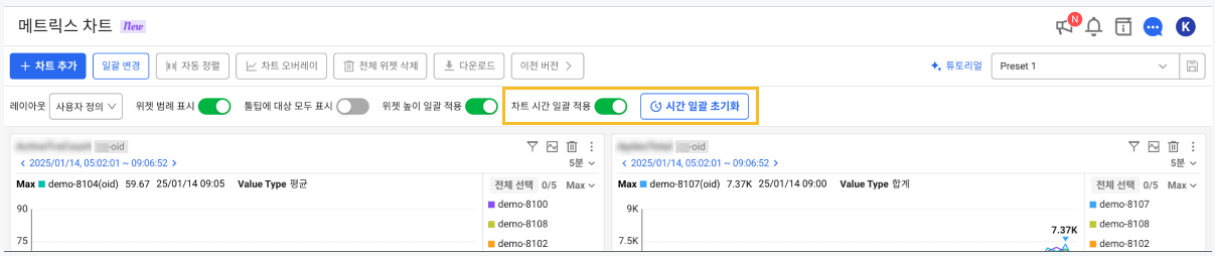
개별 위젯의 차트 조회 시간 범위를 변경하려면 위젯 상단의 < 또는 > 버튼을 선택하세요. 시간 변경 범위는 **시간** 옵션에 설정된 시간 기준과 같습니다. 예를 들어, **시간** 옵션에서 **1일**로 설정되어 있을 때, < 또는 > 버튼을 선택하면 시간을 1일 단위로 변경할 수 있습니다.



또는 차트의 특정 영역을 드래그하면, 드래그한 범위만큼 시간을 변경해 조회할 수도 있습니다.



- 화면 상단에 **차트 시간 일괄 적용** 옵션을 활성화한 상태에서 개별 위젯의 시간을 변경하면 모든 위젯의 시간을 변경할 수 있습니다.



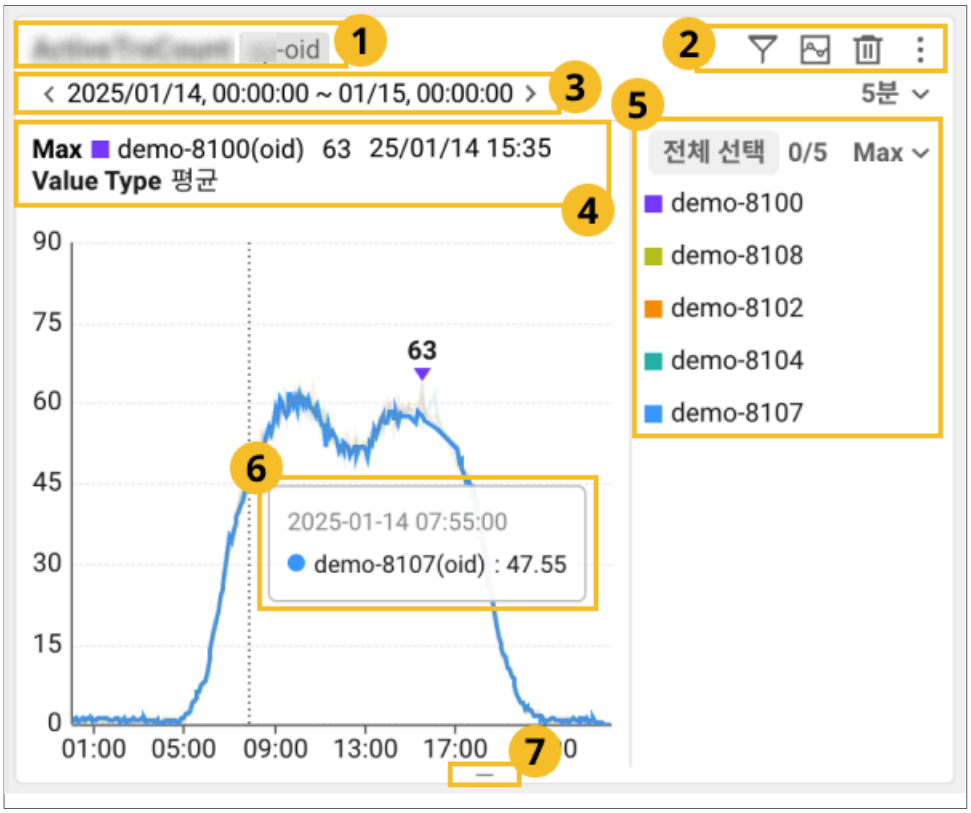
- 조회 시간 범위를 변경한 다음 최근 시간으로 다시 변경하려면 버튼을 선택하세요.

  만약 화면 상단에 **차트 시간 일괄 적용** 옵션을 활성화한 상태라면 **시간 일괄 초기화** 버튼이 표시됩니다. 이 버튼을 선택하면 모든 위젯의 시간을 초기화할 수 있습니다.

일괄 변경 기능을 이용해 전체 위젯의 시간을 변경하고, 모니터링 대상과 인터벌도 수정할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

차트 위젯 상세 안내

차트 위젯에 표시되는 각 항목에 대한 자세한 내용을 확인하세요.



- 1 **지표 이름 및 카테고리**: 차트 위젯의 지표 이름을 표시합니다. 지표의 오른쪽에는 조회 대상의 카테고리를 태그 형식으로 표시합니다. 이용하는 제품에 따라 카테고리 명칭은 다를 수 있습니다.
  - `*oid` : 개별 에이전트를 선택했을 때 표시됩니다.
  - `project` : 프로젝트 단위로 대상을 선택했을 때 표시됩니다.
  - `okind` : 에이전트 설정에서 `whatap.okind` 로 분류된 그룹을 선택했을 때 표시됩니다.
  - `onode` : 에이전트 설정에서 `whatap.onode` 로 분류된 그룹을 선택했을 때 표시됩니다.
- 2 **위젯 옵션**: 차트 위젯의 옵션 메뉴입니다. 자세한 내용은 다음 문서를 참조하세요. 인터벌 옵션에 대한 자세한 내용은 다음 문서를 참조하세요.
- 3 **시간**: 차트의 데이터를 조회하는 시간 범위를 표시합니다. < 또는 > 버튼을 선택해 조회 시간 범위를 조정할 수 있습니다.
- 4 **데이터**: 모니터링 대상 중 조회한 데이터의 최댓값(**Max**)과 데이터 병합 방식(**Value Type**)을 확인할 수 있습니다.
- 5 **범례**: 모니터링 대상을 범례로 표시합니다. 범례 표시에 대한 자세한 내용은 다음 문서를 참조하세요.
- 6 **차트**: 조회한 데이터를 시계열 그래프로 표시합니다. 마우스 포인터를 그래프 위로 가져가면 툴팁을 통해 상세 정보를 확인할 수 있습니다. 차트의 특정 영역을 드래그하면, 드래그한 범위만큼 시간을 변경해 조회할 수 있습니다.

- 7 **크기 조절**: 차트 위젯의 가운데 하단 영역을 드래그하면 위젯의 크기를 조절할 수 있습니다.

차트 위젯 옵션 이용하기

차트 위젯의 오른쪽 상단에 위치한 옵션 버튼을 통해 다양한 기능을 이용할 수 있습니다.

대상 필터링하기

화면에 추가한 차트 위젯에 모니터링 대상을 추가하거나 제외할 수 있습니다.

1. 버튼을 선택하면 모니터링 대상을 선택할 수 있는 팝업 메뉴가 나타납니다.



2. 화면의 안내에 따라 원하는 대상을 선택하세요.
3. **적용** 버튼을 선택하세요.

상세 조회하기

차트의 데이터를 더 넓은 화면에서 자세한 정보를 확인하려면 버튼을 선택하세요. **상세 조회** 창이 나타납니다. 개별 대상에 대한 차트와 평균 및 최댓값을 조회할 수 있습니다.



- **1일전**: 하루 전의 데이터와 비교해 확인할 수 있습니다.
- **1주전**: 한 주 전의 데이터와 비교해 확인할 수 있습니다.
- **CSV**: 조회한 데이터를 CSV 형식의 파일로 다운로드할 수 있습니다.
- 그래프 차트 하단의 아이콘을 드래그하면 차트의 크기를 조절할 수 있습니다.

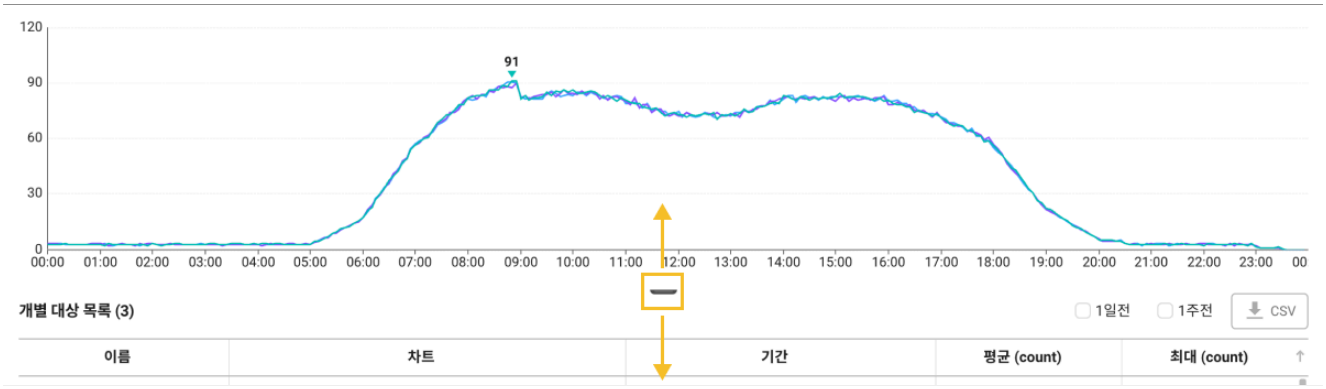
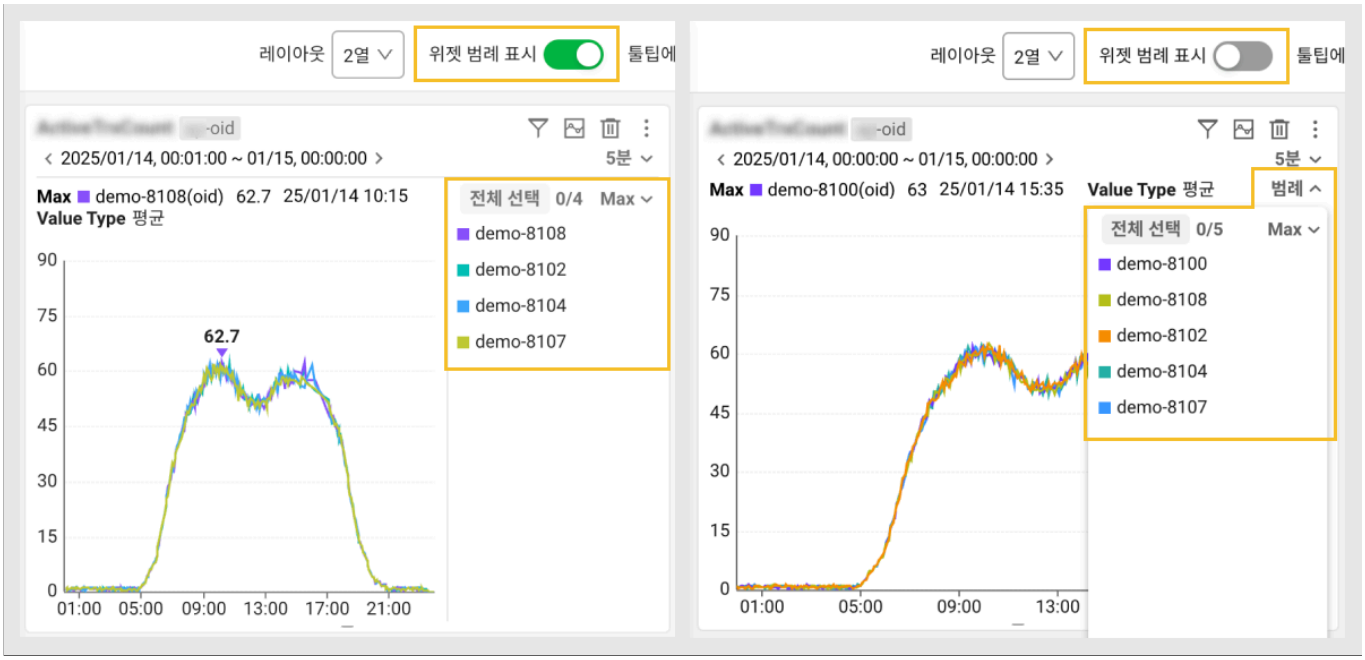


차트 위젯 삭제하기

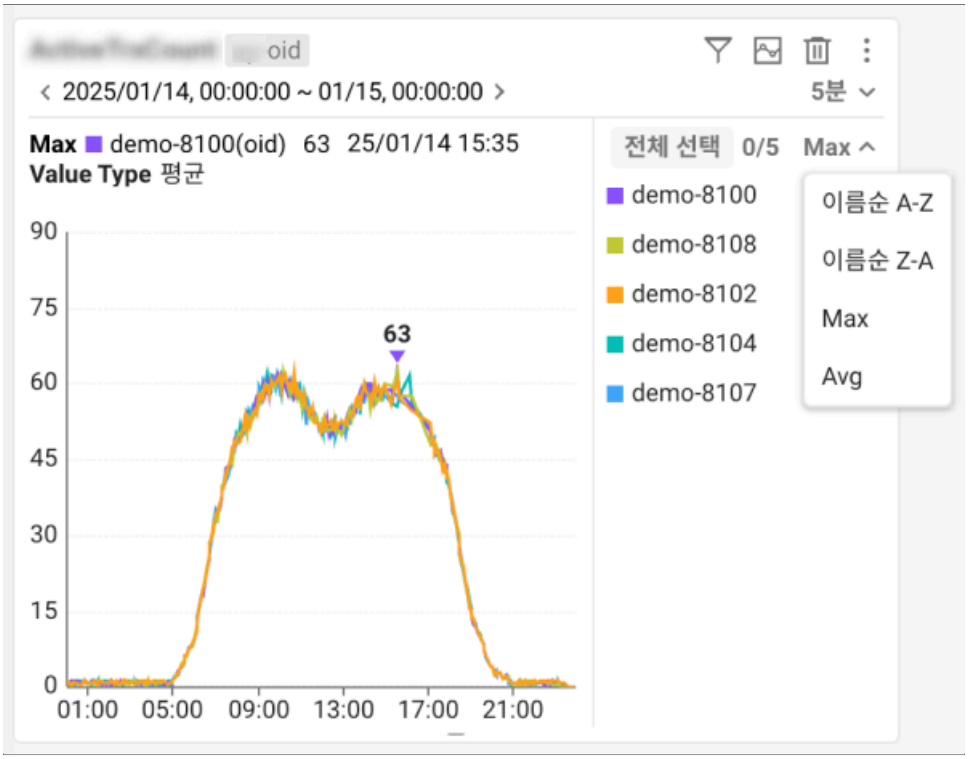
화면에 배치한 차트 위젯을 삭제하려면 버튼을 선택하세요.

범례 활용하기

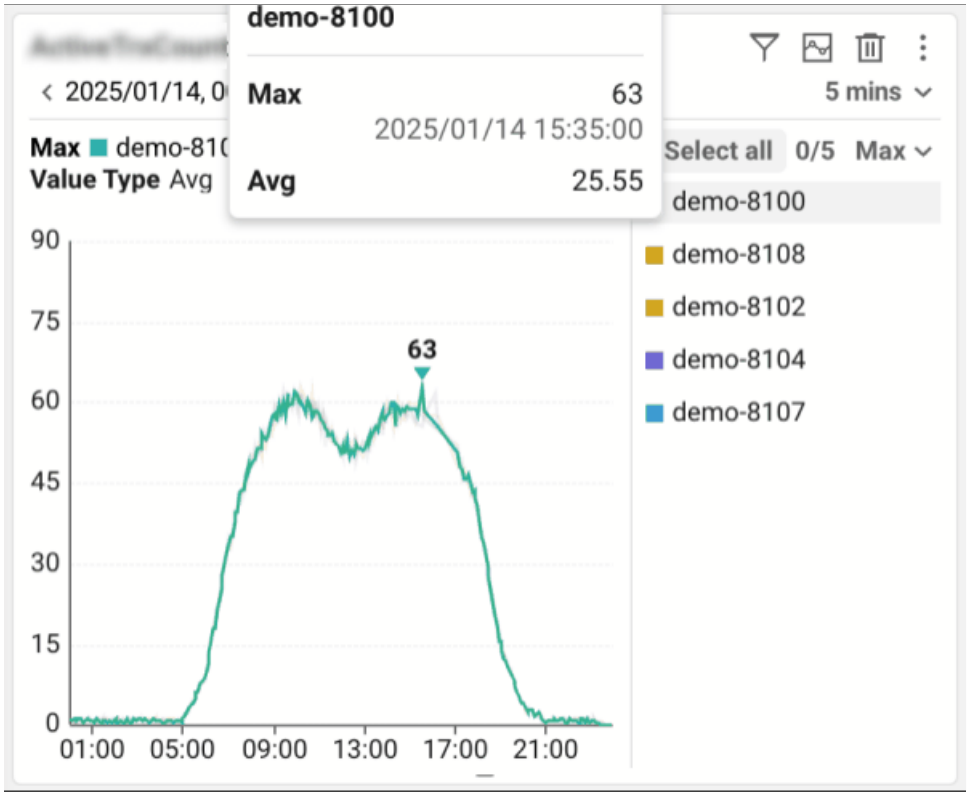
범례를 통해서 차트에 표시되는 모니터링 대상을 확인하고, 원하는 기준으로 정렬하거나 필터링할 수 있습니다. 화면 상단의 **위젯 범례 표시** 옵션을 켜거나 끄면 위젯의 차트 오른쪽에 범례를 표시하거나 숨길 수 있습니다. 옵션을 끈 상태에서는 **범례**를 클릭해 모니터링 대상을 확인하고, 선택할 수 있습니다.



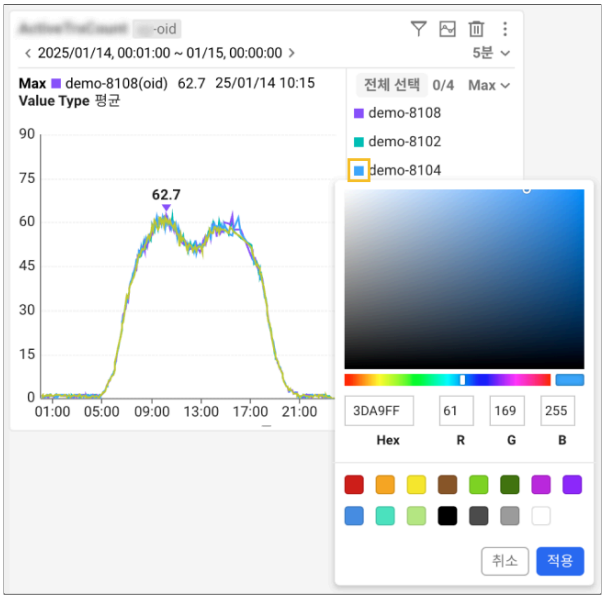
- **전체 선택/전체 해제** 버튼을 선택해 모니터링 대상을 선택하거나 해제할 수 있습니다.
- 범례 목록에서는 모니터링 대상을 이름순, 최댓값, 평균값 기준으로 정렬할 수 있습니다.



- 범례 목록에서 각 항목에 마우스를 오버하면 해당 모니터링 대상에 대한 그래프를 강조해서 확인할 수 있고, Max 값이 기록된 특정 시간을 툴팁으로 표시합니다. 또는 범례 목록의 대상을 선택해 필터링할 수도 있습니다.


demo-8100
Max 63
2025/01/14 15:35:00
Avg 25.55
5 mins
Select all 0/5 Max
demo-8100
demo-8108
demo-8102
demo-8104
demo-8107
63
90
75
60
45
30
15
0
01:00 05:00 09:00 13:00 17:00 21:00

범례 색상 변경하기

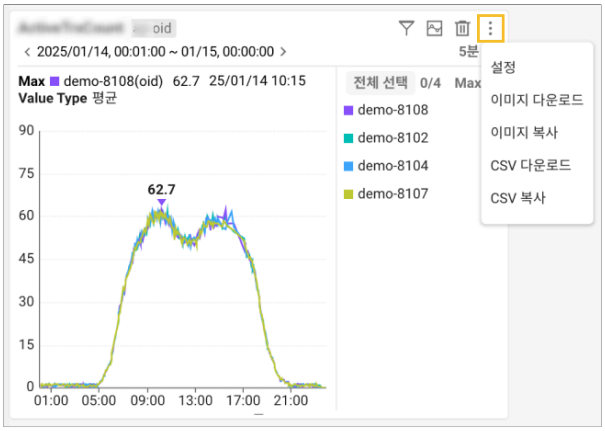

2025/01/14, 00:01:00 ~ 01/15, 00:00:00
5분
Max demo-8108(oid) 62.7 25/01/14 10:15
Value Type 평균
전체 선택 0/4 Max
demo-8108
demo-8102
demo-8104
62.7
3DA9FF
61
169
255
Hex
R
G
B
취소
적용

모니터링 대상에 해당하는 그래프의 색상을 변경할 수 있습니다. 범례 목록에서 색상을 변경하려는 대상의 왼쪽에 네모 상자를 선택하세요. 원하는 색상을 선택한 다음 **적용** 버튼을 클릭하세요.

> ⓘ 설정한 색상은 프리셋 기능을 통해서 저장되지 않습니다. 이 기능은 향후 업데이트를 통해 제공할 예정입니다.

추가 옵션 이용하기

차트 위젯 오른쪽 상단에 ⋮ 버튼을 선택하면 다음의 추가 옵션을 이용할 수 있습니다.



- **설정**
  - **자동 Y축 사용**: 조회한 데이터 범위 내에서 Y축의 크기를 자동 조정할 수 있습니다.
  - **최대값 표시**: 차트의 그래프 영역에 최댓값을 표시할 수 있습니다.
- **이미지 다운로드**: 차트를 이미지 형식의 파일로 다운로드할 수 있습니다.
- **이미지 복사**: 차트를 이미지 형식으로 클립보드에 복사할 수 있습니다.
- **CSV 다운로드**: 차트에서 조회한 데이터를 CSV 형식의 파일로 다운로드할 수 있습니다.
- **CSV 복사**: 차트에서 조회한 데이터를 CSV 형식으로 클립보드에 복사할 수 있습니다.

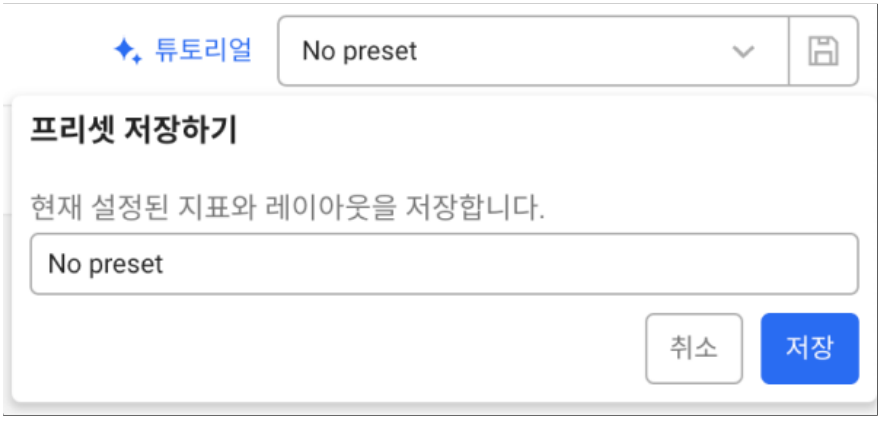
프리셋 설정하기

대시보드에서 사용자가 설정한 위젯의 설정과 레이아웃 상태를 저장하고 불러올 수 있습니다. 위젯의 크기를 조절하고, 원하는 위치에 배치해 새로운 프리셋을 만들 수 있습니다.

ⓘ **프로젝트 수정** 권한이 있는 멤버만 프리셋을 저장하거나 수정할 수 있습니다. 멤버 권한에 대한 자세한 내용은 다음 문서를 참조하세요.

새로운 프리셋 만들기

1. 대시보드에서 원하는 형식으로 위젯을 배치해 보세요. 크기를 조절하고 자주 확인하는 위젯만 배치할 수도 있습니다.
2. 화면 오른쪽 위에 버튼을 선택하세요.
3. 새로운 프리셋 이름을 입력하세요.



4. **저장** 버튼을 선택하세요.

프리셋 목록에서 새로 저장한 프리셋을 확인할 수 있습니다.

ⓘ
- 새로 만든 프리셋에 변경 사항이 생겼다면 다시 프리셋을 저장해야 합니다. 버튼을 선택한 다음 같은 이름으로 프리셋을 저장하세요. 기존의 프리셋에 변경 사항을 덮어쓰기합니다.
- 대시보드의 변경 사항을 저장하지 않고 다른 메뉴로 이동하면 변경 사항은 저장되지 않습니다.
- 프리셋은 프로젝트 단위로 저장되어 다른 사용자와 공유할 수 있습니다.
- 프리셋에 저장되는 대상은 대시보드의 레이아웃과 대시보드에 배치한 위젯, 위젯에서 수정한 옵션(**대상**, **인터벌**)입니다. 화면 상단의 **시간** 옵션은 저장되지 않습니다.
- 프리셋을 불러온 다음 위젯의 조회 시간을 변경하려면 **일괄 변경** 기능을 이용하세요. 자세한 내용은 다음 문서를 참조하세요.

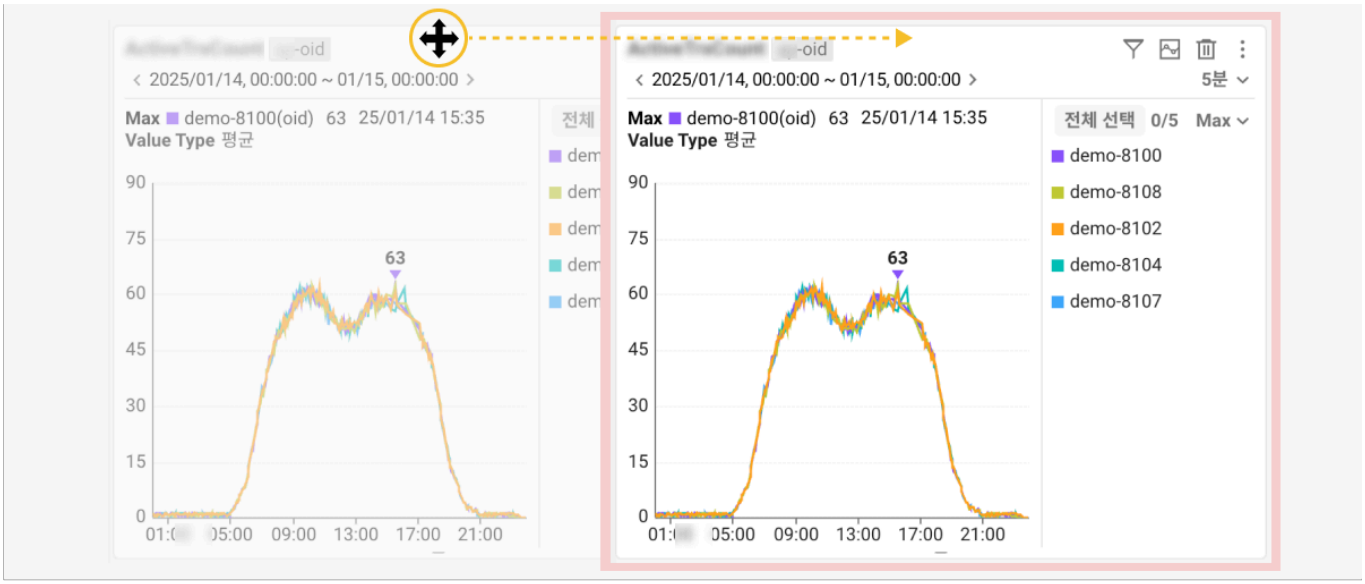
프리셋 삭제하기

사용하지 않는 프리셋이 있다면 프리셋 목록에서 삭제할 수 있습니다. 프리셋 목록에서 삭제하려는 항목의 오른쪽에 ✕ 버튼을 선택하세요.

대시보드 위젯 편집하기

대시보드에 배치한 위젯을 사용자가 원하는 위치에 배치하거나 크기를 조절할 수 있습니다.

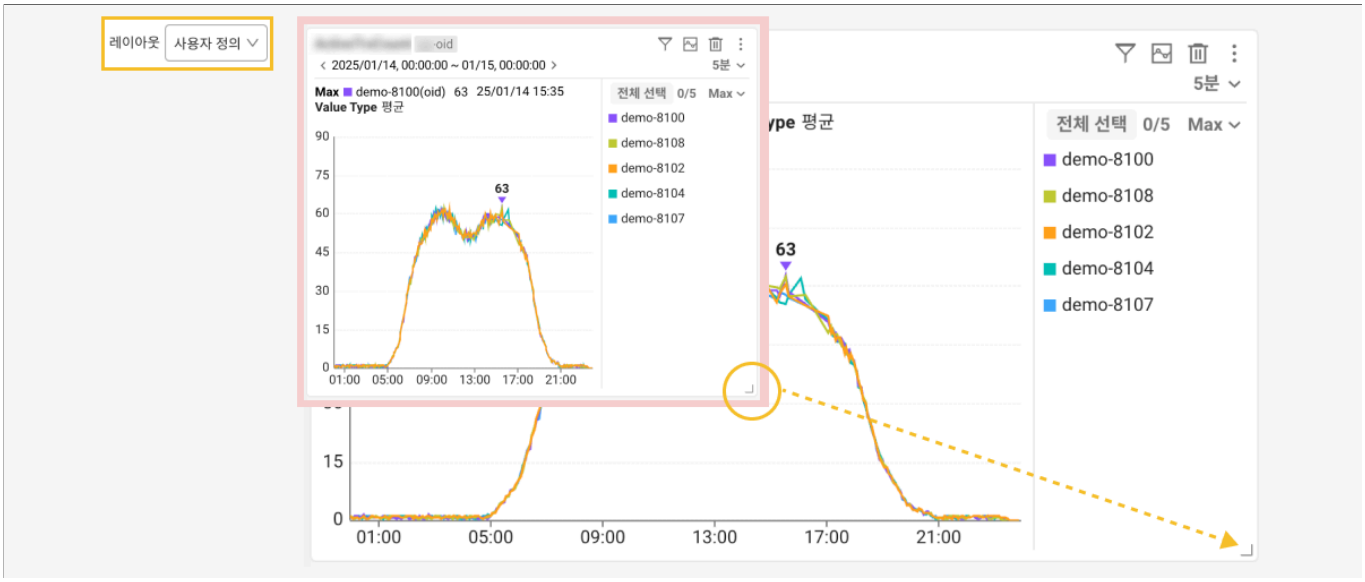
위젯 이동하기



위젯의 상단 부분을 마우스 포인터로 이동하면 ✢ 모양으로 변경됩니다. 이때 마우스로 클릭한 상태로 원하는 위치로 드래그하여 위젯을 이동할 수 있습니다.

위젯 크기 조절하기

대시보드 옵션의 **레이아웃**이 **사용자 정의**으로 설정되어 있으면 위젯의 크기를 조절할 수 있습니다.



위젯의 오른쪽 아래에 ⌟ 요소를 마우스로 클릭한 상태에서 원하는 크기로 드래그하세요.

> ⓘ **레이아웃**이 **사용자 정의**로 설정되어 있지 않다면 위젯의 높이만 조절할 수 있습니다. 위젯 하단 가운데 영역을 마우스로 클릭한 상태로 크기를 조절하세요.

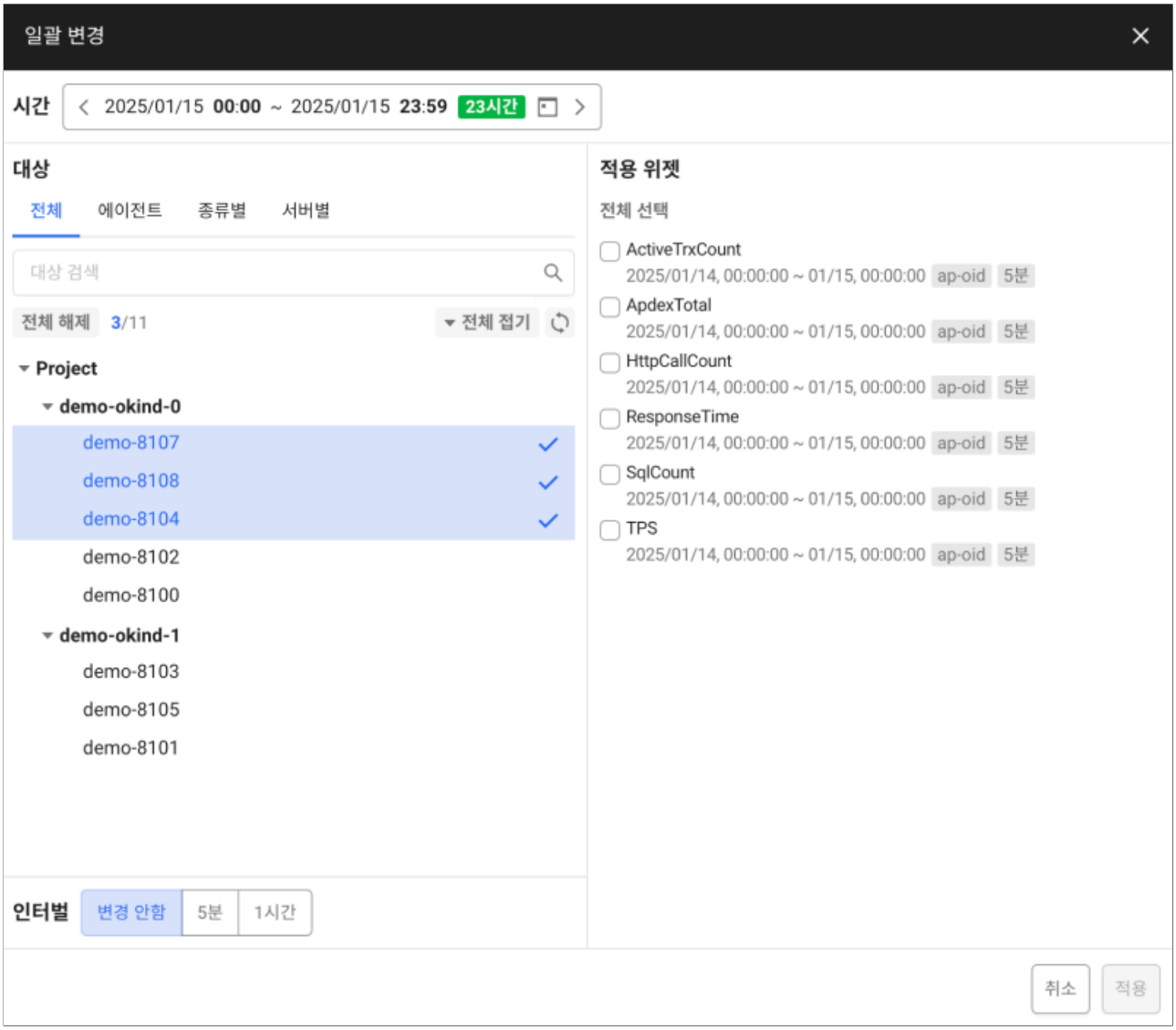
대시보드 옵션 이용하기

화면 상단에 위치한 옵션 메뉴를 이용해 차트 위젯에 적용된 다양한 설정을 한번에 적용할 수 있습니다. 또한 대시보드의 레이아웃을 원하는 형태로 설정할 수도 있습니다.



차트 위젯 옵션 일괄 변경하기

화면에 배치된 차트 위젯에 세부 옵션을 한번에 적용할 수 있는 기능을 제공합니다. 화면 왼쪽 상단에 **일괄 변경** 버튼을 선택하세요. **일괄 변경** 창이 나타나면 원하는 옵션을 설정한 다음 **적용** 버튼을 선택하세요.



- **시간**: 차트의 데이터 조회 범위를 설정할 수 있습니다.
- **대상**: 모니터링 대상을 선택하세요. 개별 에이전트를 선택하거나 개별 에이전트를 선택하거나 에이전트 설정을 통해 분류한 그룹 단위로 선택할 수 있습니다.
  - **에이전트**: 개별 에이전트를 선택하거나 모두 선택할 수 있습니다.
  - **종류별**: 에이전트 설정에서 `whatap.okind` 로 분류된 그룹 단위로 에이전트를 모니터링합니다.
  - **서버별**: 에이전트 설정에서 `whatap.onode` 로 분류된 그룹 단위로 에이전트를 모니터링합니다.
- **인터벌**: 다양한 시간 간격으로 데이터의 집계 단위를 조정할 수 있습니다.

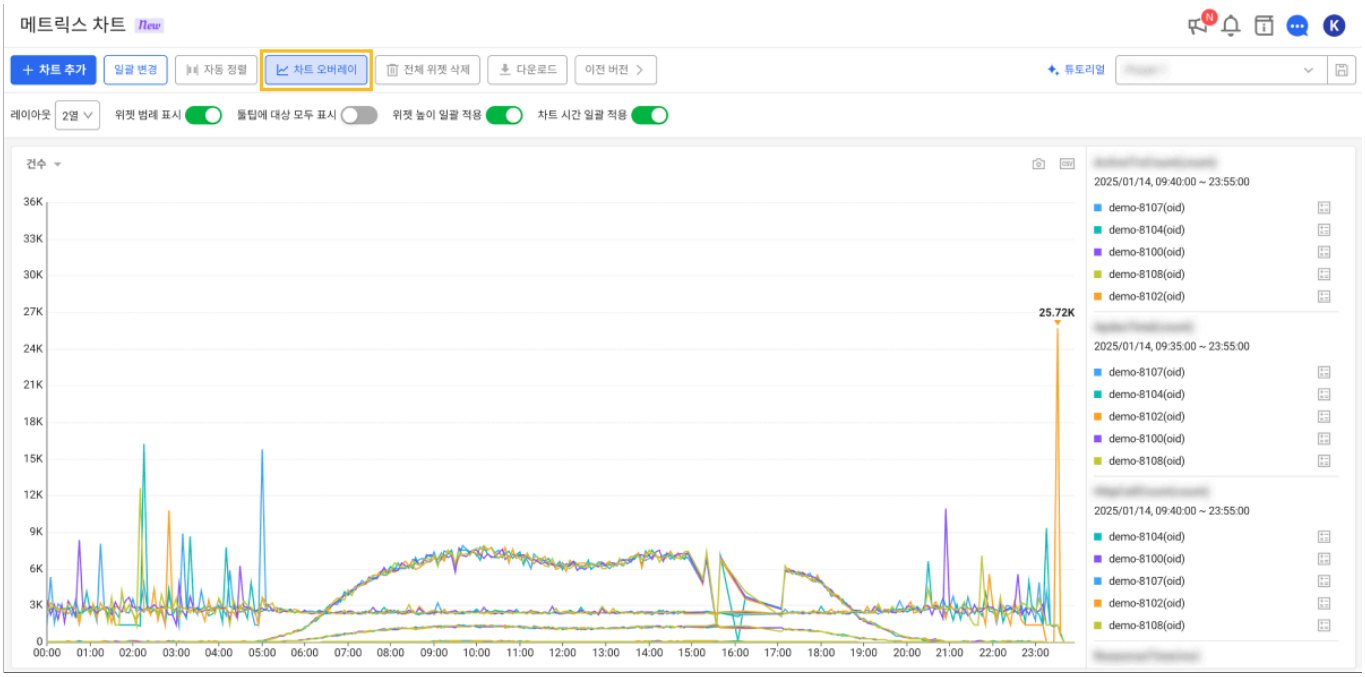
- **적용 위젯**: 변경할 옵션을 적용할 차트 위젯을 선택하세요. 선택한 위젯에 변경한 옵션을 한번에 적용할 수 있습니다.

대시보드 자동 정렬하기

대시보드에 배치한 위젯들의 위치를 자동 정렬하려면 **자동 정렬** 버튼을 선택하세요. 위젯의 너비, 높이가 작은 크기순으로 위젯의 배치를 자동 정렬합니다. 너비가 같으면 높이가 작은 순으로 정렬합니다.

여러 개의 차트 비교하기

대시보드에 배치한 여러 개의 차트 데이터를 하나의 차트에서 확인할 수 있는 기능입니다. 이 기능을 통해서 동시간대의 여러 지표 정보를 다른 지표와 비교해 가며 확인할 수 있습니다. **차트 오버레이** 버튼을 선택하세요. 화면에 오버레이 차트를 배치할 수 있습니다.



- 오버레이 차트 왼쪽 상단에 **건수**를 선택하면 x축 차트의 데이터 기준을 변경할 수 있는 목록이 나타납니다. 원하는 항목을 선택해 차트의 데이터를 비교할 수 있습니다.
- : 현재 조회 중인 데이터를 기준으로 차트를 이미지 형식의 파일로 다운로드할 수 있습니다.
- CSV : 현재 조회 중인 데이터를 기준으로 차트를 CSV 형식의 파일로 다운로드할 수 있습니다.
- 오른쪽 섹션에서는 지표별 에이전트 목록을 확인할 수 있습니다. 개별 에이전트의 색상은 차트의 그래프 색상과 일치합니다.

배치된 위젯 일괄 삭제하기

대시보드에 배치한 여러 개의 위젯을 한번에 삭제할 수 있습니다. **전체 위젯 삭제** 버튼을 선택하세요. 확인 메시지가 나타나면 **삭제** 버튼을 선택하세요.

CSV 파일로 다운로드하기

대시보드에 배치한 위젯의 데이터를 CSV 형식의 파일로 다운로드할 수 있습니다. **다운로드** > **CSV 다운로드**를 선택하세요.
시간대별 지표의 데이터를 확인할 수 있습니다.

이미지 파일로 다운로드하기

대시보드에 배치한 모든 위젯을 하나의 이미지 형식의 파일로 다운로드할 수 있습니다. **다운로드** > **이미지 다운로드**를 선택하세요.
대시보드 화면을 그대로 캡처한 이미지를 저장할 수 있습니다.

이전 버전 이용하기

이전 버전의 **메트릭스 차트** 메뉴를 이용하려면 **이전 버전** 버튼을 선택하세요. 또는 **사이트맵** > **분석** > **메트릭스 차트** *Old* 메뉴 경로에서 확인할 수 있습니다.

> **메트릭스 차트** *Old* 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.

기타 옵션 이용하기

그 밖의 다음 기타 옵션을 통해 대시보드의 레이아웃을 조정하거나 위젯의 적용된 옵션을 쉽고 간편하게 일괄 설정할 수 있습니다.



- **레이아웃**: 대시보드에서 열의 수, 즉 한 줄에 배치할 수 있는 위젯의 개수를 설정할 수 있습니다. 최대 4열까지 설정할 수 있습니다. **사용자 정의**를 선택하면 열의 수에 상관없이 위젯을 배치할 수 있습니다.

> ⓘ **사용자 정의**에서는 위젯의 너비와 높이를 자유롭게 조절할 수 있습니다. 다른 옵션을 선택하면 위젯의 높이만 조절할 수 있습니다.

- **위젯 범례 표시**: 옵션을 켜거나 끄면 위젯의 차트 오른쪽에 범례를 표시하거나 숨길 수 있습니다. 옵션을 끈 상태에서는 **범례**를 클릭해 모니터링 대상을 확인하고, 선택할 수 있습니다. 범례 활용에 대한 자세한 내용은 다음 문서를 참조하세요.
- **툴팁에 대상 모두 표시**: 차트에 마우스 포인터를 가져가면 해당 그래프의 위치에 해당하는 값을 툴팁을 통해 확인할 수 있습니다. 마우스 포인터가 위치한 시간대의 모든 대상의 데이터를 툴팁을 통해서 확인하려면 옵션을 켜세요.
- **위젯 높이 일괄 적용**: 옵션을 켜면 모든 위젯의 높이를 동일하게 설정할 수 있습니다.
- **차트 시간 일괄 적용**: 개별 차트 위젯의 데이터 조회 시간을 변경할 때, 다른 위젯의 차트 시간도 같이 변경하려면 옵션을 켜세요.

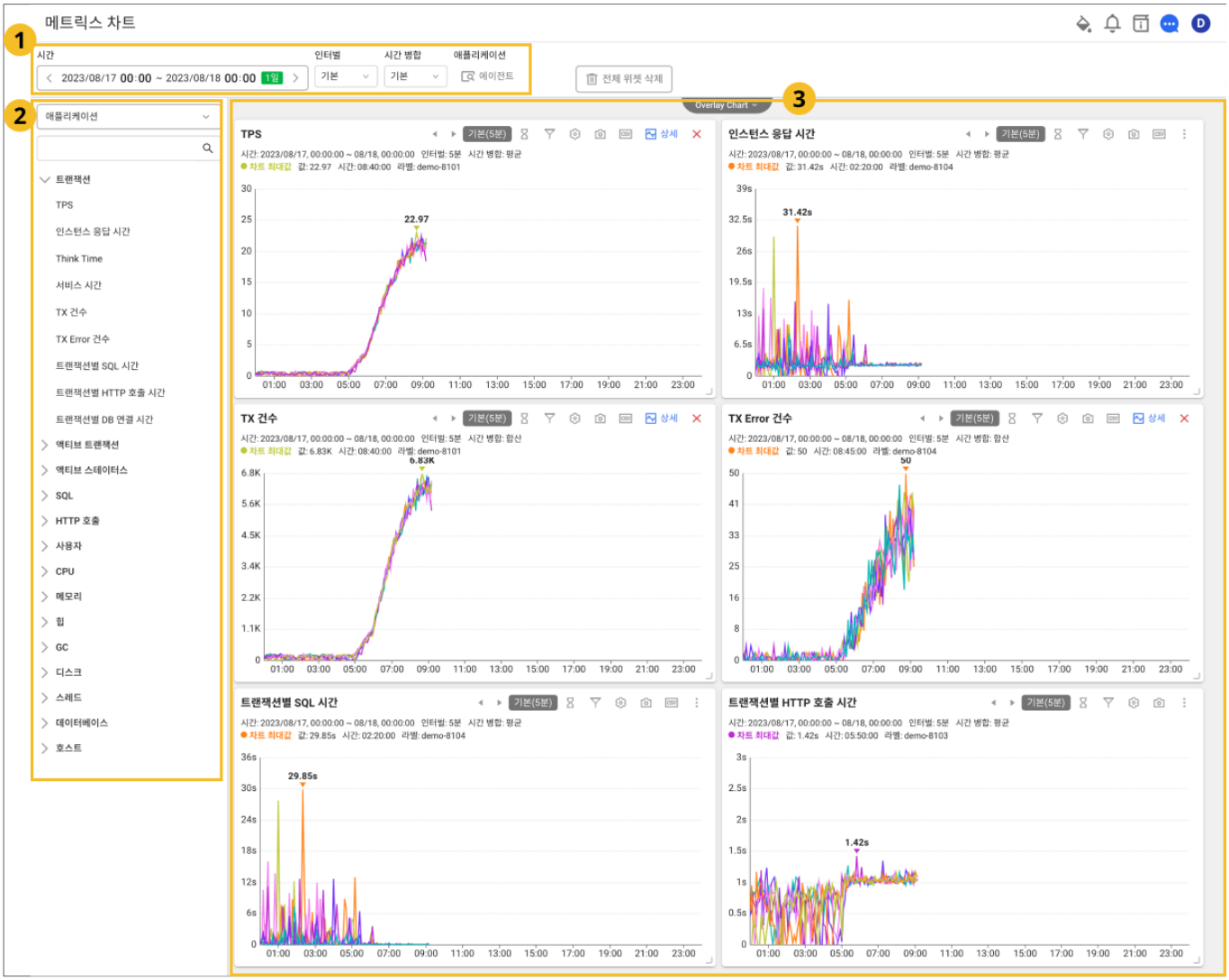
튜토리얼 이용하기

화면 오른쪽 상단에 **튜토리얼** 버튼을 선택하면 Service 2.8.0 업데이트 이후 변경 사항을 확인할 수 있습니다.

메트릭 차트

사이트맵 > 분석 > 메트릭스 차트 *Old*

메트릭스 차트 메뉴에서 모니터링 대상에서 수집된 메트릭스 데이터를 다음과 같이 차트로 조회할 수 있습니다. 시간과 지표를 지정하는 것은 필수입니다.



상단 옵션

1 영역의 **메트릭스 차트**의 상단 옵션을 통해 차트의 시간 범위와 모니터링 대상 에이전트를 지정할 수 있습니다.

- **시간**: 시간의 총 범위로 X축의 시작과 끝을 지정할 수 있습니다.
- **인터벌**: 시간 간격으로 X축 데이터 간격을 지정할 수 있습니다.
- **시간 병합**: 데이터 병합 방식 중 하나로 인터벌로 지정한 시간 내의 데이터를 병합할 수 있습니다.

  예, **평균**은 1시간 안 데이터의 평균값을 말합니다.

- **에이전트**: 조회할 에이전트를 지정할 수 있습니다. 지정하지 않으면 전체가 조회됩니다.

지표 목록

2 영역은 옵션을 조회할 지표 목록입니다. 먼저 **카테고리**를 선택하세요. 선택한 **카테고리** 하위의 지표를 조회한 후 원하는 지표를 선택하세요. **카테고리**와 지표를 선택하면 1 영역의 상단 메뉴에서 지정한 시간 범위의 데이터를 바탕으로 3 영역에서 차트 위젯을 조회할 수 있습니다.

> **데이터 병합**
>
> 데이터 병합은 **시간 병합**과 **오브젝트 병합**을 제공합니다.
>
> - **시간 병합**은 원본 데이터에서 필드 값이 같은 데이터끼리 일정한 간격으로 데이터를 병합합니다.
> - **오브젝트 병합**은 서로 다른 필드 값을 가진 데이터들 중에서 태그가 일치하는 경우 해당 데이터를 병합합니다.

차트 위젯

3 영역 차트 위젯의 좌측 상단에서 지표명을 확인할 수 있습니다. 차트 위젯의 우측 상단에서 다음과 같은 옵션을 확인할 수 있습니다.



- 시간 이동: ◀ ▶ 왼쪽 화살표 또는 오른쪽 화살표 버튼을 통해 선택한 시간 범위만큼 -1, +1 씩 이동 가능합니다.

예, 시간 범위가 2월 13일 00:00~2월 14일 00:00일 때, ◀ 왼쪽 화살표를 선택하면 2월 12일 00:00~2월 13일 00:00 데이터를 조회할 수 있습니다.

- 인터벌/시간 병합: 1 상단 메뉴에서 지정한 인터벌과 시간 병합을 수정할 수 있습니다.
- 모니터링 대상: 아이콘을 선택해 모니터링 대상을 지정할 수 있습니다. 선택하지 않으면 전체를 대상으로 조회합니다.
- 시간 비교: 아이콘을 선택하면 동일한 지표의 이전 시간대의 추이를 비교할 수 있습니다.
- 스냅샷: 아이콘을 선택해 위젯의 옵션을 제외한 차트를 스냅샷 할 수 있습니다.
- CSV: 아이콘을 선택해 차트를 그리는 데이터를 CSV 파일로 다운로드할 수 있습니다.
- **상세**: 아이콘을 선택해 상세 조회가 가능합니다. 모니터링 대상이 많은 경우 유용하며, 모니터링 대상의 지표 추이를 개별로 확인 할 수 있습니다.

> 메트릭스 차트 위젯 상단에서 옵션이 보이지 않을 경우 ⋮ 아이콘을 선택하세요.

위젯 삭제하기

화면에 배치한 모든 위젯을 삭제하려면 화면 위에 위치한 **전체 위젯 삭제** 버튼을 선택하세요.

메트릭 검색

홈 화면 > 프로젝트 선택 > **분석** > **메트릭스 조회**

메트릭스 조회 메뉴에서 태그 기반으로 특정 메트릭스를 조회할 수 있습니다.

시간과 카테고리 선택

메트릭스가 수집된 **시간 선택**과 **최대 개수** 및 **카테고리**를 지정할 수 있습니다. **시간 선택**과 **카테고리**는 반드시 지정해야 합니다.

- **시간 선택**: 메트릭스가 수집된 시간을 지정해 조회할 수 있습니다. 기본값은 `1시간` 입니다. 기본 옵션으로 제공하는 조회 시간 외 사용자가 직접 **시간 선택** 탭을 선택해 날짜와 시간을 지정할 수 있습니다.
- **최대 개수**: **메트릭스 테이블** 목록에 조회할 메트릭스 최대 개수를 지정할 수 있습니다. `10` , `50` , `100` , `200` , `300` , `1000` , `2000` , `3000` 개까지 설정할 수 있습니다.
- **카테고리**: 유관 지표들의 분류 단위입니다. **카테고리** 탭을 선택해 원하는 **카테고리**를 지정할 수 있습니다.
- **카테고리, 태그, 필드 옵션 다시 불러오기**: **카테고리**, **태그** 및 **필드** 옵션을 다시 불러올 수 있습니다.

태그와 필드 선택

태그와 **필드**를 선택합니다. 사용자가 개별적으로 지정하지 않는다면 기본 설정은 전체 선택입니다.

- **태그**: 수집된 대상을 구분할 수 있는 고유 정보 데이터입니다.
- **필드**: 모니터링 대상으로부터 수집된 지표입니다.
- **필터**: **태그값으로 필터링** 버튼을 선택하고 태그값을 설정해 필터링할 수 있습니다.

  예시, `oname` 의 값을 `demo-8101` 로 설정해 필터링한 데이터를 조회할 수 있습니다.
- **검색**: 조건을 설정 후 **검색** 아이콘을 선택하면 **메트릭스 테이블** 영역에서 해당 메트릭스의 원본 데이터를 조회할 수 있습니다.
- CSV : 해당 메트릭스 원본 데이터를 CSV 파일로 다운로드할 수 있습니다.

메트릭스 테이블

수집되는 메트릭스를 사전에 특정할 수 없기에 수집 중인 모든 메트릭스의 원본 데이터를 확인하는 것이 중요합니다. 위의 조건 영역에서 원하는 조건을 설정 후 **메트릭스 테이블** 영역에서 해당 메트릭스의 원본 데이터를 테이블 형식으로 조회할 수 있습니다. 사용자가 **태그**와 **필드** 각 조건을 지정함에 따라 테이블의 컬럼이 변경됩니다.

- 메트릭스 조회 시 **시간 선택**과 **카테고리**는 반드시 지정해야 합니다.
- 메트릭스 조회 시 **태그**와 **필드** 지정은 선택 사항입니다.

메트릭 이상 탐지

홈 화면 > 프로젝트 선택 > **사이트맵** > **메트릭스 이상 탐지**

다양한 메트릭의 패턴을 AI가 학습한 예상 패턴과 비교해 볼 수 있습니다. 예상 패턴을 벗어난 이상 탐지를 그래프 차트를 통해 확인할 수 있습니다. 과거 데이터를 바탕으로 반복되는 패턴을 확인하고 향후 지표 값 예측에 활용할 수 있습니다.

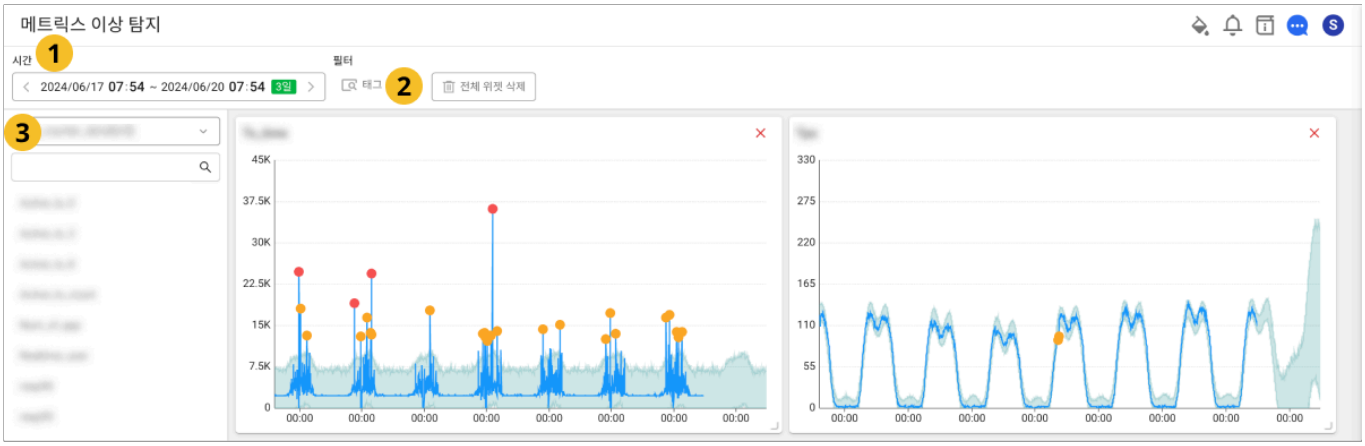
- 화면에 배치한 위젯은 다른 메뉴로 이동할 경우 저장되지 않고 초기화합니다.
- 패턴 표시와 이상치 표시 기능을 제외하면 **분석** > **메트릭스 차트** 메뉴와 유사합니다.
- **이상치 탐지(Anomaly Detection)** 경고 알림 기능의 기술 근간은 **메트릭스 이상 탐지**입니다. **이상치 탐지** 경고 알림 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

위젯 확인하기



- 1 밝은 색상의 그래프 영역은 AI가 분석한 예상 패턴입니다.
- 2 파랑색 그래프는 프로젝트의 메트릭 추이입니다.
- AI가 분석한 예상 패턴을 벗어나면 주황색, 빨간색의 단계로 그래프에 점을 표시합니다. 예상 패턴 범위를 크게 벗어난 값을 빨간색으로 표시합니다.

위젯 배치하기



1. 1 **시간**에서 원하는 시간 간격을 설정하세요. 최대 1개월 간격까지 설정할 수 있습니다.
2. 2 **필터**에서 메트릭의 범위를 선택하세요.
3. 아래 목록에서 모니터링하길 원하는 메트릭을 선택하세요.

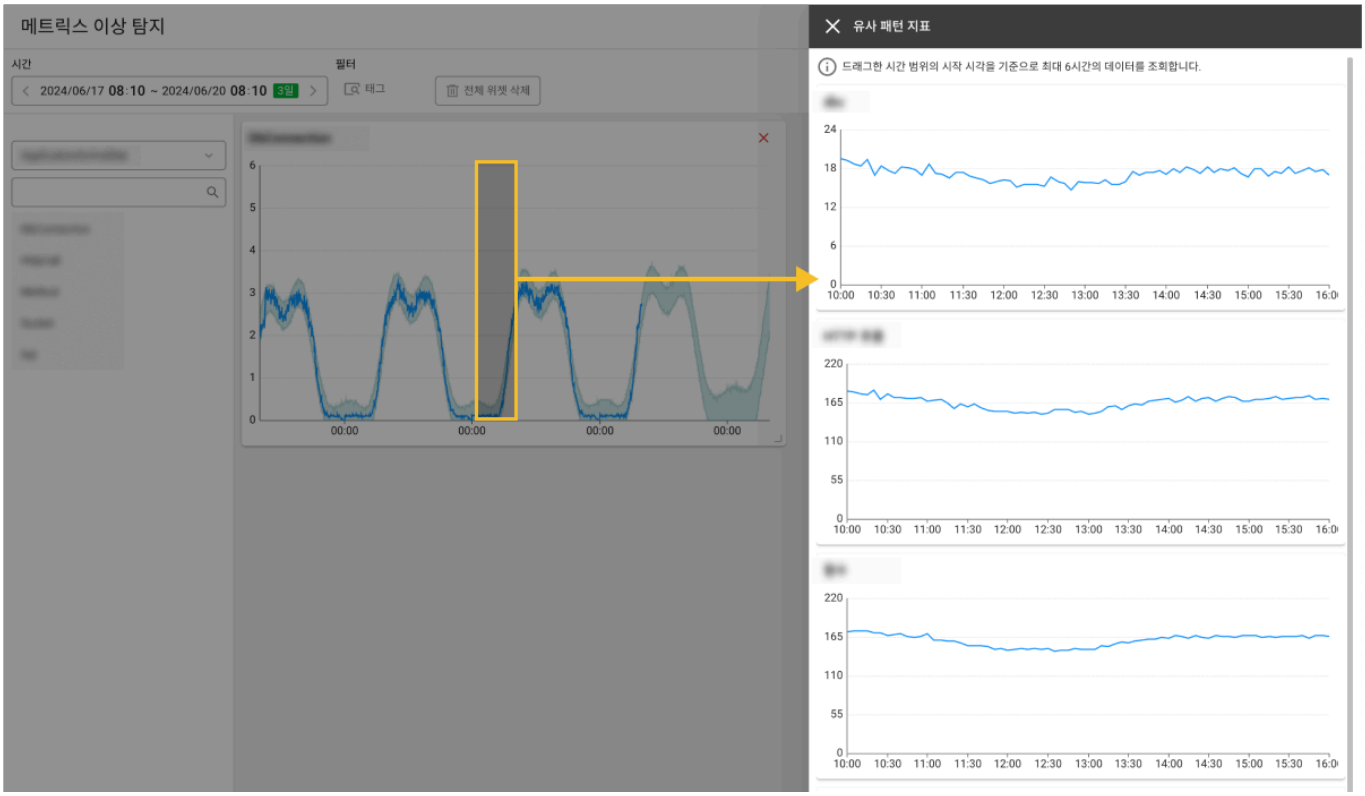
선택한 메트릭을 화면 오른쪽에 배치합니다.

- 모니터링 대상을 선택해 메트릭을 구분해서 확인하려면 2 **태그**를 선택하세요. 화면 오른쪽에 태그 선택 목록이 나타납니다. 원하는 항목을 선택한 다음 위젯을 추가하세요.
- 화면에 배치한 위젯은 **시간** 또는 **태그** 값을 변경해도 차트에 반영되지 않습니다.
- 화면에 배치한 위젯을 삭제하려면 위젯 오른쪽 위에 ✕ 버튼을 선택하세요.
- 위젯의 왼쪽 위를 선택한 상태에서 드래그해 위젯 위치를 변경할 수 있습니다.
- 위젯 오른쪽 아래를 선택한 상태에서 드래그해 위젯 크기를 조절할 수 있습니다.

위젯 삭제하기

화면에 배치한 모든 위젯을 삭제하려면 화면 위에 위치한 **전체 위젯 삭제** 버튼을 선택하세요.

연관 지표 확인하기



위젯에서 차트의 일부 영역을 드래그하세요. 화면 오른쪽에서 **유사 패턴 지표** 창이 나타납니다. 드래그한 시간 범위의 시작 시간을 기준으로 최대 6시간의 연관 지표를 조회할 수 있습니다.

트랜잭션 추적 개요

트랜잭션이란?

사용자 브라우저의 요청을 처리하기 위한 서버 사이드의 LUW(Logical Unit of Work)를 말합니다. 개별 웹서비스(URL) 요청에 대한 처리 과정이 바로 트랜잭션입니다. 웹 애플리케이션에서 트랜잭션은 웹서비스(URL)에 대한 HTTP Request를 받아 Response를 반환하는 과정입니다.

애플리케이션의 성능은 이 트랜잭션들의 성능으로 요약할 수 있습니다. 트랜잭션 성능은 트랜잭션 시작에서부터 종료 시점, 응답시간 및 자원 사용량 혹은 트랜잭션 호출자 속성 등의 정보를 포함합니다.

기본적으로 트랜잭션 응답 분포와 트랜잭션 통계를 통해서 트랜잭션 성능을 분석할 수 있습니다.

트랜잭션의 이름

트랜잭션의 이름은 URL입니다. 단 Get 파라미터(Query String)는 제외됩니다.

브라우저 요청

```
http://www.whatap.io/hr/apply.do?name='kim'
```

트랜잭션 이름

```
/hr/apply.do
```

> ⓘ 와탭에서는 **웹서비스 이름**과 **트랜잭션 이름**을 혼용하고 있습니다. 서비스 특정 URL과 그에 대한 요청을 처리하기 위한 모듈로 볼 수 있습니다. 트랜잭션 요청에 대한 처리 하나를 의미하기 때문에 둘의 이름은 동일하게 URL이라고 할 수 있습니다.

트랜잭션 이름 정규화

MSA 기반의 시스템이 발전하면서 URL + ? 인자 파라미터 형식보다 URL 패스에 파라미터를 넣는 방식을 많이 사용하게 됩니다.

```
http://www.whatap.io/hr/kim/apply.do
```

이렇게 패스 파라미터를 그대로 트랜잭션 이름으로 사용하게 되면 통계적 관점의 성능 분석이 어렵습니다. 이를 정규화할 필요가 있습니다. 와탭은 정규화를 위한 옵션과 기능을 제공합니다.

whtap.conf

```
trace_normalize_urls=/hr/{name}/apply.do
```

위와 같이 설정하면 트랜잭션 이름이 /hr/kim/apply.do → /hello/:name/apply.do로 치환되어 수집됩니다. 만약 대상 URL 설정은 그대로 두고 기능만 off 하고자 한다면 다음과 같이 옵션을 지정할 수 있습니다. 기본값은 `true` 입니다.

whatap.conf

```
trace_normalize_enabled=false
```

트랜잭션 조회

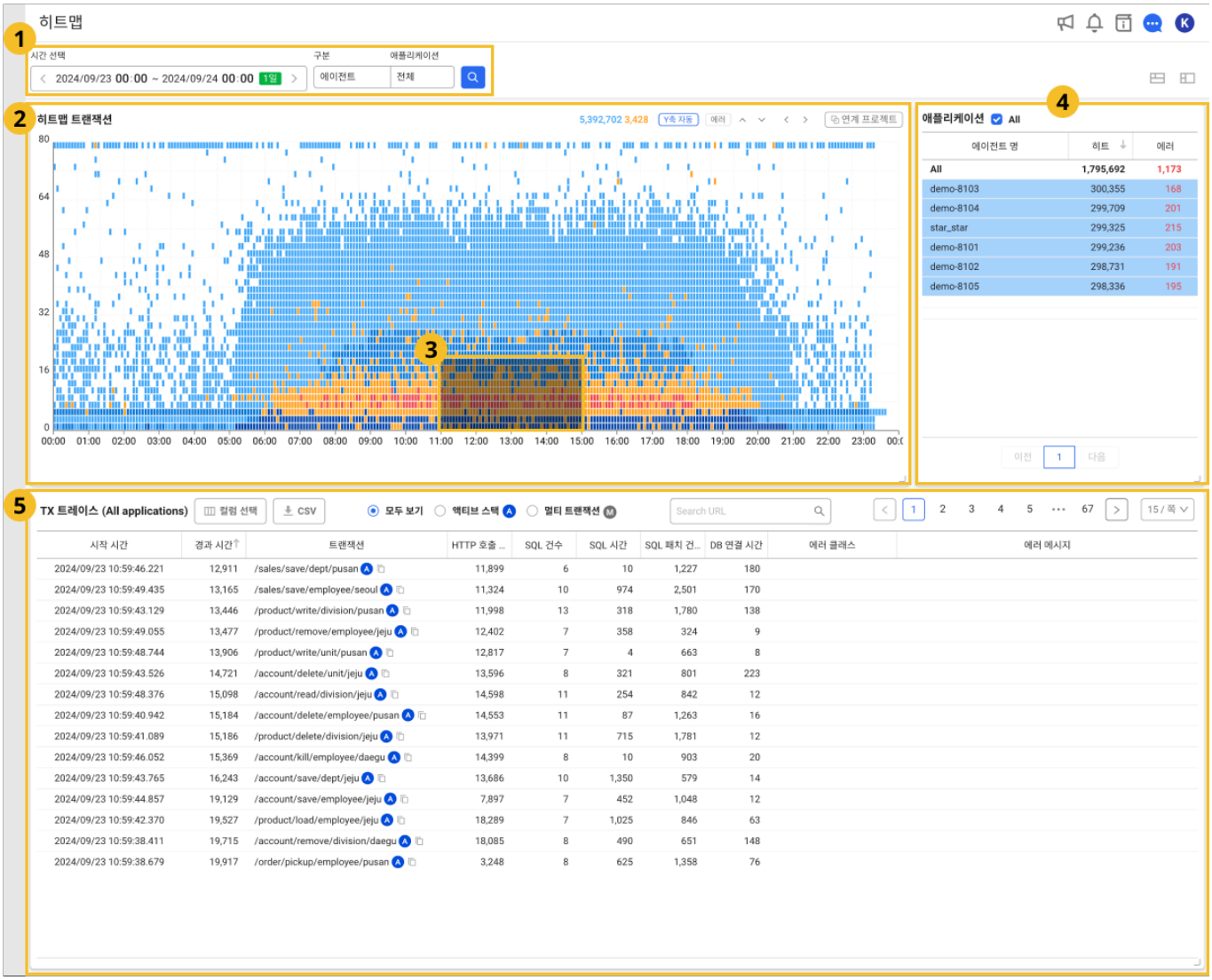
히트맵

홈 화면 > 프로젝트 선택 > **분석** > **히트맵**

와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 다음 **분석** > **히트맵** 메뉴를 선택하세요.

> **히트맵**은 **대시보드** > **애플리케이션 대시보드**의 **히트맵** 위젯을 통해 접근할 수도 있습니다.

상세 분석



1 에이전트 영역

현재 프로젝트와 연결된 에이전트를 선택해 필터링할 수 있습니다. 버튼을 선택하면 2 영역에서 **히트맵 트랜잭션** 차트를 확인할 수 있습니다.

2 히트맵 트랜잭션 차트

지연 문제가 있는 트랜잭션이 포함된 경우 **히트맵 트랜잭션** 차트에서 주황색이나 붉은색으로 표시됩니다. 정상 트랜잭션만 포함된 경우 파란색으로 표시됩니다.

3 선택 영역

실시간 히트맵 차트를 드래그하면 선택 영역의 4 **애플리케이션** 목록과 트랜잭션 정보를 확인할 수 있는 5 **TX 트레이스** 목록이 나타납니다.

4 애플리케이션 목록

차트에서 선택된 영역에 포함되는 트랜잭션 수와 에러 수를 확인할 수 있습니다. 목록 중에서 원하는 애플리케이션을 선택하면 해당 애플리케이션의 5 **TX 트레이스** 목록에 세부 정보가 나타납니다.

5 TX 트레이스 목록

TX 트레이스 목록은 세부 정보를 가지고 있습니다. 목록에서 원하는 트랜잭션을 선택하면 **트랜잭션 정보** 창이 나타납니다. **트랜잭션 정보** 창에서 해당 트랜잭션의 트레이스 상세 분석을 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

히트맵 패턴의 이해

히트맵은 트랜잭션의 종료시간은 X축, 응답시간은 Y축으로 한 분포 차트입니다. 정상적인 웹 애플리케이션이라면 수 초 이하 구간에 집중된 분포를 보입니다.

hitmap-pattern

히트맵 라인 분석하기

- **세로줄(LOCK 현상) 패턴**

  트랜잭션 처리 중 일시적인 락(Not only DB Lock)이 발생하면 이로 인해 처리를 대기합니다. 락이 해소되면 처리 대기 중 트랜잭션들은 비슷한 시간대에 함께 종료됩니다. 그러면 다음과 같이 세로로 줄이 만들어집니다.

  hitmap-lock

  세로줄 패턴으로 락을 감지하는 것은 매우 강력한 개념입니다. 특히 마이크로 서비스 아키텍처에서는 백엔드 시스템에서

발생하는 LOCK도 동일하게 감지될 수 있습니다.

hitmap-front-api-db

Front 애플리케이션의 응답 패턴 세로줄은 Back-End 시스템이 사용하는 DB에서 락이 발생한 경우에도 감지됩니다.

- **가로줄(타임아웃) 패턴**

  10초 타임아웃 조건에서 해당 자원이 부족하면 트랜잭션들은 10초 대기 후 타임아웃 에러가 발생합니다. 이때 히트맵 10초 부근에 다음과 같이 가로줄이 생깁니다.

  hitmap-timeout

  타임아웃 이후 재시도하는 로직이 있다면 그림처럼 가로라인이 10초 단위로 반복됩니다. 다음은 실제 **장애 상황**의 히트맵입니다.

  hitmap-error

  (1) 구간에서 응답시간이 급증했고 (2) 구간의 빨간 라인은 전형적인 **가로 라인 패턴**입니다. (1) 구간 부하 발생으로 ConnectionPool이 소진되고 (2) 구간은 ConnectionPool 부족으로 2차 **타임아웃** 장애가 발생한 상황입니다.

패턴 분석 활용

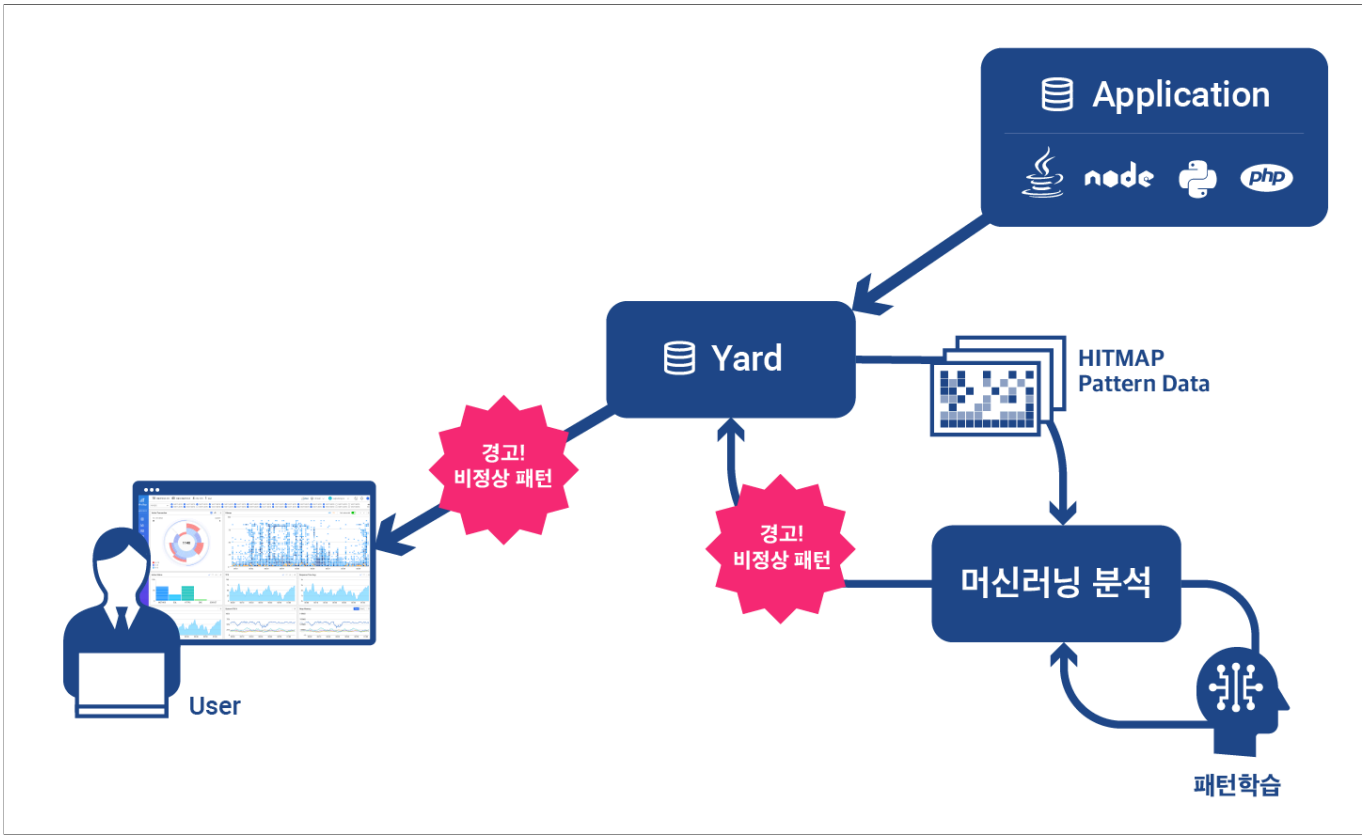
트랜잭션 응답분포에 줄이 보인다는 것은 병목이 있다는 것입니다. 일시적인 락킹이면 세로줄이 그 병목이 타임아웃으로 빠지면 가로 줄이 만들어집니다.

hitmap-pattern-analysis

문제를 분석할 때 라인에 포함된 트랜잭션만을 선택적으로 분석해서 문제를 빠르게 찾아낼 수 있습니다.

머신러닝 기반 응답 패턴 분석

머신러닝 기술을 통해 히트맵 패턴을 분석 후 비정상 여부를 자동 감지해 경고를 발행하는 기능입니다.



비정상 패턴 예시

월 수백 TB의 성능 데이터로부터 비정상 패턴을 학습하고 학습된 비정상 패턴과 유사한 패턴이 발생하는 경우 이에 대한 알람을 발행합니다.

- **세로줄 패턴**

  hitmap-abnormal1

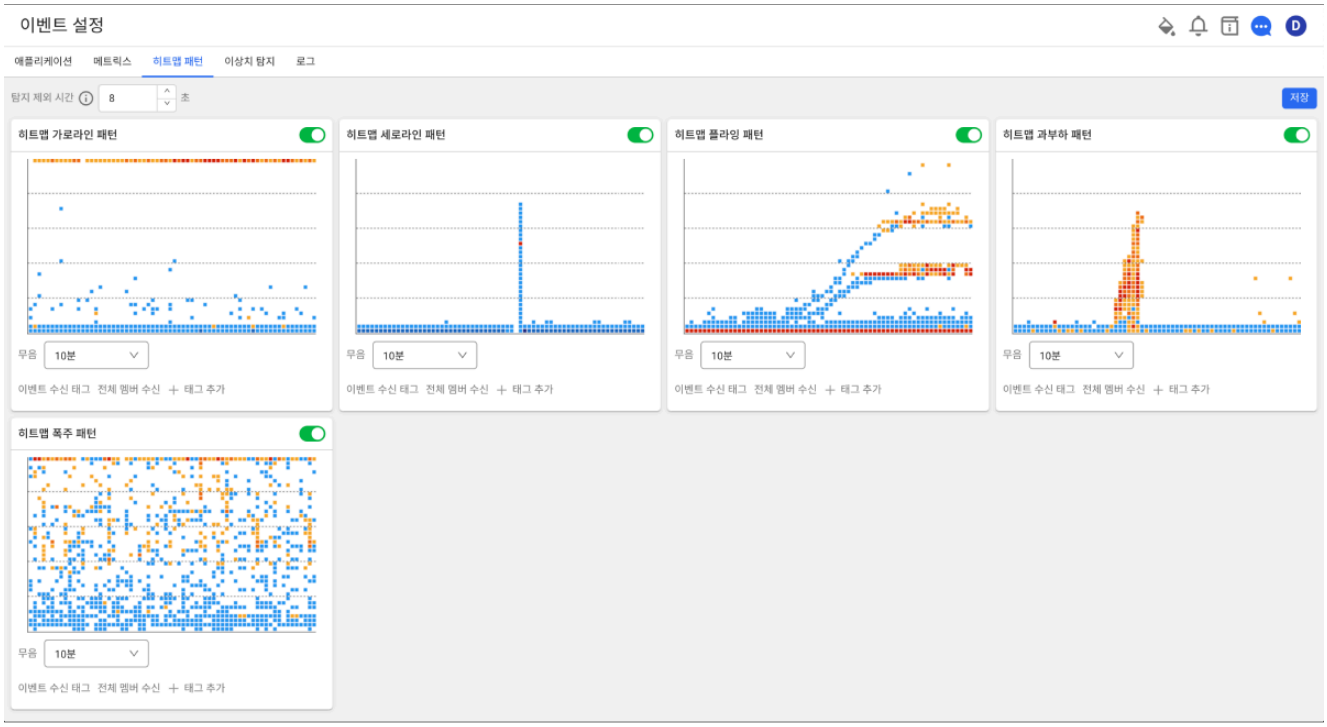
- **가로줄 패턴**

  hitmap-abnormal2

- **복합 패턴**

  hitmap-abnormal3

- **히트맵 알람**

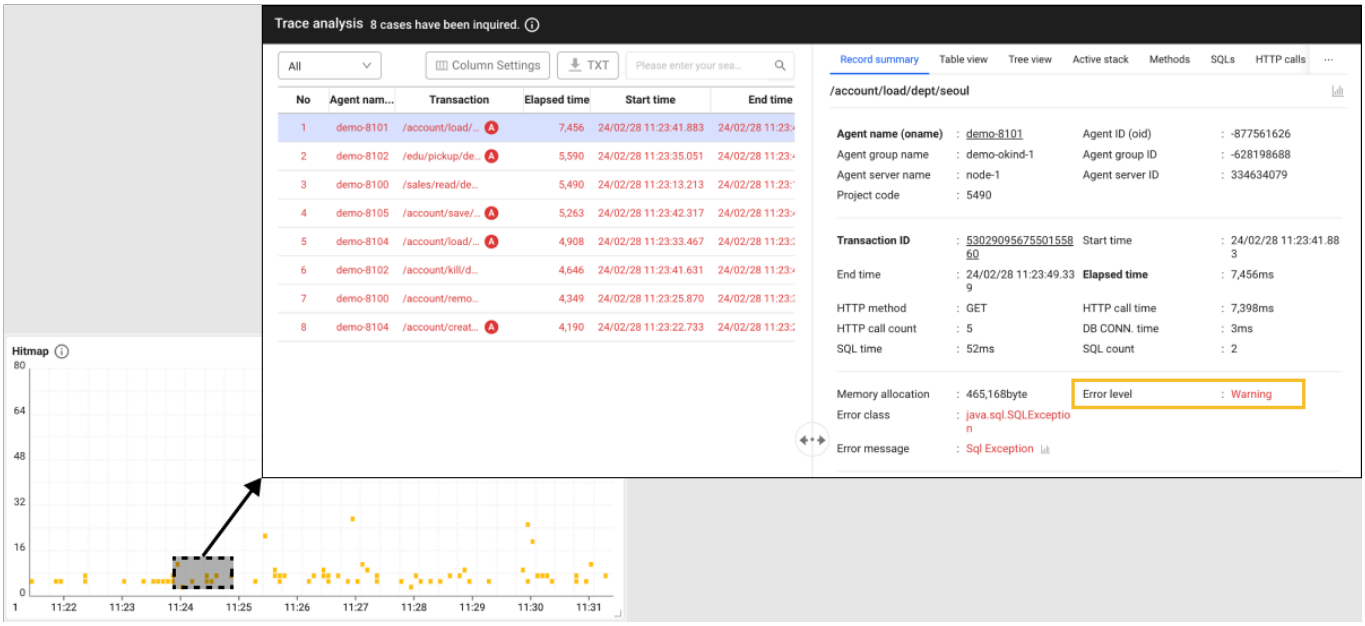


ⓘ **히트맵 트랜잭션** 차트를 분석하는 방법에 대한 자세한 설명은 다음 문서를 참조하세요.

Java 에이전트의 예외 처리(WARNING) 기준

Java 언어를 사용하는 애플리케이션 환경에서 발생하는 에러를 Java 에이전트가 예외 처리하는 기준에 대해서 안내합니다. 다음 내용은 Java 에이전트 v2.2.27 기준으로 작성되었습니다.

서비스 화면에서 에러 표시



히트맵 위젯, **트레이스 분석** 창을 통해서 확인할 수 있는 **에러 단계**는 대부분 WARNING 수준에 해당합니다.

Spring ExceptionHandler 통과 시 에러 처리

Spring 프레임워크에서는 `org.springframework.web.servlet.DispatcherServlet.processHandlerException` 메소드를 통과하는 경우 에러로 처리합니다. 다만 개발자가 Try-Catch 문을 사용하는 경우 `ExceptionHandler` 를 통과하지 않으므로 와탭에서는 에러 처리하지 않습니다.

Status Code 400 이상 에러 처리

HTTP 응답 코드가 400 이상인 경우에는 예외 클래스(Exception Class)가 발생하지 않더라도 에러로 처리합니다.

JDBC driver 호출 시 에러 처리

JDBC 연결 호출이나 SQL 실행 호출 시에 발생하는 에러는 에러로 처리됩니다.

HTTP call 에러 처리

HTTP 호출 중 발생하는 에러는 `SLOW_HTTPC` , `httpc error` 등의 형태로 표시됩니다.

와탭에서 정의한 에러 처리

와탭 모니터링 서비스에서는 특정한 조건에 따라 에러로 처리합니다.

- **TOO_MANY_RECORDS**: fetch 건수 10,000건 이상

  관련한 에이전트 옵션은 다음 문서를 참조하세요. `profile_error_jdbc_fetch_max=10000`
- **SLOW_SQL**: SQL 시간 30,000ms 이상

  관련한 에이전트 옵션은 다음 문서를 참조하세요. `profile_error_sql_time_max=30000`
- **SLOW_HTTPC**: HTTP call 경과 시간이 10,000ms 이상

  관련한 에이전트 옵션은 다음 문서를 참조하세요. `profile_error_httpc_time_max=10000`
- **httpc error**: HTTP call status code 400 이상인 경우

  관련한 에이전트 옵션은 다음 문서를 참조하세요. `httpc_status_error_enable=true`
- **status error**: 트랜잭션 status code 400 이상인 경우

  관련한 에이전트 옵션은 다음 문서를 참조하세요. `transaction_status_error_enable=true`

트랜잭션 에러 단계 INFO 처리

에이전트 설정을 통해서 일부 에러를 정상 상태로 표시하거나 무시할 수 있습니다. **히트맵** 위젯에서 표시되는 레벨은 INFO(파란색)입니다.

- `biz_exceptions` : 레코드 에러 표시

  whatap.conf

  ```
  biz_exceptions=io.home.test.baseapp.base.exception.custom.CustomParameterizedException
  ```
- `ignore_exceptions` : 레코드 에러 미표시

whatap.conf

```
ignore_exceptions=org.springframework.web.util.NestedServletException
```

- `status_ignore`

  whatap.conf

  ```
  status_ignore=408,500,200
  ```

- `httpc_status_ignore`

  whatap.conf

  ```
  httpc_status_ignore=403,401
  ```

- `status_ignore_set`

  whatap.conf

  ```
  status_ignore_set=/api/posts/test/timeout:200,/error:500
  ```

- `httpc_status_url_ignore_set`

  whatap.conf

  ```
  httpc_status_url_ignore_set=/api/posts/test/timeout:408,/api/apache/httpclient/posts/timeout:500
  ```

> ⓘ 에이전트 설정은 whatap.conf 파일을 편집하거나 **관리** > **에이전트 설정** 메뉴에서 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

메소드 예외 처리

`hook_method_patterns`, `hook_service_patterns` 옵션을 통해 설정한 메소드가 수행 중 Exception을 발생시키는 경우, 해당 예외는 에러로 처리됩니다.

Try-Catch 구문을 사용하여 메소드 내부에서 Exception을 처리한 경우, 자바 에이전트는 해당 Exception을 수집 대상에서 제외하여 에러로 처리하지 않습니다.

트랜잭션 프로파일

트랜잭션 트레이싱이란?

트랜잭션 성능이 트랜잭션 시작과 종료 사이의 요약 지표들이나 속성들을 의미한다면 트랜잭션 트레이스는 트랜잭션이 수행되는 과정 중인 스텝들을 추적하는 것입니다. 트랜잭션이 느리거나 오류가 있다면 그 원인을 추적하기 위해서 수행 이력을 스텝별로 추적해야 합니다. 이것을 **트랜잭션 트레이싱**이라고 합니다.

트랜잭션 성능 추적을 위해 수집하는 스텝의 종류는 다음과 같습니다.

- **DB 연결 스텝** START-END

  RDB에 대한 연결에 대한 성능을 포함합니다. 스텝 정보에는 이름, 응답시간, 에러를 포함합니다.

- **SQL 스텝** START-END

  JDBC SQL에 대한 성능을 포함합니다. 스텝 정보에는 연결 정보, SQL문, 에러가 포함되어 있습니다.

- **HTTP Call 스텝** START-END

  외부 http 서비스 호출에 대한 성능을 포함합니다. 스텝 정보에는 url, host, port, 응답시간, 에러가 포함됩니다.

- **Message 스텝** ADD

  트레이스를 수집하는 과정에서 비정형적인 모든 구간에 대한 이력을 수집할 때 메시지 스텝을 사용합니다.

- **SOCKET 스텝** ADD

  Socket 오픈을 표현하는 스텝입니다.

- **METHOD 스텝** START-END

  메소드 응답시간을 추적합니다.

- **ACTIVE STACK 스텝** START-END

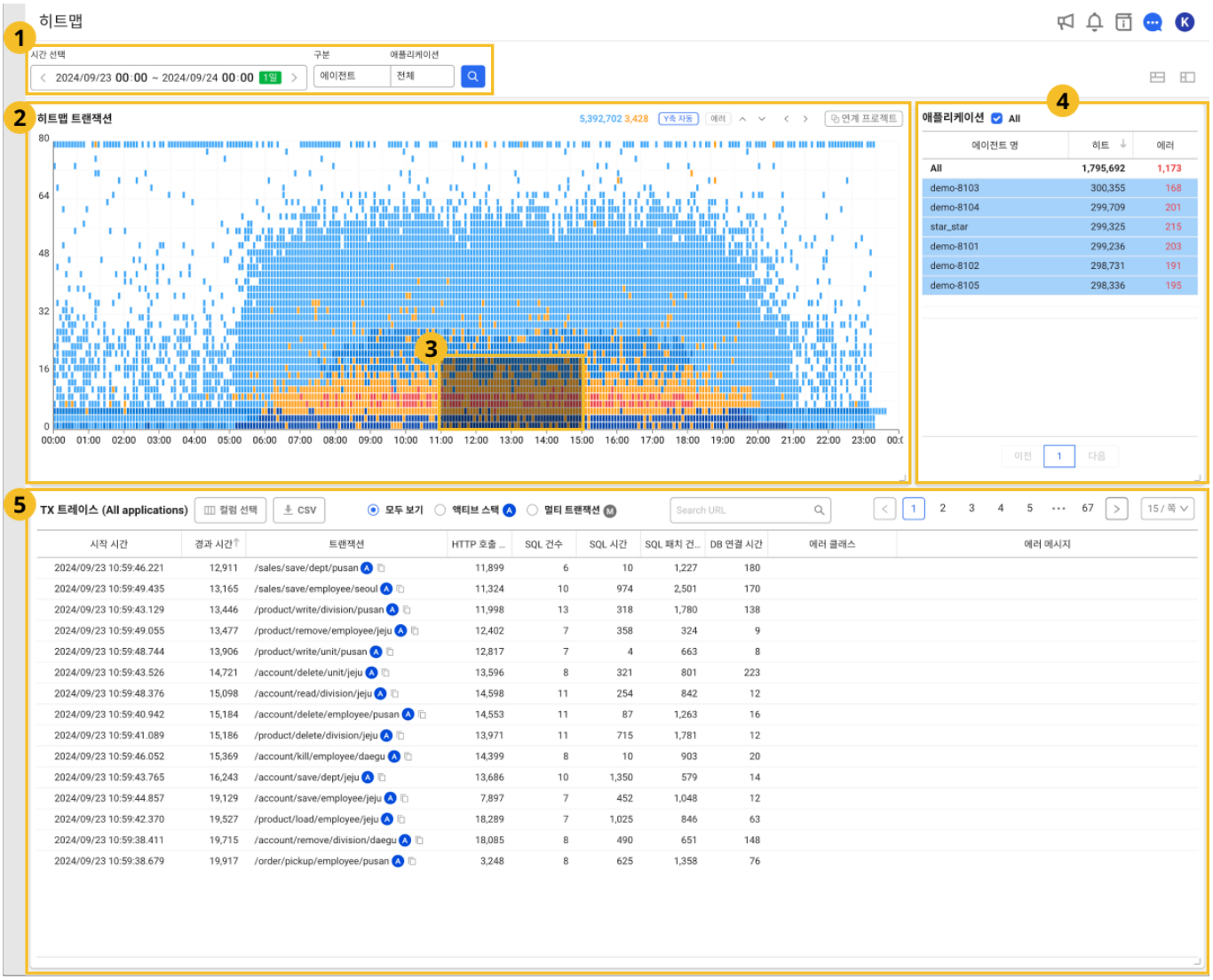
  액티브 스택에 대한 정보를 포함합니다. 별도 스레드가 생성하여 트레이스에 추가하는 방식으로 수집합니다.

> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면 연계한 프로젝트의 성능 지표를 조회할 수 있습니다. 연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

ⓘ 와탭 모니터링 서비스에서 **스텝(Step)**은 **스팬(Span)**과 같은 뜻으로 사용됩니다.

상세 분석

홈 화면 > 프로젝트 선택 > **분석** > **히트맵**



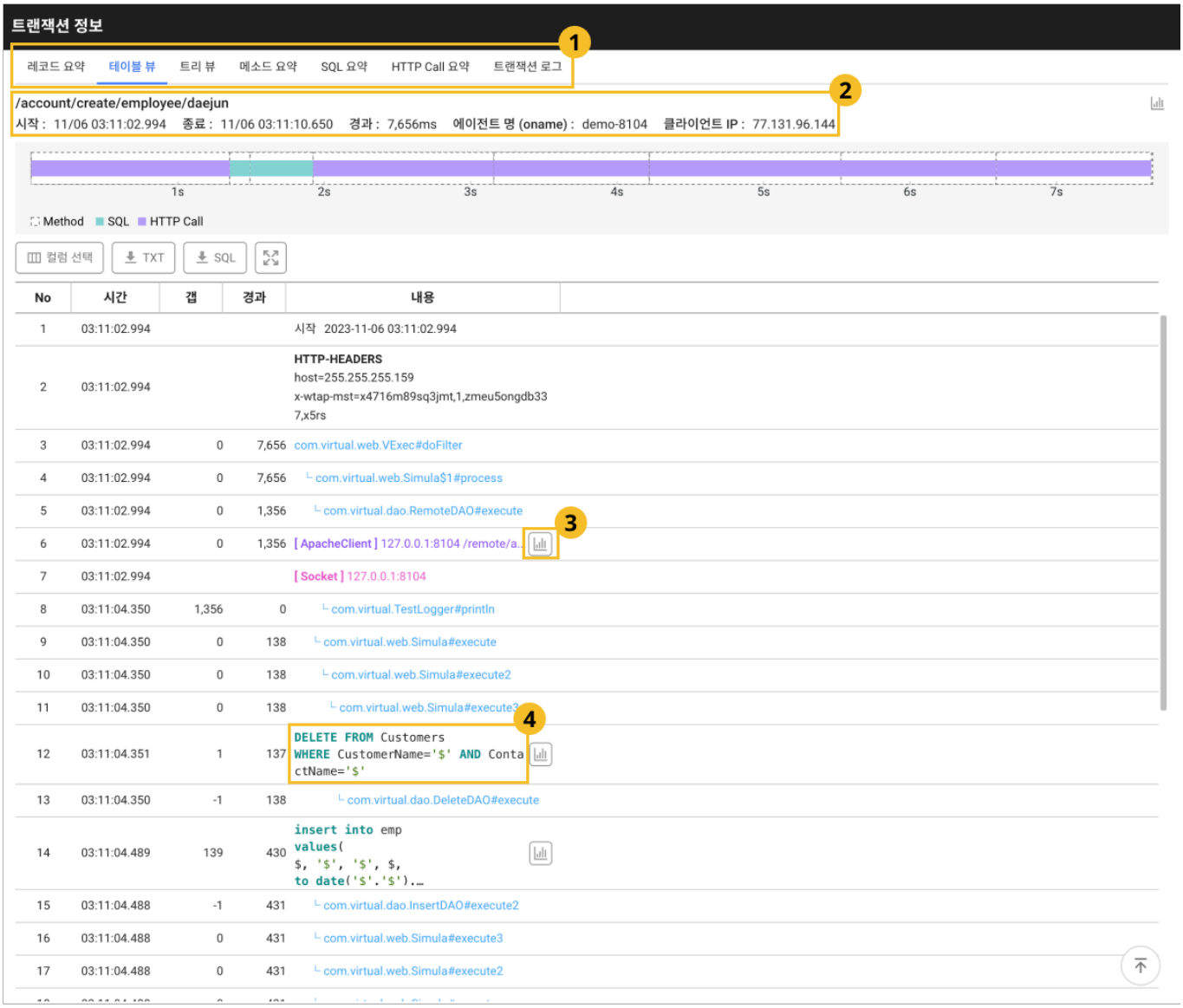
1. 1 **시간 선택**에서 히트맵을 조회할 기간을 선택하세요.
2. 1 **구분, 애플리케이션**에서 조회 대상을 선택하세요.

3. 버튼을 선택하세요.
4. 2 **히트맵 트랜잭션**에서 원하는 영역을 3과 같이 드래그하세요.
5. 5 **TX 트레이스**에 목록이 나타나면 분석하기 원하는 트랜잭션 항목을 선택하세요.

트랜잭션의 성능 분석을 위한 클라이언트 정보 등의 속성, 트랜잭션의 처리 성능, 각 구간별 상세 수행 이력 등을 확인할 수 있는 **트랜잭션 정보** 창이 나타납니다.

> - **히트맵** 메뉴의 기능에 대한 자세한 내용은 다음 문서를 참조하세요.
> - 애플리케이션 대시보드의 히트맵 위젯에 대한 자세한 내용은 다음 문서를 참조하세요.

트랜잭션 정보 창 안내



- 1 영역에서는 스텝 정보를 다양한 형식으로 확인할 수 있습니다. 원하는 형식의 탭을 선택하세요.
- 2 영역에서는 트랜잭션에 대한 기본 정보를 확인할 수 있습니다.
- 3 : 해당 URL 또는 각 수행 구간의 통계 데이터 창이 나타납니다. 통계 데이터 창의 그래프 차트에서 원하는 시간을 클릭하면 **통계** 또는 **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간 기준으로 통계 데이터를 조회할 수 있습니다.

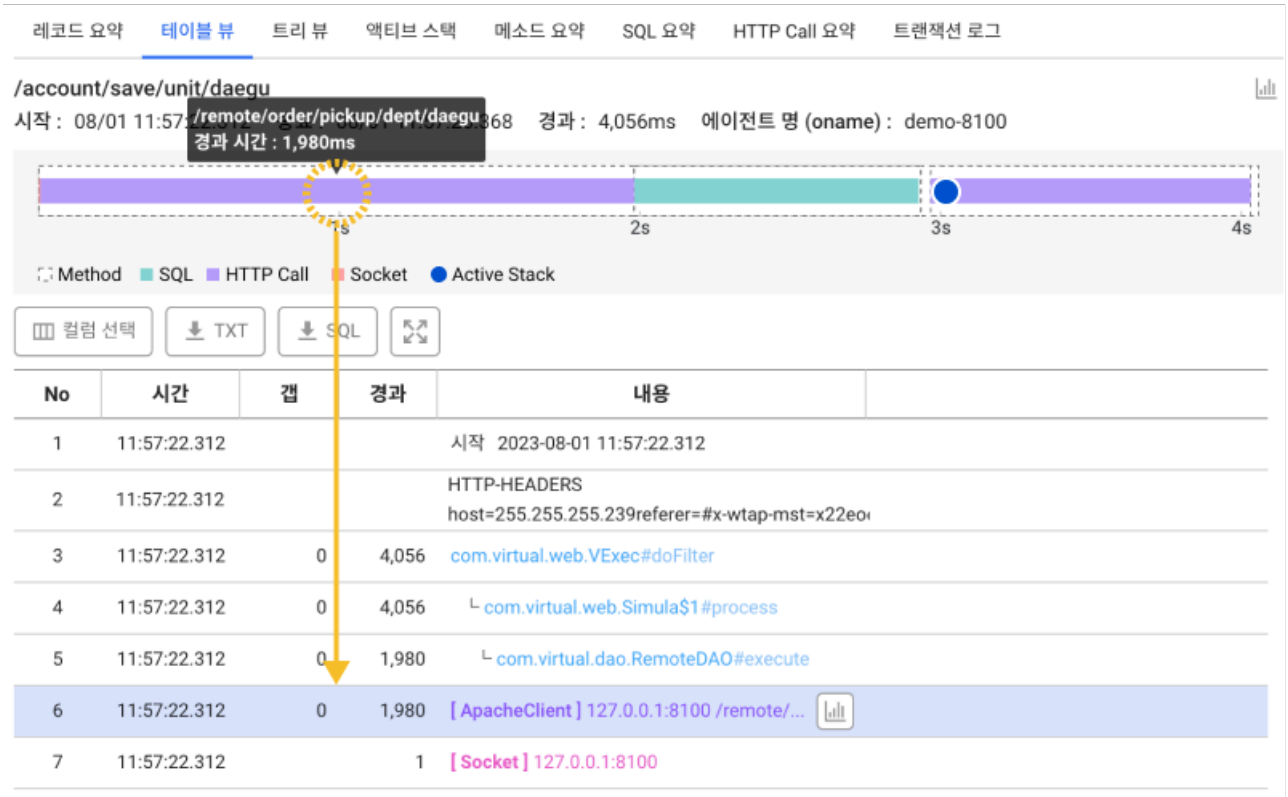
> ◦ **통계** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
> ◦ **트랜잭션 검색** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.

- 4 SQL 스텝을 선택하면 SQL 변수와 HTTP 쿼리를 조회할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

테이블 뷰

테이블 뷰 탭에서는 트랜잭션의 수행 과정을 시간의 순서대로 확인할 수 있습니다.

- 트랜잭션을 수행 구간별로 분류해 다이어그램을 통해 확인할 수 있습니다. 전체 경과 시간 중 구간별 소요된 시간, 가장 오래 소요된 구간을 빠르게 파악할 수 있습니다.
- 다이어그램의 각 구간을 선택하면 해당 스텝이 위치한 테이블 목록으로 이동합니다.

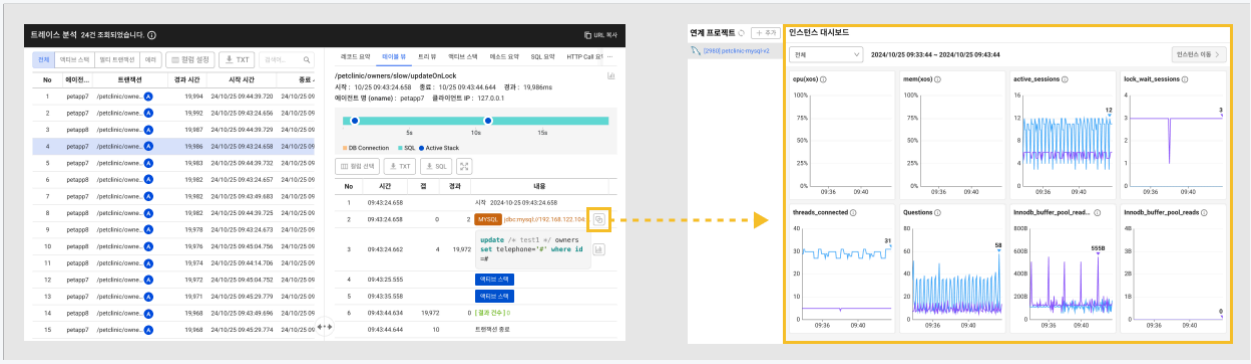


- 파란색 원(🔵)이 위치한 영역은 액티브 스택이 수집된 순간입니다. 파란색 원을 선택하면 **액티브 스택** 버튼이 위치한 테이블 목록으로 이동할 수 있습니다. **액티브 스택** 버튼을 선택하면 해당 구간 동안 수행된 스텝 정보를 확인할 수 있습니다.

> ◦ 트랜잭션 목록에서 A 아이콘이 표시된 항목에서 확인할 수 있습니다.
> ◦ 액티브 스택에 대한 자세한 내용은 다음 문서를 참조하세요.

- **컬럼 선택**: 테이블 목록에 컬럼으로 메모리 누적 정보 및 CPU 누적 정보를 추가하거나 감출 수 있습니다. 컬럼 정보는 다음과 같습니다.
  - **No**: 스텝의 발생 순서
  - **시간**: 각 스텝의 시작 시각
  - **갭**: 직전 스텝의 시작 시각부터 현재 스텝으로 넘어가기까지 대기 시간, 외부 요인으로 지연될 경우 경과 시간과 차이가 있을 수 있습니다.
  - **경과**: 각 메소드 시작부터 종료까지 총 소요 시간
  - **내용**: 해당 스텝의 세부 수행 내용
- **TXT**: 트랜잭션 기본 정보 및 구간별 수행 정보를 TXT 형식의 파일로 다운로드할 수 있습니다.
- **SQL**: 트랜잭션 기본 정보 및 SQL 수행 정보를 TXT 형식의 파일로 다운로드할 수 있습니다.
  - **치환값 포함 다운로드**: 바인드 변수값을 원래의 값으로 치환하여 SQL 통계 데이터를 다운로드합니다. 보안키(`paramkey`)가 설정되어 있다면 보안키 입력창이 나타납니다. 보안키를 입력를 입력해야 다운로드할 수 있습니다. 보안키와 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.
  - **치환값 미포함 다운로드**: 바인드 변수값을 치환하지 않은 상태로 SQL 통계 데이터를 다운로드합니다.
- : 트랜잭션 요약 정보와 다이어그램을 감추고 테이블 목록만 확인할 수 있습니다. 버튼을 선택하면 감춰진 정보를 다시 표시합니다. 테이블 목록이 긴 경우 이 기능을 이용하면 유용합니다.
- SQL 스텝을 선택하면 파라미터를 조회할 수 있는 **SQL** 창이 나타납니다. 자세한 내용은 다음 문서를 참조하세요.

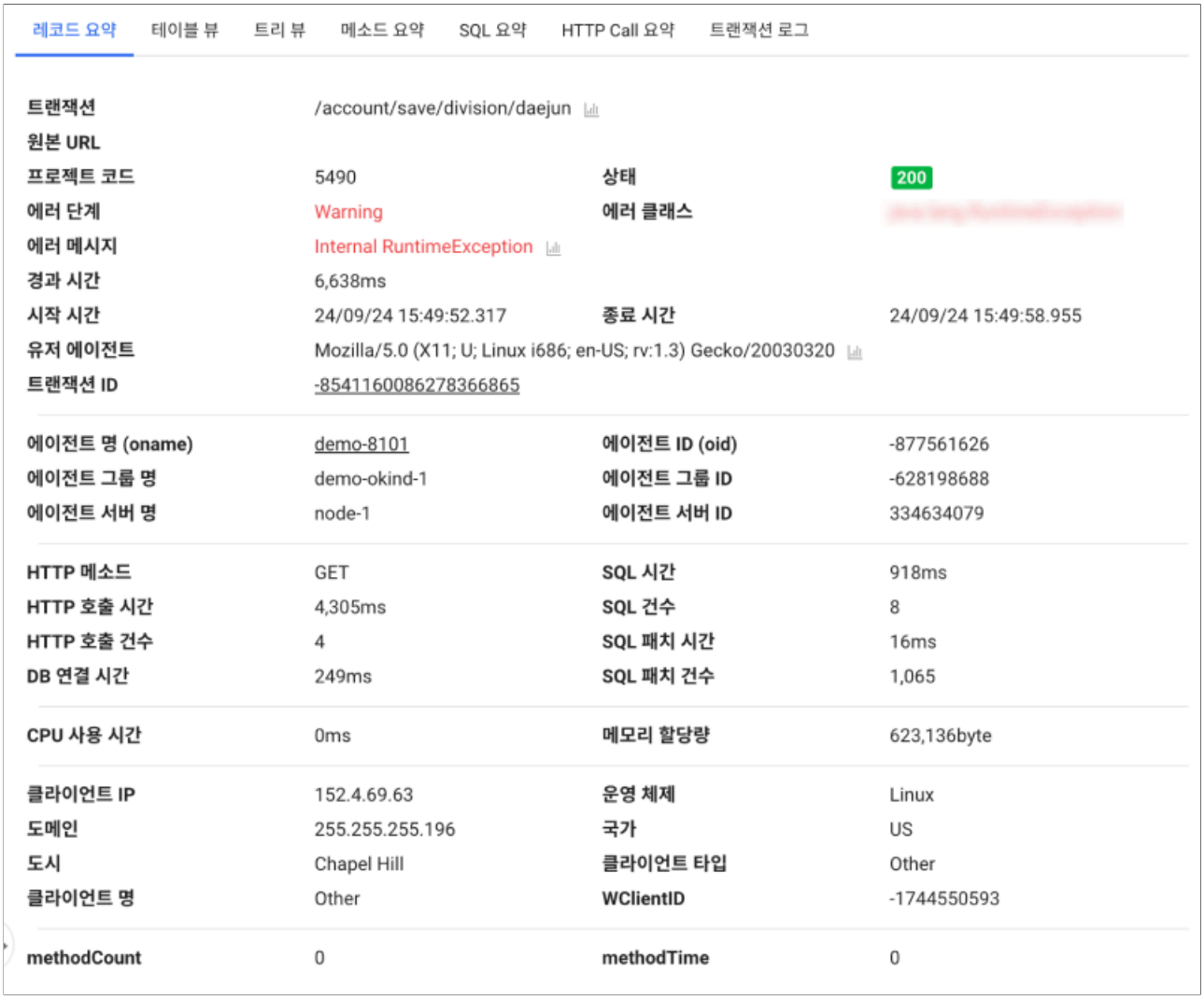
> **연계 프로젝트 관리** 메뉴를 통해 데이터베이스 프로젝트를 추가하면, **DB Connection 스텝**에서 연계한 프로젝트의 성능 지표를 조회할 수 있습니다.



연계 프로젝트 추가 및 분석에 대한 자세한 내용은 다음 문서를 참조하세요.

레코드 요약

트랜잭션의 기본 정보를 확인하려면 **레코드 요약** 탭을 선택하세요. 트랜잭션의 기본 정보 및 해당 트랜잭션이 수행된 에이전트 관련 정보, 메모리 할당 사용량, 클라이언트 관련 정보를 확인할 수 있습니다.



분류	속성	설명
트랜잭션	트랜잭션	애플리케이션에서 실행된 API 또는 URL 호출, 해당 트랜잭션의 경로입니다. 정규 표현식을 사용해 일정한 형식으로 변환합니다.
	원본 URL	실제로 호출된 URL 경로입니다. 변수를 포함하지 않은 구체적인 요청 경로입니다. 원본 URL이 있을 경우 원본 URL 표시, 원본 URL이 없을 경우 표시하지 않습니다.
	프로젝트 코드	와탭 모니터링 서비스에 등록한 프로젝트의 식별 코드입니다.
	상태	해당 트랜잭션의 HTTP 응답 상태 코드입니다.

분류	속성	설명
	에러 단계	해당 트랜잭션에서 발생한 에러 레벨입니다. Warning 또는 Critical로 표시됩니다.
	에러 클래스	해당 트랜잭션에서 발생한 에러 관련 클래스입니다.
	에러 메시지	해당 트랜잭션에서 발생한 에러 메시지입니다.
	경과 시간	트랜잭션 수행 시간입니다.
	시작 시간	트랜잭션 시작 시각입니다.
	종료 시간	트랜잭션 종료 시각입니다.
	유저 에이전트	클라이언트 관련 정보를 추출하는데 이용하는 브라우저 정보입니다.
	Referer	클라이언트가 브라우저에서 이전에 이용한 페이지 주소 또는 유입 경로입니다.
	트랜잭션 ID	해당 트랜잭션의 고유 식별자(ID)입니다.
	멀티 트랜잭션 ID	멀티 트랜잭션의 고유 식별자(ID)입니다. 다른 애플리케이션의 트랜잭션과 병렬로 실행된 경우 표시됩니다.
에이전트	에이전트 명 (oname)	에이전트 이름입니다.
	에이전트 ID (oid)	에이전트의 고유 식별자(ID)입니다.
	에이전트 그룹 명	`okind` 옵션으로 설정된 에이전트 그룹 이름입니다.
	에이전트 그룹 ID	에이전트 그룹의 고유 식별자(ID)입니다.
	에이전트 서버 명	`ondoe` 옵션으로 설정한 에이전트 서버 이름입니다.

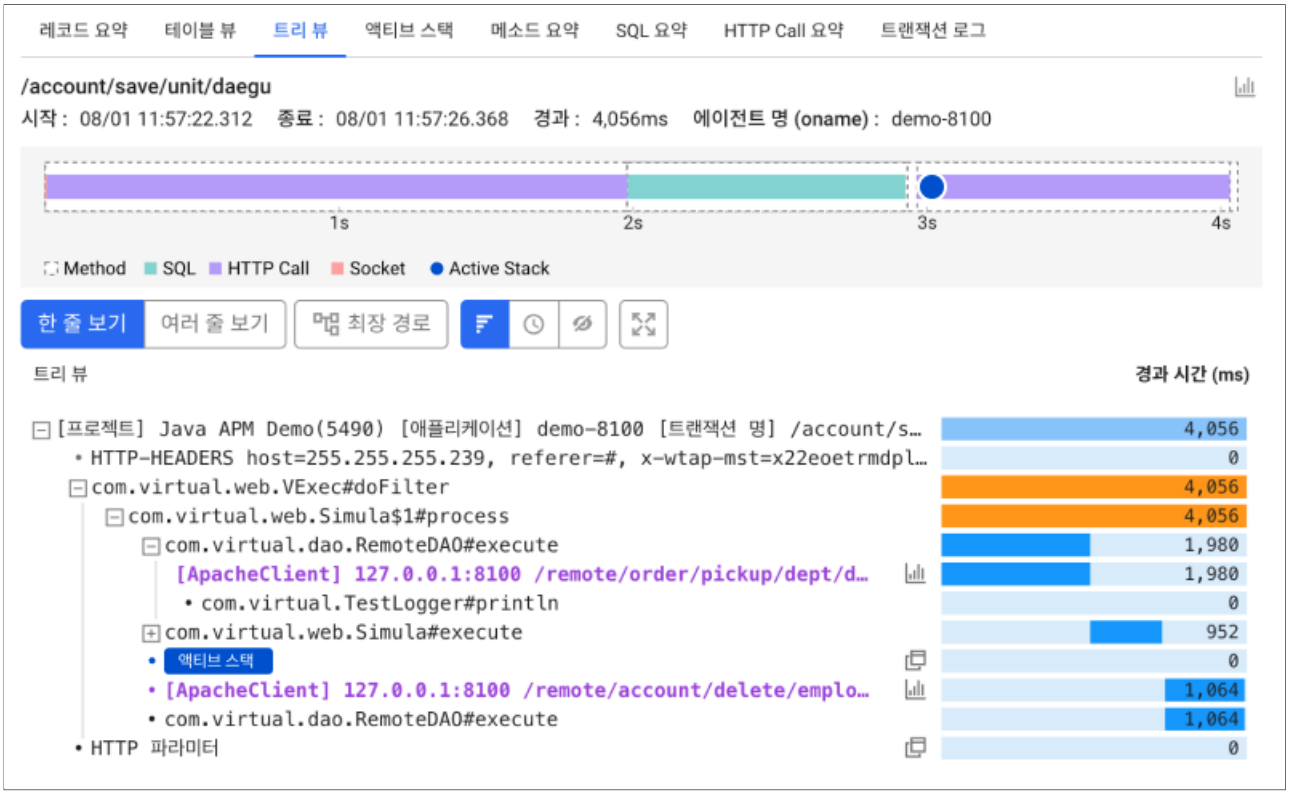
분류	속성	설명
	에이전트 서버 ID	에이전트 서버의 고유 식별자(ID)입니다.
스텝(Step)	HTTP 메소드	HTTP 메소드입니다. (GET, POST, PUT, HEAD 등)
	HTTP 호출 시간	외부 Http Call 시간입니다.
	HTTP 호출 건수	외부 HTTP Call 건수입니다.
	DB 연결 시간	데이터베이스에 연결되기까지 걸린 시간입니다.
	SQL 시간	SQL 수행 시간입니다.
	SQL 건수	SQL 수행 건수입니다.
	SQL 패치 시간	레코드를 조회하는 데 걸린 시간입니다. 중첩이 발생하거나 타 로직이 포함될 수 있습니다.
	SQL 패치 건수	SQL에서 데이터를 조회한 레코드 건수입니다.
자원(Resource)	CPU 사용 시간	트랜잭션 처리에 사용된 CPU 시간입니다.
	메모리 할당량	해당 트랜잭션에서 사용된 메모리 양입니다.
컨테이너	Pod 명	해당 트랜잭션을 처리한 Kubernetes Pod 이름입니다.
	컨테이너 ID	Pod 내에서 실행 중인 특정 컨테이너의 고유 식별자입니다.
클라이언트	클라이언트 IP	클라이언트의 IP 주소입니다.
	운영체제	브라우저가 실행되는 운영 체제 환경입니다.
	도메인	클라이언트가 접속한 IP 주소에 지정된 인터넷 주소입니다.
	국가	클라이언트의 국가 정보입니다.

분류	속성	설명
	도시	클라이언트의 국가 내 도시 정보입니다.
	클라이언트 타입	클라이언트가 이용한 브라우저 종류입니다.
	클라이언트 명	클라이언트가 이용한 기기 이름입니다.
	WClientID	클라이언트의 고유 식별자입니다.
Method	methodCount	트랜잭션에서 호출된 메소드의 수입니다.
	methodTime	메소드 실행에 소요된 시간입니다.

- 애플리케이션 종류나 설정, 스텝의 종류에 따라 수집하는 정보는 달라질 수 있습니다. **컨테이너** 항목은 애플리케이션이 컨테이너 환경에서 실행 중일 경우에만 표시됩니다.
- 에러 관련 항목은 해당 트랜잭션에서 에러가 발생할 경우 표시됩니다.
- 트랜잭션 속성 중 일부 항목은 통계 기능을 제공합니다. 버튼을 선택하면 시계열 차트를 통해 조회 시간 동안 발생한 트랜잭션 현황을 파악할 수 있는 창이 표시됩니다. 차트 내 막대 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동해 상세 조회할 수 있습니다.
- 와탭은 클라이언트와 관련한 정보를 기본 저장합니다. 사용자 데이터 수집과 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

트리 뷰

트랜잭션 수행 과정을 트리 형식으로 확인하려면 **트리 뷰** 탭을 선택하세요. 각 트랜잭션과 그에 속한 트레이스의 세부 정보, 트레이스의 시작 시간 및 소요 시간, 호출 관계를 확인할 수 있습니다. 다이어그램의 각 구간을 선택하면 해당 스텝이 위치한 트리뷰로 이동합니다.



- **한 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 한 줄로 표시해 트리 형식을 간격하게 정리할 수 있습니다.
- **여러 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 줄바꿈해 모두 표시합니다.
- **최장 경로**: 가장 긴 경로로 이동할 수 있습니다.
- **시간바 표시**: 경과 시간을 막대 형식의 차트로 표시합니다.
- **시간 표시**: 각 구간별 타임 스탬프, 갭, 경과 시간을 텍스트 형식으로 표시합니다.
  - 8초 이상: 초과 지연 상태로 빨간색으로 표현합니다.
  - 3초 이상 8초 미만: 지연 상태로 주황색으로 표현합니다.
  - 3초 미만: 정상 상태로 파란색으로 표현합니다.
- **시간 숨기기**: 시간 정보를 숨깁니다.
- : 트랜잭션 요약 정보와 다이어그램을 감추고 트리뷰만 확인할 수 있습니다. 버튼을 선택하면 감춰진 정보를 다시 표시합니다. 트리뷰 목록이 긴 경우 이 기능을 이용하면 유용합니다.
- : SQL 변수와 HTTP 쿼리를 조회할 수 있는 창이 나타납니다. 자세한 내용은 다음 문서를 참조하세요.

> 시작 및 소요 시간의 경우 트랜잭션 호출 환경에 따라 발생하는 시차를 상위 트랜잭션 내 트레이스와 매핑을 통해 보정하여 표현하기 때문에 실제 수집된 시간 데이터와 차이가 발생할 수 있습니다.

액티브 스택



액티브 스택 탭을 선택하면 해당 트랜잭션에서 수집한 액티브 스택 목록을 확인할 수 있습니다.

- 왼쪽의 **액티브 스택** 목록에서 수집 시점(No, 시간)을 선택하면 오른쪽에 해당 시점의 스택 정보를 표시합니다.
- **스택**: 액티브 스택 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

> - 트랜잭션 목록에서 A 아이콘이 표시된 항목에서 액티브 스택을 확인할 수 있습니다.
> - 액티브 스택에 대한 자세한 내용은 다음 문서를 참조하세요.

AI 액티브 스택 분석

Beta

액티브 스택 분석 버튼을 클릭하면 수집한 액티브 스택을 AI가 자동으로 분석합니다. 진행 중인 트랜잭션의 스택 덤프를 바탕으로

성능 문제를 찾고 개선 방법을 안내합니다.

분석 결과는 다음 4개 섹션으로 구성됩니다.

- **전체 요약**: 분석한 스택 수와 전체 실행 흐름을 요약합니다.
- **공통 패턴**: 반복적으로 나타나는 실행 패턴을 표시합니다. 패턴명, 발생 횟수, 설명을 포함합니다.
- **발견된 이슈**: 찾아낸 성능 문제를 심각도(**High**, **Medium**, **Low**)와 함께 표시하며, 판단의 바탕이 된 스택 정보도 제공합니다.

심각도	기준
High	데드락, 완전 블로킹, 동일 블로킹 4회 이상 반복
Medium	I/O 대기, 2~3회 락 경합, 느린 외부 호출
Low	단일 발생 대기, 가벼운 최적화 기회

- **권장 사항**: 우선순위에 따라 구체적인 개선 방법을 안내합니다.

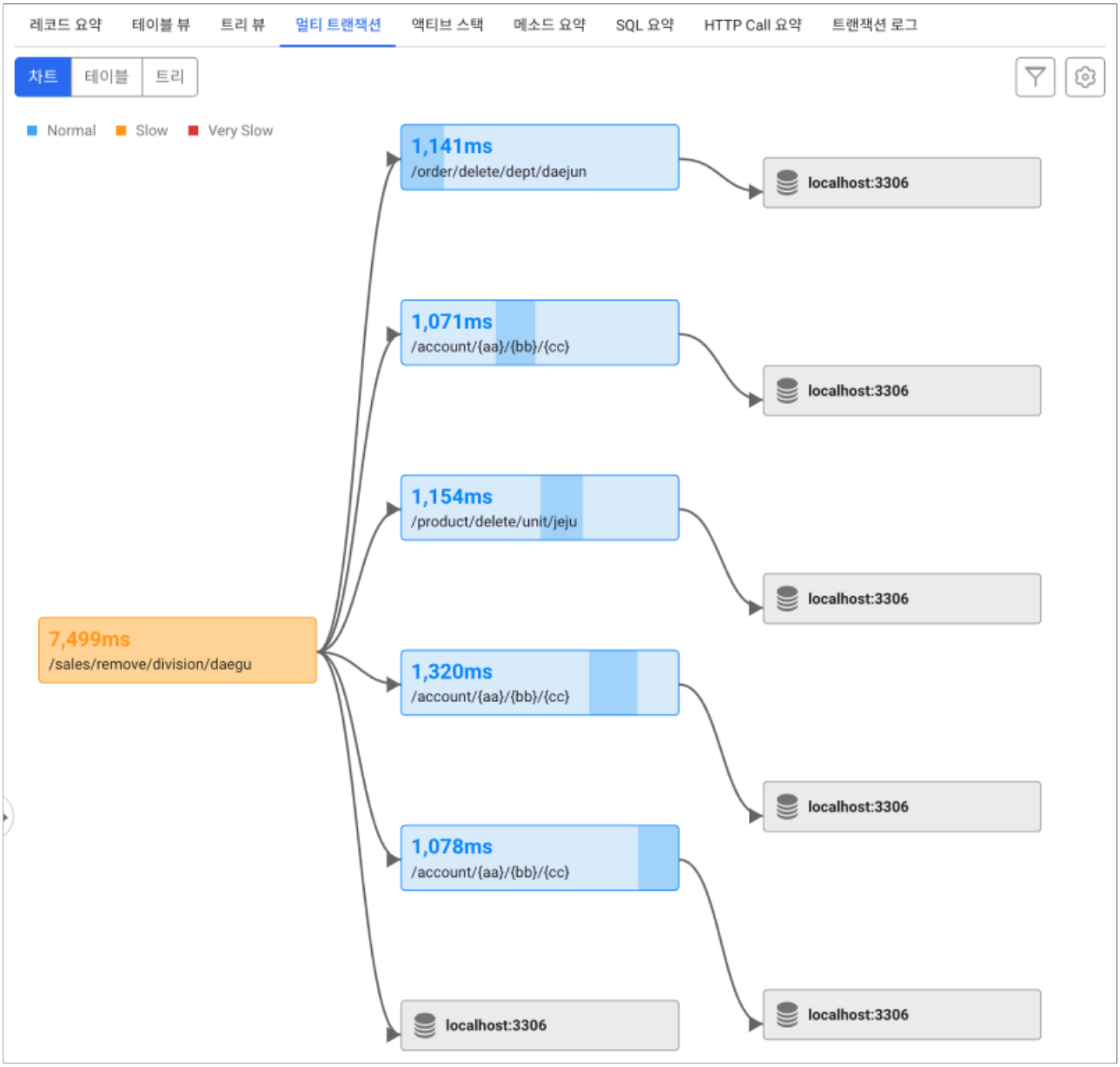
각 분석 항목에는 수집한 스택 번호(`No.N`)가 표시됩니다. 스택 번호를 클릭하면 해당 스택 원본을 확인할 수 있습니다.

> - AI가 만든 분석 결과로, 실제 상황과 다를 수 있습니다. 참고 자료로 활용하세요.
> - 프로젝트 **멤버** 권한 이상이 필요합니다.

멀티 트랜잭션

멀티 트랜잭션은 다른 에이전트나 프로젝트와의 연관된 트랜잭션을 의미합니다. **멀티 트랜잭션** 탭에서는 와탭 모니터링 서비스에 등록한 애플리케이션 간의 호출 관계를 확인할 수 있습니다.

> 멀티 트랜잭션을 추적하려면 **관리** > **에이전트 설정** 메뉴에서 `mtrace_enabled` 옵션을 `true` 로 설정하세요.
> 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.



- **차트**: 각 트랜잭션의 호출 관계를 플로우 차트 형식으로 제공합니다. 트랜잭션 노드를 선택하면 해당 트랜잭션 노드에 대한 트레이스 분석 정보를 확인할 수 있습니다. **차트** 모드에서는 마우스를 이용해 원하는 위치로 이동하거나 스크롤을 통해서 확대, 축소할 수 있습니다.
  - ⚙ **차트 뷰 설정**: 차트에 표시할 요소를 표시하거나 숨길 수 있습니다.
- **테이블**: 테이블 형식으로 멀티 트랜잭션 내에 포함된 각 트랜잭션 별 정보를 확인할 수 있습니다. ▥ 컬럼 선택 아이콘을 선택해 테이블 헤더 컬럼을 편집할 수 있습니다. 각 트랜잭션 항목을 선택하면 트레이스 분석 정보를 확인할 수 있습니다.

- **트리**: 트리 형식으로 트랜잭션 간의 호출 관계를 파악할 수 있습니다. 관련한 부가 기능은 **트리 뷰** 탭의 기능과 같습니다.
- **프로젝트 선택**: 차트에 표시될 프로젝트를 선택하거나 해제할 수 있습니다.

> 트랜잭션 목록에서 M 아이콘이 표시된 항목에서 확인할 수 있습니다.
>
> - 멀티 트랜잭션에 대한 자세한 내용은 다음 문서를 참조하세요.

메소드 요약



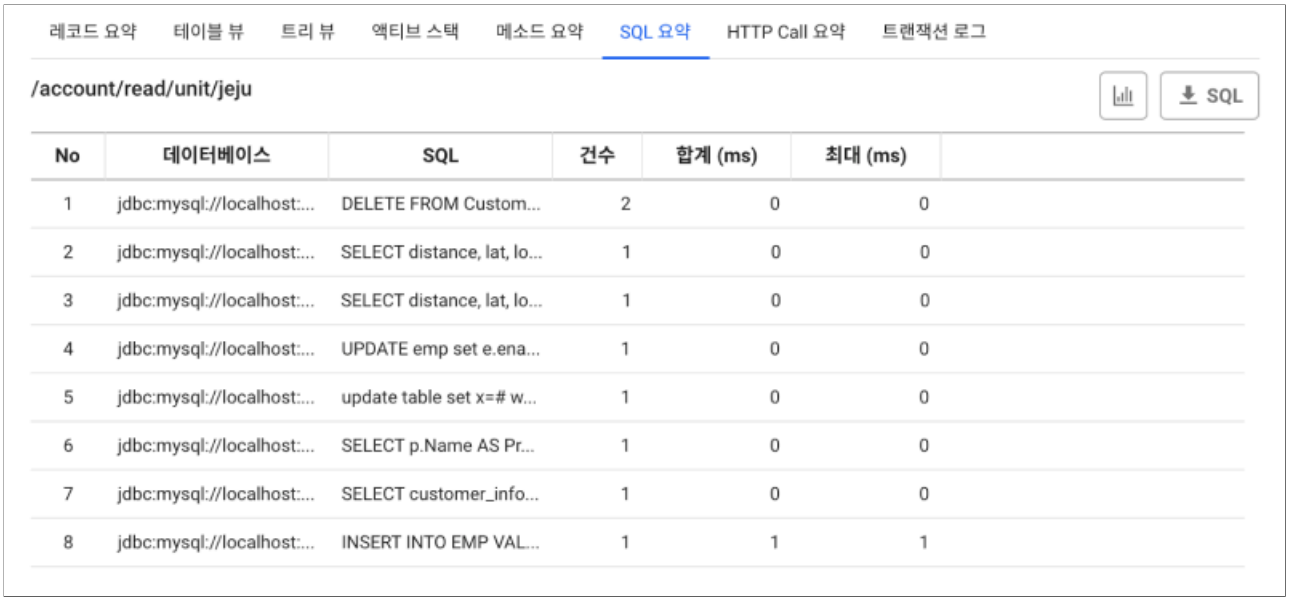
No	클래스	메소드	파라미터/리턴	건수	합계 (ms)	최대 (ms)
1	com.virtual.dao.Remot...	execute	()V	27	39,063	2,776
2	com.virtual.dao.PIITest	exec	(Ljava/sql/PreparedSta...	3	0	0
3	com.virtual.dao.InsertD...	execute2	(Ljava/lang/String;)V	2	4	3
4	com.virtual.dao.Update...	execute2	(Ljava/lang/String;)V	2	18	14
5	com.virtual.dao.Delete...	execute	()V	2	3	2
6	com.virtual.dao.Select...	execute2	(Ljava/lang/String;)V	1	5	5
7	com.virtual.dao.Update...	execute	()V	1	4	4
8	com.virtual.dao.Select...	execute2	()V	1	7	7
9	com.virtual.dao.Select...	execute	()V	1	9	9
10	com.virtual.dao.InsertD...	execute2	()V	1	3	3

메소드 정보만을 확인하려면 **메소드 요약** 탭을 선택하세요. 에이전트에 추적이 설정된 메소드 이름과 소요 시간을 표시합니다. 불필요한 로직이 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 메소드 로직 개선을 위한 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **메소드**: 메소드 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

> ⓘ 메소드(method)와 관련한 에이전트 설정 옵션에 대한 자세한 내용은 다음 문서를 참조하세요. 이용 중인 제품에 따라 메소드 추적 지원 여부는 다를 수 있습니다.

SQL 요약



No	데이터베이스	SQL	건수	합계 (ms)	최대 (ms)
1	jdbc:mysql://localhost:...	DELETE FROM Custom...	2	0	0
2	jdbc:mysql://localhost:...	SELECT distance, lat, lo...	1	0	0
3	jdbc:mysql://localhost:...	SELECT distance, lat, lo...	1	0	0
4	jdbc:mysql://localhost:...	UPDATE emp set e.ena...	1	0	0
5	jdbc:mysql://localhost:...	update table set x=# w...	1	0	0
6	jdbc:mysql://localhost:...	SELECT p.Name AS Pr...	1	0	0
7	jdbc:mysql://localhost:...	SELECT customer_info...	1	0	0
8	jdbc:mysql://localhost:...	INSERT INTO EMP VAL...	1	1	1

SQL 문에 대한 정보를 확인하려면 **SQL 요약** 탭을 선택하세요. 불필요하게 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 쿼리 성능 개선을 위한 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **SQL**: SQL 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

HTTP Call 요약



No	원격 서버	URL	건수	합계 (ms)	최대 (ms)
1	127.0.0.1:8105	/remote/account/save...	2	4,116	2,776
2	127.0.0.1:8105	/remote/order/save/di...	1	238	238
3	127.0.0.1:8105	/remote/sales/pickup/...	1	922	922
4	127.0.0.1:8105	/remote/sales/remove...	1	1,980	1,980
5	127.0.0.1:8105	/remote/order/save/de...	1	2,639	2,639
6	127.0.0.1:8105	/remote/edu/kill/unit/p...	1	1,519	1,519
7	127.0.0.1:8105	/remote/order/read/de...	1	1,467	1,467
8	127.0.0.1:8105	/remote/order/delete/...	1	485	485
9	127.0.0.1:8105	/remote/sales/kill/emp...	1	2,210	2,210

HTTP 호출의 호출 건수, 합계 시간, 평균 시간 등을 확인하려면 **HTTP Call 요약** 탭을 선택하세요. 불필요한 외부 호출이 반복 실행되는 경우, 낮은 건수에 비해 실행 시간이 오래 걸리는 경우를 파악해 트랜잭션 지연 요인이 외부인지 내부인지를 파악하는 분석 정보로 활용할 수 있습니다.

- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **HTTP**: HTTP Call 관련 상세 정보를 TXT 파일로 다운로드할 수 있습니다.

트랜잭션 로그

트랜잭션과 관련한 로그 정보를 확인하려면 **트랜잭션 로그** 탭을 선택하세요.



- : **트랜잭션 통계** 정보를 확인할 수 있습니다. 차트의 그래프를 선택하면 **분석** > **트랜잭션 검색** 메뉴로 이동합니다. 선택한 시간대의 트랜잭션 목록을 확인할 수 있습니다.
- **로그 검색기**: 로그 검색으로 이동해 원하는 로그 데이터를 쉽게 조회할 수 있습니다.
- **키워드 검색**: 키워드 검색란에 검색하려는 텍스트를 입력한 다음 엔터 키를 입력하거나 버튼을 선택하세요. 입력한 텍스트와 일치하는 키워드를 하이라이트 표시합니다.
- **키워드 하이라이트**: 키워드와 색상을 설정하면 자동으로 로그 목록에서 키워드와 일치하는 텍스트를 하이라이트 표시합니다. 자세한 내용은 다음 문서를 참조하세요.
- **컬럼 설정**: 로그 목록 테이블에 표시되는 컬럼을 추가하거나 순서를 변경할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- **로그 표시 상세 설정**: 로그 목록에 표시되는 내용을 설정할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.

> - **트랜잭션 로그** 탭을 활성화하려면 로그와 관련한 에이전트 설정 옵션을 적용해야 합니다. 로그 설정에 대한 자세한 내용은 다음 문서를 참조하세요.
> - **트랜잭션 로그** 탭은 **로그 조회** 권한을 가진 멤버만 진입할 수 있습니다. 멤버 권한에 대한 자세한 내용은 다음 문서를 참조하세요.

키워드 하이라이트 설정하기

1. 하이라이트 표시할 키워드를 추가하려면 버튼을 선택하세요.



2. 원하는 키워드를 입력하세요.
3. 색상 아이콘을 선택한 다음 원하는 색상을 선택하세요.
4. 엔터를 입력하세요. 추가한 키워드가 목록에 생성됩니다.

키워드 하이라이트 삭제하기

하이라이트 목록에서 이용하지 않는 항목을 삭제하려면 — 버튼을 선택하세요.

테이블 컬럼 표시하거나 숨기기

1. 테이블 오른쪽 위에 ▥ 버튼을 선택하세요.
2. **컬럼 설정** 창이 나타나면 왼쪽 목록에서 원하는 항목을 선택하거나 선택을 해제하세요.

   오른쪽 목록에서 × 버튼을 선택해 컬럼을 숨길 수도 있습니다.
3. 모든 설정을 완료한 다음 **확인** 버튼을 선택하세요. 설정한 내용이 테이블에 적용됩니다.

테이블 컬럼 순서 바꾸기

1. 테이블 오른쪽 위에 ▥ 버튼을 선택하세요.
2. **컬럼 설정** 창이 나타나면 오른쪽 목록에서 원하는 항목을 드래그해서 순서를 변경하세요.
3. 모든 설정을 완료한 다음 **확인** 버튼을 선택하세요. 설정한 내용이 테이블에 적용됩니다.

테이블 콘텐츠 표시 설정

1. 테이블 오른쪽 위에 ⚙ 버튼을 선택하세요.
2. **로그 표시 상세 설정** 창이 나타나면 원하는 옵션을 설정하세요.
   - **content** 또는 **태그** 옵션을 선택해 로그의 콘텐츠 표시 여부를 선택하세요.
   - **태그 관리**: 태그는 추가한 순서대로 로그에 표시됩니다. 태그의 순서는 드래그로 변경할 수 있습니다. 비활성화된 태그는 로그에서 보이지 않습니다.
3. 모든 설정을 완료한 다음 ✕ 버튼을 클릭하세요. 설정한 내용이 테이블에 적용됩니다.

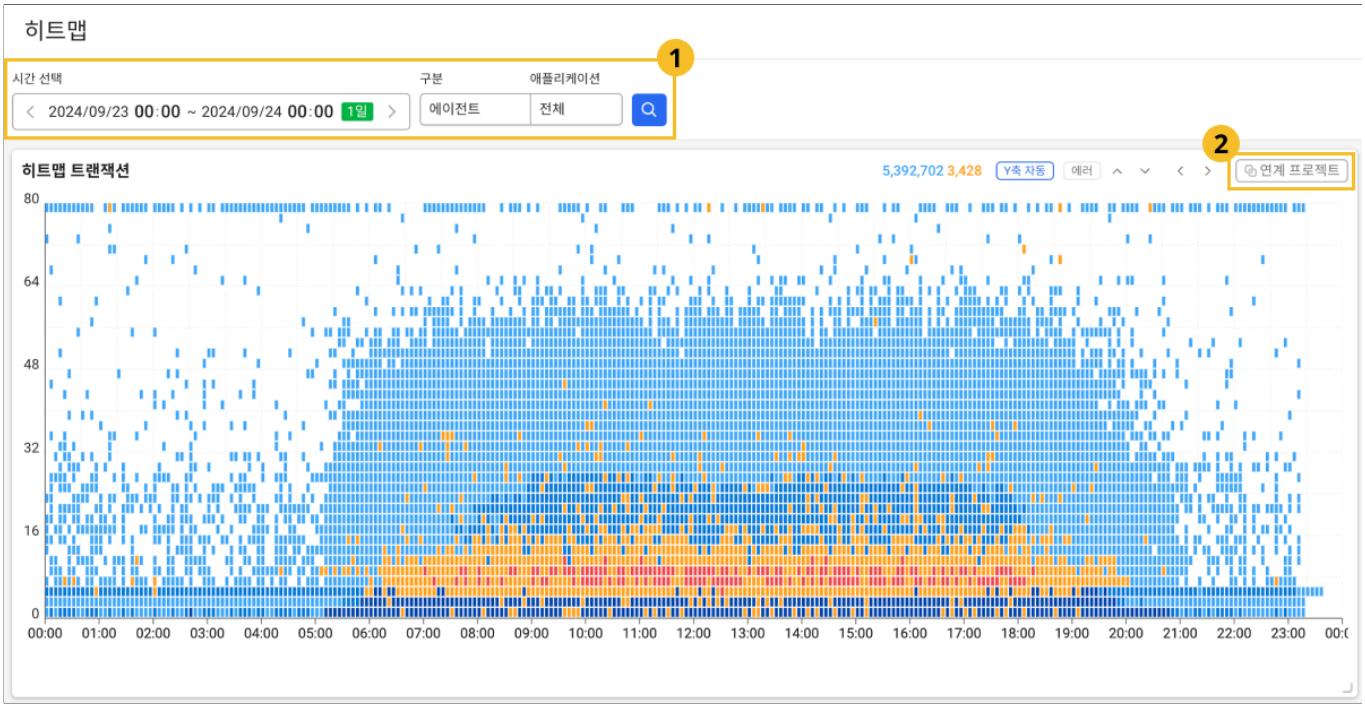
> ⓘ **로그 표시 상세 설정**에 대한 자세한 내용은 다음 문서를 참조하세요.

부가 기능

연계 프로젝트 지표 확인하기

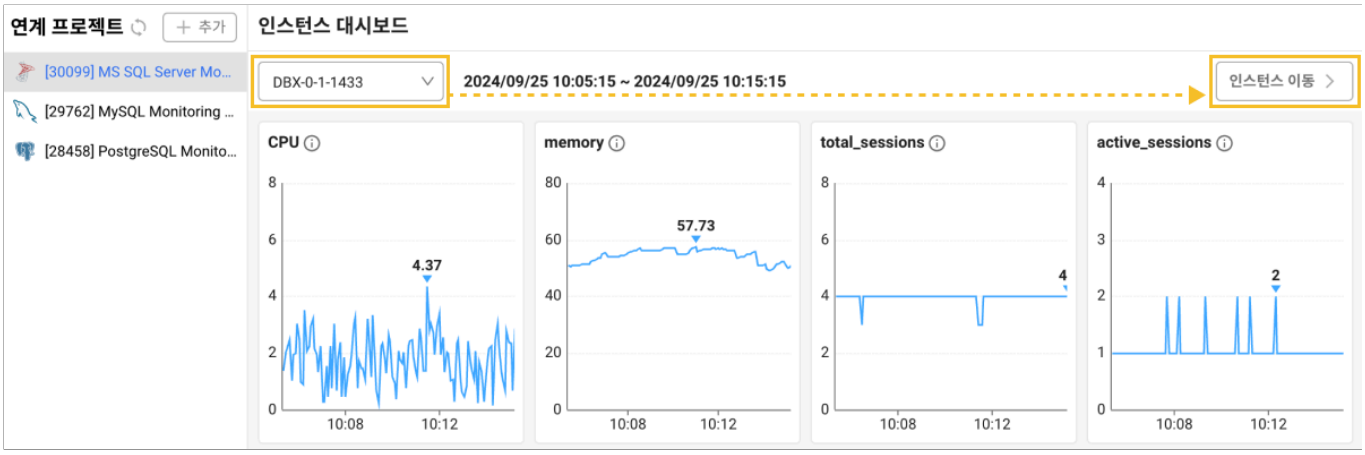
데이터베이스 모니터링(DPM) 프로젝트와 애플리케이션 모니터링(APM) 프로젝트를 생성한 경우, 두 프로젝트를 연계하여 DPM에서 수집한 모니터링 데이터를 APM 프로젝트에서 확인할 수 있습니다. **히트맵 트랜잭션** 섹션의 오른쪽 상단의 **연계 프로젝트** 기능을 통해 트랜잭션을 조회하는 동안 연계된 프로젝트의 성능 지표를 함께 확인할 수 있습니다.

연계 프로젝트 기능은 APM 프로젝트에서 트랜잭션을 분석할 때, 느린 SQL을 발견했을 경우 비슷한 여러 DB를 사용하는 환경에서 정확히 어느 DB에서 문제가 발생했는지 확인하기 어려운 상황에 유용합니다.



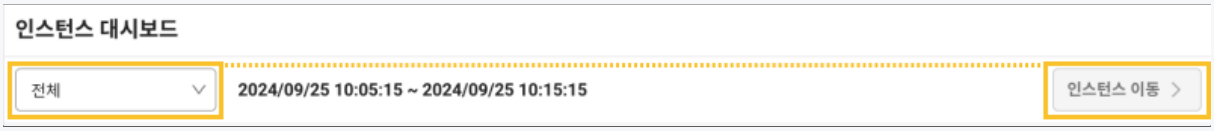
1. 트랜잭션을 조회할 시간과 **애플리케이션**을 설정한 후 버튼을 선택하세요.
2. **히트맵 트랜잭션** 섹션에서 오른쪽 상단의 **연계 프로젝트** 버튼을 선택하세요.
3. 연계 프로젝트의 지표를 확인할 수 있는 새창이 나타납니다.

왼쪽 목록에서 연계 프로젝트를 선택하고 **인스턴스 대시보드** 섹션의 지표를 확인하세요.



선택한 프로젝트의 대시보드 메뉴로 이동하려면 인스턴스를 선택한 후, 오른쪽 상단의 **인스턴스 이동** 버튼을 클릭하세요. **대시보드 > 인스턴스 모니터링** 메뉴로 이동합니다.

- **히트맵 트랜잭션** 섹션의 **연계 프로젝트** 버튼은 연계된 프로젝트가 있을 때만 표시됩니다. 연계 프로젝트 추가에 대한 자세한 내용은 다음 문서를 참조하세요.
- 왼쪽 **연계 프로젝트** 목록 상단의 **추가** 버튼을 선택해 프로젝트를 추가할 수 있습니다.
- **인스턴스 이동** 버튼은 **전체**를 선택한 상태에서는 비활성화됩니다.



- 데이터베이스 제품 종류에 따라 **분석** > **카운트 추이** 메뉴로 이동할 수 있습니다.

HTTP 파라미터 조회

테이블 뷰 탭에서 해당 트랜잭션의 HTTP 파라미터를 조회할 수 있습니다.

1. 페이지 아래로 스크롤해 트랜잭션 수행의 가장 마지막 단계로 이동하세요.
2. **HTTP 파라미터** 항목을 선택하세요.
3. **HTTP-PARAMETERS** 창이 나타나면 **비밀번호** 버튼을 선택하세요.
4. 설정한 Param Key를 입력하세요. 가려진 매개 변수를 확인할 수 있습니다.

- HTTP 파라미터와 관련한 에이전트 설정은 다음 문서를 참조하세요.
- **비밀번호**: 복호화된 파라미터 값을 확인할 수 있습니다. 비밀번호는 `WHATAP_HOME` /security.conf (paramkey.txt) 파일 내 6자리 문자열입니다. 다른 문자열로 변경 가능합니다. SQL 변수 조회, HTTP 쿼리 조회, Thread 중지에 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

SQL 파라미터 조회

테이블 뷰 탭에서 SQL 스텝을 선택하거나 **트리 뷰** 탭에서 버튼을 선택하세요. 파라미터를 조회할 수 있는 **SQL** 창이 나타납니다.



- **비밀번호**: 복호화된 파라미터 값을 확인할 수 있습니다. 비밀번호는 `WHATAP_HOME` /security.conf 파일 내 6자리 문자열(`paramkey`)입니다. 다른 문자열로 변경 가능합니다.

  > ⓘ **비밀번호**는 SQL 변수 조회, HTTP 쿼리 조회, Thread 중지에 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

- **SQL 포멧팅**: SQL 문장을 들여쓰기 및 포맷팅하여 가독성을 높일 수 있습니다.
- : SQL 문장을 클립보드에 복사할 수 있습니다.
- **통계**: **통계** > **SQL** 메뉴로 이동합니다.

> ✓ SQL 변수와 HTTP 쿼리를 조회하려면 다음 옵션을 에이전트 설정에 추가하세요.
>
> - SQL 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
> - HTTP 파라미터 정보 기록과 관련한 에이전트 설정은 다음 문서를 참조하세요.
>
> whatap.conf
>
> ```
> # SQL 파라미터 조회 옵션: 옵션이 적용되면 SQL 파라미터를 암호화하여 수집합니다.
> profile_sql_param_enabled=true
>
> # HTTP 파라미터 조회 옵션: 옵션이 적용되면 HTTP 쿼리 파라미터를 암호화하여 수집합니다.
> profile_http_parameter_enabled=true
> ```

트레이스 목록 컬럼 설정하기

TX 트레이스 섹션의 컬럼 설정을 통해 조회된 결과에 대한 세부 항목을 추가 확인할 수 있습니다.

1. **TX 트레이스** 영역의 **컬럼 선택** 버튼을 클릭하세요.
2. 원하는 컬럼 항목을 선택하거나 해제하세요. 설정한 상태가 테이블에 바로 반영됩니다.
   - **전체 선택**: 모든 컬럼 항목을 선택
   - **전체 해제**: 모든 컬럼 항목을 해제
   - **초기화(기본값)**: 컬럼 항목을 기본값으로 설정
3. 설정을 모두 완료했다면 오른쪽 상단의 × 아이콘을 클릭하세요.

트레이스 목록 필터링하기

TX 트레이스 섹션에서 조회된 결과를 다음 옵션을 통해 필터링할 수 있습니다.



- **모두 보기**: 조회된 결과에서 모든 트랜잭션을 확인할 수 있습니다.
- **액티브 스택**: 조회된 결과에서 액티브 스택을 포함한 트랜잭션만 확인할 수 있습니다.
- **멀티 트랜잭션**: 조회된 결과에서 멀티 트랜잭션을 포함한 트랜잭션만 확인할 수 있습니다.
- **Search**: 입력한 문자열과 일치하는 항목을 포함한 트랜잭션만 표시합니다. 원하는 문자열을 입력한 후 키보드의 `Enter` 키를 누르세요.
  - **트랜잭션**
  - **클라이언트 IP**
  - **도메인**
  - **에러 클래스**
  - **에러 메시지**

조회 목록 다운로드하기

TX 트레이스 섹션에서 조회된 결과를 CSV 형식의 파일로 다운로드할 수 있습니다. 다운로드한 CSV 파일 이름은 profile_

`{project_code}_YYYYMMDD.csv` 형식입니다.

1. 트랜잭션을 조회한 후에 CSV 버튼을 선택하세요.
2. **최대 CSV 라인 수** 옵션에 원하는 값을 입력하세요.
3. **다운로드** 버튼을 선택하세요.

액티브 트랜잭션

진행 중인 트랜잭션을 액티브 트랜잭션이라고 합니다. **액티브 트랜잭션에서 정기적으로 덤프한 스택을 액티브 스택**이라 합니다.

> 와탭 에이전트는 매 10초(옵션 가능)마다 액티브 트랜잭션에 대해서 액티브 스택을 덤프하고 이것을 서버에 전송합니다. `active_stack_second=10`

컴팩트한 액티브 스택 수집

액티브 스택은 스레드 덤프를 정기적으로 수행하기 때문에 잘못 구현되면 에이전트에 오버헤드가 커질 수 있습니다. 와탭은 에이전트 부하를 최소화하면서 액티브 스택을 수집하기 위해 다양한 옵션들을 가지고 있습니다.

> **인스턴스 성능 관리** > **스레드 목록/덤프** 메뉴에서 스레드 목록 중 **WhaTap-ActiveStackDump** 스레드의 **CPU 시간**을 확인하면 오버헤드를 판단할 수 있습니다.

액티브 스택 예시

JAVA

```
java.lang.StringBuffer.append(StringBuffer.java:309)
java.util.regex.Matcher.appendReplacement(Matcher.java:839)
java.util.regex.Matcher.replaceAll(Matcher.java:906)
java.lang.String.replaceAll(String.java:2162)
core.log.triggers.TriggerRegister.changeNotify(TriggerRegister.java:114)
core.log.aop.handler.DaoInfo.log(DaoInfo.java:141)
core.log.aop.handler.DaoInfo.doAround(DaoInfo.java:102)
core.log.aop.reflection.profiler.AroundProfiler.invoke(AroundProfiler.java:19)
com.sun.proxy.$Proxy39.getUpdateCount(Unknown Source)
org.apache.ibatis.executor.resultset.DefaultResultSetHandler.getNextResultSet(DefaultResultSetHandler.java:256)
org.apache.ibatis.executor.resultset.DefaultResultSetHandler.handleResultSets(DefaultResultSetHandler.java:193)
org.apache.ibatis.executor.statement.PreparedStatementHandler.query(PreparedStatementHandler.java:64)

* * *
```

```
sun.reflect.GeneratedMethodAccessor140.invoke(Unknown Source)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:606)
org.springframework.web.method.support.InvocableHandlerMethod.doInvoke(InvocableHandlerMethod.java:221)
org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerMethod.java:136)
org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHandle(ServletInvocableHandlerMe
org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerMethod(RequestMappingHand
```

최적화된 데이터 수집

- 트랜잭션을 수행 중인 스레드에 대해서만 스택을 덤프합니다.
- 액티브 스택 덤프 시간 간격을 조정할 수 있습니다.

  whatap.conf

  ```
  active_stack_second=10
  ```

- 액티브 스택의 최대 라인에 제한되어 있습니다. Top 라인에서부터 기본 50라인을 수집합니다.

  whatap.conf

  ```
  trace_active_callstack_depth=50
  ```

- 액티브 스택의 각 라인은 해시 처리되어 수집됩니다. text는 한 번만 수집됩니다.
- 한 타임에 수집되는 최대 액티브 스택 개수도 제한되어 있습니다.

  whatap.conf

  ```
  active_stack_count=100
  ```

Background Thread에 대한 액티브 스택

기본적으로 **액티브 스택**은 트랜잭션이 수행되고 있는 스레드의 스택을 의미합니다. 하지만 일부 백그라운드 스레드에 대해서도 스택을 분석할 필요가 있을 수 있습니다. 이때 옵션을 통해서 백그라운드 스레드에 대한 **액티브 스택**을 확보할 수 있습니다. 이 기능은 Java 에이전트 1.6.2 버전 이후부터 지원합니다.

- `async_stack_enabled`의 값을 `true`로 설정하면 활성화됩니다.

  whatap.conf

  ```
  async_stack_enabled=false
  ```

- 스택 덤프 간격은 포그라운드 액티브 스택 설정에 따라갑니다.

  whatap.conf

  ```
  active_stack_second=10
  ```

- 대상 스레드 이름을 지정할 때는 `*`를 사용하여 문자열 패턴을 지정합니다.

  whatap.conf

  ```
  async_thread_match=http*,abc*
  ```

  스레드 이름으로 스택 덤프 대상을 식별합니다. 쉼표(,)를 구분자로 사용하여 match를 여러 개 지정할 수 있습니다.

- 스택의 Top 메소드가 `async_thread_parking`에 등록된 클래스 또는 메소드일 때 스레드가 파킹 상태에 있다고 판단하고 덤프를 생성하지 않습니다.

  whatap.conf

  ```
  async_thread_parking_class=sun.misc.Unsafe
  async_thread_parking_method=park
  ```

참고 자료

- 액티브 트랜잭션
- 장애를 가장 빠르게 알아내는 액티브 트랜잭션

멀티 트레이스

멀티 트랜잭션은 다른 에이전트나 프로젝트와 연관된 트랜잭션을 의미합니다. 와탭 프로젝트에 등록된 애플리케이션 서비스 간의 호출을 추적하는 것이 **멀티 트랜잭션 추적**입니다.

> Java 에이전트는 세 개의 HTTP 헤더 키값(`x-wtap-po` , `x-wtap-mst` , `x-wtap-sp1`)으로 멀티 트랜잭션을 추적합니다. 게이트웨이를 통과하는 HTTP 트랜잭션이 연계 추적이 안 된다면 HTTP 헤더 조건을 확인하세요.

> **멀티 트랜잭션 활성화**
>
> 멀티 트랜잭션을 추적하려면 **관리** > **에이전트 설정** 메뉴에서 `mtrace_enabled` 옵션을 `true` 로 설정하세요.
> 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

멀티 트랜잭션 추적 기능 이용하기

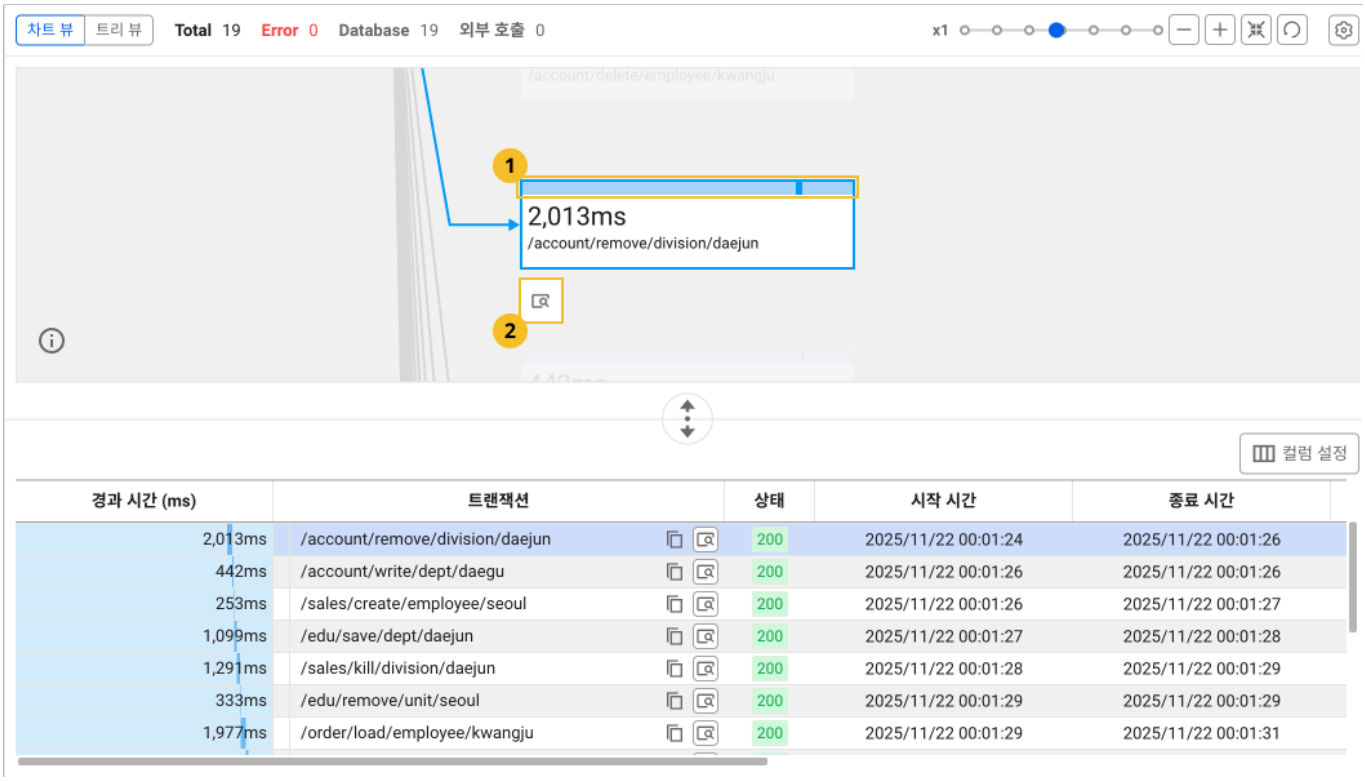
1. **분석** > **멀티 트랜잭션 추적**로 이동하세요.
2. **트랜잭션 정보** 창에서 확인한 **MTID** 값을 **MTID / CUSTID 조회** 항목에 입력하세요.
3. 조회할 날짜와 프로젝트를 선택하세요.
4. 화면 아래에 **적용** 버튼을 클릭하세요.
    - 다음 조건에서 화면 하단의 **적용** 버튼이 비활성화됩니다.
        - 이전 검색과 동일한 필터값(MTID/CUSTID, 날짜, 선택 프로젝트)을 설정한 경우
        - 필터값(MTID/CUSTID, 날짜, 선택 프로젝트)을 하나도 입력하지 않은 경우
5. 오른쪽 **차트** 탭에 각 트랜잭션의 호출 관계를 파악할 수 있는 다이어그램과 테이블이 표시됩니다.

    > 다이어그램과 테이블 영역은 상하로 구분되어 그 비율을 직접 조절할 수 있습니다.
    >
    > - 위쪽 화살표를 클릭하면 테이블이 전체 영역을 차지합니다. 단, 다이어그램 영역이 전체 영역을 차지하고 있었다면 다이어그램과 테이블이 각각 절반의 영역으로 나뉩니다.

- 아래쪽 화살표를 클릭하면 차트 영역이 전체 영역을 차지합니다. 단, 테이블 영역이 전체 영역을 차지하고 있었다면 다이어그램과 테이블이 각각 절반의 영역으로 나뉩니다.

차트

차트는 각 트랜잭션의 호출 관계를 빠르고 명확하게 사용자에게 제공합니다. 동일한 **멀티 트랜잭션 ID**를 갖는 트랜잭션 서비스들의 개별 수행 시간을 확인할 수 있습니다. 트랜잭션 노드의 상단에 표현되어 있는 소요시간(1 **타임바**)를 통해 트랜잭션 간 호출 관계를 확인할 수 있습니다.



차트는 마우스로 자유롭게 이동하거나 확대, 축소할 수 있습니다.

- 차트 영역을 드래그하면 원하는 위치로 이동할 수 있습니다.
- 차트 영역을 스크롤하면 원하는 위치로 이동할 수 있습니다.

- 상하 스크롤: 상하 이동
- 좌우 스크롤: 좌우 이동

- Ctrl + 상하 스크롤로 차트를 확대하거나 축소할 수 있습니다.
  - 터치패드의 확대, 축소 동작으로도 가능합니다.

트랜잭션을 선택하면 연결된 노드만 활성화되고, 연결 관계없는 모든 노드는 흐리게 표시됩니다. 선택된 트랜잭션의 하단의 2 **상세보기** 버튼이 활성화되며, 클릭 시 **트랜잭션 정보** 창이 열립니다. 해당 트랜잭션의 상세 내역을 확인할 수 있습니다. **트랜잭션 정보** 창을 활용한 트랜잭션 트레이스 상세 분석에 관한 자세한 내용은 다음 문서를 참고하세요.

- **확대**, **축소**: 현재 확대 배율과 확대 축소 동작이 가능함
- **화면에 맞춤**: 모든 노드를 화면에 맞게 표시
- **확대/축소 초기화**: 차트의 표시 위치와 확대 정도를 최초 상태로 초기화
- **차트 뷰 설정**: 차트에 표시할 요소를 표시하거나 숨김
  - **상세 정보**: 애플리케이션 명, 프로젝트 정보, IP 정보, 상태 코드를 표시하거나 숨김
  - **데이터베이스 / 외부 호출**: 해당 트랜잭션에서 발생한 다른 데이터베이스 커넥션 요청이나 HTTP Call의 정보 또한 차트의 노드로 확인 가능함

테이블

테이블 탭에서 멀티 트랜잭션 내에 포함된 각 트랜잭션 별 정보를 테이블 형식으로 확인할 수 있습니다.

각 트랜잭션 항목을 선택하면, 해당 트랜잭션 **차트**의 **노드**가 활성화되고, 연결된 노드가 하이라이트 됩니다. 트랜잭션 컬럼의 상세보기 버튼을 클릭하면, **트랜잭션 정보** 창이 나타납니다. 트랜잭션 트레이스에서 트랜잭션의 상세 내역을 확인할 수 있습니다. **트랜잭션 정보** 창을 활용한 트랜잭션 트레이스 상세 분석에 관한 자세한 내용은 다음 문서를 참조하세요.

- **컬럼 설정**: 테이블 내 컬럼 편집

트리

트리 버튼을 선택하면 각 트랜잭션과 그에 속해 있는 트레이스의 세부 정보를 확인할 수 있습니다. 전체 트랜잭션 소요 시간 내의 각 하위 트랜잭션이나 트레이스의 시작 및 소요 시간을 시각화해 트랜잭션 호출 관계를 트리 형식으로 제공합니다.

- **프로젝트 선택**: 차트에 표시될 프로젝트를 선택하거나 해제할 수 있습니다.

- **한 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 한 줄로 표시해 트리 형식을 간격하게 정리할 수 있습니다.
- **여러 줄 보기**: 각 구간 별 수행 정보에 표시된 텍스트를 줄바꿈해 모두 표시합니다.
- **최장 경로**: 가장 긴 경로로 이동할 수 있습니다.
- **시간바 표시**: 경과 시간을 막대 형식의 차트로 표시합니다.
- **시간 표시**: 각 구간별 타임 스탬프, 갭, 경과 시간을 텍스트 형식으로 표시합니다.
  - 8초 이상: 초과 지연 상태로 빨간색으로 표현합니다.
  - 3초 이상 8초 미만: 지연 상태로 주황색으로 표현합니다.
  - 3초 미만: 정상 상태로 파란색으로 표현합니다.

> 시작 및 소요 시간의 경우 트랜잭션 호출 환경에 따라 발생하는 시차를 상위 트랜잭션 내 트레이스와 매핑을 통해 보정하여 표현하기 때문에 실제 수집된 시간 데이터와 차이가 발생할 수 있습니다.

- **시간 숨기기**: 시간 정보를 숨깁니다.

> - 해당 트레이스의 버튼 또는 버튼을 선택하면 **HTTP 호출 통계** 및 **액티브 스택**, **SQL** 등에 대한 상세 정보를 확인할 수 있습니다.
> - SQL 스텝의 버튼을 선택하면 해당 스텝의 SQL 문을 복사할 수 있습니다.

엔드포인트 설정

트랜잭션 엔드포인트는 트랜잭션의 시작 메소드입니다. HTTP 트랜잭션의 경우에는 `HttpServlet.service()` 혹은 `Filter.doFilter()`가 트랜잭션의 시작점이고 이곳을 **트랜잭션 엔드 포인트**라고 부릅니다.

NON HTTP 추적

트랜잭션 엔드포인트로 지정된 메소드가 시작해서 종료될 때까지의 성능을 트랜잭션 성능이라고 합니다. Non http 트랜잭션을 추적하기 위해서는 **엔드포인트**를 지정해야 합니다.

> **엔드포인트** 찾기
>
> 1. 트랜잭션이 호출될 것으로 추정되는 모든 메소드에 대해 트레이스 추적을 설정하세요.
> 2. 다시 시작 후 트랜잭션을 발생시켜 모니터링하세요.
> 3. `back stack` 옵션을 켜서 진입 메소드를 정확히 파악하세요.

대상 선정

1. **메소드 트레이스 설정**

   메소드 트레이스를 설정하세요. 확실하게 트랜잭션에서 호출되는 클래스를 지정하세요. DB를 사용하는 프로그램에서는 **JDBC 드라이버**가 유용합니다.

   whatap.conf

   ```
   hook_method_patterns=jdbc.*.*
   hook_method_access_public_enabled=true
   hook_method_access_protected_enabled=true
   hook_method_access_none_enabled=true
   ```

2. **트랜잭션 시작 옵션 설정**

메소드가 호출되면 트랜잭션을 시작하세요. 트랜잭션 시작 옵션과 트랜잭션 시작 시 스택을 덤프하는 옵션을 켜세요.

whatap.conf

```
trace_auto_transaction_enabled=true
trace_auto_transaction_backstack_enabled=true
```

3. **재시작 후 트레이스 분석**

다시 시작하세요. 이후 서비스를 호출하면 트랜잭션이 추적되는 것을 볼 수 있습니다.

트랜잭션을 조회하면 `jdbc.*` 로 시작하는 모든 클래스의 메소드가 이 트랜잭션으로 나타나는 것을 알 수 있습니다. 트랜잭션 트레이스를 조회하면 **TRANSACTION BACKSTACK**라는 메시지 스텝을 확인할 수 있습니다.

TRANSACTION BACKSTACK

```
jdbc.FakePreparedStatement.executeQuery(FakePreparedStatement.java),
com.virtual.dao.SelectDAO.execute2(SelectDAO.java:29),
com.virtual.web.SimulaNonHttp.execute(SimulaNonHttp.java:147),
com.virtual.web.SimulaNonHttp.process(SimulaNonHttp.java:76),
com.virtual.web.SimulaNonHttp.run(SimulaNonHttp.java:100)
```

4. **스택 내용 확인**

스택 내용을 확인하면 어떤 메소드로부터 출발하고 있는지 추정할 수 있습니다.

Example

```
com.virtual.web.SimulaNonHttp.execute(SimulaNonHttp.java:147),
com.virtual.web.SimulaNonHttp.process(SimulaNonHttp.java:76),
com.virtual.web.SimulaNonHttp.run(SimulaNonHttp.java:100)
```

위 3개의 메소드 중에 하나를 트랜잭션 시작점으로 판단할 수 있습니다. 이 상황에서는 역 컴파일을 수행해서 적절한 **트랜잭션 엔드포인트**를 결정해야 합니다.

로직을 보면 `SimulaNonHttp.run` 내에서 `while()` 가 돌면서 `SimulaNonHttp.process()` 을 호출하고 `SimulaNonHttp.execute()` 가 수행됩니다. `process()` 가 적당하다는 것을 알 수 있습니다. 이 부분은 소스를 보고 판단해야 합니다.

> **엔드포인트**의 가장 **중요한 기준은 종료되어야 한다는 것**입니다. 정상적인 상황에서 지연되지 않고 곧바로 종료되어야 성능적인 판단을 할 수 있습니다.

트랜잭션 엔드포인트 지정

1. 트랜잭션 시작 지점을 다음과 같이 설정하세요.

   whatap.conf

   ```
   hook_service_patterns=com.virtual.web.SimulaNonHttp.process
   ```

2. 애플리케이션을 다시 시작하세요.

`process()` 메소드가 새로운 트랜잭션의 **엔드포인트**가 됩니다.

> `hook_service_patterns` 옵션은 와일드카드 문자를 사용할 수 없으며, 2개 이상의 값을 설정하려면 쉼표(,)를 구분자로 이용하세요.

트랜잭션 이름 정의

일반적으로 메소드 명칭으로 트랜잭션을 구분할 수 있습니다.

whatap.conf

```
service_name_mode=[full,class,method,string,arg]
service_name_index=0
```

- **service_name_mode**

  `full` , `class` , `method` , `string` , `arg` 5가지 옵션을 지정할 수 있습니다.

  - `full` : Full Class 이름을 서비스 명으로 사용합니다.

- `class` : Class 이름을 서비스 명으로 사용합니다.
- `method` : Method 이름을 서비스명으로 사용합니다.
- `string` : 문자열 중에서 첫 번째 파라미터를 서비스 명으로 사용합니다.
- `arg` : 파라미터 중에서 `service_name_index` 옵션에 지정한 인덱스에 파라미터를 서비스 명으로 사용합니다.

플러그인 사용

> 플러그인은 충분히 이해한 경우에만 사용하길 권장합니다.

WHATAP_HOME 경로 하위에 plugin 폴더를 생성하고, plugin로 이동하세요. `vi` 명령어로 AppServiceStart.x 파일을 생성하세요.

플러그인 사용 예시

`println("test");` 라고 타이핑하고 저장하면 화면에 "test"라는 문자열이 출력되는 것을 확인할 수 있습니다. 확인 후 파라미터에서 정보를 추출하세요.

다음 예시에서는 `HashMap` 이 인수로 전달되고, 이 `HashMap` 에는 'url'이라는 키가 포함되어 있습니다.

```
Object url =((java.util.HashMap)$point.getArgs()[0]).get("url");
$ctx.service((String)url);
//println("url="+url);
```

이렇게 플러그인을 만들면 트랜잭션 이름을 변경할 수 있습니다.

스택 수집

애플리케이션 모니터링 에이전트는 선형 수집 방식과 환형 수집 방식을 제공합니다.

선형 수집

개별 트랜잭션 트레이스는 무한정 스텝을 수집할 수 없습니다. 트랜잭션별로 길이가 제한된 버퍼에 트레이스 스텝을 저장합니다. 각 트랜잭션은 최대로 수집할 수 있는 스텝의 수가 옵션으로 지정되어 있습니다.

- **profile_step_max_count**

  기본값 `1024`

  최대로 수집 가능한 스텝 수입니다.

- **profile_step_normal_count**

  기본값 `800`

  일반적으로 아무런 제약없이 수집되는 스텝 수입니다.

- **profile_step_heavy_count**

  기본값 `1000`

  normal count를 초과한 경우 스텝은 응답시간이 느린 스텝과 액티브 스택 스텝만 수집합니다.

- **profile_step_heavy_time**

  기본값 `100`

  heavy count 이내에서의 수집되는 스텝의 기준 시간은 `profile_step_heavy_time` 입니다.

  step_buffer

  수집되는 트레이스의 스텝 수가 heavy count를 초과하는 경우에는 액티브 스택만이 수집됩니다. 이 경우에도 최대 스텝 수는 `profile_step_max_count` 를 넘지 않습니다.

환형 수집

선형 수집은 트레이스에서 앞부분을 수집하고 버퍼 사이즈를 넘으면 나중 내용을 버립니다. 반면 환형 수집은 앞부분의 스텝을

버리는 방식입니다.

```
circular_profile_enabled=true
```

circular_profile

버퍼 사이즈는 `profile_step_max_count` 에 설정합니다.

```
profile_step_max_count=1024
```

애플리케이션 토폴로지

애플리케이션 토폴로지는 트랜잭션의 흐름과 관련된 데이터베이스 및 외부 서비스 호출을 시각적으로 보여줍니다. 사용자 요청부터 데이터베이스 호출, 외부 서비스 연동까지 전체 경로를 한눈에 파악할 수 있어, 성능 병목 지점을 쉽게 확인할 수 있습니다.

- 사용자 요청 → DB/외부 서비스까지의 경로 시각화
- URL 패턴을 기준으로 한 필터링 조회
- 트랜잭션 성능 상태(느림/에러 등) 확인
- 병목 지점 식별 및 상세 정보 확인

토폴로지 조회

시간 범위와 애플리케이션을 선택한 후, 아래 옵션으로 조회 범위를 설정합니다. 조회할 때 마다 새로운 탭에 표시됩니다. 탭에 마우스를 올리면 조회 조건이 툴립으로 나타나며, 툴팁 내 새 창 버튼을 클릭하면 해당 조건으로 조회된 토폴로지가 팝아웃으로 열립니다.

트랜잭션 URL 검색

특정 URL 패턴의 트랜잭션을 호출한 도메인만 표시합니다. 와일드카드(`*`)를 사용하여 패턴을 설정할 수 있습니다.

실제 사용 예시

- `/api/*` : /api/로 시작하는 모든 URL
- `*/users/*` : users가 포함된 모든 URL
- `/api/v1/users` : 정확히 일치하는 URL

> ⚠ **슬래시(`/`)와 와일드카드(`*`) 조합에 주의하세요.**
>
> - `/api/*` : `/api` 는 매칭되지 않음, `/api/` 뒤에 경로가 있어야 매칭(예: 🔵 `/api/users` , ❌ `/api`)
> - `/api*` : `/api` 자체도 매칭됨(예: 🔵 `/api` , `/api/users` , `/apitest`)

패턴 제한사항

- **필수 문자열**: 패턴에 와일드카드(`*`) 외에 최소 1개 이상의 문자가 포함되어야 합니다. 와일드카드만으로 패턴을 설정할 수 없습니다.
  - ❌ 불가: `***` , `**`
  - 🔵 가능: `a*` , `*a` , `*a*` , `/api*`
- **빈 문자열**: 빈 문자열이나 null은 매칭되지 않습니다.
- **연속된 와일드카드**: `**` 는 `*` 와 동일하게 처리됩니다.
- **복잡한 패턴**: 3개 이상의 `*` 가 포함된 패턴은 완전히 지원되지 않습니다.
- **정규식 미지원**: 문자 클래스(`[a-z]`), 수량자(`+` , `?`), 그룹화(`()`) 등은 지원하지 않습니다.
- **대소문자 구분**: 모든 패턴 매칭은 대소문자를 구분합니다.
- **이스케이프 불가**: `*` 문자 자체를 매칭할 방법은 없습니다.

조회 제한 건수

조회할 최대 트랜잭션 트레이스(프로파일) 건수입니다. 건수가 늘어날수록 더 많은 데이터를 볼 수 있지만 조회 시간이 길어집니다.

- 기본값: 10,000건
- 최대값: 1,000,000건

> Limit을 늘려도 필터 조건에 맞는 트랜잭션 전체를 조회하지 못할 수 있습니다. 설정된 건수에 도달하면 화면에 경고 메시지가 표시됩니다.

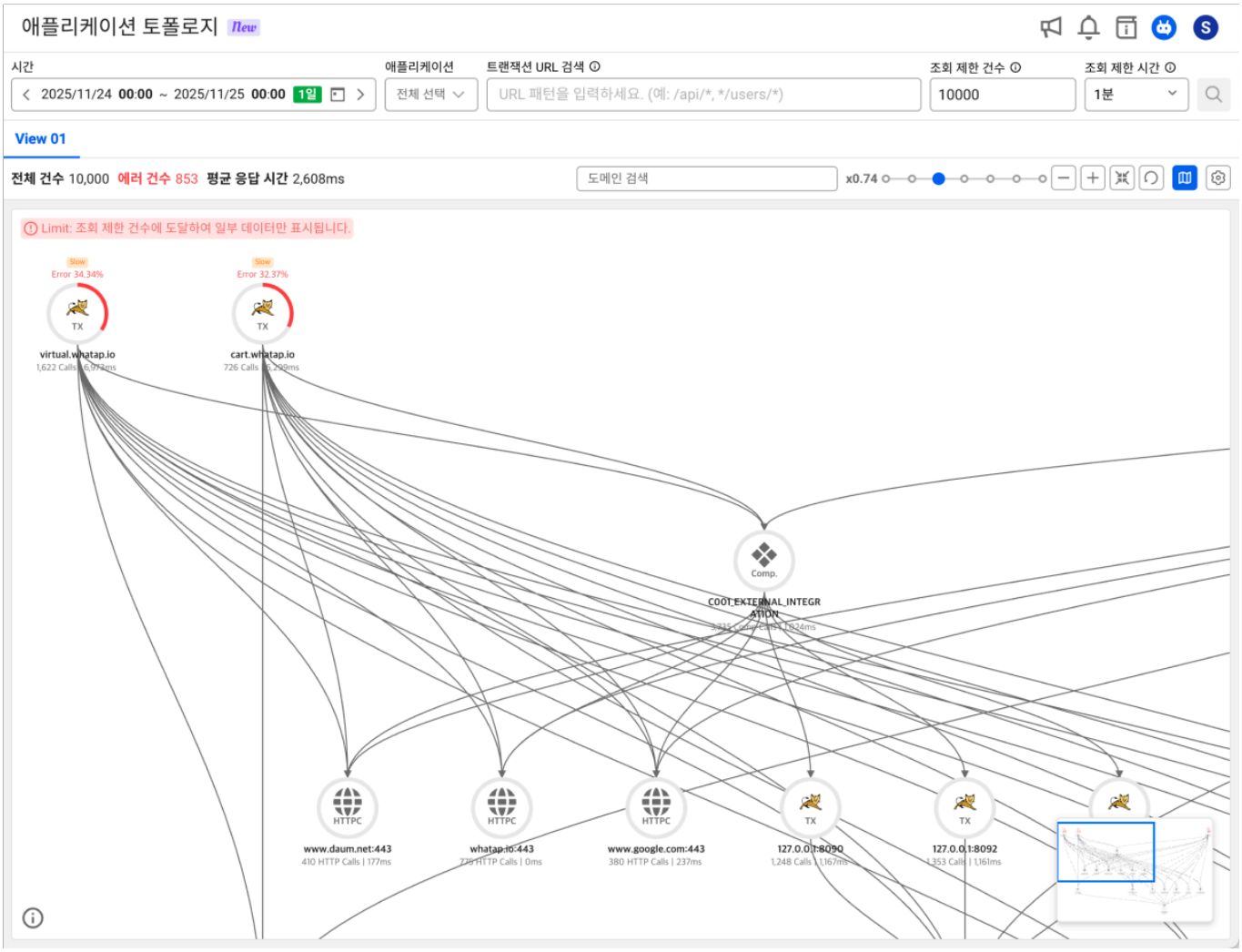
조회 제한 시간

조회 제한 시간 도달 시 해당 시점까지의 데이터만 표시되며 화면에 경고 메시지가 표시됩니다.

- 범위: 30초 ~ 5분
- 기본값: 1분

화면 구성 및 시각화

화면에는 원형의 **노드**와 노드를 연결하는 **연결선**으로 트랜잭션 흐름을 시각화합니다.



노드 구조

노드는 트랜잭션 실행 흐름에 따라 위에서 아래로 배치되며, 상위 노드가 하위 노드를 호출합니다.

노드 타입

노드는 위에서 아래로 User → TX → DB/HTTPC 순서로 배치됩니다.

노드 타입	설명
User	트랜잭션을 호출한 사용자 수
TX (Transaction)	애플리케이션 트랜잭션(HTTP 도메인/URL)
DB (Database)	데이터베이스 쿼리 및 연결
HTTPC (HTTP Call)	외부 HTTP 호출

노드 시각 요소

- **에러 게이지**: 노드 외곽선에 원형 게이지로 에러율 표시 (에러 발생 시)
- **성능 태그**: 평균 경과 시간에 따라 노드 상단에 "Slow"(주황색) 또는 "Very Slow"(빨간색) 배지 표시
- **호버 효과**: 마우스 올리면 회색 테두리 + 연결 경로 강조
- **선택 효과**: 클릭 시 파란색 테두리 + 연결 경로 강조

연결선

연결선 평균 경과 시간에 따라 색상이 변경됩니다.

- **회색**: 일반 상태
- **주황색**: 느림(Slow 임계값 이상, 기본 3초)
- **빨간색**: 매우 느림(Very Slow 임계값 이상, 기본 8초)

결과 요약

화면 상단에 조회 결과의 핵심 지표가 표시됩니다.

- **총 건수**: 조회된 전체 트랜잭션 수

- **에러 수**: 에러가 발생한 트랜잭션 수
- **평균 응답시간**: 전체 트랜잭션의 평균 응답시간

화면 조작 방법

구분	동작	설명
스크롤 및 줌	**마우스 휠**	세로 스크롤(위/아래)
	Shift + **마우스 휠**	가로 스크롤(좌/우)
	Ctrl + **마우스 휠**	줌 확대/축소
드래그	**캔버스 드래그**	빈 공간 드래그로 전체 토폴로지 이동
	노드 드래그	개별 노드 드래그로 위치 조정
노드 인터랙션	**노드 호버**	마우스를 올리면 해당 노드와 연결된 모든 경로 강조
	노드 클릭	선택한 노드 파란색 테두리로 강조, 연결 경로 표시 및 상세 정보창 열림
	재클릭	선택 해제
줌 컨트롤	**확대(+) / 축소(-)**	줌 확대 및 축소(범위: 0.1x ~ 2.0x)
	화면 맞춤	전체 토폴로지를 화면에 자동 맞춤
	리셋	줌 초기화
	줌 슬라이더	드래그로 세밀한 줌 조정
이미지 다운로드	**PNG 다운로드**	현재 차트 화면을 PNG로 저장
미니맵	**위치 확인**	우측 하단에 현재 보기 영역을 파란색 사각형으로 표시
	지도 아이콘	표시/숨김 전환

구분	동작	설명
도메인 검색	**검색창 입력**	키워드 입력 시 일치 노드 강조, 일치하지 않는 노드는 투명 처리

화면 표시 설정 옵션

토폴로지 표시 방식을 세밀하게 조정할 수 있습니다.

노드 라벨

각 노드에 표시할 정보를 선택합니다.

- **호출 건수**: 노드 타입별로 형식화 (예: “1,234 Calls”, “567 SQL”, “89 HTTP Calls”)
- **에러 건수**:
  - **표시 모드**: 백분율(%) 또는 건수
  - **임계값**: 설정값 초과 시만 에러 색상 표시 (빨간색 텍스트)
  - 예: 백분율 5% 설정 시, 에러율 5% 초과 노드만 에러 색상 표시
- **평균 경과 시간**: 밀리초 단위
- **이름**: 도메인명, DB URL 등
- **타입**: TX, DB, HTTPC, User

연결선 라벨

연결선에 배지 형태로 정보를 표시합니다.

- **표시 모드**
  - **항상 표시**: 모든 연결선 라벨 항상 표시
  - **선택 시 표시**: 노드 호버/클릭 시 관련 경로의 연결선 라벨만 표시 (기본값)
- **표시 정보**:
  - **호출 건수**: 타겟 노드 타입에 따라 형식화 (예: “1,234 Calls”, “567 SQL”)

- **에러 건수/에러율**:
  - 임계값 초과 시 빨간색 배경으로 강조
  - 하단에 에러율 게이지 바 표시
- **평균 경과 시간**:
  - 일반: 흰색 배경
  - 느림: 주황색 배경
  - 매우 느림: 빨간색 배경

기타 설정

- **연결선 애니메이션**: Slow/Very Slow 상태 연결선 애니메이션 표시 여부 제어
- **사용자 노드 표시/숨김**: User 노드 표시 여부 제어
- **고립된 노드 숨기기**: 연결되지 않은 독립 노드 제거
- **동일 노드 병합**: 도메인이 동일한 노드 병합, 성능 지표 합산. 클릭 시 소스 도메인별 분석 테이블 제공
- **HTTPC/TX 노드 병합**: HTTP 클라이언트 호출과 대응 트랜잭션 노드 병합

경과 시간 임계값

노드 성능 태그와 연결선 색상을 결정하는 기준값입니다. Slow는 Very Slow보다 작아야 합니다.

- **Slow(느림)**: 기본값 3초, 주황색
- **Very Slow(매우 느림)**: 기본값 8초, 빨간색

노드 상세 정보

툴팁

노드에 마우스를 올리면 정보가 툴팁으로 표시됩니다.

- 노드 타입 및 이름

- 호출 건수
- 에러 건수 및 에러율 (에러 시 빨간색)
- 평균 경과 시간
- 병합된 노드: 상위 10개 소스 도메인의 성능 테이블(에러율 > 에러 건수 > 평균 경과 시간 > 건수 > 이름 순 정렬)

상세 드로어

노드 클릭 시 우측에 상세 정보가 표시됩니다.

- **성능 지표**
  - 호출 수
  - 에러 수 / 에러율
  - 평균 경과 시간
  - 노드 타입별 추가 정보 (도메인, DB 서버, 드라이버 등)
- **병합된 노드**
  - 소스 도메인별 성능 분석 테이블
- **Caller 트랜잭션 목록** (최대 1,000건)
  - 상태 필터 (전체/액티브 스택/멀티 트랜잭션/에러)
  - 텍스트 검색
  - 컬럼별 정렬
  - CSV 다운로드
  - 행 클릭 시 트레이스 상세 팝업

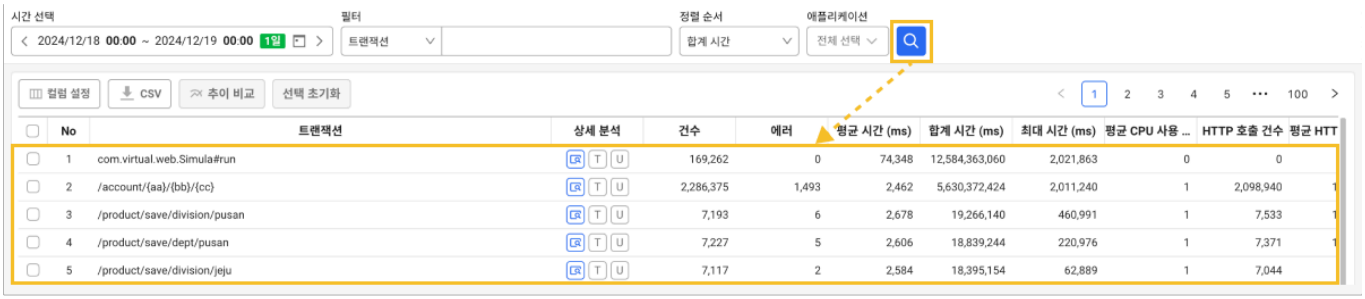
트랜잭션 분석

홈 화면 > 프로젝트 선택 > **통계**

와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 다음 **통계**의 하위 메뉴를 선택하세요. 와탭 에이전트는 트랜잭션 정보, SQL과 같은 주요 트레이스 정보에 대해 통계 정보를 수집합니다. 5분마다 목록을 만들고 서버로 전송합니다.

기본 화면 안내

시간 선택 및 **필터**, **정렬 순서**, **애플리케이션** 항목을 설정한 다음 🔍 버튼을 선택하세요.



- 각 **통계** 메뉴는 **합계**, **최대** 및 **평균 시간** 및 **건수** 기준의 정렬 옵션을 제공해 원하는 관점으로 상위 목록을 조회할 수 있습니다.
- **컬럼 설정** 버튼을 통해 기본값 외에 목록 내 조회할 컬럼을 추가하거나 제외할 수 있습니다.
- 🔍 버튼을 선택하면 조회 결과와 함께 조회 목록 위에 CSV 버튼이 나타납니다. 버튼을 선택하면 조회 목록을 CSV 형식의 파일로 다운로드할 수 있습니다.

> 통계 조회 목록이 표시된 이후에 조회 조건을 변경한 다음 🔍 버튼을 선택하지 않고 CSV 버튼을 클릭하면 변경된 조건으로 CSV 파일을 다운로드할 수 있습니다.

- 통계 목록에서 **상세 보기** 아이콘을 선택하면 건수와 평균 시간 차트를 확인할 수 있고, 에이전트 간 건수 및 에러 건수 비교, 응답 시간 비교 차트를 추가로 제공합니다. 해당 차트에서 조회를 원하는 시점을 선택하면 **트랜잭션 검색** 메뉴로 이동합니다.
- 통계 목록에서 T 버튼을 선택하면 **탑 스택 URL** 메뉴로 이동합니다.
- 통계 목록에서 U 버튼을 선택하면 **유니크 스택 URL** 메뉴로 이동합니다.

- 최대 31일 동안의 데이터를 조회할 수 있습니다. 다만, **클라이언트 IP** 메뉴에서는 최대 7일 동안의 데이터를 조회할 수 있습니다.

트랜잭션 통계

홈 화면 > 프로젝트 선택 > **통계** > **트랜잭션**

트랜잭션 통계를 수집합니다. 5분마다 최대 5,000개의 URL 별 수행 통계를 수집하여 서버에 전송합니다. 만약 서로 다른 URL의 수가 5분 동안 5,000개가 넘으면 무시됩니다.

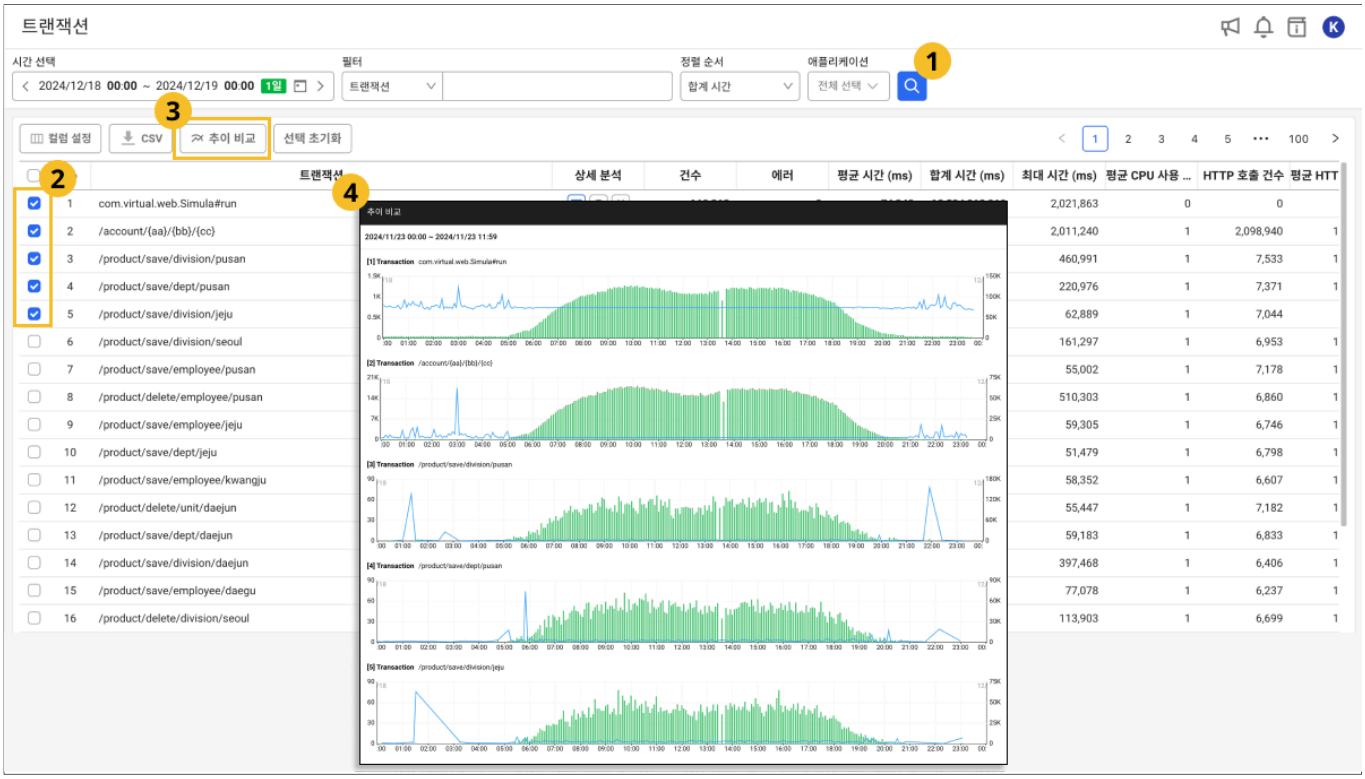
컬럼	설명
트랜잭션	트랜잭션 주소
건수	수행 건수
에러	수행된 트랜잭션 중 에러 발생 건수
평균 시간 (ms)	응답시간 합계에 대한 평균
합계 시간 (ms)	응답시간의 합
최대 시간 (ms)	최대 응답시간
평균 CPU 사용 시간	CPU 사용 시간에 대한 평균
HTTP 호출 건수	HTTP 호출 건수
평균 HTTP 호출 시간 (ms)	총 HTTP 호출 시간 / 트랜잭션 발생 건수
HTTP 호출 당 평균 시간 (ms)	총 HTTP 호출 시간 / 트랜잭션에서 발생한 총 HTTP 호출 건수
SQL 건수	SQL 수행 건수
평균 SQL 시간 (ms)	총 SQL 시간 / 트랜잭션 발생 건수

컬럼	설명
SQL 당 평균 시간 (ms)	총 SQL 시간 / 트랜잭션에서 발생한 총 SQL 건수
SQL 패치	SQL 패치 건수
평균 SQL 패치 시간 (ms)	총 SQL 패치 시간 / 트랜잭션 발생 건수
SQL 패치 당 평균 시간 (ms)	총 SQL 패치 시간 / 트랜잭션에서 발생한 총 SQL 패치 건수

> ⓘ **평균 메모리 할당량**은 에이전트 설정(whatap.conf)에서 `trace_malloc_enabled=true` 옵션을 설정해야 수집합니다. 기본값은 `false` 입니다.

추이 비교 기능 이용하기

추이 비교 기능을 이용해 다수의 트랜잭션 추이를 한눈에 비교할 수 있습니다. 이 기능은 확인해야할 트랜잭션 건수가 많은 경우, 주요 문제를 빠르게 식별하고 트래픽 이슈를 해결하는 데 유용합니다.



1. 조회하길 원하는 시간과 모니터링 대상을 선택한 다음 버튼을 선택하세요.
2. 트랜잭션 목록에서 비교하길 원하는 트랜잭션을 하나 이상 선택하세요.
3. **추이 비교** 버튼을 선택하세요.
4. 선택한 트랜잭션의 추이 그래프가 나열된 **추이 비교** 창이 나타납니다.

추이 비교 창에서 추이 그래프를 확인해 트랜잭션 간의 트래픽 패턴을 비교하세요. 특정 시간대, 서비스 경로의 트래픽 분포를 비교할 수 있습니다.

- 최대 100개의 트랜잭션을 선택해 조회할 수 있습니다.
- 선택한 트랜잭션이 많은 경우, 로딩 시간이 증가할 수 있습니다.
- 정확한 분석을 위해 **필터** 옵션을 적절히 활용하세요.

트랜잭션 도메인 통계

홈 화면 > 프로젝트 선택 > **통계** > **도메인**

와탭 에이전트는 도메인별 트랜잭션 통계를 수집할 수 있습니다. 하나의 서버에 여러 개의 도메인을 분리하여 서비스하는 시스템에서는 도메인 별 분석이 필요할 수 있습니다. 다음 옵션을 설정하세요.

whatap.conf

```
stat_domain_enabled=true
```

수집되는 데이터는 도메인별 URL의 처리 현황을 파악할 수 있습니다.

컬럼	설명
도메인	서비스 도메인
트랜잭션	트랜잭션 주소
건수	수행 건수
에러	수행된 트랜잭션 중 에러 발생 건수
평균 시간 (ms)	응답시간 합계에 대한 평균
합계 시간 (ms)	응답시간의 합

트랜잭션 Caller 통계

홈 화면 > 프로젝트 선택 > **통계** > **Caller**

멀티 서버가 rest 호출로 연결되면 **Caller**와 **Callee** 간의 연관 통계를 수집할 수 있습니다. 이 데이터를 수집하기 위해 다음의 옵션을 설정하세요.

whatap.conf

```
mtrace_rate=100
mtrace_spec=v1
stat_mtrace_enabled=true
```

msa-system

위와 같은 아키텍처에서 **Caller & Callee** 통계는 api1, api2에서만 조회할 수 있습니다. 사용자 브라우저에서 호출되는 시스템에서는 **Caller** 통계를 조회할 수 없습니다.

하지만 **Caller** 쪽 서버에서 데이터를 전송해야 하므로 모든 서버에 적절한 설정이 들어가야 합니다.

whatap.conf of [front]

```
mtrace_rate=100
mtrace_spec=v1
stat_mtrace_enabled=true
```

whatap.conf of [api1] & [api2]

```
mtrace_spec=v1
stat_mtrace_enabled=true
```

수집되는 통계 데이터는 다음과 같습니다. **Callee** 쪽에서 조회되어야 합니다.

컬럼	설명
Caller Pcode	Caller에 해당하는 프로젝트 코드
Caller 에이전트 종류	Caller에 해당하는 에이전트가 속한 그룹 단위
Caller URL	Caller 트랜잭션의 트랜잭션 주소
트랜잭션	Callee 트랜잭션 주소
건수	수행 건수
에러	수행된 트랜잭션 중 에러 발생 건수
평균 시간 (ms)	응답시간 합계에 대한 평균

Caller 메뉴에서 한 번에 다운로드할 수 있는 CSV 라인 수는 최대 10,000줄입니다.

IP 트랜잭션 통계

홈 화면 > 프로젝트 선택 > **통계** > **IP 트랜잭션**

특정 IP 주소별로 애플리케이션에서 호출한 URL과 호출 건수를 조회할 수 있는 기능을 제공합니다. 이 기능은 비정상적으로 많은 요청이 발생하는 IP 주소를 확인하고, 해당 IP 주소에서 호출된 URL을 분석하여 잠재적인 보안 위협이나 성능 이슈를 사전에 식별하는 데 유용합니다. 사용자는 이 정보를 바탕으로 불필요하거나 악의적인 요청을 차단할 수 있습니다.

필터 옵션을 통해 IP 주소 또는 트랜잭션 주소를 필터링하여 조건에 맞는 데이터를 빠르게 조회할 수 있습니다. 특정 IP나 트랜잭션 주소 패턴을 분석할 때 유용합니다.

IP 트랜잭션 통계를 수집하기 위해 다음 옵션 설정이 필요합니다. 기본값은 `false` 입니다.

whatap.conf

```
stat_ipurl_enabled=true
```

컬럼	설명
IP	요청을 발생시킨 클라이언트의 IP 주소
트랜잭션	호출된 트랜잭션 주소, 특정 경로 변수는 `{}`로 표시
건수	해당 IP 주소에서 발생한 요청의 총 건수
에러	요청 중 발생한 에러 수
평균 시간 (ms)	해당 요청의 평균 응답 시간(밀리초, ms)

- **비정상적인 트래픽 감지**: 비정상적으로 호출 건수가 많은 IP를 탐지하여 해당 IP에서 발생한 트랜잭션 주소 패턴을 분석하고 잠재적인 공격을 차단할 수 있습니다. 예를 들어, 특정 IP에서 `/account/{aa}/{bb}/{cc}` URL에 대해 과도한 요청이 발생한 경우 해당 IP를 블락하거나 요청을 제한하는 조치를 취할 수 있습니다.

- **응답 시간 분석**: 평균 응답 시간이 비정상적으로 높은 IP를 확인하여 네트워크 또는 서버 측의 성능 문제를 파악하고 개선할 수 있습니다.
- **에러 발생 분석**: 특정 IP에서 에러가 다수 발생할 때 해당 IP의 요청에 문제가 있는지 또는 시스템에 특정 문제가 있는지를 진단할 수 있습니다.

트랜잭션 Referer 통계

홈 화면 > 프로젝트 선택 > **통계** > **리퍼러**

와탭 에이전트는 Referer 별 트랜잭션 통계를 수집할 수 있습니다. 다음 옵션을 설정하세요. 기본값은 `false` 입니다.

whatap.conf

```
stat_referer_enabled=true
```

수집되는 데이터는 Referer 정보를 통계로 확인할 수 있습니다.

컬럼	설명
Referer	Referer 주소
트랜잭션	트랜잭션 주소
건수	수행 건수
에러	수행된 트랜잭션 중 에러 발생 건수
평균 시간 (ms)	응답시간 합계에 대한 평균
합계 시간 (ms)	응답시간의 합

에러 통계

홈 화면 > 프로젝트 선택 > **통계** > **에러**

5분 동안 발생한 서비스 에러에 대한 통계입니다. 서로 다른 에러와 트랜잭션 이름을 키로 발생 건수를 수집합니다. 에러 발생 시점의 스택을 제공합니다. 5분 당 최대 1,000 가지 서로 다른 에러를 통계화합니다.

컬럼	설명
클래스	에러 클래스
트랜잭션	트랜잭션 주소
메시지	에러 메시지
건수	에러 발생 건수
에이전트	에이전트 이름
에이전트 서버	`whatap.onode`로 분류된 에이전트 그룹 단위
에이전트 종류	`whatap.okind`로 분류된 에이전트 그룹 단위
최초 생성 시간	조회 기간 중 트랜잭션의 에러가 최초 생성된 통계시간(5분 또는 1시간)

SQL 수행 통계

홈 화면 > 프로젝트 선택 > **통계** > **SQL**

5분 동안의 SQL 수행 통계를 수집합니다. 5분 동안 서로 다른 SQL 문장이 최대 5,000까지만 허용됩니다. 만약 하나의 Java 프로세스에서 한계를 넘는 SQL이 발생하면 통계 데이터에서는 버려집니다.

컬럼	설명
데이터베이스	데이터베이스 종류
SQL	SQL 쿼리문
트랜잭션	해당 SQL을 호출한 URL 중 하나를 임의로 수집한 정보
CRUD	Create, Read, Update, Delete
건수	쿼리 실행 건수
에러 건수	실행된 쿼리 중 에러 발생 건수
평균 시간 (ms)	쿼리 실행 시간 합계에 대한 평균
합계 시간 (ms)	쿼리 실행 시간의 합
표준 편차	쿼리 실행 시간에 대한 표준 편차

ⓘ 트랜잭션은 5분 동안 해당 SQL을 호출한 URL 중 하나(마지막 호출 URL)를 분석 활용을 위해 수집합니다.

HTTP Call 수행 통계

홈 화면 > 프로젝트 선택 > **통계** > **HTTP 호출**

HTTP 및 HTTPS 프로토콜을 통해 5분 동안의 애플리케이션 외부 호출 통계를 수집합니다. 5분 동안 서로 다른 Http Call 문장이 최대 5,000까지만 허용됩니다. 만약 하나의 애플리케이션 프로세스에서 한계를 넘는 외부 Http Call이 발생하면 통계 데이터에서는 버려집니다.

컬럼	설명
HTTP 호출 URL	HTTP 호출 URL 해시(hash)
트랜잭션	해당 HTTP API를 요청한 URL 중 하나를 임의로 수집한 정보

컬럼	설명
호스트	Host 또는 IP 주소
포트	TCP 포트
전체	전체 수행 건수
에러	에러 발생 건수
평균 시간 (ms)	응답시간 합계에 대한 평균
합계 시간 (ms)	응답시간의 합
최소 시간 (ms)	최소 응답시간
최대 시간 (ms)	최대 응답시간
표준 편차 (ms)	HTTP 호출 시간에 대한 표준 편차

클라이언트 IP 지역별 호출 건수

홈 화면 > 프로젝트 선택 > **통계** > **클라이언트 IP**

서버와 클라이언트의 물리적 거리는 데이터 전송 시간과 비례합니다. 클라이언트 IP의 지역별 호출 건수를 통계적으로 수집합니다. 5분당 수집할 수 있는 서로 다른 IP 수는 인스턴스 당 최대 70,000개입니다.

컬럼	설명
클라이언트 IP	클라이언트의 IP 주소
국가	사용자 접속 국가
도시	사용자 접속 도시
건수	전체 호출 건수

- 클라이언트 IP에 대한 데이터는 최대 7일 동안의 데이터를 조회할 수 있습니다.
- 와탭은 클라이언트와 관련한 정보를 기본 저장합니다. 사용자 데이터 수집과 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

지역별 점유 비중 차트

점유 비중 차트는 사용자의 접속 위치를 국가와 도시 단위로 수집해 제시합니다.

- 국가 영역 선택 시 해당 국가의 도시와 접속 유저의 IP 등을 확인할 수 있는 상세 차트로 이동합니다.
- 중심 원 선택 시 기존 국가 영역 선택 차트로 돌아갑니다.

유저 에이전트별 호출 건수

홈 화면 > 프로젝트 선택 > **통계** > **유저 에이전트**

유저 에이전트 문자열의 Hash별로 호출 건수를 수집합니다. 5분당 수집할 수 있는 서로 다른 User Agent Hash는 인스턴스 당 최대 500개입니다.

컬럼	설명
유저 에이전트	브라우저의 유저 에이전트 정보
운영체제	브라우저가 실행되는 운영 체제 환경
브라우저	브라우저 종류
건수	수집된 전체 건수

클라이언트 브라우저 통계

홈 화면 > 프로젝트 선택 > **통계** > **클라이언트 브라우저**

유저 에이전트의 문자열을 분석하여 최종 사용자가 사용하는 브라우저의 종류, 운영체제 기준으로 집계한 통계 정보를 제공합니다.

수집한 통계 정보를 기준으로 생성된 파이 차트를 통해 점유 비중을 확인할 수 있습니다.

컬럼	설명
유저 에이전트	브라우저의 유저 에이전트 정보
운영체제	브라우저가 실행되는 운영 체제 환경
브라우저	브라우저 종류
건수	수집된 전체 건수

Java 보고서

홈 화면 > 프로젝트 선택 > **보고서**

보고서는 개별 프로젝트에 대해 서비스 이용 현황과 장애 발생 기록을 보고하는 문서입니다. 서비스 모니터링 담당자는 보고서를 통해 관련 부서 담당자들과 현황을 공유합니다.

모니터링 데이터 분석은 서비스의 개선 방향을 정하는 지표가 되기 때문에 중요합니다. 하지만 여러 대시보드의 데이터를 취합해서 문서화하는 일은 번거롭습니다.

와탭의 **보고서** 메뉴는 **보고서 작성 업무 자동화**를 지원합니다. 매주 보고서를 작성해야 하는 일, 정해진 시간에 보고서를 공유하는 일, 여러 가지 서식을 관리하는 일 모두 **보고서** 메뉴에서 할 수 있습니다.

> **보고서**를 다운로드하거나 인쇄, 메일 발송 예약을 원하면 해당 버튼을 클릭하세요. HTML 형식으로 다운로드할 수 있습니다.

보고서와 통합 보고서의 차이

- **보고서**는 하나의 프로젝트에 대한 보고서를 생성하는 메뉴입니다. 따라서 프로젝트를 선택한 다음 사이드 메뉴의 **보고서** 메뉴를 통해 진입할 수 있습니다.
- **통합 보고서**는 여러 프로젝트에 대한 보고서를 생성하는 메뉴입니다. 프로젝트를 선택하지 않고 홈 화면의 사이드 메뉴에서 진입할 수 있습니다. **통합 보고서**에 대한 자세한 내용은 다음 문서에서 확인할 수 있습니다.

보고서의 종류

기본 보고서는 대기업, 공공기관 및 IT 서비스 기업에서 실제로 사용하고 있는 양식입니다. 원하는 양식이 있으면 support@whatap.io로 요청해 주세요. 요청한 보고서는 **보고서** 목록에 추가됩니다.

보고서 양식 추가

- 요청 양식이 범용성이 있을 경우 **보고서** 목록에 추가로 제공합니다.
- 요청 시 HTML 및 CSS 양식이 함께 제공될 경우 추가가 용이합니다.

요청 양식 예시

보고서 양식 추가 요청 시 예시와 같이 원하는 HTML 및 CSS 양식 첨부를 권장합니다. 세부 사항은 다음 예시 파일을 참조하세요.

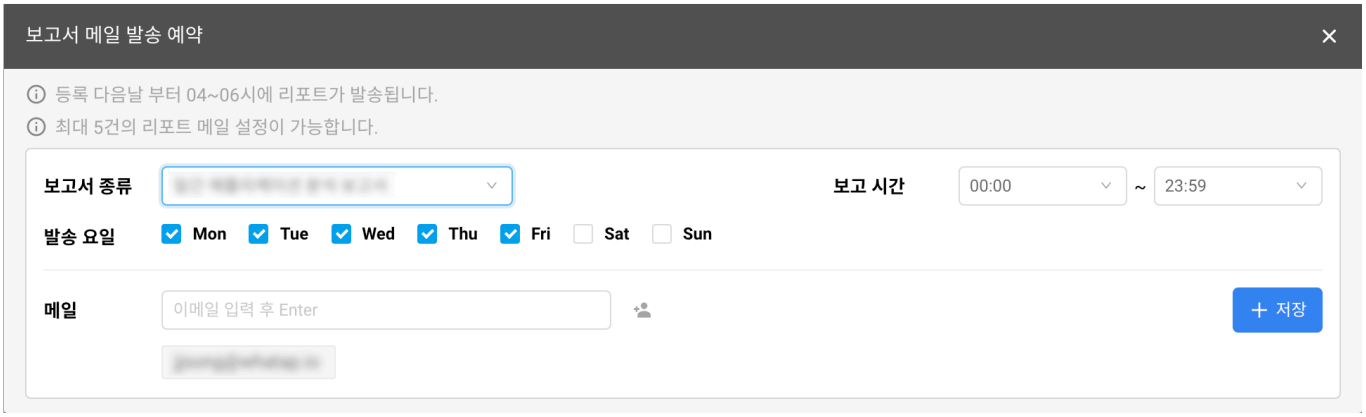
- 예시 파일: 다운로드 후 ZIP 형식의 압축을 풀어 HTML 형식의 예시 파일을 확인하세요. 파일 내 주석에 커스터마이징을 위한 CSS 영역과 HTML 영역 안내가 기술되어있습니다.

제약 사항

- 와탭 모니터링에서 현재 제공하고 있는 보고서 템플릿의 각 항목을 추가 또는 수정, 삭제가 필요한 경우 제공할 수 있습니다.
- 현재 제공하는 보고서 템플릿에 없는 데이터나 분석 결과를 추가하려면 개발자의 검토가 필요합니다.
- 데이터 추가 및 가공은 와탭 모니터링에서 수집 및 저장된 데이터(메트릭스)로 한정합니다. 다른 도구 및 외부 데이터는 추가할 수 없습니다.
- 특정 고객사의 요구를 충족하는 범용적이지 않은 보고서 양식은 제공하기 어렵습니다.
- 자세한 내용은 support@whatap.io 또는 영업 담당자에게 문의하세요.

메일 발송 예약

정기적으로 보고서를 이메일로 받으려면 **보고서 메일 발송 예약**을 선택하세요. 출근 직후 수행하던 여러 가지 서비스 점검 절차를 메일 확인으로 대체할 수 있습니다.



APM 보고서

홈 화면 > 프로젝트 선택 > **보고서**

화면 위에 **보고서**에서 ⌄ 버튼을 클릭하면 일간, 주간, 월간으로 분류된 각종 분석 보고서를 확인할 수 있습니다. 원하는 보고서 양식을 선택한 다음 조회하길 원하는 날짜와 시간, 필터 옵션을 설정하세요. 🔍 버튼을 선택하면 선택한 보고서를 확인할 수 있습니다.

> ⓘ 주간 및 월간 보고서를 생성할 때는 데이터의 조회 범위에 따라 다소 시간이 소요될 수 있습니다.

일간 보고서

하루 동안 애플리케션 성능의 변화를 확인할 수 있는 보고서를 제공합니다. 일간 보고서로 제공하는 보고서 양식은 다음과 같습니다.

- **일간 애플리케이션 보고서**
- **애플리케이션 피크 성능 비교**
- **일간 애플리케이션 사용자 통계**
- **일간 메트릭 추이현황(Avg/Max)**
- **일간 애플리케이션 사용자정의 보고서**

일간 애플리케이션 보고서

하루 동안 애플리케이션 성능에 대한 주요 지표를 종합적으로 분석할 수 있는 보고서입니다.

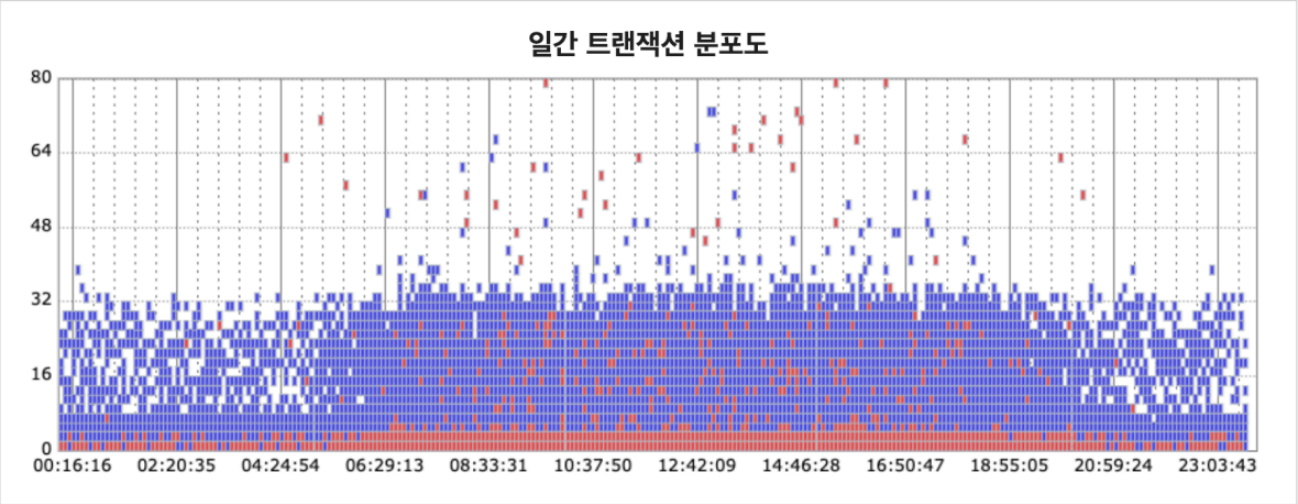
> ⓘ 주요 용어 및 지표에 대한 자세한 내용은 다음 문서를 참조하세요.

- **개요**: 애플리케이션의 주요 성능 지표를 확인할 수 있습니다.
- **피크타임 성능요약**: 하루 동안 처리량(TPS)이 가장 높았던 시간과 해당 시간의 지푯값을 확인할 수 있습니다.
- **지표별 평균/최대값**: 하루 동안 주요 지표별 평균값과 발생 시간 기준 최댓값을 확인할 수 있습니다.

- **성능추이차트**: 시스템의 주요 지표들을 시각적으로 표현한 시계열 차트입니다. 시간의 흐름에 따라 주요 지표들의 수치를 확인할 수 있습니다. 시스템의 안정성 및 성능을 평가하는데 중요한 지표입니다.

  ⓘ ◦ **주간 애플리케이션 보고서**는 일별 변화량을 추가 제공합니다.
  ◦ 차트 오른쪽 위에 **CSV** 버튼을 선택하면 데이터를 CSV 형식 파일로 다운로드할 수 있습니다.

- **일간 트랜잭션 분포도**: 하루 동안 시간대별 트랜잭션 분포와 에러 발생 상황을 한눈에 파악할 수 있습니다. 그래프의 x축은 각 트랜잭션의 종료 시각, y축은 응답시간(초)을 의미합니다. 정상 트랜잭션의 경우 파란색으로 표시되고, 에러가 발생한 경우 빨간색으로 표시됩니다.



- **애플리케이션별 현황**: 모니터링 대상 에이전트별 현황을 확인할 수 있습니다. **액티브건수**는 진행 중인 트랜잭션 건수입니다.
- **트랜잭션 TOP 10**: **트랜잭션 수**, **에러건수**, **평균응답시간** 기준으로 상위 10개의 트랜잭션을 확인할 수 있습니다.
- **탑스택 TOP 10**: Stack Trace 상의 스텝을 기준으로 스텝 간의 호출 빈도를 비율로 계산한 정보를 간소화하여 볼 수 있습니다.

  ⓘ **분석** > **스택** > **탑 스택** 메뉴에서 스택 상세 내용을 조회할 수 있습니다.

- **SQL 평균응답시간 TOP 10**: 사용된 SQL 쿼리문에 대해 평균 응답시간을 기준으로 정렬한 통계 정보입니다.

  ⓘ ◦ **증가%**: 조회 전일까지 3일간의 평균 데이터와 조회 당일 데이터를 비교하여 증가한 비율을 계산한 수치입니다.

예, 4월 25일 00시 00분 ~ 23시 59분 데이터 조회 → 4월 22일 00시 00분 ~ 4월 24일 23시 59분 데이터와 비교
해당 수치가 10% 이상인 경우 붉은색으로, 5%~10% 사이인 경우 주황색으로 표시됩니다. 5% 미만의 증가율은 표시하지 않습니다.

- **통계** > **SQL** 메뉴에서 각 쿼리의 발생 시간 및 성능을 상세 조회할 수 있습니다.

애플리케이션 피크 성능 비교

모니터링 대상 에이전트별로 애플리케이션의 최대 성능 지표를 일간(빨간색), 전일(파란색), 그리고 지난주(녹색) 데이터와 비교하여 제공합니다. 각 지표는 특정 시간대의 최대값과 그에 대응하는 시간의 지푯값을 함께 표시합니다. 이를 통해 각 애플리케이션의 성능 변동을 한눈에 파악할 수 있습니다.

주요 용어 및 지표에 대한 자세한 내용은 다음 문서를 참조하세요.

일간 애플리케이션 사용자 통계

10초, 5분, 30분 단위 동시 접속 사용자와 1시간 방문자 수를 시계열 차트로 제공하는 보고서입니다. 이 보고서는 사용 패턴을 이해하고 사용자 트래픽을 효과적으로 관리하는 데 중요한 데이터로 활용할 수 있습니다.

- **10s 동시접속 사용자**: 실시간 트래픽 변화를 세밀하게 모니터링하여 사용자 수가 갑작스럽게 증가하는 시간대를 확인할 수 있습니다. 또한 사용자가 집중되는 시간대에 콘텐츠 및 서비스를 제공하는 전략을 수립할 수도 있습니다.
- **5m / 30m 동시접속 사용자**: 중단기적 사용자 패턴을 분석하여 서버 자원 및 네트워크 대역폭을 효율적으로 배분할 수 있습니다. 피크 시간대를 예측하여 해당 시간대에 더 많은 자원을 할당하는 계획을 수립할 수 도 있습니다.
- **1h 방문자 수**: 시간별로 사용자의 방문 패턴을 분석해 마케팅 전략 및 운영 계획에 활용할 수 있습니다.

일간 메트릭 추이현황(Avg/Max)

애플리케이션의 다양한 성능 지표를 시계열 차트로 제공하는 보고서입니다. 각 지표의 평균 및 최댓값을 시각적으로 표현합니다. 이를 통해 시스템의 성능 추이를 모니터링하고 성능 최적화 및 문제 해결에 필요한 데이터로 활용할 수 있습니다.

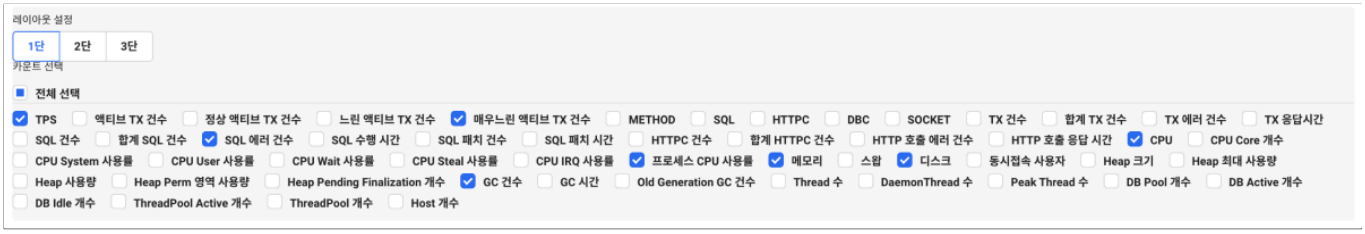
차트의 제목은 **Category[Field]** 형식으로 구성되어 있습니다. 애플리케이션의 성능 지표에 대한 자세한 내용은 다음 문서를

참조하세요.

> ⓘ
> - **Avg.**: 파란색 선으로 표시되며, 각 시간대의 평균값입니다.
> - **Max.**: 빨간색 선으로 표시되며, 각 시간대의 최댓값입니다.

일간 애플리케이션 사용자정의 보고서

사용자가 원하는 지표를 선택해 보고서를 생성할 수 있습니다. 선택한 지표를 기준으로 시계열 차트를 제공하는 보고서를 만들 수 있습니다.



1. 화면 위에 **보고서**에서 **일간 애플리케이션 사용자정의 보고서**를 선택하세요.
2. **레이아웃 설정**에서 원하는 단수를 설정하세요.
3. 원하는 지표를 선택하세요. 전체를 선택하려면 **전체 선택**을 클릭하세요.
4. 버튼을 선택하세요.

보고서가 생성됩니다.

주간 애플리케이션 보고서

애플리케이션의 한 주간 주요 지표를 종합적으로 분석할 수 있는 보고서입니다.

> ⓘ 주요 용어 및 지표에 대한 자세한 내용은 다음 문서를 참조하세요.

- **개요**: 한주간 총 방문자와 트랜잭션 건수를 제공합니다.
- **가동률**: 애플리케이션이 주어진 시간 동안 동작한 시간을 백분율로 나타낸 것입니다. 예를 들어, 가동률이 100%라면 해당

애플리케이션이 조회 기간에 단 한 번도 중단되지 않고 계속 동작했음을 의미합니다. 반면에 가동률이 낮으면 해당 애플리케이션이 조회 기간에 중단되었거나 비정상적으로 동작한 시간이 있었음을 나타냅니다.

- **성능추이차트**: 시스템의 주요 지표들을 시각적으로 표현한 시계열 차트입니다. 시간의 흐름에 따라 주요 지표들의 수치를 확인할 수 있습니다. 시스템의 안정성 및 성능을 평가하는데 중요한 지표입니다.

> ◦ 일별 변화량과 일별 평균 및 최댓값을 제공합니다.
> ◦ 차트 오른쪽 위에 **CSV** 버튼을 선택하면 데이터를 CSV 형식 파일로 다운로드할 수 있습니다.

- **트랜잭션 TOP 10 비교**: **트랜잭션 수**, **에러건수**, **평균응답시간** 기준으로 상위 10개의 트랜잭션을 확인할 수 있습니다. 각 트랜잭션의 일주일 전과 조회 주의 실행 건수, 증감 건을 제공합니다.
- **탑스택 TOP 10 비교**: 스택 트레이스 상위 10개를 일주일 전과 조회 주간의 퍼센트와 건수를 비교한 목록입니다. 가장 많이 호출되는 스택 항목을 식별할 수 있습니다.

> **분석** > **스택** > **탑 스택** 메뉴에서 스택 상세 내용을 조회할 수 있습니다.

- **SQL 평균응답시간 TOP 10 비교 (실행 건수 100건 이상)**: SQL 평균응답시간을 기준으로 상위 10개의 SQL 문을 비교한 목록입니다. 각 SQL 문에 대해 일주일 전과 조회 주의 실행 건수, 평균 응답 시간, 증감 시간을 제공합니다.
- **프로젝트 에러현황(에러메시지 포함)**: 한 주간의 프로젝트의 에러 통계를 에러 클래스와 에러 메시지 기준으로 제공하는 통계 정보입니다.

월간 애플리케이션 보고서

한 달간 애플리케이션 성능에 대한 주요 지표를 종합적으로 분석할 수 있는 보고서입니다.

> 주요 용어 및 지표에 대한 자세한 내용은 다음 문서를 참조하세요.

- **개요**: 애플리케이션의 주요 성능 지표를 확인할 수 있습니다.
- **애플리케이션별 현황**: 에이전트별 **IP 주소**, **OS 정보**, **가동률**을 확인할 수 있습니다.
- **성능추이차트**: 시스템의 주요 지표들을 시각적으로 표현한 시계열 차트입니다. 시간의 흐름에 따라 주요 지표들의 수치를 확인할 수 있습니다. 시스템의 안정성 및 성능을 평가하는데 중요한 지표입니다.

> ⓘ 차트 오른쪽 위에 **CSV** 버튼을 선택하면 데이터를 CSV 형식 파일로 다운로드할 수 있습니다.

- **트랜잭션 TOP 10 비교**: **트랜잭션 수**, **에러건수**, **평균응답시간** 기준으로 상위 10개의 트랜잭션을 확인할 수 있습니다. 각 트랜잭션의 한달 전과 조회 당월의 실행 건수, 증감 건을 제공합니다.
- **탑스택 TOP 10 비교**: 스택 트레이스 상위 10개를 한달 전과 조회 당월의 퍼센트와 건수를 비교한 목록입니다. 가장 많이 호출되는 스택 항목을 식별할 수 있습니다.

> ⓘ **분석** > **스택** > **탑 스택** 메뉴에서 스택 상세 내용을 조회할 수 있습니다.

- **SQL 평균응답시간 TOP 10 비교**: SQL 평균 응답 시간을 기준으로 상위 10개의 SQL 문을 비교한 목록입니다. 각 SQL 문에 대해 한달 전과 조회 당월의 실행 건수, 평균 응답 시간, 증감 시간을 제공합니다.

주요 용어 및 지표 안내

각 보고서에 표시된 주요 지표들에 대한 설명입니다.

항목	설명
일간 사용자 수 (**DAU**, Daily Active User) / **일일 사용자**	일일 활성 사용자 수로, 하루동안 애플리케이션을 사용한 사용자 수입니다.
주간 총 방문자수	한 주간 애플케이션을 사용한 총 방문자 수입니다.
월간 사용자수 (**MAU**, Monthly Active User)	한달간 애플리케이션을 한 번 이상 사용한 고유 사용자 수입니다.
동시접속 사용자	중복을 제거한 동시 접속 사용자 수입니다.
사용자 수 / **실시간 사용자**	특정 시간대에 애플리케이션에 접속한 사용자 수입니다.
트랜잭션 건수	요청된 전체 트랜잭션 수입니다.
일일 TX / **일간 트랜잭션 수**	하루 동안 처리된 전체 트랜잭션 수입니다.

항목	설명
주간 트랜잭션건수	한 주간 요청된 트랜잭션 수입니다.
일간 트랜잭션 분포도	하루 동안 시간대별 트랜잭션 분포와 에러 발생 상황을 한눈에 파악할 수 있는 분포도 차트입니다.
에러비율(%)	전체 트랜잭션 건수 중 에러 비율입니다.
이벤트 건수	메트릭스 이벤트 중 Critical 레벨인 이벤트의 수입니다.
에러건수	트랜잭션 처리 중 발생한 오류로 인해 실패한 트랜잭션 수입니다.
애플리케이션 수	조회 기간 모니터링한 애플리케이션 수(에이전트 수)입니다.
전체 코어	조회 기간 사용된 CPU 코어 수입니다.
APDEX*100** (Application Performance Index)	조회 기간 동안 애플리케이션에 대한 고객 성능 만족도입니다.
처리량 (TPS, Transaction Per Second)	초당 처리된 트랜잭션의 수입니다.
평균응답시간(ms) / 응답시간	트랜잭션에 대한 평균 응답 시간입니다.
CPU 코어	사용된 CPU 코어 수입니다.
시간당 방문자 수	조회 기간의 시간당 방문자 수입니다.
서버 CPU 사용률	서버의 CPU 사용률입니다.
서버 메모리 사용률	서버의 메모리 사용률입니다.
액티브건수 / 액티브 트랜잭션	진행 중인 트랜잭션 건수입니다.
CPU	CPU 사용률입니다.
힙(최소값)	최소 힙 메모리 사용량입니다.

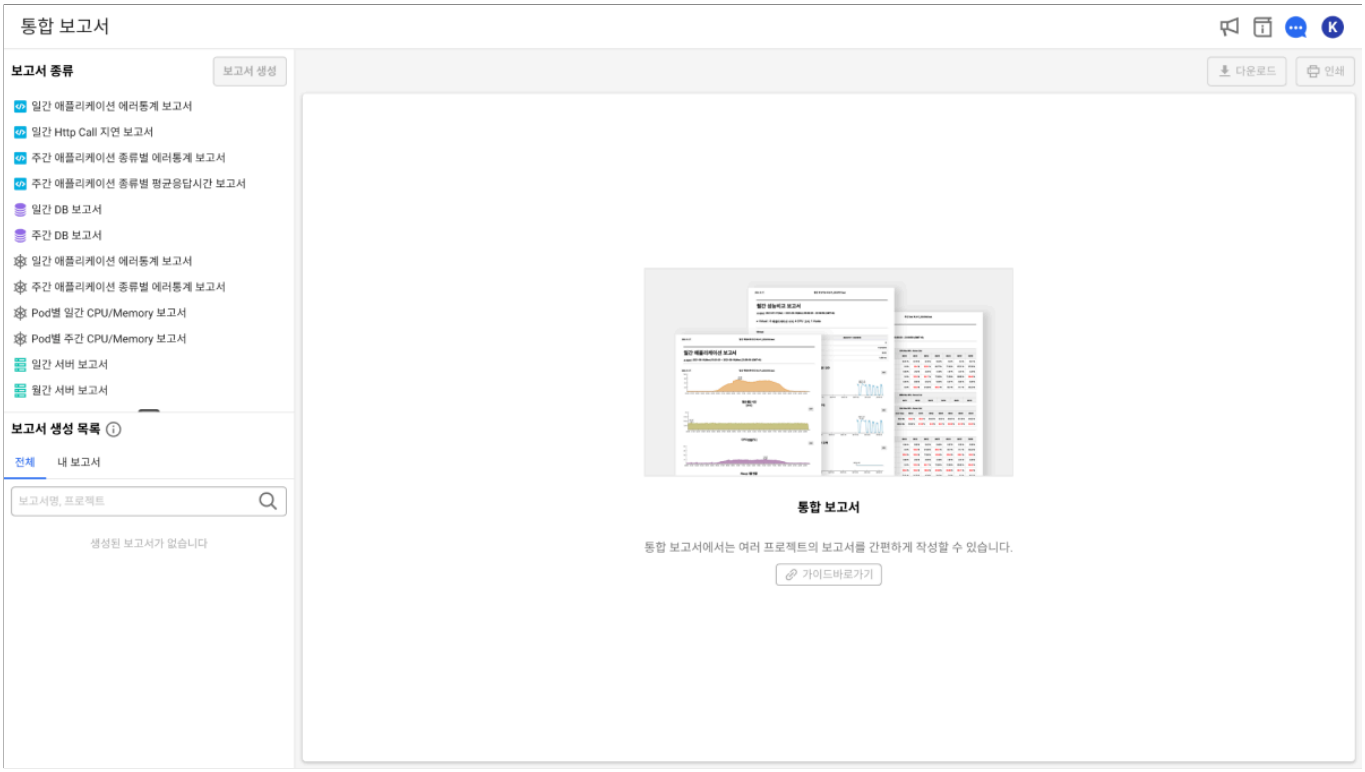
항목	설명
가동률	애플리케이션이 주어진 시간 동안 정상적으로 동작한 시간을 백분율로 나타낸 것입니다.

- **자바 CPU 사용률**: Java 애플리케이션의 CPU 사용률입니다.
- **자바 힙 메모리 사용량**: JVM의 힙 메모리 영역에서 할당된 메모리 양입니다.

통합 보고서

홈 화면 > **통합 보고서**

사용자는 여러 프로젝트를 통합한 보고서를 생성하고 공유할 수 있습니다. 보고서 종류와 포함할 프로젝트를 선택해 보고서를 생성할 수 있으며, 생성 목록에서 작업 진행 상태를 확인할 수 있습니다. 특정 프로젝트에서 에러가 발생하면 에러 내용을 확인할 수 있습니다. 생성이 완료된 보고서는 즉시 조회하거나, 다운로드 및 인쇄 기능으로 공유할 수 있습니다.



> ⓘ **통합 보고서** 메뉴는 **Application**, **Server**, **Kubernetes**, **Database** 제품의 프로젝트에 대한 보고서만 생성할 수 있습니다.

주요 기능 안내

통합 보고서 메뉴에서 제공하는 주요 기능은 다음과 같습니다.

- **멀티 프로젝트 지원**

  여러 프로젝트를 선택해 한 번에 보고서를 생성할 수 있어 편리합니다. 프로젝트에서 이용할 수 있는 **보고서** 메뉴는 해당 프로젝트에 대한 보고서만 생성할 수 있습니다.

- **업무 단위 보고서 생성**

  보고서 생성 기능은 업무 단위로 프로젝트를 선택해 템플릿으로 저장할 수 있습니다. 보통 업무 단위로 프로젝트를 나누어 사용하는 마이크로 서비스 아키텍처(Micro Service Architecture, MSA) 환경에서 유용한 기능입니다.

- **대용량 데이터 최적화**

  보고서 생성 시간이 획기적으로 줄어들었습니다. 그리고 일부 프로젝트 보고서 작성 과정에서 오류가 발생하더라도 보고서 결과를 조회할 수 있습니다.

- **보고서 생성 중 동시 작업 가능**

  보고서를 생성하는 동시에 다른 업무를 진행할 수 있습니다. 프로젝트에서 이용할 수 있는 **보고서** 메뉴는 보고서 생성 작업을 완료할 때까지 기다려야 합니다.

  통합 보고서 메뉴에서 보고서 생성을 시작한 후 다른 페이지로 이동할 수 있습니다. 또한 보고서 결과 목록에서 진행 상태와 완료된 보고서를 조회할 수 있습니다.

- **보고서 작업 공유**

  동일 프로젝트 권한을 가진 사용자라면 누구든 생성한 보고서를 조회할 수 있습니다. 관련 보고서가 이미 생성됐는지 확인할 수 있어 중복으로 작성하지 않을 수 있습니다.

보고서와 통합 보고서의 차이

프로젝트에서 이용할 수 있는 **보고서** 메뉴에서는 해당 프로젝트에 대한 보고서를 생성할 수 있습니다. 초기 화면에서 프로젝트를 선택 후 사이드 메뉴의 **보고서** 또는 **통계/보고서** 메뉴를 통해 접근할 수 있습니다.

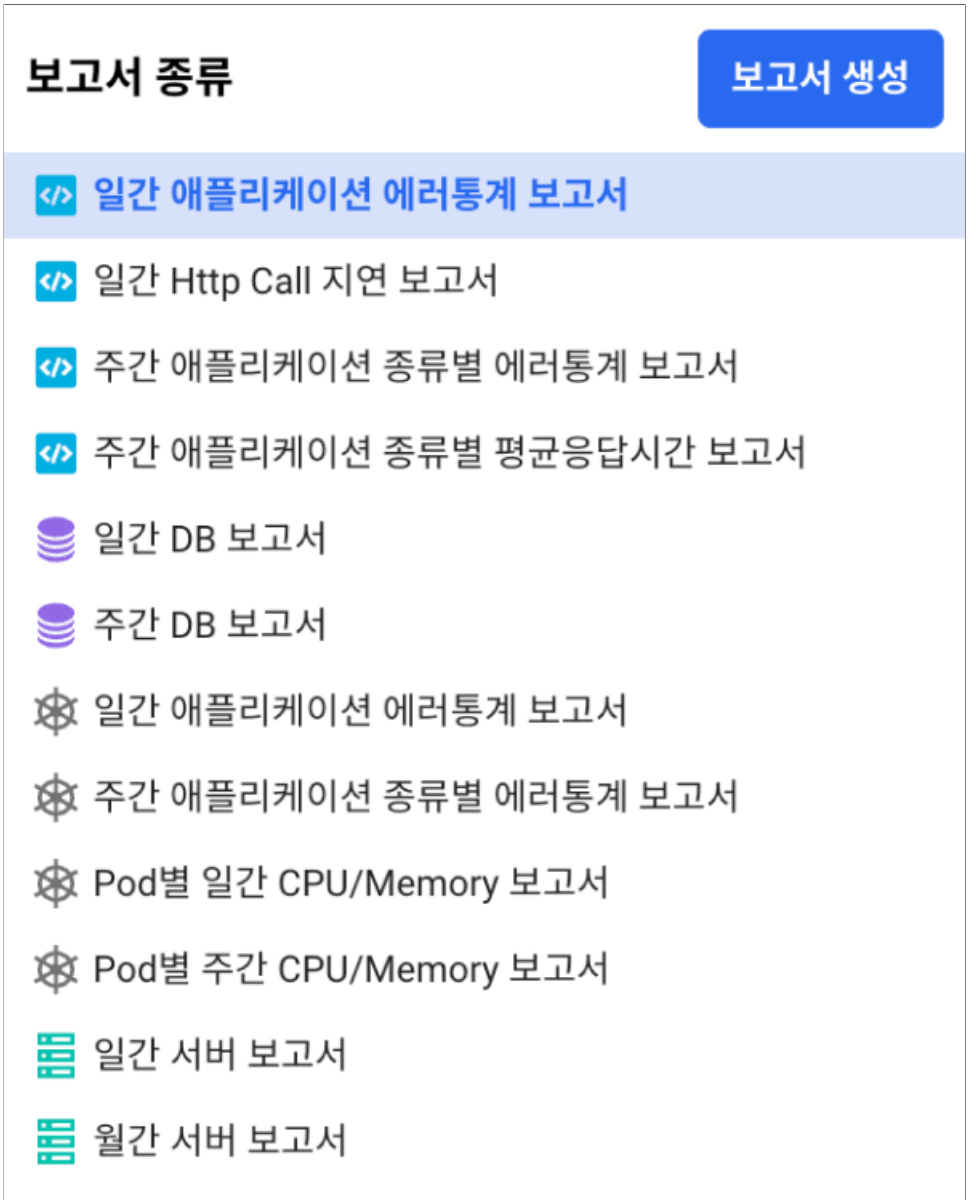
통합 보고서 메뉴에서는 여러 프로젝트에 대한 보고서를 생성할 수 있습니다. 프로젝트를 선택하지 않고 초기 화면에서 접근할 수 있습니다.

보고서 생성하기

보고서 종류 선택하기

보고서 종류 목록에서 원하는 보고서 유형을 선택하면 **보고서 생성** 버튼이 활성화됩니다. 보고서 이름 앞의 **아이콘**은 보고서 생성을 지원하는 프로젝트 플랫폼을 의미합니다.

- **Application** 프로젝트 플랫폼을 나타냅니다.
- **Database** 프로젝트 플랫폼을 나타냅니다.
- **Kubernetes** 프로젝트 플랫폼을 나타냅니다.
- **Server** 프로젝트 플랫폼을 나타냅니다.



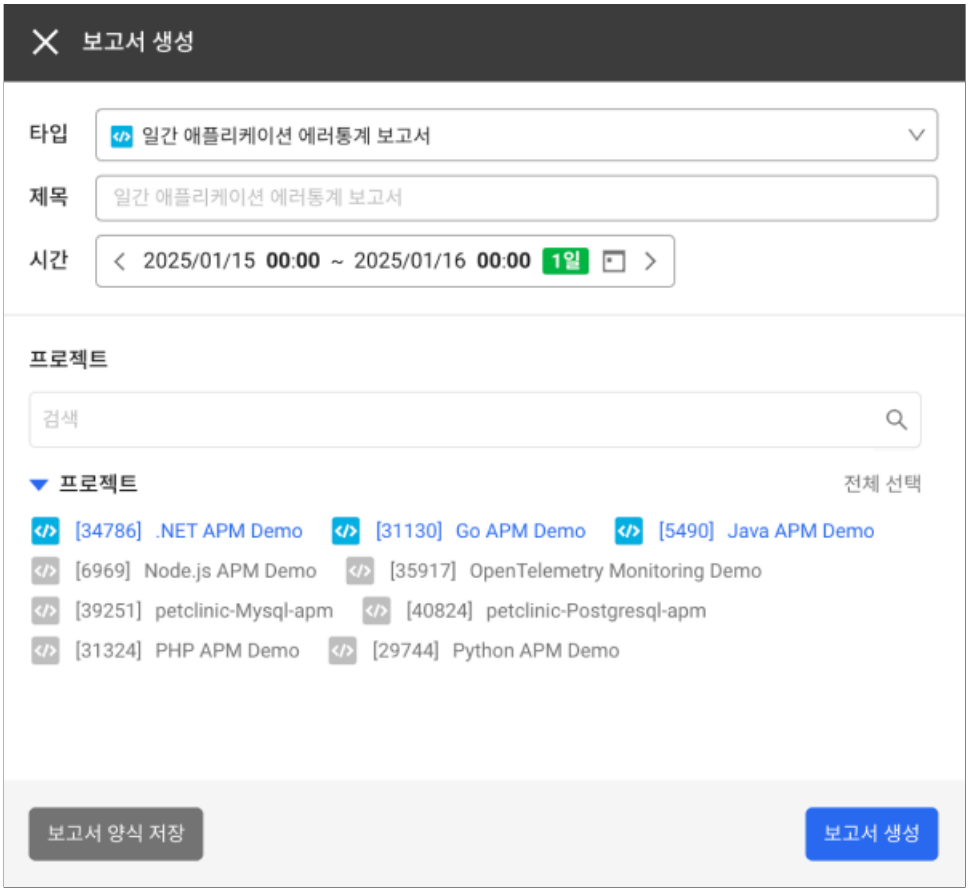
상세 설정하기

보고서 생성 창이 나타나면 각 항목을 설정하세요.

- **제목**: 같은 타입의 보고서가 여러 개인 경우 이름을 지정하면 보고서 결과를 구분하기 쉽습니다.
- **시간**: 보고서에 사용될 데이터의 기간을 선택하세요. 기간은 보고서 타입(일, 주, 월)에 따라 달라집니다.

- **프로젝트**: 보고서를 작성할 프로젝트를 선택하세요. 그룹 및 개별 프로젝트 단위로 선택할 수 있습니다. **검색** 필드를 통해 원하는 프로젝트를 검색할 수 있습니다.
- **보고서 양식 저장**: 현재 입력한 보고서의 설정을 템플릿으로 저장할 수 있습니다. 이후 새로운 보고서 생성 시 **타입** 목록에서 불러올 수 있습니다.

설정을 완료했다면 **보고서 생성** 버튼을 선택하세요.



선택한 보고서의 양식에 따라 조회 기간을 설정하는 방식은 다음과 같습니다.

- **일간 보고서**: 시작과 종료 날짜, 시간을 선택할 수 있습니다.
- **주간 보고서**: 시작 날짜만 선택하면 종료 날짜를 계산해 자동 선택합니다.
- **월간 보고서**: 시작 날짜만 선택하면 종료 날짜를 계산해 자동 선택합니다.

- 시작 날짜가 1일이면 해당 월의 월말까지 자동으로 선택합니다.
- 시작 날짜가 1일이 아니면 다음 달의 전일까지 자동으로 선택합니다.

예시 1, 시작날짜가 **1월 1일**이면, **1월 1일 00시 00분 00초**부터 **1월 31일 23시 59분 59초**를 자동 선택합니다.

예시 2, 시작날짜가 **1월 10일**이면, **1월 10일 00시 00분 00초**부터 **2월 9일 23시 59분 59초**를 자동 선택합니다.

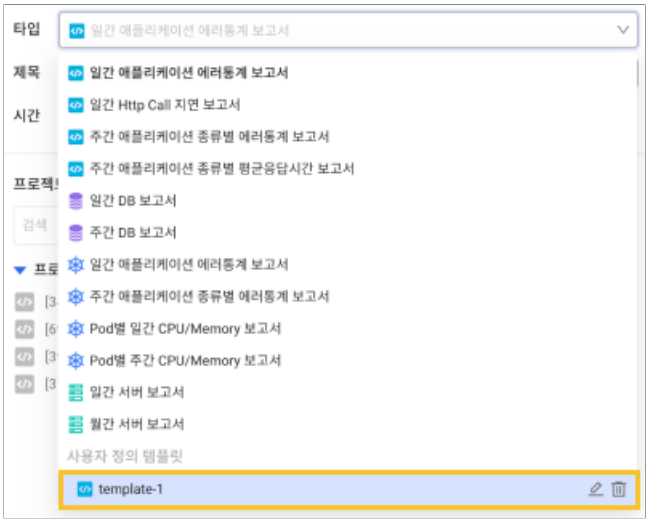
- **시간** 옵션 이용 방법에 대한 자세한 내용은 다음 문서를 참조하세요.
- 저장한 보고서 양식을 불러오는 방법에 대한 자세한 내용은 다음 문서를 참조하세요.

보고서 양식 불러오기

새로운 보고서를 생성할 때 **보고서 양식 저장** 기능을 사용해 저장한 양식은 이후 보고서 생성 시 **타입** 항목에서 확인할 수 있습니다.

업무 단위로 프로젝트를 자주 사용하는 사용자는 유용하게 활용할 수 있습니다.

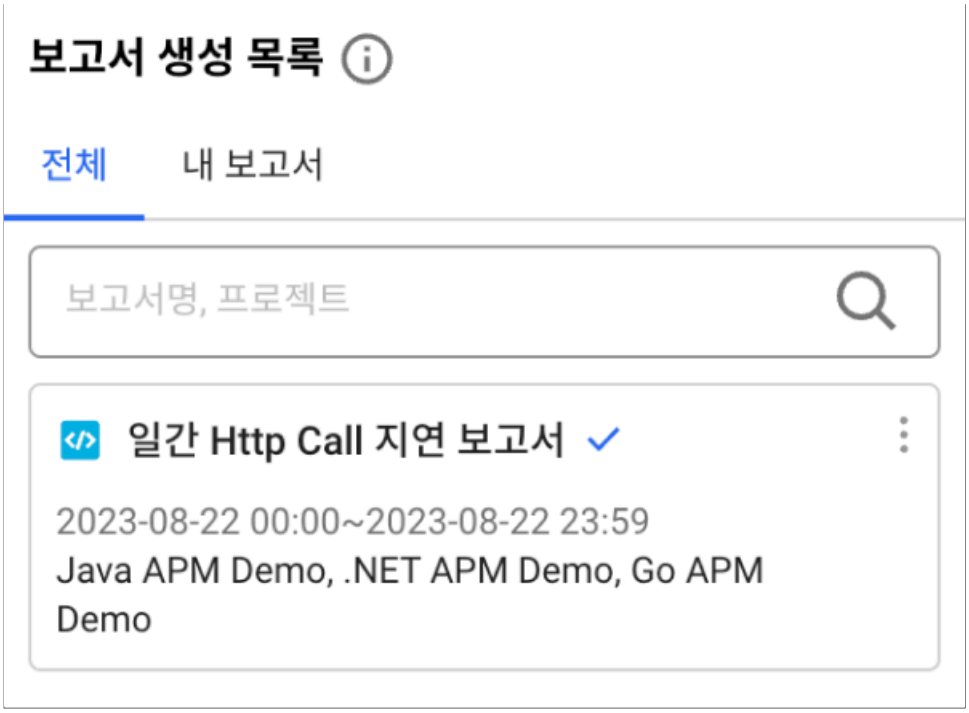
- : 저장한 보고서 양식을 수정할 수 있습니다.
- : 저장한 보고서 양식을 삭제할 수 있습니다.



보고서 목록 확인하기

보고서 생성 목록에서 이전에 만들었던 보고서와 현재 작업 중인 보고서를 확인할 수 있습니다.

- **전체** 탭에서는 사용자 본인이 생성한 보고서 외에도 같은 권한을 가진 사용자가 생성한 보고서를 함께 확인할 수 있습니다.
- **내 보고서** 탭에서는 사용자 본인이 생성한 보고서만 확인할 수 있습니다.
- 보고서 항목의 오른쪽 상단에 ⋮ 버튼을 선택하면, 해당 보고서를 복사하거나 삭제할 수 있는 기능을 이용할 수 있습니다.



- 에러 없이 생성된 보고서는 ✓ 아이콘으로 표시되며, 에러가 포함된 경우는 ⚠ 아이콘이 함께 표시됩니다. ⚠ 아이콘을 선택하면 어떤 프로젝트에서 에러가 발생했는지 확인할 수 있습니다.
- 생성한 보고서는 최대 100개까지 7일간 보관합니다.

조회 및 공유하기

- **화면 조회**

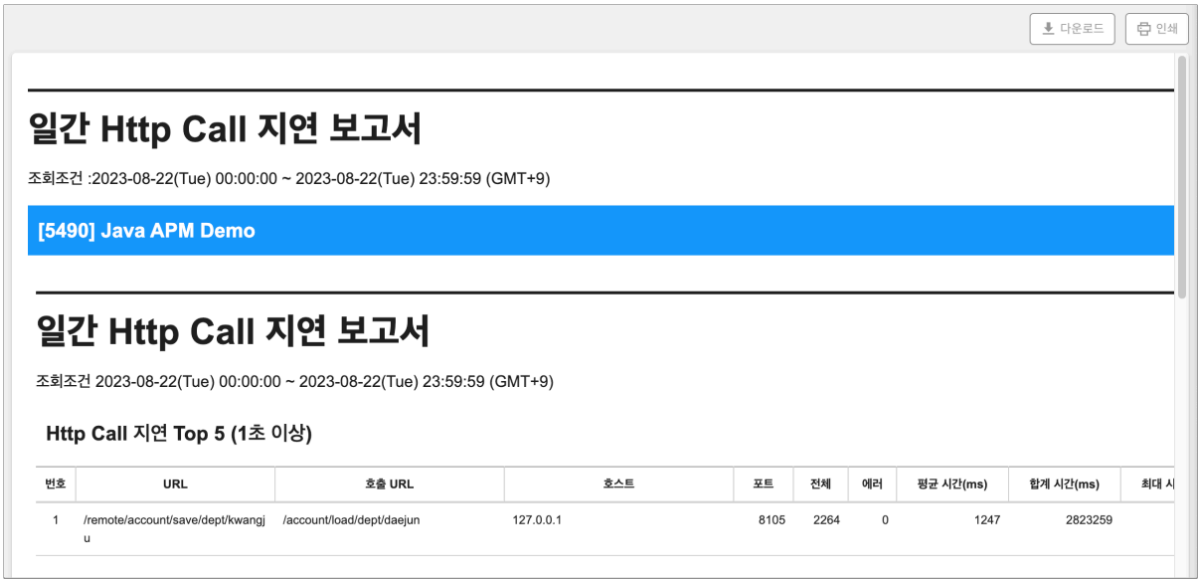
  보고서 생성 목록 목록에서 생성한 보고서를 선택하면 보고서의 상세 내용을 확인할 수 있습니다.

- **다운로드**

  현재 조회 중인 보고서를 html 형식의 파일로 다운로드하려면 **다운로드** 버튼을 선택하세요. 모니터링 관리 권한이 없는 사람에게 보고서를 공유할 수 있습니다.

- **인쇄 및 PDF 저장**

  보고서를 프린터를 통해 인쇄하거나 PDF 형식의 파일로 저장하려면 **인쇄** 버튼을 선택하세요.



Java 경고 알림 개요

모니터링 대상 시스템 또는 애플리케이션에서 문제가 발생했을 때, 미리 설정한 조건에 따라 사용자에게 알림을 전달하는 기능입니다. 알림은 이메일, SMS, 메신저, 앱 푸시 등 다양한 채널로 전송됩니다.

예를 들어, CPU 사용률, 메모리 사용량, 디스크 I/O, 네트워크 트래픽, 트랜잭션 수, 오류 발생, 응답 시간 지연 등 주요 성능 지표에 알림 기준을 설정해두면, 해당 기준을 초과하거나 이상 징후가 감지될 때 알림이 전송됩니다.

와탭의 AI 기반 알림 기능을 사용하면 복잡한 조건을 직접 설정하지 않아도 됩니다. AI는 애플리케이션의 정상적인 동작 패턴을 학습해 이상 패턴을 자동으로 감지하여 알려줍니다. 태그 카운트 알림 기능을 활용하면 특정 태그 기반으로 집계된 값에 조건을 설정하여 복잡한 상황도 정밀하게 감지할 수 있습니다.

주요 기능

- **이벤트 설정**: 임계치 기반 알림 조건과 수신 채널 설정
- **이벤트 수신 설정**: 사용자별 수신 채널, 시간대 설정
- **이벤트 기록**: 알림 발생 기록 조회

권한

이벤트를 추가하거나 설정하려면 **알림 설정 권한**이 필요합니다.

- **조회**: 메트릭스 이벤트 설정 내용을 확인하려면 **조회 분석 권한**이 필요합니다.
- **추가/수정/삭제**: 설정을 변경하려면 **알림 설정 권한**이 필요합니다.
- **알림 수신**: 알림을 수신하려면 **알림 수신 권한**이 필요합니다.

> **참고 문서**
>
> 자세한 내용은 멤버 권한을 참고하시기 바랍니다.

이벤트 설정

홈 화면 > 프로젝트 선택 > **경고 알림** > **이벤트 설정**

이벤트 설정에서 모니터링 대상 시스템 또는 애플리케이션에서 발생할 수 있는 이상 징후를 기준으로 이벤트를 설정하고 관리하는 기능을 제공합니다. 설정된 조건이 충족되면 이메일, SMS, 메신저, 앱 푸시 등 원하는 채널로 알림을 받을 수 있습니다. 모니터링 목적에 따라 이벤트를 구성할 수 있습니다.

메트릭스

Java 메트릭 기반 이벤트 조건을 설정하고 경고 알림을 발생시키는 방법을 안내합니다.

복합 메트릭스

여러 메트릭을 조합하여 이벤트 조건을 설정하는 복합 메트릭 이벤트 기능을 안내합니다.

히트맵 패턴

히트맵 기반 패턴 분석으로 이상 상황을 탐지하는 이벤트 설정을 안내합니다.

이상 탐지

Java 메트릭의 이상 패턴을 자동으로 탐지하여 경고 알림을 발생시키는 이벤트 설정을 안내합니다.

실시간 로그

Java 로그 내용을 기준으로 경고 알림을 발생시키는 실시간 로그 이벤트를 설정합니다.

복합 로그

Java 로그 내용을 기준으로 경고 알림을 발생시키는 복합 로그 이벤트를 설정합니다.

트랜잭션

Java 트랜잭션 조건을 기준으로 경고 알림을 발생시키는 이벤트 설정을 안내합니다.

참고

- 이벤트 설정 시 적정 임계값을 사용하여 불필요한 이벤트 발생을 방지하세요.
- 알림 설정 시 해소 알림(문제 해결 시 알림) 여부도 함께 고려하면 운영 대응에 도움됩니다.

공통 기능

모든 이벤트 설정을 **JSON 또는 Excel 형식으로 다운로드**할 수 있습니다. 같은 유형의 제품 설정을 공유하면 반복되는 이벤트 설정 작업을 줄일 수 있습니다.

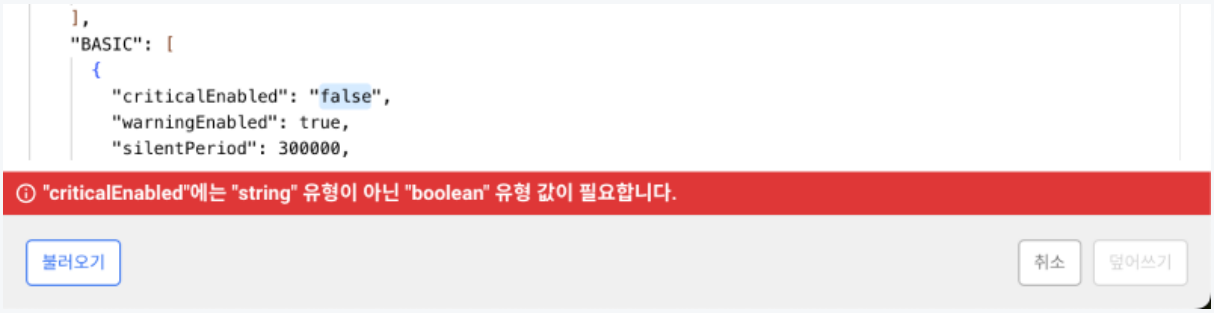
- **JSON 일괄 수정**: 모든 이벤트 조건을 JSON 형식으로 일괄 수정할 수 있습니다.
- **JSON 일괄 다운로드**: 현재 등록된 이벤트 조건을 JSON 파일로 저장할 수 있습니다.
- **Excel 일괄 다운로드**: 이벤트 목록을 Excel 파일로 내려받아 외부 보고나 백업에 활용할 수 있습니다.

JSON 일괄 수정

1. 화면 상단의 [JSON 일괄 수정] 버튼을 클릭하세요.
2. **JSON 일괄 다운로드** 기능을 통해 다운로드 받은 JSON 파일을 가져오거나 직접 수정하세요.
3. [덮어쓰기] 버튼을 클릭해 저장합니다.
4. 확인 메시지가 나타나면 [덮어쓰기] 버튼을 클릭하세요.

덮어쓰기 후 이전의 이벤트 설정으로 복구할 수 없습니다.

- `productType` 또는 `platform` 속성의 값을 임의로 수정하면 다른 프로젝트에서 이벤트가 정상 동작하지 않을 수 있습니다.
- JSON 편집창은 JSON 검증(Validation) 기능이 내장되어 있습니다. 형식이나 값이 올바르지 않은 경우, 편집창 아래에 오류 메시지가 표시되며 [덮어쓰기] 버튼은 비활성화됩니다.



JSON 데이터 구조

```
{
 "productType": "{productType}",
 "platform": "{platform}",
 "events": {
  "HITMAP": {
   ...
  },
  "LOG_REALTIME": [
   {...}
  ],
```

```
    "METRICS": [
      {...}
    ],
    "COMPOSITE_LOG": [
      {...}
    ],
    "BASIC": [
      {...}
    ],
    "COMPOSITE_METRICS": [
      {...}
    ],
    "ANOMALY": [
      {...}
    ]
  }
}
```

JSON 필드	설명
productType	제품 타입(예: APM, Infra 등)
platform	플랫폼 정보(예: java, node 등)
events	이벤트 데이터
ㄴ HITMAP	히트맵 패턴
ㄴ LOG_REALTIME	실시간 로그 이벤트
ㄴ METRICS	메트릭스
ㄴ COMPOSITE_LOG	복합 로그 이벤트
ㄴ BASIC	기본 이벤트
ㄴ COMPOSITE_METRICS	복합 메트릭스

JSON 필드	설명
ㄴ ANOMALY	이상치 탐지

JSON 일괄 다운로드

화면 상단의 [JSON 일괄 다운로드] 버튼을 클릭하세요.

- JSON 파일 이름은 integrated-event- `YYYYMMDD` .json 형식입니다.
- JSON 데이터의 구조와 `events` 속성의 하위 속성은 제품(`platform`) 또는 제품 유형(`productType`)에 따라 다를 수 있습니다.

> JSON 데이터의 속성 중 `productType` 또는 `platform` 속성의 값을 임의로 수정하면 다른 프로젝트에서 이벤트가 동작하지 않을 수 있습니다.

Excel 일괄 다운로드

화면 상단의 [Excel 일괄 다운로드] 버튼을 클릭하세요.

메트릭 이벤트 설정

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **메트릭스** 탭

프로젝트에 설정된 모든 메트릭스 기반 이벤트 목록을 확인할 수 있습니다. 사용자가 설정한 메트릭스 이벤트를 확인, 수정, 삭제하거나, 새로운 이벤트를 추가할 수 있습니다.

기본 옵션

- **이벤트 레벨**: 이벤트의 심각도를 의미하며 정상(Info), 경고(Warning), 위험(Critical) 세 단계로 구분
- **에이전트 중복 알림**: 설정한 시간(30초, 1분, 3분, 5분, 10분)에 같은 이름의 에이전트가 동시에 2개 이상 동작하는 경우, 선택한 시간 동안 이벤트 발생 안함
- **검색**: 이벤트 이름을 기준으로 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **+ 이벤트 추가**: 새로운 메트릭스 이벤트 추가
- **컬럼 크기 조정**: 각 컬럼의 경계를 드래그하여 너비 조정 가능

표 | 메트릭스 이벤트 목록 구성

항목	설명
No.	이벤트의 순번
	이벤트의 활성화 여부 – 활성화: 조건을 충족하면 이벤트 발생 – 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음
	이벤트 설정 편집 및 삭제
이벤트 이름	사용자가 지정한 이벤트명 – 최대 255자까지 입력 가능
규칙	이벤트 발생 조건

항목	설명
대상	이벤트가 적용되는 모니터링 대상
발생 횟수	이벤트 발생 기준이 되는 기간과 횟수 – 선택한 시간 동안 설정한 횟수만큼 조건을 충족하면 이벤트 발생
일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 – 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 – 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가 발생해도 알림이 발생하지 않음
해소된 알림	Critical과 Warning 레벨의 이벤트가 해소되면 RECOVERED(정상) 상태의 알림 수신 여부 – 활성화: 진행 중 → 정상 상태로 전환되면 RECOVERED 알림을 보내고, 이벤트 기록에서 진행 중/해소 상태를 구분해 볼 수 있음 – 비활성화: 임곗값을 초과할 때마다 단발성 이벤트 발생
이벤트 수신	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)

이벤트 설정

메트릭스 이벤트는 모니터링 대상에서 발생하는 이벤트를 조건에 따라 정의하고, 효율적으로 알림을 받을 수 있도록 설정할 수 있는 기능을 제공합니다.

이벤트 추가

메트릭스 이벤트를 추가하려면 **이벤트 설정** > **메트릭스** 탭으로 이동한 후, 오른쪽 상단의 [+ 이벤트 추가] 버튼을 클릭하세요.

메트릭스 이벤트 추가 방법
Step 1. 템플릿 선택: 이벤트의 기본 구조를 선택합니다.
Step 2. 이벤트 조건 정의: 이벤트를 어떤 기준으로 발생시킬지 조건을 설정합니다.

메트릭스 이벤트 추가 방법
Step 3. 이벤트 대상 선택: 이벤트가 발생하는 모니터링 대상을 선택합니다.
Step 4. 기본 정보 및 수신 설정: 이벤트 이름 및 메시지, 수신 대상을 설정합니다.

템플릿 선택

1. 이벤트를 어떻게 구성할지에 따라 화면 왼쪽의 3가지 템플릿 중 하나를 선택하세요.
    - **새로 만들기**: 카테고리와 지표를 처음부터 자유롭게 구성할 수 있으며, 복잡한 조건이나 세밀한 모니터링 규칙이 필요한 고급 사용자에게 적합
    - **빠른 설정**: 사전 정의된 카테고리, 지표, 기본 임계치가 제공되며, 임곗값만 조정하여 빠르게 적용할 수 있어 초급 사용자에게 적합
    - **고급 설정**: 사전 정의된 카테고리와 지표를 기반으로 조건을 추가하거나 자유롭게 변경

> ⓘ 메트릭스 이벤트에서는 다양한 설정 방식을 지원합니다. 모니터링 플랫폼에 따라 제공하는 템플릿은 다를 수 있습니다.

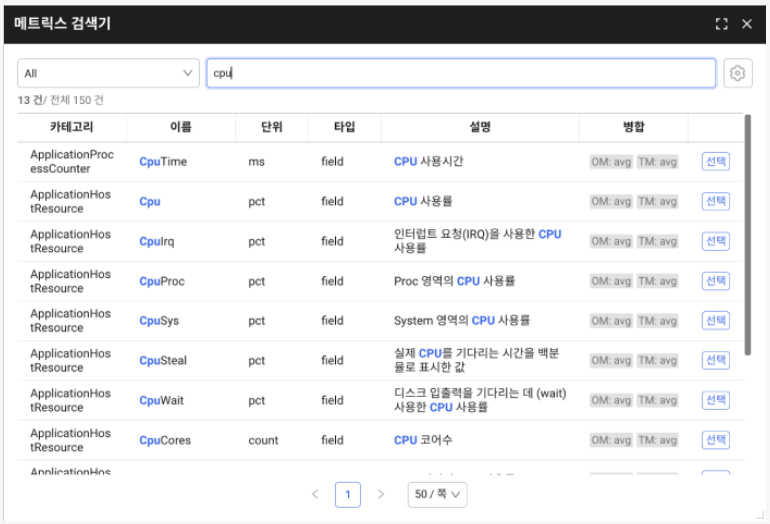
이벤트 조건 정의

이벤트 조건 정의 단계에서 이벤트가 언제 발생할지 결정하는 지표 설정, 발생 횟수, 일시 중지, 해소된 알림, 이벤트 동작 시간을 설정합니다.

지표 설정

1. 검색창에서 카테고리를 입력하거나, 카테고리 목록에서 카테고리(지표)를 선택하세요.
   카테고리를 선택해야 이후 지표 설정을 진행할 수 있습니다.
    - 카테고리 목록은 카테고리 이름, 데이터 수집 주기, 키를 확인할 수 있습니다. 최근 3시간 동안 프로젝트에서 수집된 메트릭스 데이터가 표시됩니다.
2. 목록에 없는 지표를 사용하려면 **직접 입력하기**를 선택하여 카테고리를 직접 입력하세요.

> **카테고리와 지표를 모르는 경우 메트릭스 검색기를 활용하세요.**
>
> 처음 지표를 설정하는 경우 메트릭스 검색기를 활용하면 더 빠르게 지표를 찾을 수 있습니다.
>
> - 메트릭스 검색기에서 원하는 지표를 검색하면 관련 카테고리와 상세 정보를 확인할 수 있습니다.
> - 검색 결과에서 필요한 메트릭스의 [선택] 버튼을 클릭하면, 선택한 값이 이벤트 설정 화면에 자동 반영됩니다.



3. 이벤트 레벨(Critical, Warning, Info)을 선택하세요. 하나의 지표에 여러 레벨을 설정할 수 있습니다.

> **이벤트 레벨 동작**
>
> 여러 조건을 만족하더라도 가장 높은 우선 순위(Critical → Warning → Info)의 레벨만 이벤트가 발생합니다. 레벨 동작 방식은 레벨 동작 가이드를 참고하세요.
>
> - 이벤트 레벨 수준은 위험(Critical)과 경고(Warning), 정상(Info)으로 구분합니다.
> - 같은 지표로 레벨을 설정하고 단계별 임계치를 설정하는 것을 권장합니다.

4. 선택한 이벤트 레벨의 필드, 연산자, 값을 입력하세요.

   a. **+ 추가**를 클릭해 조건을 추가할 수 있습니다.

   - `&&` : **모든 조건을 동시에 만족**해야 이벤트 발생

- `||`: **하나 이상의 조건만 만족**하면 이벤트 발생

b. — 를 클릭하면 해당 조건이 삭제됩니다.

> ⓘ 조건식 작성 방법과 지원되는 연산자는 조건 설정 가이드를 참고하세요.

발생 횟수

조건이 몇 번 충족되었을 때 이벤트를 발생시킬지 설정합니다. 지속적인 문제만 감지하고 싶을 때 유용합니다.

1. 발생 빈도를 선택하세요.
   - **연속**: 설정한 횟수만큼 이벤트가 연속으로 발생하는 경우 이벤트 발생
   - **최근**: 선택한 시간 내에 설정한 횟수만큼 이벤트가 발생하는 경우 이벤트 발생
     - 선택 가능 시간: `5초`, `10초`, `15초`, `30초`, `1분`, `2분`, `3분`, `4분`, `5분`, `10분`, `30분`, `1시간`
2. 발생 횟수를 입력하세요.
   - **Interval** 값은 선택한 카테고리의 데이터 수집 간격입니다. (Interval: 10s)

일시 중지

과도한 이벤트 발생을 방지하기 위해, 이벤트 동작을 일시 중지할 시간을 선택하세요. 이벤트 발생 후 설정한 시간 동안 동일한 이벤트가 발생하지 않습니다.

- 선택 가능 시간: `사용 안 함`, `5분`, `10분`, `15분`, `20분`, `30분`, `1시간`, `2시간`, `3시간`, `6시간`, `12시간`, `1일`

> ⓘ **해소된 알림 활성화 시**
>
> 알림은 정상 상태 알림(**RECOVERED**)을 받은 시점을 기준으로 설정한 시간이 지나기 전까지 동일한 이벤트가 발생해도 알림이 발생하지 않습니다.

해소된 알림

해소된 알림 기능을 활성화하면, 이벤트 발생 이후 상태 변화(진행 중 → 해소됨)를 추적할 수 있습니다. **이벤트 기록** 메뉴에서 진행

중인 이벤트로 표시됩니다.

- 해소된 알림은 Critical과 Warning 레벨에만 적용되며, Info 레벨은 항상 단발성 이벤트로 동작합니다.
- Critical과 Warning 레벨의 이벤트가 해소되면, 정상(RECOVERED) 상태의 알림이 전송됩니다.
- 토글 버튼을 클릭해 **활성화** 또는 **비활성화**하세요.
  - **활성화**: 이벤트는 상태 기반으로 동작하며, 임계치 초과 시 진행 중, 임계치 이하로 복구 시 해소 상태로 전환됨
  - **비활성화**: 임곗값을 초과할 때마다 단발성 이벤트 발생

이벤트 동작 시간

이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. 이벤트 동작 시간에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 삭제됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

시뮬레이션

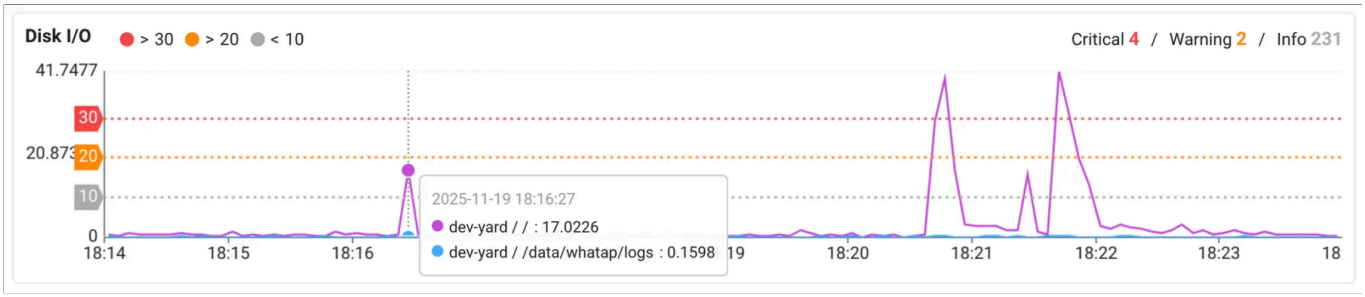
시뮬레이션 기능은 설정한 이벤트 조건을 과거 데이터에 기반으로 사전 테스트 할 수 있습니다. 예상 이벤트 수가 지나치게 많거나 부족하다면 조건을 조정해 최적의 임곗값을 찾을 수 있습니다.

- **규칙 검증**: 새로운 이벤트 규칙을 만들었을 때, 과거 데이터에 적용하면 어떤 결과가 나오는지 확인 가능
- **임곗값 튜닝**: 이벤트가 자주 발생하거나 전혀 발생하지 않을 때 적절한 수준으로 임곗값 조정

시뮬레이션 시작하기

1. 이벤트 조건 설정이 완료되면 [시뮬레이션] 버튼을 클릭하세요.
2. 예상 이벤트 건수와 시각화된 결과를 확인합니다.

시뮬레이션 결과 이해하기



1. **시계열 데이터**

   지표의 시간대별 변화와 각 레벨의 임계선이 그래프로 표시되어 이벤트 발생 시점을 직관적으로 확인할 수 있습니다.

2. **레벨별 발생 현황**

   화면 우측에 Critical, Warning, Info 레벨의 이벤트 발생 횟수가 표시됩니다. 전체 이벤트 분포와 빈도를 파악할 수 있습니다.

3. **대상별 상세 정보**

   차트의 특정 지점에 마우스를 올리면 해당 시점의 정확한 지표 값과 모니터링 대상이 표시됩니다. 여러 대상이 있는 경우 색상으로 구분되며, 대상별 패턴이 다른 경우 필터링 기능을 사용해 개별적으로 분석할 수 있습니다.

4. **피크 구간 식별**

   그래프에서 지표가 급격히 상승하는 시간을 확인하여 문제가 자주 발생하는 시간대나 패턴을 파악할 수 있습니다.

시뮬레이션 활용 팁

- **다양한 시간대 시뮬레이션**

  평일/주말, 피크/비피크 시간대를 각각 시뮬레이션하여 시간대별 최적 임곗값을 설정하세요.

- **실제 장애 기간 포함**

  과거 장애가 발생했던 기간을 포함하여 시뮬레이션을 실행하면 현재 규칙이 장애를 정확히 감지하는지 확인하세요.

- **대상 필터링 추가**

모니터링 대상이 많으면 시뮬레이션 결과가 복잡해집니다. 필터링 기능으로 특정 대상만 선택하여 명확한 결과를 확인할 수 있습니다.

- **임곗값 단계적 조정**

  임곗값을 한 번에 크게 변경하기보다 10% 단위로 점진적으로 조정하면서 최적값을 탐색하는 것이 좋습니다.

이벤트 대상 선택

이벤트 대상을 지정하지 않으면, 프로젝트에 포함된 전체 대상으로 이벤트가 발생되어 많은 이벤트가 발생할 수 있습니다. 따라서 이벤트가 적용될 대상을 명확히 설정하는 것을 권장합니다.

이벤트 대상을 변경한 경우, 이벤트 건수가 달라질 수 있습니다. **시뮬레이션** 버튼을 클릭한 후, 다시 실행해 예상 이벤트 수를 확인하세요.

대상 선택

이벤트가 적용될 대상을 **선택 입력** 또는 **직접 입력** 방식으로 지정할 수 있습니다.

- 선택 입력 **직접 입력**

1. 태그, 연산자, 값을 선택 또는 입력하세요. a. 태그를 찾기 어렵다면, **메트릭스 검색기**에서 태그를 검색할 수 있습니다.
2. **+ 추가**를 클릭해 조건(`&&` , `||`)으로 이벤트 대상을 추가하세요.

모니터링할 대상의 태그, 연산자, 값을 직접 입력하세요.

- 직접 입력을 선택하면 입력란 아래에 사용 가능한 태그 목록(**태그 선택 입력**)이 함께 제공됩니다.

```
ex. endsWith(okindName, 'example_name') && container == 'prod.billing'
ex. ${4xxErrorType} == '401'
```

입력 방법을 변경하면, 작성한 내용이 초기화됩니다.

대상 선택 연산자 목록

선택 입력 방식에서 문자열 태그를 선택하면 연산자 목록에 다음 두 항목이 함께 표시됩니다. 목록에 나타나는 이름은 함수 표기 그대로이며, `startsWith(key, 'match_text')` 와 같은 기존 항목과 형태가 같습니다.

- `equalsIgnoreCase(key, 'match_text')`
- `notEqualsIgnoreCase(key, 'match_text')`

두 연산자는 대소문자를 구분하지 않고 값을 비교합니다. **직접 입력** 방식에서는 같은 함수를 조건식에 직접 작성해 사용할 수 있습니다. 조건식 문법과 지원되는 함수 전체 목록은 조건 설정 가이드를 참고하세요.

> ⓘ 이벤트 발생 조건과 이벤트 대상 선택은 같은 문법을 사용하므로 두 함수를 양쪽 조건식에 모두 작성할 수 있습니다.

기본 정보 및 수신 설정

이벤트의 이름, 메시지, 수신 대상을 설정합니다. 설정한 내용은 이벤트 기록과 실제 알림 메시지에 그대로 표시되므로, 식별하기 쉽고 명확하게 작성하는 것이 중요합니다.

이벤트 활성화

설정한 조건을 충족했을 때 이벤트 발생 여부를 설정합니다.

- **활성화**: 설정한 조건을 충족하면 이벤트가 발생됨
- **비활성화**: 조건을 검사하지 않음

이벤트 이름

이벤트 제목으로 사용될 이벤트 이름을 입력하세요. 설정한 이벤트 이름은 **이벤트 목록** 및 **이벤트 기록** 메뉴에 표시되며, 해당 이름으로 검색할 수 있습니다.

메시지

이벤트 메시지를 입력하세요. 입력한 메시지는 이벤트 기록 및 알림에 사용됩니다. 을 클릭하면 이전에 작성한 메시지 기록을 확인할 수 있습니다. 메시지에 변수를 사용하여 이벤트 발생 시점의 실제 값을 포함할 수 있습니다.

- **변수 사용 규칙**
  - 변수 형식: `${지표명}`

- 설정한 **카테고리와 동일한 카테고리** 내 지표만 변수로 사용할 수 있습니다.
- 여러 변수를 조합해 상세한 메시지를 구성할 수 있습니다.
- 사용 가능한 변수 목록은 **메트릭스 조회** 메뉴에서 확인할 수 있습니다.

- **변수 활용 예시**

	이벤트 설정	이벤트 발생 시
제목	Disk 사용량 증가	Disk 사용량 증가
메시지	`Disk = ${disk}%`	`Disk = 89.9%`

이벤트 수신

이벤트 알림을 받을 멤버를 지정합니다.

- **전체 수신** **태그 선택 수신**

프로젝트에서 이벤트 수신 설정이 활성화된 모든 대상(멤버, 채널)에게 알림을 전송합니다.

해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다.

1. **태그 선택 수신**을 선택하세요.
2. **수신 태그** 항목에서 [+ 태그 추가] 또는 [+] 버튼을 클릭하세요.
3. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
4. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
5. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
6. **이벤트 수신 태그** 창에서 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> **수신 태그**
>
> 수신자 태그는 알림을 받을 사용자 또는 채널을 **그룹 단위로 관리하기 위한 기능**입니다. 이벤트에 수신자 태그를 설정하면, 해당 태그가 포함된 사용자에게만 알림이 전달됩니다. 수신 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 설정**의 **메트릭스** 탭으로 이동하세요.
2. 이벤트 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 규칙 수정** 창에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 규칙 수정** 창의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules- `YYYYMMDD` .json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.
3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

> ⓘ 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.
>
> Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)

JSON 데이터 구조


```
{
  "version": 2,
  "eventId": "zcef7s7f27rum1",
  "enabled": true,
  "stateful": false,
  "title": "Active Transaction",
  "message": "Active Transaction = ${active_tx_count}",
  "category": "app_counter",
  "alertLabel": [
    "oid"
  ],
  "repeatCount": 1,
  "repeatDuration": 0,
  "silent": 300000,
  "receiver": [],
  "timeTag": [],
  "selectString": "",
  "conditions": [
    {
      "level": 20,
      "enabled": true,
      "rule": "active_tx_count > 100"
```

```
    }
  ],
  "createTime": 1763632674345,
  "lastModifiedTime": 1763632674347,
  "lastModifiedUser": "support@whatap.io",
  "basic": true,
  "metaId": "java004",
  "selectCondition": {
        "oid": [
          "-1010758404",
          "-250906941"
        ]
  }
}
```


표 | JSON 데이터 구조

JSON 필드	타입	설명	비고
version	Integer	이벤트 규칙의 버전	변경 제한
eventId	String	이벤트 규칙의 고유 식별자	
enabled	boolean	이벤트 활성화 여부	
stateful	boolean	상태 기반 이벤트 여부	
title	String	사용자가 설정한 이벤트 규칙 이름	
message	String	사용자가 설정한 이벤트 규칙 메시지	
category	String	데이터 카테고리	
alertLabel	`List<String>`	이벤트의 상태 관리를 위해 카테고리마다 지정되는 기본 식별자(primary key) 값	변경 제한
repeatCount	Integer	반복 횟수	
repeatDuration	Long	반복 기간	

JSON 필드	타입	설명	비고
silent	Integer	일시 중지 시간	
receiver	List<String>	수신자 태그의 키값 목록	
timeTag	List<String>	동작시간 태그의 키값 목록	
selectString	String	대상 선택	
conditions	List<Object>	레벨별 발생 조건 목록	
ㄴ level	byte	레벨 값	Critical : 30 Warning : 20 Info : 10
ㄴ enabled	boolean	레벨 활성화 여부	
ㄴ rule	String	레벨 발생 조건	
createTime	Long	이벤트 규칙 최초 생성 시간	
lastModifiedTime	Long	이벤트 규칙 마지막 수정 시간	
lastModifiedUser	String	이벤트 규칙 마지막 수정 계정	
basic	boolean	빠른 설정으로 생성한 이벤트 여부	변경 제한
metaId	String	빠른 설정 템플릿 고유 식별자	변경 제한: 이 값은 빠른 설정으로 생성한 이벤트에만 존재합니다. (basic=true)
selectCondition	Object	빠른 설정 대상 선택값	이 값은 빠른 설정으로 생성한 이벤트에만 존재합니다. (basic=true)

'변경 제한'으로 표시된 필드는 이벤트 동작에 영향을 줄 수 있으므로 값을 변경하지 않는 것을 권장합니다.

부록

발생 조건, 대상 선택 가이드

메트릭스 경고 알림의 이벤트 발생 조건과 이벤트 대상 선택은 동일한 문법을 사용합니다. 단, 이벤트 발생 조건은 필드(Field)의 Key를 변수로 사용하고, 이벤트 대상 선택은 태그(Tag)의 Key를 변수로 사용합니다.

레벨 동작 가이드

이벤트 레벨은 위험(Critical)과 경고(Warning), 정상(Info)으로 구분합니다.

1. **우선순위 기반 이벤트 발생**

   여러 레벨의 조건을 동시에 충족할 경우, 가장 높은 우선순위의 이벤트만 발생합니다.

   예시

   ```
   설정:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   현재 상태: CPU 95%
   → 결과: Critical 이벤트만 발생 (Warning은 억제됨)
   ```

2. **레벨 상승**

   진행중 상태를 갖는 이벤트에서 낮은 레벨 이벤트가 진행 중일 때 높은 레벨 조건을 충족하면, 두 레벨 모두 진행 중 상태가 됩니다.

   예시

   ```
   설정:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   시나리오:
   1) CPU 80% → Warning 발생 (진행 중)
   ```

```
2) CPU 95% → Warning (진행 중), Critical (진행 중)
```

3. **레벨 하강**

   진행중 상태를 갖는 이벤트에서 높은 레벨에서 낮은 레벨로 전환되면 높은 레벨은 해소되고 낮은 레벨이 유지됩니다.

   예시

```
설정:
- Warning: CPU > 70%
- Critical: CPU > 90%

시나리오:
1) CPU 97% → Critical 발생
2) CPU 85% → Critical 해소, Warning 유지
3) CPU 65% → Warning 해소
```

4. **동작 흐름도**

```
- Warning: CPU > 70%
- Critical: CPU > 90%

메트릭 값: 60%  →  75%  →  92%  →  85%  →  60%
상태:      정상    Warning   Critical  Warning   정상
                   (발생)    (발생)    (유지)    (해소)
                             Warning
                             (유지)
```

> 진행 중 상태를 갖는 이벤트는 조건이 충족되는 동안 계속 활성화 상태를 유지하며, 조건이 해제될 때까지 지속됩니다.

조건 설정 가이드

> 지표명이 숫자로 시작하거나 특수문자가 포함된 경우 `${지표명}` 형태로 작성해야 합니다.
>
> - 괄호 `()`, `[]`

- 연산자 `+`, `-`, `*`, `/`, `%`
- 구분자 `:`, `@`, `#`, `,`, (공백)
- 기타 특수문자 `!`, `^`, `&`, `|`, `~`, ```, `=`

`${}` 사용 예시:

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` 없이 사용 가능: 영문자, 언더스코어(`_`), 점(`.`)만 포함된 경우

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. 비교 연산자

숫자 비교

```
크다:          cpu > 80
크거나 같다:    cpu >= 80
작다:          memory < 1000
작거나 같다:    memory <= 1000
같다:          status == 200
같지 않다:      error != 0
```

문자열 비교

```
status == 'OK'
region == "us-east-1"
```

2. 산술 연산자

- 기본 산술 연산

```
덧셈:   cpu + 10 >= 90
뺄셈:   memory - 100 >= 500
곱셈:   cpu * 2 >= 100
나눗셈:  disk / 1024 >= 100
나머지:  value % 10 == 0
```

- 괄호를 이용한 우선순위 제어

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 음수 표현

```
cpu > -100
```

3. 논리 연산자

- AND 연산 (&&)

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 연산 (||)

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 복합 논리 연산

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. 패턴 매칭 연산자

- LIKE 연산자

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 연산자

```
oname not like 'test-*'
url not like '*debug*'
```

5. 내장 함수

- Null 체크 함수

```
isNull(value)           # null인지 확인
isNotNull(value)        # null이 아닌지 확인
nvl(value, 0)           # null이면 기본값 반환
isEmpty(str)            # 빈 문자열인지 확인
isNotEmpty(str)         # 빈 문자열이 아닌지 확인
```

- 집계 함수

```
sum(cpu, memory, disk)          # 합계
avg(cpu, memory)                # 평균
max(cpu, memory, disk)          # 최대값
min(cpu, memory, disk)          # 최소값
count(value1, value2, value3)   # 개수
```

- 수학 함수

```
round(cpu, 2)           # 반올림 (소수점 2자리)
```

- 문자열 함수

```
length(str)             # 문자열 길이
startsWith(str, 'prefix')  # 접두사 확인
```

```
endsWith(str, 'suffix')    # 접미사 확인
indexOf(str, 'search')     # 문자열 위치
substring(str, 0, 10)      # 부분 문자열
trim(str)                  # 공백 제거
replace(str, 'old', 'new') # 문자열 치환
hasStr(str, 'search')      # 문자열 포함 여부
```

문자열 함수

```
equalsIgnoreCase(str, 'match_text')     # 대소문자를 구분하지 않는 일치 비교
notEqualsIgnoreCase(str, 'match_text')  # 대소문자를 구분하지 않는 불일치 비교
```

`==` 와 `!=` 는 대소문자를 구분하므로 `WebServer` 와 `webserver` 를 다른 값으로 판단합니다. 표기가 일정하지 않은 태그나 필드를 비교할 때 위 두 함수를 사용하면 대소문자 차이를 무시하고 값을 맞출 수 있습니다.

예시

```
equalsIgnoreCase(okindName, 'WebServer')
notEqualsIgnoreCase(container, 'PROD.billing')
```

이벤트 발생 조건과 이벤트 대상 선택은 같은 문법을 사용하므로 두 함수를 양쪽 조건식에 모두 작성할 수 있습니다.

- 조건 함수
  - **if:** 조건식이 참(`true`)이면 `true값` 을 반환하고, 거짓(`false`)이면 `false값` 을 반환하는 조건 함수

    형식

    ```
    if(조건, true값, false값)
    ```

    예시

    ```
    if(value >= 90, 'High', 'Medium') == 'High'
    ```

  - **decode**: 값과 조건을 비교하여 일치하는 조건의 결과를 반환, 없으면 기본값 반환

    형식

```
decode(값, 조건1, 결과1, 조건2, 결과2, ..., 조건N, 결과N, 기본값)
```

예시

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

- **in**: 값이 지정한 후보값 중 하나와 일치하면 `true`, 아니면 `false`

형식

```
in(값, 후보값1, 후보값2, ..., 후보값N)
```

예시

```
in(status, 200, 201, 204) == false
```

작성 시 주의사항

- 권장사항
  - 간단한 변수명은 `${}` 없이 사용 (예: cpu, memory)
  - 특수문자 포함 시 반드시 `${}` 사용
  - 복잡한 연산은 괄호로 명확하게 구분
  - 함수명은 정확히 입력 (대소문자 구분)
- 피해야 할 사항
  - 불완전한 표현식 작성 (예: `cpu >`, `memory &&`)
  - 괄호 미스매칭 (예: `(cpu > 80 && cpu > 70)`)
  - 연속된 연산자 사용 (예: `cpu >> 80`, `cpu >< 80`)

복합 메트릭 이벤트 설정

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **복합 메트릭스** 탭

복잡한 규칙을 활용해 이벤트를 생성하고 경고 알림을 보낼 수 있습니다.

기본 옵션

- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **알림 메시지 설정**: 수신할 메시지의 상세 항목 선택 및 항목 별 내용 편집
- **+이벤트 추가**: 새로운 복합 메트릭스 이벤트 추가

> **권한**
>
> **복합 메트릭스** 이벤트를 설정하려면 **알림 설정** 권한이 있어야 합니다. 알림 설정 권한은 **홈 > 프로젝트 선택 > 관리 > 통합 멤버 관리**에서 권한을 부여할 멤버의 아이콘을 클릭해 설정할 수 있습니다.

표 | 복합 메트릭스 이벤트 목록 구성

항목	설명
이벤트 제목	사용자가 지정한 이벤트명 - 최대 255byte까지 입력 가능
조건	이벤트 발생 조건
이벤트 상태가 해소되면 추가 알림	Critical과 Warning 레벨의 이벤트가 해소되면 RECOVERED(정상) 상태의 알림 수신 여부 – 활성화: 진행 중 → 정상 상태로 전환되면 RECOVERED 알림을 보내고, 이벤트 기록에서 진행 중/해소 상태를 구분해 볼 수 있음 – 비활성화: 임곗값을 초과할 때마다 단발성 이벤트 발생
이벤트 수신 태그	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)

항목	설명
	이벤트 수정 및 삭제
	이벤트의 활성화 여부 – 활성화: 조건을 충족하면 이벤트 발생 – 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음

이벤트 설정

복합 메트릭스 이벤트는 메트릭스 데이터 질의 언어인 **MXQL**을 기반으로 이벤트 조건을 생성합니다. **차트로 생성하기** 기능은 **MXQL**의 자동완성을 위한 콤보박스 기능을 제공합니다.

이벤트 추가

1. **경고 알림** > **이벤트 설정**에서 **복합 메트릭스** 탭을 클릭하세요.
2. 화면 오른쪽의 [+ 이벤트 추가] 버튼을 클릭하세요.
3. **복합 메트릭스** 창에서 차트로 생성하기를 클릭하세요.

   a. 제공하는 템플릿을 활용할 수 있습니다.
4. **이벤트 설정** 화면에서 먼저 이벤트 데이터를 조회하고, 이벤트 조건을 설정하세요.

이벤트 데이터 조회

이벤트 데이터를 조회하여 차트를 구성합니다. **이벤트 설정** 화면의 오른쪽 **이벤트 데이터 조회**의 **위젯** 또는 **텍스트** 옵션을 선택해 이벤트를 설정하세요.

- **위젯** **텍스트**

시계열 차트를 구성하는 옵션을 통해 이벤트 설정 시 사용할 **MXQL**을 자동 완성할 수 있습니다.

1. 데이터를 조회할 날짜를 선택하세요.
2. 데이터를 조회할 **카테고리**(대상)를 선택하세요.
3. **필터** 항목의 **+ 필터 추가**를 클릭해 이벤트 조건을 생성하세요.

a. 필터 창에서 연산식, 태그, 필터 값을 선택하세요.
b. + **추가**를 클릭해 조건을 추가할 수 있습니다.
c. 조건을 삭제하고 싶다면 조건 옆의 – 를 클릭하세요.

4. **그룹화** 항목에서 그룹화된 메트릭스 데이터를 선택하세요. 다중 선택할 수 있습니다.
5. **타임 유닛** 항목에서 `초`, `분`, `시간` 단위로 그룹화된 데이터를 나눌 시간 기준을 설정하세요.
6. **필드** 항목에서 이벤트 발행 조건에 사용할 필드를 선택하세요. 다중 선택할 수 있습니다.

> ⚠ 텍스트 모드로 입력한 mxql은 위젯 모드와 호환되지 않습니다.

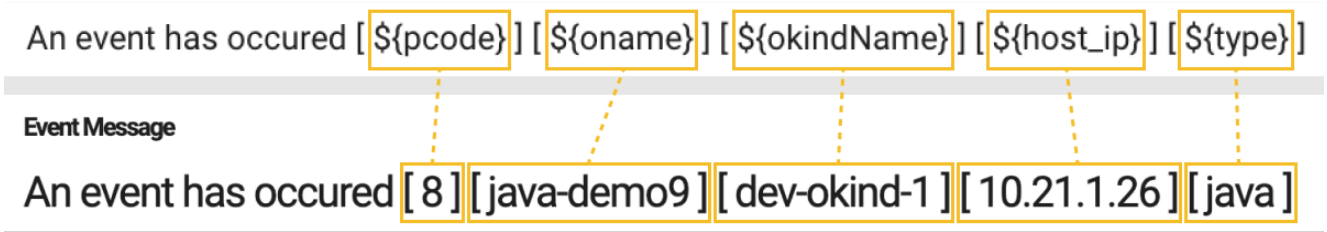
MXQL을 그대로 수정할 수 있는 편집창이 나타납니다.

1. **MXQL**를 직접 수정하고 **조회** 버튼을 클릭하세요.
2. 날짜를 선택하면 해당 내역을 차트 또는 그래프로 확인할 수 있습니다.

알림

경고 알림 설정의 기본 정보를 입력합니다.

- **이벤트 활성화**: 토글 버튼을 활성화하면 이벤트가 동작합니다.
- **레벨**: 이벤트 레벨(Critical, Warning, Info)을 선택하세요.
  - **이벤트 상태가 해소되면 추가 알림**: 토글 버튼을 활성화하면 이벤트가 해소되어 정상(RECOVERED) 상태의 알림이 전송됩니다.
- **제목**: 이벤트 규칙의 제목을 입력하세요.
- **메시지**: 이벤트 발생 시 전달받을 알림 메시지의 내용을 입력하세요. 메시지에 변수를 사용하여 이벤트 발생 시점의 실제 값을 포함할 수 있습니다.



변수 사용 규칙

- 변수 형식: `${지표명}`
- 설정한 **카테고리와 동일한 카테고리** 내 지표만 변수로 사용 가능합니다.
- 여러 변수를 조합해 상세한 메시지를 구성할 수 있습니다.
- 사용 가능한 변수 목록은 **메트릭스 조회**에서 확인할 수 있습니다.

이벤트 발행 조건

- **데이터 조회 범위:** 실시간 데이터 조회 범위를 선택하세요.
  - 조회 가능 범위: `30초` , `1분` , `2분` , `5분` , `10분` , `30분` , `1시간` , `2시간` , `3시간`
- **조건:** 이벤트 발생 조건을 입력하세요. 이벤트 데이터 조회에 포함된 필드만 사용할 수 있습니다.

부가정보

- **인터벌**: 선택한 시간(`1분` , `5분` , `10분` , `30분` , `1시간` , `12시간`) 간격으로 알림 조건을 확인합니다.
- **무음**: 이벤트가 발생한 후, 선택한 시간(`unset` , `1분` , `5분` , `10분` , `30분` , `1시간` , `12시간`) 동안 같은 이벤트가 발생되지 않습니다.
- **이벤트 동작 시간**: 이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.
  1. **+ 추가**를 클릭하세요.
  2. **이벤트 동작 시간**에서 **+ 새로운 태그 생성**을 클릭하세요.
  3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
  4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
  5. 수정 또는 삭제할 태그의 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 제거됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

- **이벤트 수신 태그**: 이벤트에 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신 태그를

설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

1. **+ 태그 추가** 또는 +를 클릭하세요.
2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
3. **태그 생성** 창에서 태그 이름을 입력하고 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
4. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
5. **이벤트 수신 태그** 창에서 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> 이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. **경고 알림** > **이벤트 수신 설정**에서 프로젝트 멤버와 3rd-party 플러그인에 각각 태그를 지정할 수 있습니다.

이벤트 규칙 테스트

설정한 조건을 과거 데이터에 기반으로 테스트를 진행해, 몇 건의 이벤트가 발생할지 예측할 수 있습니다.

1. 이벤트 규칙을 테스트할 날짜를 선택하고 [확인] 버튼을 클릭하세요.
2. [실행] 버튼을 클릭해 이벤트 발생 건수를 확인하세요.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 수신 설정**의 **복합 메트릭스** 탭으로 이동하세요.
2. 복합 메트릭스 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 수신 설정** 화면에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 수신 설정** 화면의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules- `YYYYMMDD` .json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.
3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

> 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.
>
> ```
> Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)
> ```

알림 메시지 설정

수신할 메시지의 상세 항목을 선택하고, 항목별 내용을 편집할 수 있습니다.

부록

MXQL 문법 가이드

복합 메트릭스는 MXQL 쿼리를 기반으로 동작합니다. 자세한 문법은 MXQL 문법 가이드를 참고하시기 바랍니다.

히트맵 패턴 이벤트 설정

홈 > 프로젝트 선택 > 🔔 **경고 알림** > **이벤트 설정** > **히트맵 패턴** 탭

히트맵 패턴은 애플리케이션 실행 구간(응답 시간 분포, 호출 밀도 등)의 시간대별 패턴 변화를 기반으로 이상 징후를 자동 탐지하는 기능입니다.

복잡한 조건이나 임계값을 직접 설정하지 않아도, 시스템이 실시간으로 수집한 히트맵에서 정상 패턴과 다른 이상 패턴을 감지하면 자동으로 경고 알림이 발생합니다. 히트맵 패턴 경고 알림은 다음 상황에서 유용합니다.

- 특정 트랜잭션이 평소보다 특정 시간대에만 과도하게 몰리는 경우
- 트래픽 분포나 응답 패턴이 평소와 달라진 경우
- 조건 설정만으로는 포착하기 어려운 패턴 기반 이상 탐지

이벤트 설정

탐지할 이벤트 패턴의 옵션을 설정하세요.

활성화

토글을 켜면 해당 패턴이 감지될 때 경고 알림이 전송됩니다.

- **활성화**: 조건을 충족하면 이벤트 발생
- **비활성화**: 이벤트가 발생하지 않으며, 조건을 검사하지 않음

무음

이벤트 발생 후, 선택한 시간 동안 이벤트가 발생되지 않고 **이벤트 기록**에 남지 않습니다.

이벤트 수신 태그

이벤트에 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

1. **+ 태그 추가** 또는 +를 클릭하세요.
2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
3. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
4. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
5. **이벤트 수신 태그** 창에서 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

이벤트 동작 시간

이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. **이벤트 동작 시간**에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 제거됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

히트맵 패턴 유형

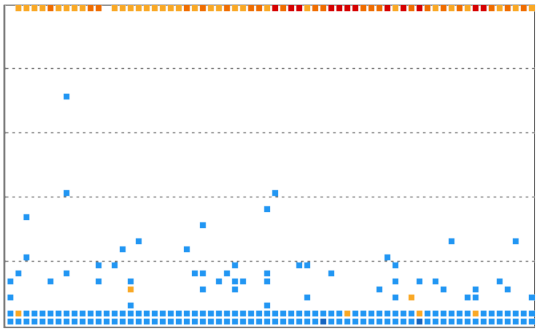
각 패턴의 토글을 켜면 해당 패턴이 감지되었을 때 이벤트가 발생됩니다.

히트맵 가로라인 패턴

여러 트랜잭션이 비슷한 응답 시간 구간에서 일괄적으로 종료되는 패턴입니다. 특정 응답 시간대에 몰림 현상이 발생하며, 서비스의 지연 증가 또는 외부 자원 응답 불안정을 의미합니다.

- 외부 API/DB/HTTPS 호출에서 Timeout 또는 지연 발생

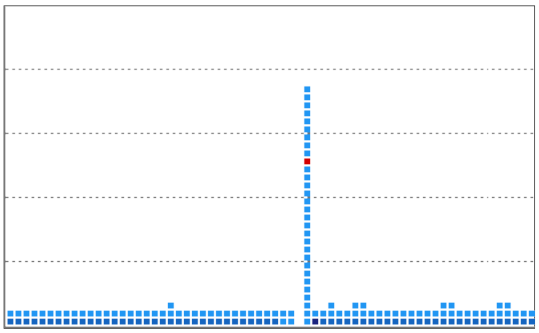
- 특정 리소스를 사용하려던 요청이 일정 시간 대기 후 한번에 처리되는 경우
- 공통 I/O 또는 네트워크 구간에서 반복적인 지연 발생



히트맵 세로라인 패턴

트랜잭션의 시작 시점은 다르지만 같은 시점에 동시에 종료되는 패턴입니다. 여러 요청이 특정 순간에 한 번에 처리되며 공통 자원의 병목(Bottleneck)을 의미합니다.

- DB Lock, File Lock 등 공유 자원에서 대기 시간 증가
- Queue 또는 Thread Pool에서 동시 처리 지연
- GC Pause 등 시스템 정지로 인해 요청이 잠시 멈췄다가 동시에 재개

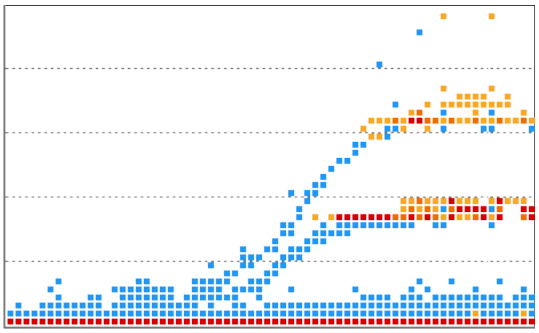


히트맵 세로라인 패턴은 락 경쟁 또는 병목 자원의 상태 점검이 필요한 신호입니다.

히트맵 플라잉 패턴

특정 리소스나 로그와 같은 공통 자원 부족으로 응답 시간이 일정 간격을 두고 파도처럼 오르내리는 패턴입니다.

- GC, 커넥션 풀, 파일 핸들 등 공통 리소스 부족
- 로그 flush, 스토리지 I/O 등 주기적 지연 요소 존재
- 배치성 작업 또는 백그라운드 job이 주기적으로 리소스 점유

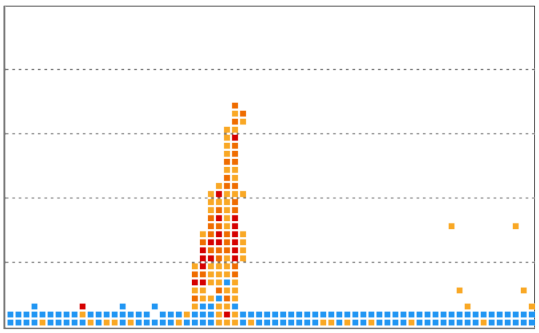


히트맵 플라잉 패턴은 주기적 병목 또는 리소스 부족의 가능성이 높습니다.

히트맵 과부하 패턴

특정 응답 시간대에 트랜잭션이 비정상적으로 밀집되는 패턴입니다. 일시적인 부하 또는 장애 전조로 인해 요청이 정상적으로 분산되지 못하고 특정 구간에 집중될 때 나타납니다.

- 일시적 장애 또는 지연으로 인해 트랜잭션이 대기열에 쌓였다가 일괄 처리
- 백엔드 서비스 응답 지연
- 특정 시간대 트래픽 집중으로 처리 여유 부족

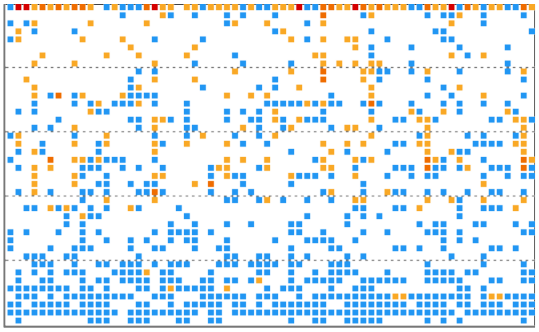


과부하 패턴은 일시적 서비스 품질 저하 또는 부하 분산 실패를 의미합니다.

히트맵 폭주 패턴

과도한 요청 또는 부하로 인해 전체 응답 시간이 전반적으로 증가하는 패턴입니다. 서비스 전반의 성능 저하가 발생하고 있음을 의미합니다.

- 사용자 트래픽 또는 자동화 트래픽 급증
- CPU, 메모리, GC 등 시스템 자원 부족
- 백엔드 병목으로 전체 응답 품질이 악화되는 경우



폭주 패턴은 즉각적인 점검이 필요합니다.

이상 탐지 이벤트 설정

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **이상치 탐지** 탭

이상치 탐지는 메트릭스 데이터에서 비정상적인 상승 또는 하락 패턴을 식별해, 평소와 다른 움직임이 감지되면 이벤트를 발생시키는 기능입니다.

기본 옵션

- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **+ 이벤트 규칙 추가**: 새로운 이상치 탐지 이벤트 추가

표 | 이상치 탐지 이벤트 목록 구성

항목	설명
이벤트 메시지	이벤트 발생 시 전송되는 메시지
카테고리	메트릭스 데이터를 구분하는 단위로 메트릭스 이벤트 설정 기준이 되는 카테고리
필드	이상치를 감지할 지표(메트릭스 필드)
필터	이상치 탐지 대상이 될 데이터 범위를 제한하는 조건
상승/하락 패턴	상승 또는 하락 패턴의 민감도
이벤트 수신 태그	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)
	이벤트 설정 수정
	이벤트 삭제
	이벤트의 활성화 여부 - 활성화: 조건을 충족하면 이벤트 발생 - 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음

이벤트 설정

메트릭스의 데이터를 기반으로 값의 움직임이 평소와 다르게 상승하거나 하락할 때 경고 알림을 보내도록 설정할 수 있습니다.

이벤트 규칙 추가

1. 화면 오른쪽의 [+ 이벤트 규칙 추가] 버튼을 클릭하세요.
2. **이벤트 기본 정보, 메트릭스, 이상치 탐지, 부가정보**를 차례로 설정하세요.

이벤트 기본 정보

1. 이벤트 레벨(Critical, Warning, Info)을 선택하세요.
   - 이벤트의 중요도와 알림 레벨이 결정됩니다.
2. 이벤트 발생 시 전달받을 알림 메시지를 입력하세요.
   - 메시지에 변수를 사용하여 이벤트 발생 시점의 실제 값을 포함할 수 있습니다.

> ⓘ **사용 가능 변수**
>
> - `$pcode` : 프로젝트 코드
> - `$category` : 카테고리
> - `$field` : 필드
> - `$value` : 발생값
> - `$oname` : 이벤트 대상

메트릭스

1. **카테고리**를 선택하세요.
   - 이상치를 감지할 메트릭스 데이터의 기준이 되는 카테고리입니다.
2. 이벤트 발생 조건에 사용할 **필드**를 선택하세요. 다중 선택할 수 있습니다.

3. 이벤트 대상을 필터링할 이벤트 조건 대상인 필터를 선택하세요. 다중 선택할 수 있습니다.

이상치 탐지

1. **상승 태턴**과 **하락 패턴**의 민감도를 선택하세요.
2. 각 패턴의 토글 버튼을 클릭해 켜거나 끌 수 있습니다.
   - 상승 패턴만 감지하고 싶은 경우: 상승 패턴 **ON**, 하락 패턴 **OFF**
   - 하락 패턴만 감지하고 싶은 경우: 하락 패턴 **ON**, 상승 패턴 **OFF**

부가 정보

이벤트 동작 시간

이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. **이벤트 동작 시간**에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 ✎ 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 삭제할 수 있습니다.

> ⚠ 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 삭제됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

이벤트 수신 태그

이벤트 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

1. **+ 태그 추가** 또는 +를 클릭하세요.
2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.

3. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
4. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
5. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> 이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. **경고 알림** > **이벤트 수신 설정**에서 프로젝트 멤버와 3rd-party 플러그인에 각각 태그를 지정할 수 있습니다.

이벤트 수정

1. **경고 알림** > **이벤트 설정**의 **이상치 탐지** 탭으로 이동하세요.
2. **이상치 탐지** 목록에서 수정할 이벤트의 오른쪽 아이콘 클릭하세요.
3. **이상치 탐지** 창에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

이벤트 삭제

1. **경고 알림** > **이벤트 설정**의 **이상치 탐지** 탭으로 이동하세요.
2. **이상치 탐지** 목록에서 수정할 이벤트의 오른쪽 아이콘 클릭하세요.
3. 삭제 확인 창에서 [삭제] 버튼을 클릭하세요.

실시간 로그 이벤트

홈 > 프로젝트 선택 > 경고 알림 > **이벤트 설정** > **실시간 로그** 탭

수집한 로그 데이터를 조건에 맞춰 필터링하려 경고 알림을 설정할 수 있습니다. 로그 이벤트를 사용하려면 로그 모니터링이 활성화되어야 합니다. **로그 모니터링 활성화**에 대한 자세한 내용은 각 제품 로그 문서를 참고하세요.

- **실시간 로그 이벤트**: 실시간으로 수집한 로그에서 설정한 이벤트 조건이 충족되면 이벤트가 발생합니다.

기본 옵션

- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **알림 메시지 설정**: 수신할 메시지의 상세 항목 선택 및 항목 별 내용 편집
- **+ 이벤트 추가**: 새로운 로그 이벤트 추가

표 | 실시간 로그 이벤트 목록 구성

항목	설명
이벤트 이름	알림 제목으로 사용되는 이벤트 이름 - 최대 255byte까지 입력 가능
카테고리	로그 구분 명칭(로그 폴더명)
검색 키	로그 데이터에서 찾고 싶은 항목 - 예: HTTP 응답 상태 코드를 나타내는 값에 접근하고자 할 경우 검색 키 `status`
검색 값	검색 키에 해당하는 실제 데이터로 로그에서 입력한 단어를 포함할 경우 경고 알림을 보냄 - 예: 검색 키 `status` 검색 값 `200`을 설정한 경우 HTTP 응답 상태 코드 200을 포함하는 로그 데이터 수집 시 경고 알림 발생
이벤트 발생 일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 - 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 - 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가

항목	설명
	발생해도 알림이 발생하지 않음
이벤트 대상 필터링	이벤트가 적용될 대상의 조건 - 대상 조건을 설정하지 않으면 수집되는 모든 데이터에 대해 알림 여부 판단 - 선택 입력: 태그, 연산자, 값을 선택해 조건 구성 - 직접 입력: 조건을 직접 입력해 대상 지정
이벤트 수신 태그	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)
	이벤트 수정 및 삭제
	이벤트의 활성화 여부 - 활성화: 조건을 충족하면 이벤트 발생 - 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음

이벤트 설정

로그 이벤트는 실시간 로그 또는 일정 조건을 만족하는 로그가 누적될 때 경고 알림을 설정할 수 있는 기능입니다.

실시간 로그 이벤트 추가

1. **경고 알림** > **이벤트 설정**의 **실시간 로그** 탭을 클릭하세요.
2. 화면 오른쪽의 [+ 이벤트 추가] 버튼을 클릭하세요.
3. **이벤트 규칙 추가** 창에서 로그 이벤트 조건을 설정하세요.
    - **이벤트 이름**: 이벤트 제목으로 사용할 이름을 입력하세요.
    - **이벤트 활성화**: 토글 버튼 활성화 시, 토글 버튼 활성화 시, 조건을 충족하면 이벤트 발생합니다.
    - **레벨**: 이벤트 레벨(Critical, Warning, Info)을 선택하세요.
    - **메시지**: 이벤트 발생 시 전달받을 알림 메시지를 입력하세요. 메시지에 변수를 사용하여 이벤트 발생 시점의 실제 값을 포함할 수 있습니다.

> **변수 사용 규칙**
>
> - 변수 형식: `${지표명}`
> - 설정한 **카테고리와 동일한 카테고리** 내 지표만 변수로 사용할 수 있습니다.
> - 여러 변수를 조합해 상세한 메시지를 구성할 수 있습니다.
> - 사용 가능한 변수 목록은 **메트릭스 조회**에서 확인할 수 있습니다.

- **카테고리**: 로그 구분 명칭(로그 폴더명)을 목록에서 선택하거나 직접 입력하세요. 카테고리를 선택해야 검색 키와 검색 값을 입력할 수 있습니다.
- **검색 키**: 로그 데이터에서 찾고 싶은 항목을 입력하세요.
  - 예: HTTP 응답 상태 코드를 나타내는 값에 접근하고자 할 경우 **검색 키** `status`
- **검색 값**: **검색 키**에 해당하는 실제 데이터를 입력하세요. 로그에서 입력한 단어를 포함할 경우 경고 알림을 보냅니다.
  - 예: **검색 키** `status` **검색 값** `200` 을 설정한 경우 HTTP 응답 상태 코드 200을 포함하는 로그 데이터 수집 시 경고 알림 발생
- **이벤트 대상 필터링**: 이벤트가 적용될 대상의 조건을 입력하세요. 입력값이 없으면, 수집되는 모든 데이터에 대해 알림 여부 판단합니다.
  - **선택 입력**: 태그, 연산자, 값을 선택해 대상 조건을 설정하세요.

    > 조건식 작성 방법과 지원되는 연산자는 조건 설정 가이드를 참고하세요.

  - **직접 입력**: 조건을 직접 입력하세요.
- **이벤트 발생 일시 중지**: 알림을 일시 중지할 시간을 선택하세요. 이벤트 발생 후 설정한 시간 동안 동일한 이벤트가 발생하지 않습니다.
  - 선택 가능 시간: `사용 안 함` , `1분` , `2분` , `3분` , `5분` , `10분` , `20분` , `30분` , `1시간` , `3시간` , `6시간` , `12시간`
- **이벤트 동작 시간**: 이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. **이벤트 동작 시간**에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 제거됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

- **이벤트 수신 태그**: 이벤트에 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신자 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

  1. **+ 태그 추가** 또는 +를 클릭하세요.
  2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
  3. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
  4. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
  5. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

  > 이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. **경고 알림 > 이벤트 수신 설정**에서 프로젝트 멤버와 3rd-party 플러그인에 각각 태그를 지정할 수 있습니다.

4. **이벤트 규칙 추가** 창에서 설정이 완료되면 [저장] 버튼을 클릭하세요.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 설정**의 **실시간 로그** 탭으로 이동하세요.

2. 로그 이벤트 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 설정** 화면에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 설정** 화면의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules-`YYYYMMDD`.json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.

3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

ⓘ 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.

```
Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)
```

부록

발생 조건, 대상 선택 가이드

메트릭스 경고 알림의 이벤트 발생 조건과 이벤트 대상 선택은 동일한 문법을 사용합니다. 단, 이벤트 발생 조건은 필드(Field)의 Key를 변수로 사용하고, 이벤트 대상 선택은 태그(Tag)의 Key를 변수로 사용합니다.

레벨 동작 가이드

이벤트 레벨은 위험(Critical)과 경고(Warning), 정상(Info)으로 구분합니다.

1. **우선순위 기반 이벤트 발생**

   여러 레벨의 조건을 동시에 충족할 경우, 가장 높은 우선순위의 이벤트만 발생합니다.

   예시

   ```
   설정:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   현재 상태: CPU 95%
   → 결과: Critical 이벤트만 발생 (Warning은 억제됨)
   ```

2. **레벨 상승**

   진행중 상태를 갖는 이벤트에서 낮은 레벨 이벤트가 진행 중일 때 높은 레벨 조건을 충족하면, 두 레벨 모두 진행 중

상태가 됩니다.

예시

```
설정:
- Warning: CPU > 70%
- Critical: CPU > 90%

시나리오:
1) CPU 80% → Warning 발생 (진행 중)
2) CPU 95% → Warning (진행 중), Critical (진행 중)
```

3. 레벨 하강

진행중 상태를 갖는 이벤트에서 높은 레벨에서 낮은 레벨로 전환되면 높은 레벨은 해소되고 낮은 레벨이 유지됩니다.

예시

```
설정:
- Warning: CPU > 70%
- Critical: CPU > 90%

시나리오:
1) CPU 97% → Critical 발생
2) CPU 85% → Critical 해소, Warning 유지
3) CPU 65% → Warning 해소
```

4. 동작 흐름도

```
- Warning: CPU > 70%
- Critical: CPU > 90%

메트릭 값: 60%  →  75%  →  92%  →  85%  →  60%
상태:      정상     Warning   Critical  Warning   정상
                    (발생)    (발생)    (유지)    (해소)
                              Warning
                              (유지)
```

진행 중 상태를 갖는 이벤트는 조건이 충족되는 동안 계속 활성화 상태를 유지하며, 조건이 해제될 때까지 지속됩니다.

조건 설정 가이드

지표명이 숫자로 시작하거나 특수문자가 포함된 경우 `${지표명}` 형태로 작성해야 합니다.

- 괄호 `()`, `[]`
- 연산자 `+`, `-`, `*`, `/`, `%`
- 구분자 `:`, `@`, `#`, `,`, (공백)
- 기타 특수문자 `!`, `^`, `&`, `|`, `~`, `` ` ``, `=`

`${}` 사용 예시:

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` 없이 사용 가능: 영문자, 언더스코어(`_`), 점(`.`)만 포함된 경우

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. 비교 연산자

숫자 비교

```
크다:          cpu > 80
크거나 같다:    cpu >= 80
작다:          memory < 1000
작거나 같다:    memory <= 1000
같다:          status == 200
```

```
같지 않다:      error != 0
```

문자열 비교

```
status == 'OK'
region == "us-east-1"
```

2. 산술 연산자

- 기본 산술 연산

```
덧셈:   cpu + 10 >= 90
뺄셈:   memory - 100 >= 500
곱셈:   cpu * 2 >= 100
나눗셈:  disk / 1024 >= 100
나머지:  value % 10 == 0
```

- 괄호를 이용한 우선순위 제어

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 음수 표현

```
cpu > -100
```

3. 논리 연산자

- AND 연산 (&&)

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 연산 (||)

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 복합 논리 연산

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. 패턴 매칭 연산자

- LIKE 연산자

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 연산자

```
oname not like 'test-*'
url not like '*debug*'
```

5. 내장 함수

- Null 체크 함수

```
isNull(value)          # null인지 확인
isNotNull(value)       # null이 아닌지 확인
nvl(value, 0)          # null이면 기본값 반환
isEmpty(str)           # 빈 문자열인지 확인
isNotEmpty(str)        # 빈 문자열이 아닌지 확인
```

- 집계 함수

```
sum(cpu, memory, disk)          # 합계
avg(cpu, memory)                # 평균
max(cpu, memory, disk)          # 최대값
```

```
min(cpu, memory, disk)             # 최소값
count(value1, value2, value3)      # 개수
```

- 수학 함수

```
round(cpu, 2)          # 반올림 (소수점 2자리)
```

- 문자열 함수

```
length(str)              # 문자열 길이
startsWith(str, 'prefix')  # 접두사 확인
endsWith(str, 'suffix')    # 접미사 확인
indexOf(str, 'search')     # 문자열 위치
substring(str, 0, 10)      # 부분 문자열
trim(str)                # 공백 제거
replace(str, 'old', 'new') # 문자열 치환
hasStr(str, 'search')      # 문자열 포함 여부
```

문자열 함수

```
equalsIgnoreCase(str, 'match_text')     # 대소문자를 구분하지 않는 일치 비교
notEqualsIgnoreCase(str, 'match_text')  # 대소문자를 구분하지 않는 불일치 비교
```

`==` 와 `!=` 는 대소문자를 구분하므로 `WebServer` 와 `webserver` 를 다른 값으로 판단합니다. 표기가 일정하지 않은 태그나 필드를 비교할 때 위 두 함수를 사용하면 대소문자 차이를 무시하고 값을 맞출 수 있습니다.

예시

```
equalsIgnoreCase(okindName, 'WebServer')
notEqualsIgnoreCase(container, 'PROD.billing')
```

이벤트 발생 조건과 이벤트 대상 선택은 같은 문법을 사용하므로 두 함수를 양쪽 조건식에 모두 작성할 수 있습니다.

- 조건 함수
  - **if:** 조건식이 참(`true`)이면 `true값` 을 반환하고, 거짓(`false`)이면 `false값` 을 반환하는 조건 함수

형식

```
if(조건, true값, false값)
```

예시

```
if(value >= 90, 'High', 'Medium') == 'High'
```

- **decode**: 값과 조건을 비교하여 일치하는 조건의 결과를 반환, 없으면 기본값 반환

형식

```
decode(값, 조건1, 결과1, 조건2, 결과2, ..., 조건N, 결과N, 기본값)
```

예시

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

- **in**: 값이 지정한 후보값 중 하나와 일치하면 `true`, 아니면 `false`

형식

```
in(값, 후보값1, 후보값2, ..., 후보값N)
```

예시

```
in(status, 200, 201, 204) == false
```

작성 시 주의사항

- 권장사항
  - 간단한 변수명은 `${}` 없이 사용 (예: cpu, memory)
  - 특수문자 포함 시 반드시 `${}` 사용
  - 복잡한 연산은 괄호로 명확하게 구분
  - 함수명은 정확히 입력 (대소문자 구분)

- 피해야 할 사항
  - 불완전한 표현식 작성 (예: `cpu >` , `memory &&`)
  - 괄호 미스매칭 (예: `(cpu > 80 && cpu > 70)`)
  - 연속된 연산자 사용 (예: `cpu >> 80` , `cpu >< 80`)

복합 로그 이벤트 설정

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **복합 로그** 탭

수집한 로그 데이터를 조건에 맞춰 필터링하려 경고 알림을 설정할 수 있습니다. 로그 이벤트를 사용하려면 로그 모니터링이 활성화되어야 합니다. **로그 모니터링 활성화**에 대한 자세한 내용은 (각 제품 로그 문서 링크)를 참고하세요.

- **복합 로그 이벤트**: 설정한 **데이터 조회 범위** 내 로그를 기준으로, 로그 검색 조건을 충족하는 로그 수가 임계 개수 이상일 경우 이벤트가 발생합니다.

> **권한**
>
> 복합 메트릭스 이벤트를 설정하려면 **알림 설정** 권한이 필요합니다. 알림 설정 권한은 **홈 화면 > 프로젝트 선택 > 관리 > 통합 멤버 관리**에서 권한을 부여할 멤버의 아이콘을 클릭해 설정할 수 있습니다.

기본 옵션

- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **알림 메시지 설정**: 수신할 메시지의 상세 항목 선택 및 항목 별 내용 편집
- **+ 이벤트 추가**: 새로운 로그 이벤트 추가

표 | 복합 로그 이벤트 목록 구성

항목	설명
이벤트 이름	알림 제목으로 사용될 이벤트 이름
템플릿	복합 로그 템플릿
카테고리	로그 구분 명칭(로그 폴더명)
규칙	이벤트 발생하는 조건

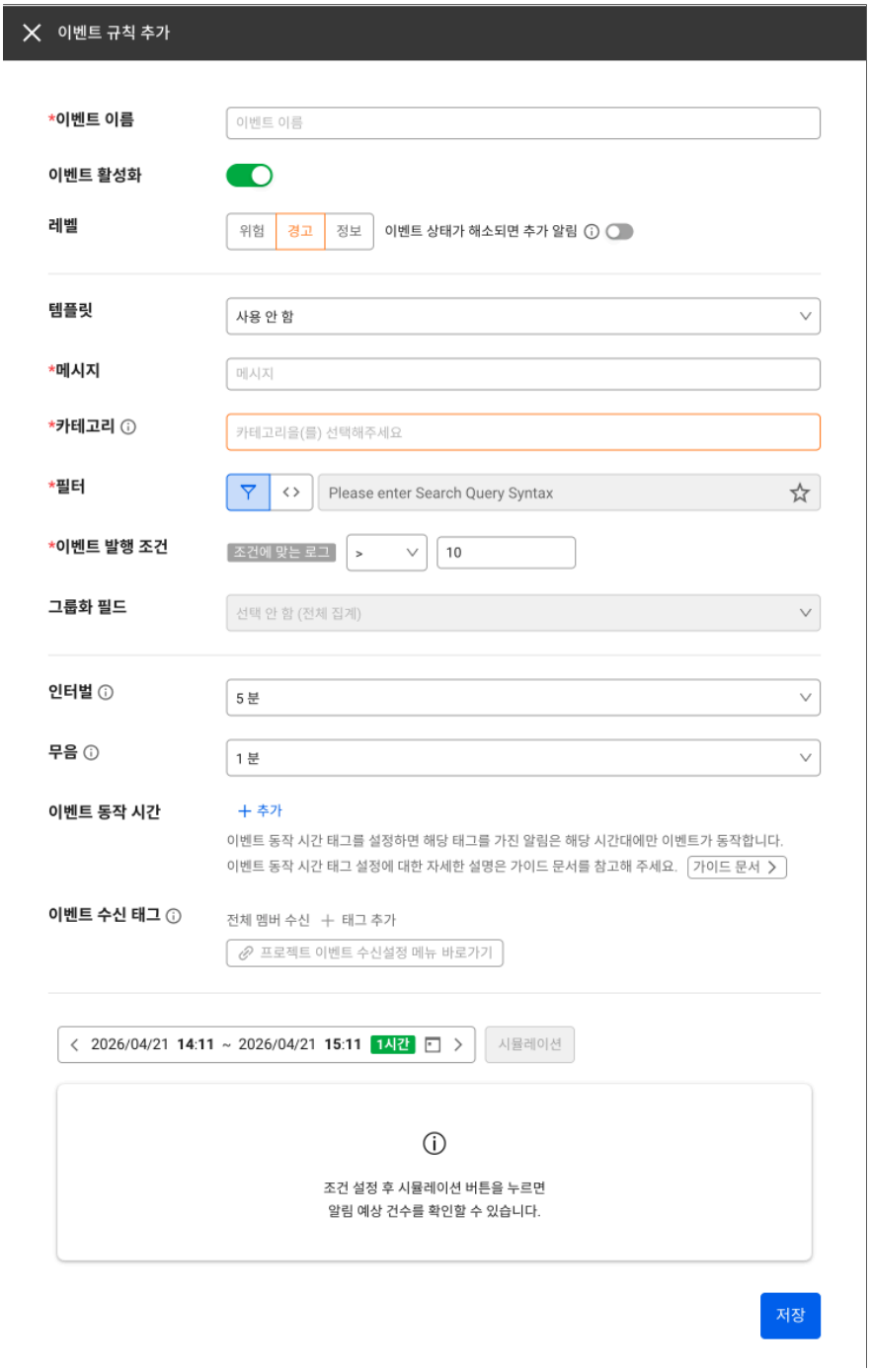
항목	설명
이벤트 상태가 해소되면 추가 알림	Critical과 Warning 레벨의 이벤트가 해소되면 정상(RECOVERED) 상태의 알림 수신 여부 - 활성화: 진행 중 → 정상 상태로 전환될 때 RECOVERED 알림을 보내며, 이벤트 기록에서 진행 중/해소 상태를 구분해 볼 수 있음 - 비활성화: 임계값을 초과할 때 마다 단발성 이벤트 발생
이벤트 발생 일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 - 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 - 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가 발생해도 알림이 발생하지 않음
인터벌	로그 데이터를 조회할 시간 범위
이벤트 수신 태그	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)
	이벤트 수정 및 삭제
	이벤트의 활성화 여부 - 활성화: 조건을 충족하면 이벤트 발생 - 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음

이벤트 설정

복합 로그 이벤트는 설정한 **데이터 조회 범위** 내 로그를 기준으로, 로그 검색 조건을 충족하는 로그 수가 임계 개수 이상일 경우 경고 알림을 설정할 수 있는 기능입니다.

복합 로그 이벤트 추가

1. **경고 알림** > **이벤트 설정**의 **복합 로그** 탭을 클릭하세요.
2. 화면 오른쪽의 [+ 이벤트 추가] 버튼을 클릭하세요.
3. **이벤트 규칙 추가** 창에서 로그 이벤트 조건을 설정하세요. 자세한 내용은 기본 설정, 상세 설정, 추가 설정를 참고하세요.



4. **이벤트 규칙 추가** 창에서 설정이 완료되면 [저장] 버튼을 클릭하세요.

기본 설정

- **이벤트 이름**: 이벤트 제목으로 사용할 이름을 입력하세요.
- **이벤트 활성화**: 토글 버튼 활성화 시, 조건을 충족하면 이벤트 발생합니다.
- **레벨**: 이벤트 레벨(Critical, Warning, Info)을 선택하세요.

상세 설정

템플릿(프리셋), 메시지, 카테고리, **필터**, 이벤트 발생 조건 항목을 설정합니다.

템플릿(프리셋)

미리 정의된 프리셋을 선택하면 **필터, 쿼리 타입, 이벤트 발행 조건**이 자동으로 설정됩니다. 프리셋을 선택한 후 필요에 맞게 수정할 수 있습니다.

복합 로그 템플릿 유형

프리셋	쿼리 타입	기본 규칙
사용 안 함	none	`rows > 10`
2xx 상태코드 건수	count	`count > 10`
3xx 상태코드 건수	count	`count > 10`
4xx 상태코드 건수	count	`count > 10`
5xx 상태코드 건수	count	`count > 10`
정상 상태코드(2xx, 3xx) 건수	count_v2	`include_minus_exclude_count > 10`
에러 상태코드(4xx, 5xx) 건수	count_v2	`include_minus_exclude_count > 10`
에러 수신 건수	count	`count > 10`
예외 수신 건수	count	`count > 10`

프리셋	쿼리 타입	기본 규칙
초당 로그 발생건수	rps	`rps > 100`
필드 집계 (Aggregation)	aggr	`avg > 1000`
비율 (Rate)	rate	`rate > 10`
고유값 수 (Cardinality)	cardinality	`cardinality > 100`
백분위 (Percentile)	percentile	`p99 > 3000`

메시지

이벤트 발생 시 전달받을 알림 메시지를 입력하세요. 메시지에 변수를 사용하여 동적 메시지를 구성할 수 있습니다.

- 예시: 메시지 `${host}`에서 에러 `${count}`건 발생 → 알림 메시지 web-01에서 에러 15건 발생

표 | 메시지 변수

변수	설명	사용 조건
`${count}`	조건에 맞는 로그 건수	항상 사용 가능
`${groupField}`	그룹화 필드 이름	그룹화 알림만
`${groupValue}`	그룹화 필드 값	그룹화 알림만
`${필드명}`	그룹화 필드 값 (필드명으로 직접 참조)	그룹화 알림만

`${count}` 변수는 그룹화 여부와 관계없이 모든 알림에서 사용할 수 있습니다. `${groupField}`, `${groupValue}`, `${필드명}` 변수는 그룹화 필드를 설정한 경우에만 치환됩니다.

카테고리

로그 구분 명칭(로그 폴더명)을 목록에서 선택하거나 직접 입력하세요. 카테고리를 선택해야 검색 키와 검색 값을 입력할 수 있습니다.

필터

Lucene 쿼리 문법으로 로그 검색 조건을 입력합니다. 두 가지 입력 모드를 지원합니다.

- **필터 모드** : 필드와 값을 선택하여 쿼리를 구성합니다.
- **코드 모드** : Lucene 쿼리를 직접 입력합니다.

표 | Lucene 쿼리 예시

쿼리	설명
`level:ERROR`	level 필드가 ERROR인 로그
`status_nxx:status_4xx OR status_nxx:status_5xx`	4xx 또는 5xx 상태코드
`appender:accessLog`	accessLog appender의 로그
`response_time.n:*`	response_time.n 필드가 존재하는 로그
`host:web-01 AND level:ERROR`	web-01 호스트의 에러 로그

이벤트 발생 조건

쿼리 타입별로 설정합니다. 선택한 템플릿(프리셋) 또는 쿼리 타입에 따라 추가 설정 항목이 표시됩니다.

표 | 이벤트 발행 조건 키 의미

키	의미
`rows` / `count`	필터 조건에 맞는 로그 건수
`rps`	초당 로그 발생건수 (건수 / 인터벌 초)

키	의미
avg	대상 필드 값의 평균
sum	대상 필드 값의 합계
min	대상 필드 값의 최솟값
max	대상 필드 값의 최댓값
rate	분자 건수 / 분모 건수 × 100 (%)
cardinality	대상 필드의 고유한 값 개수
p99 / p95 / p90 / p75 / p50	대상 필드의 백분위 값
include_minus_exclude_count	포함 필터 건수 - 제외 필터 건수

- **건수 (count / none)**: 설정한 필터 조건에 맞는 로그의 건수를 카운트합니다.

표 | 건수 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
이벤트 발행 조건	건수 또는 조건에 맞는 로그 {연산자} {임계값} (예: count > 10)

- **초당 로그 발생건수 (rps)**: 설정한 필터 조건에 맞는 로그의 초당 발생건수(Requests Per Second)를 계산합니다. 인터벌 시간 동안의 총 건수를 인터벌 초로 나누어 산출합니다.

표 | 초당 로그 발생건수 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
이벤트 발행 조건	RPS {연산자} {임계값} (예: rps > 0.5)

- **필드 집계 (aggr)**: 지정한 숫자 필드의 집계 값(평균, 합계, 최솟값, 최댓값)을 계산합니다. 이벤트 발행 조건에서 집계 방식을 드롭다운으로 선택할 수 있습니다.

표 | 필드 집계 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
대상 필드	집계 대상 숫자 필드 (예: `response_time.n`)
이벤트 발행 조건	`평균` / `합계` / `최솟값` / `최댓값` `{연산자}` `{임계값}` (예: `avg > 1000`)

- **비율 (rate)**: 분자 필터와 분모 필터의 건수 비율(%)을 계산합니다. 전체 로그 대비 특정 조건의 비율을 모니터링할 때 유용합니다.

표 | 비율 설정 항목

설정 항목	설명
필터 (분자)	비율 계산의 분자가 되는 로그 조건 (예: `level:ERROR`)
분모 필터	비율 계산의 분모가 되는 로그 조건 (예: `level:*`)
이벤트 발행 조건	`비율 (%)` `{연산자}` `{임계값}` (예: `rate > 10`)

  - **예시**: 필터 `level:ERROR`, 분모 필터 `level:*`, rule `rate > 10` → 전체 로그 중 에러 비율이 10%를 초과하면 알림 발생

- **고유값 수 (cardinality)**: 지정한 필드의 고유한 값 개수를 계산합니다. 비정상적으로 많은 종류의 값이 발생하는 경우를 감지할 때 유용합니다.

표 | 고유값 수 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
대상 필드	고유값을 계산할 필드 (예: `host`)

설정 항목	설명
이벤트 발행 조건	`고유값 수` `{연산자}` `{임계값}` (예: `cardinality > 100`)

- **백분위 (percentile)**: 지정한 숫자 필드의 백분위(Percentile) 값을 계산합니다. 응답 시간의 tail latency를 모니터링할 때 유용합니다. Percentile 선택을 변경하면 이벤트 발행 조건의 키가 자동으로 업데이트됩니다.

표 | 백분위 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
대상 필드	백분위를 계산할 숫자 필드 (예: `response_time.n`)
Percentile	P99, P95, P90, P75, P50 중 선택
이벤트 발행 조건	`P99` / `P95` /... `{연산자}` `{임계값}` (예: `p99 > 3000`)

표 | 백분위 의미

백분위	의미
P50	중앙값. 전체 데이터의 50%가 이 값 이하
P75	전체 데이터의 75%가 이 값 이하
P90	전체 데이터의 90%가 이 값 이하
P95	전체 데이터의 95%가 이 값 이하
P99	전체 데이터의 99%가 이 값 이하

- **포함/제외 건수 (count_v2)**: 포함 필터의 건수에서 제외 필터의 건수를 뺀 순수 건수를 계산합니다.

표 | 포함/제외 건수 설정 항목

설정 항목	설명
필터 (포함)	포함할 로그 조건 (예: `status_nxx:status_4xx OR status_nxx:status_5xx`)
필터 제외	제외할 로그 조건 (예: `status_nxx:status_4xx`)
분모 필터	비율 계산용 분모 (선택사항)
이벤트 발행 조건	`포함-제외 건수` `{연산자}` `{임계값}` (예: `include_minus_exclude_count > 10`)

- **예시**: 필터 `status_nxx:status_4xx OR status_nxx:status_5xx`, 필터 제외 `status_nxx:status_4xx` → 4xx+5xx에서 4xx를 제외한 순수 5xx 건수가 10을 초과하면 알림 발생

그룹화 필드(Group By)

그룹화 필드를 설정하면 해당 필드의 값별로 독립적으로 알림 조건을 판정합니다. 드롭다운에서 그룹화할 필드를 선택합니다. 드롭다운을 클릭하면 현재 카테고리에서 사용 가능한 필드 목록이 자동완성됩니다.

- **예시:** 필터 `level:ERROR`, 그룹화 필드 `host`, rule `rows > 3`

표 | 그룹화 설정 비교

항목	그룹화 없음	그룹화 사용
판정 단위	전체 로그를 하나로 집계	그룹 값별 독립 집계
알림 발생	1건	조건 충족 그룹 수만큼
알림 제목	`이벤트 제목`	`[그룹 값] 이벤트 제목`
Stateful	전체 기준 상태 전이	그룹별 독립 상태 전이

- `[web-01] 에러 알림` — web-01에서 에러 8건 → 알림 발생
- `[web-02] 에러 알림` — web-02에서 에러 5건 → 알림 발생
- `[api-01] 에러 알림` — api-01에서 에러 2건 → 조건 미충족, 알림 미발생

> **이벤트 상태가 해소되면 추가 알림**을 활성화한 경우, 그룹별로 독립적으로 ABNORMAL → RECOVERED 상태가 전이됩니다. 예를 들어 `web-01` 에서 에러가 해소되면 `[web-01] RECOVERED` 알림이 발생하며, 다른 호스트의 상태에는 영향을 미치지 않습니다.

추가 설정

- **인터벌**: 선택한 시간(`unset` , `1분` , `5분` , `10분` , `30분` , `1시간` , `12시간`) 간격으로 알림 조건을 확인합니다.
- **무음**: 이벤트 발생 후, 선택한 시간(`unset` , `1분` , `5분` , `10분` , `30분` , `1시간` , `12시간`) 동안 같은 이벤트가 발생되지 않습니다.
- **이벤트 동작 시간**: 이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.
    1. **+ 추가**를 클릭하세요.
    2. **이벤트 동작 시간**에서 **+새로운 태그 생성**을 클릭하세요.
    3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
    4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
    5. 수정 또는 삭제할 태그의 ✎ 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 제거됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

- **이벤트 수신 태그**: 이벤트에 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신자 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.
    1. **+ 태그 추가** 또는 **+**를 클릭하세요.
    2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
    3. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
    4. 생성된 태그는 태그 목록의 ✎ 아이콘을 클릭해 수정 또는 삭제하세요.
    5. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> 이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. **경고 알림 > 이벤트 수신 설정**에서 프로젝트 멤버와 3rd-party 플러그인에 각각 태그를 지정할 수 있습니다.

시뮬레이션

알림을 저장하기 전에 과거 데이터로 시뮬레이션하여 알림 발생 예상 건수를 확인할 수 있습니다.

1. 알림 설정을 완료한 후 하단의 시간 범위를 선택하세요.
2. [시뮬레이션] 버튼을 클릭하세요.
3. 차트에 시간대별 데이터와 알림 발생 횟수가 표시됩니다.

   > 시뮬레이션 시간 범위를 조절하여 다양한 시간대의 알림 발생 패턴을 확인할 수 있습니다. 최근 1~2분의 데이터는 인덱스 빌드 지연으로 조회되지 않을 수 있습니다.

4. **이벤트 규칙 추가** 창에서 설정이 완료되면 [저장] 버튼을 클릭하세요.

로그 탐색기에서 알림 생성

로그 탐색기에서 직접 알림을 생성할 수 있습니다.

1. **로그** > **로그 탐색** 메뉴로 이동하세요.
2. 카테고리와 검색 쿼리를 설정하세요.
3. 화면 오른쪽 위의 [이벤트 추가] 버튼을 클릭하세요.
4. Drawer에서 알림 설정 폼이 열리며, 현재 카테고리와 검색 쿼리가 자동으로 채워집니다.
5. 추가 설정을 완료하고 [저장] 버튼을 클릭하세요.
6. 저장 완료 후 [이동] 버튼을 클릭하면 알림 목록으로 이동합니다.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 설정**의 **복합 로그** 탭으로 이동하세요.
2. 로그 이벤트 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 설정** 화면에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 설정** 화면의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules-`YYYYMMDD`.json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.
3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

> 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.
>
> ```
> Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)
> ```

트랜잭션 이벤트 설정

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **트랜잭션** 탭

트랜잭션 이벤트는 특정 URL, 에러, 상태 코드 등을 기준으로 트랜잭션을 세밀하게 모니터링하는 기능입니다. 설정한 조건을 만족하는 트랜잭션이 감지되면 이벤트가 발생합니다. 트랜잭션 데이터를 1분 단위로 분석하며, 조건을 충족하는 트랜잭션 수가 임계값을 초과할 경우 알림을 전송합니다.

최신 에이전트 버전

트랜잭션 이벤트를 사용하려면 최소 지원 버전 이상의 에이전트가 필요합니다.

언어	최소 지원 버전
Java	2.2.50 버전 이상
PHP	버전 미정 (최신 버전 이상 사용 권장, 별도 문의 필요)
Node.js	0.5.21 버전 이상
Python	1.7.8 버전 이상
.NET	2.3.5 버전 이상
Go	버전 미정 (최신 버전 이상 사용 권장, 별도 문의 필요)

기본 옵션

- **이벤트 레벨**: 이벤트의 심각도를 의미하며 Info, Warning, Critical 세 단계로 구분
- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **+ 이벤트 추가**: 새로운 트랜잭션 이벤트 추가
- **컬럼 크기 조정**: 각 컬럼의 경계를 드래그하여 너비 조정 가능

표 | 트랜잭션 이벤트 목록 구성

항목	설명
No.	이벤트의 순번
	이벤트의 활성화 여부 - 활성화: 조건을 충족하면 이벤트 발생 - 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음
	이벤트 설정 수정 및 삭제
이벤트 이름	사용자가 지정한 이벤트명 - 최대 255byte까지 입력 가능
조건 유형	이벤트 발생 조건 유형(경과 시간, 에러 발생, 패턴만)
집계 기준	트랜잭션 수를 집계하는 기준
대상 패턴	이벤트가 적용되는 포함/제외 패턴 (URL, 상태 코드 등)
발생 횟수	이벤트 발생 기준이 되는 횟수 - 집계 주기: 1분 고정(매 정시 00~59초 기준, 기간 선택 불가) - 해당 1분 동안 조건을 충족한 트랜잭션이 설정한 횟수 이상 발생하면 이벤트 발생
일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 - 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 - 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가 발생해도 알림이 발생하지 않음
이벤트 수신	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)

이벤트 설정

트랜잭션 이벤트는 **이벤트 조건, 대상 패턴, 기본 정보 및 수신 설정** 단계를 통해 구성됩니다. 이벤트 설정을 통해 특정 트랜잭션 패턴을 감지하고 알림을 받을 수 있습니다.

이벤트 추가

트랜잭션 이벤트를 추가하려면 **이벤트 설정** > **트랜잭션** 탭으로 이동한 후, 화면 오른쪽 상단의 [+ 이벤트 추가] 버튼을 클릭하세요.

트랜잭션 이벤트 추가 방법
Step 1. 이벤트 조건 정의: 이벤트의 기본 구조를 선택합니다.
Step 2. 이벤트 대상 선택: 이벤트가 발생하는 모니터링 대상을 선택합니다.
Step 3. 기본 정보 및 수신 설정: 이벤트 이름 및 메시지, 수신 대상을 설정합니다.

이벤트 조건 정의

트랜잭션 이벤트 발생 조건을 설정합니다.

조건 유형

이벤트를 발생시킬 기준을 선택하세요.

- **경과 시간:** 지정한 시간(ms) 이상 소요된 트랜잭션
  - **경과 시간 임계값**: 조건 유형이 경과 시간인 경우, ms 단위로 입력하세요. 설정한 시간(ms) 이상 소요된 트랜잭션이 발생할 경우 이벤트가 발생합니다.
- **에러 발생:** 에러가 발생한 트랜잭션
- **패턴만:** 특정 패턴(히트맵 패턴 등)에 해당하는 트랜잭션

집계 기준

알림 조건을 어떤 단위로 집계할지 선택하세요.

- 집계 기준: 에이전트별, 각 에이전트의 URL별, 각 에이전트의 에러 클래스별, 각 에이전트의 업무별, URL별, 에러 클래스별, 업무별, 전체

발생 횟수(필수)

조건을 충족한 트랜잭션이 몇 번 발생했을 때 알림을 보낼지 설정합니다.

예: 발생 횟수 **1회** → 1분 내 조건 충족 트랜잭션이 1건이라도 있으면 즉시 알림 / 발생 횟수 **3회** → 같은 1분 동안 조건 충족 트랜잭션이 3건 이상 누적되어야 알림

일시 중지

과도한 이벤트 알림을 방지하기 위해, 알림을 일시 중지할 시간을 선택하세요.

- 이벤트 발생 후 설정한 시간 동안 동일한 이벤트가 발생하지 않습니다.
- 선택 가능 시간: `1분` , `5분` , `10분` , `15분` , `30분` , `1시간` , `2시간` , `3시간` , `6시간` , `12시간` , `1일`

이벤트 동작 시간

이벤트가 어떤 시간대에 동작할지 설정합니다. **비근무 시간, 점검 시간대 등의 알림 발생을 제한**할 수 있습니다. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. 이벤트 동작 시간에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 ✎ 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> ⚠ 태그 삭제 시 해당 태그가 적용된 모든 알림에서 삭제됩니다. 단, 이벤트 설정에서 사용 중인 태그는 삭제할 수 없습니다.

이벤트 대상 선택

트랜잭션 필터링을 위한 패턴을 설정합니다.

패턴 설정 방법

- `Enter`로 구분하여 여러 패턴을 입력할 수 있습니다.
- 별도로 설정하지 않으면, 프로젝트 내 모든 트랜잭션을 대상으로 이벤트가 발생합니다.
- 같은 유형의 패턴은 **OR** 조건으로 적용됩니다.
- 다른 유형의 패턴은 **AND** 조건으로 적용됩니다.
- 와일드카드(`*`)를 사용하여 패턴을 설정할 수 있습니다. 와일드카드 위치는 최대 2곳까지 허용됩니다.
- 와일드카드(`*`)와 슬래시(`/`) 외에 다른 문자가 함께 포함되어야 합니다.

> **슬래시(`/`)와 와일드카드(`*`) 조합에 주의하세요**
>
> - `/api/*` : `/api` 는 매칭되지 않음, `/api/` 뒤에 경로가 있어야 매칭 (예: ✅ /api/users, ❌ /api)
> - `/api*` : `/api` 자체도 매칭됨 (예: ✅ /api, ✅ /api/users, ✅ /apitest)

일치 패턴

지정한 항목과 일치하는 데이터를 필터링합니다. 일치 패턴은 적용 후 파란색 태그로 표시됩니다.
각 항목의 패턴은 최대 10개까지 입력할 수 있습니다.

항목	설명
업무명	업무명 패턴
URL	트랜잭션 URL 패턴 - 예: `/api/*`, `*/users/*`
에러 클래스	에러 클래스명 패턴 - 예: `*Exception`, `*Error`
에러 메시지	에러 메시지 내용 패턴
상태 코드	HTTP 상태 코드 패턴 - 예: `4**`, `5**`

제외 패턴

지정한 항목과 일치하는 데이터를 필터링 대상에서 제외합니다. 제외 패턴은 적용 후 빨간색 태그로 표시되고, 일치 패턴과 동일한 형식을 사용합니다. 각 항목의 패턴은 최대 10개까지 입력할 수 있습니다.

패턴 예시

- **완전 매칭**

```
패턴: "hello"
"hello"    → 일치
"Hello"    → 불일치 (대소문자 구분)
```

- **접두사 매칭**

```
패턴: "hello*"
"hello"      → 일치
"helloworld" → 일치
"hi"         → 불일치
```

- **접미사 매칭**

```
패턴: "*world"
"world"      → 일치
"helloworld" → 일치
"hello"      → 불일치
```

- **부분 문자열 매칭**

```
패턴: "*ell*"
"hello"      → 일치 (h'ell'o)
"excellent"  → 일치 (exc'ell'ent)
"welcome"    → 불일치 (ell이 포함되지 않음)
```

- **복합 패턴**

```
패턴: "hello*world"
"hellobeautifulworld"  → 일치

패턴: "hello*test*"
```

```
"hellomytestcase"    → 일치

패턴: "*test*world"
"mytestcaseworld"    → 일치
```

표 | 실제 사용 예시

패턴 유형	예시
URL 패턴	- `/api/*` : /api/로 시작하는 모든 URL - `/users/*` : users가 포함된 모든 URL - `/api/v1/users` : 정확히 일치하는 URL만
에러 클래스 패턴	- `Exception` : Exception으로 끝나는 모든 에러 - `java.lang.*` : java.lang으로 시작하는 모든 클래스 - `Timeout*` : Timeout이 포함된 모든 에러
상태 코드 패턴	- `4**` : 400번대 모든 상태 코드 - `404` : 404 상태 코드만 - `5**` : 500번대 모든 상태 코드

패턴 제한사항

- **필수 문자열**: 패턴에는 와일드카드(`*`) 외에 최소 1개 이상의 문자가 포함되어야 합니다. 와일드카드만으로는 패턴을 설정할 수 없습니다.
  - ❌ 불가: `**` , `*`
  - ✅ 가능: `a*` , `a` , `a*` , `/api*`
- **빈 문자열**: 빈 문자열이나 null 값은 매칭되지 않습니다.
- **연속된 와일드카드**: 연속된 `*` 는 하나의 와 동일하게 처리됩니다.
- **복잡한 패턴**: 3개 이상 포함된 패턴은 지원하지 않습니다.
- **정규식 미지원**: 문자 클래스(`[a-z]`), 수량자(`+` , `?`), 그룹화(`()`) 등은 지원하지 않습니다.
- **대소문자 구분**: 모든 패턴 매칭은 대소문자를 구분합니다.
- **이스케이프 불가**: 문자 자체를 매칭할 방법은 없습니다.

기본 정보 및 수신 설정

이벤트의 기본 정보와 알림 수신 방식을 설정합니다.

이벤트 활성화

이벤트가 발생했을 때 알림 전송 여부를 설정합니다.

- **활성화**: 설정한 조건을 충족하면 이벤트의 알림이 전송됨
- **비활성화**: 설정한 조건을 충족하더라도 알림이 전송되지 않음

이벤트 이름(필수)

알림 제목으로 사용될 이벤트 이름을 입력하세요.

이벤트 레벨(필수)

이벤트 레벨(Info, Warning, Critical)을 선택하세요.

이벤트 수신

이벤트 알림을 받을 멤버를 지정합니다.

- **전체 수신** **태그 선택 수신**

프로젝트에서 이벤트 수신 설정이 활성화된 모든 대상(멤버, 채널)에게 알림을 전송합니다.

해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다.

1. **태그 선택 수신**을 선택하세요.
2. **수신 태그** 항목에서 [+ 태그 추가] 또는 [+] 버튼을 클릭하세요.
3. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
4. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
5. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
6. **이벤트 수신 태그** 창에서 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> **수신 태그**
>
> 수신자 태그는 알림을 받을 사용자 또는 채널을 **그룹 단위로 관리하기 위한 기능**입니다. 이벤트에 수신자 태그를 설정하면, 해당 태그가 포함된 사용자에게만 알림이 전달됩니다. 수신 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

참고. 이벤트 동작 예시

트랜잭션 이벤트가 실제로 발생한 예시입니다.

> **설정 조건**
>
> - 집계 기준: 에이전트별
> - 발생 횟수: 2회
> - 일시 중지: 5분
> - URL 패턴: `/api/posts/**` , `/delete/*` , `/*test*/`
> - 에러 클래스 패턴: `*TimeoutException*` , `*NullPointerException`
> - 상태 코드 패턴: `4**`
> - 제외 에러 클래스: `*IOException`
> - 제외 상태 코드: `5**`

발생한 알림 메시지 예시

이벤트 메시지의 트랜잭션 정보는 조건에 부합한 마지막 트랜잭션의 값을 표시합니다. 이벤트 메시지는 설정한 조건과 실제 발생한 트랜잭션의 정보를 모두 포함하여, 어떤 조건에 의해 알림이 발생했는지 명확히 확인할 수 있습니다.

```
(ignored count: 3)
[Per Agent] limit: 2, count: 3
[Condition]
url patterns: /api/posts/**, /delete/*, */test/*
error class patterns: *TimeoutException*, *NullPointerException
status patterns: 4**
exclude error class patterns: *IOException
```

```
exclude status patterns: 5**
[Transaction]
url: /api/posts/test/timeout
txid: 408093121470636540
status: 408
error class: java.util.concurrent.TimeoutException
```

이벤트 메시지 구성 설명

- **(ignored count: n)**: 일시 중지 시간 동안 조건에 부합했지만 알림이 발생하지 않은 트랜잭션 수입니다. 일시 중지 시간 동안 무시된 건이 없으면 이 항목은 표시되지 않습니다.
- **집계 정보**: 설정한 집계 방식에 따라 다르게 표시됩니다.
  - **에이전트별**: `[Per Agent] limit: n, count: n`
  - **URL별**: `[Per Url] limit: n` 후 각 URL과 발생 건수 나열
  - **에러 클래스별**: `[Per Error Class] limit: n` 후 각 에러 클래스와 발생 건수 나열
  - **컨텍스트별**: `[Per Context] limit: n` 후 각 컨텍스트와 발생 건수 나열
  - **각 에이전트의 URL별**: `[Per Agent & Url] limit: n` 후 각 URL과 발생 건수 나열
  - **각 에이전트의 에러 클래스별**: `[Per Agent & Error Class] limit: n` 후 각 에러 클래스와 발생 건수 나열
  - **에이전트별 컨텍스트별**: `[Per Agent & Context] limit: n` 후 각 컨텍스트와 발생 건수 나열
  - **전체**: `[Total] limit: n, count: n`
- **[Condition]**: 이벤트 발생 시점에 설정되어 있던 조건들을 표시합니다. 설정된 패턴만 표시됩니다.
  - **context patterns**: 컨텍스트 일치 패턴
  - **url patterns**: URL 일치 패턴
  - **error class patterns**: 에러 클래스 일치 패턴
  - **error message patterns**: 에러 메시지 일치 패턴
  - **status patterns**: 상태 코드 일치 패턴
  - **exclude context patterns**: 컨텍스트 제외 패턴
  - **exclude url patterns**: URL 제외 패턴
  - **exclude error class patterns**: 에러 클래스 제외 패턴
  - **exclude error message patterns**: 에러 메시지 제외 패턴

  - **exclude status patterns**: 상태 코드 제외 패턴
- **[Transaction]**: 조건에 부합한 마지막 트랜잭션의 상세 정보입니다.
  - **url**: 트랜잭션 URL
  - **txid**: 트랜잭션 고유 식별자
  - **status**: HTTP 응답 상태 코드 (HTTP 트랜잭션이 아닌 경우 미표시)
  - **error class**: 에러 클래스 (에러 클래스가 없는 경우 미표시)
  - **error message**: 에러 메시지 (에러 메시지가 없는 경우 미표시)

> **참고**
>
> - 트랜잭션 이벤트는 1분 단위로 집계되어 처리됩니다.
> - 조건 유형이 '패턴만'인 경우, 반드시 하나 이상의 일치 또는 제외 패턴을 설정해야 합니다.
> - 패턴 매칭은 대소문자를 구분합니다.
> - 알림 메시지는 조건에 부합한 마지막 트랜잭션의 상세 정보가 포함됩니다.

애플리케이션 이벤트 설정 (클래식)

홈 화면 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** *classic*

애플리케이션의 **서버 자원 사용량(CPU, 메모리, 디스크)**, **진행 중 트랜잭션 수**, **트랜잭션 에러**, **응답 시간 증가** 등의 조건을 기반으로 경고 알림을 설정할 수 있습니다.

> 이벤트 설정 *classic* 은 기존 방식의 이벤트 설정을 계속 사용할 수 있도록 분리된 메뉴입니다.
> 더 다양한 기능은 이벤트 설정에서 이용하실 수 있습니다.

기본 옵션

에이전트 비활성 경고

설정한 시간 동안 에이전트로부터 데이터를 수신하지 못한 경우 이벤트가 발생합니다. 토글 버튼을 클릭해 기능을 켜거나 끌 수 있습니다.

- **활성화**: 일정 기간 데이터 미수신 시 에이전트 비활성 이벤트 발생
- **비활성화**: 데이터 미수신 여부과 관계없이 이벤트 발생하지 않음

> 기존의 에이전트 비활성 확인 간격 설정은 **관리 > 에이전트 목록**에서 설정할 수 있습니다.

이벤트 상태가 해소되면 추가 알림

조건이 충족되어 이벤트가 발생한 후, 상태가 정상으로 복구되면 **해소(RECOVERED) 알림**을 보낼지 여부를 선택할 수 있습니다. 토글 버튼을 클릭해 기능을 켜거나 끌 수 있습니다.

- **활성화**: 이벤트는 상태 기반으로 동작하며, 임계치 초과 시 진행중, 임계치 이하로 복구 시 해소 상태로 전환됨
- **비활성화**: 임계값을 초과할 때 마다 단발성 이벤트 발생

에이전트 중복 알림

동일한 이름의 에이전트가 동시에 2개 이상 기동될 경우 정상적인 모니터링이 어려울 수 있어 이벤트가 필수적으로 발생합니다.
단, 설정한 시간 동안은 중복 에이전트 이벤트가 발생하지 않습니다.

- 에이전트 이름을 중복으로 설정하지 않았는지 확인하려면 **이벤트 설정** 메뉴에서 관련 옵션을 검토하세요.
- 선택 가능 시간: 30초 , 1분 , 3분 , 5분 , 10분

조건

항목별로 이벤트 발생 조건을 설정할 수 있으며, 항목에 따라 입력 형식이 다릅니다.

- **서버 CPU/서버 메모리/서버 디스크** **액티브 트랜잭션/트랜잭션 에러** **트랜잭션 응답시간**

조건(%)

Warning과 Critical 두 단계의 임계값을 설정하세요.
설정값을 초과 시 경고 알림이 발생하며, Critical이 우선 적용됩니다.

- 왼쪽 값: 경고(Warning) 임계값
- 오른쪽 값: 위험(Critical) 임계값

조건(건수)

이벤트 발생 기준이 되는 건수를 입력하세요. 지정된 지속 시간 동안 설정한 건수를 초과하는 상태가 유지되면 이벤트가 발생합니다.

조건(건수)

조건을 설정하세요. 설정한 응답시간을 초과한 트랜잭션이 지정 개수 이상 발생하면 이벤트 조건 충족합니다.

조건(응답시간)

기준이 되는 응답시간을 선택하세요.

- 선택 가능 시간: 1초 , 2초 , 3초 , 5초 , 10초 , 15초 , 30초 , 1분 , 2분 , 3분 , 5분 , 10분 , 30분 , 1시간
- 예시: 건수 기준이 **10건,** 응답시간 기준이 **1초,** 지속 시간이 **5초**인 경우 → 1초를 넘는 트랜잭션이 10건 이상 발생하고 그 상태가 5초 지속되면 이벤트가 발생합니다.

지속

선택한 시간 동안 이벤트 조건이 지속될 경우 이벤트가 발생합니다.

- 선택 가능 시간: 5초 , 10초 , 15초 , 30초 , 1분 , 2분 , 3분 , 5분 , 10분 , 30분 , 1시간

무음

이벤트 발생 후, 선택한 시간 동안 동일한 조건의 이벤트가 발생하지 않습니다. 적절한 무음시간 설정으로 과도한 이벤트 발생을 방지할 수 있습니다.

- 선택 가능 시간: 1분 , 2분 , 3분 , 5분 , 10분 , 30분 , 1시간 , 2시간 , 3시간 , 6시간 , 12시간 , 1일

이벤트 활성화

토글 버튼을 클릭해 이벤트의 동작을 제어할 수 있습니다.

- **활성화**: 설정한 조건을 충족하면 이벤트가 발생됨
- **비활성화**: 조건을 검사하지 않음

이벤트 설정

다음 항목에 이벤트를 설정하여 시스템 상태를 실시간으로 모니터링 할 수 있습니다. 각 항목을 설정한 후 화면 오른쪽 상단의 [저장] 버튼을 클릭하세요.

서버 CPU / 서버 메모리 /서버 디스크

서버 자원 사용량이 설정한 임계값(%)을 초과할 경우 이벤트가 발생합니다. Warning과 Critical 단계별로 다른 임계치를 설정할 수 있습니다.

- 예: CPU 사용률이 Warning: 80%, Critical: 90%로 설정되었을 경우, 90%를 초과하면 Critical 이벤트가 발생합니다.

액티브 트랜잭션

현재 처리 중인 트랜잭션의 수를 모니터링합니다. 설정한 기준 건수를 초과하고, 이 상태가 **지속 시간** 이상 유지되면 이벤트가

발생합니다. 대량의 트랜잭션이 동시에 처리되고 있는 상황을 감지하는 데 유용합니다.

- 발생 건수 기본값: 100개 기준

트랜잭션 에러

정상적으로 종료되지 않은 트랜잭션을 에러로 판단합니다. 에러 건수가 설정한 기준 건수를 초과하고, 해당 상태가 지속되면 이벤트가 발생합니다.

- 수집 주기: 5초

트랜잭션 응답시간

각 트랜잭션이 완료되는 데 걸린 시간(응답 시간)을 기준으로 이벤트를 설정합니다. 사용자가 기대하는 응답시간을 기준으로 느려진 서비스 감지를 설정할 수 있습니다.

- 예: 응답시간 기준을 10초, 건수 기준을 10건으로 설정한 경우
  “응답시간 10초를 넘긴 트랜잭션이 10건 이상 발생하고, 이 상태가 5초간 지속되면 이벤트 발생”

이벤트 수신 설정

사용자별 이벤트 수신 설정

> **사용자별 이벤트 수신 설정** 의 다른 멤버의 정보는 **멤버 관리** 권한을 가진 멤버만 확인할 수 있습니다. 그 외 멤버에게는 필수 정보가 마스킹 처리됩니다.
>
> - 이벤트 알림의 일괄 수신설정 및 접근 설정을 위한 모바일 기기 관리는 **계정 정보**에서 할 수 있습니다.
> - 멤버 권한에 대한 자세한 설명은 다음 문서를 참고하세요.

수신 수단 설정

이메일, **SMS**, **WhatsApp**, **모바일** 알림을 제공합니다.

사용자별 이벤트 수신 설정 섹션에서 원하는 알림 수신 수단의 체크 박스를 선택하면 경고 알림을 받을 수 있습니다. 알림 수신 수단의 체크 박스를 해제하면 경고 알림을 보내지 않습니다.

> 프로젝트 최고 관리자를 제외한 모든 사용자는 자신의 수신 설정만 변경할 수 있습니다.

이메일 알림

이메일 알림은 회원 가입 시 입력한 이메일 주소로 알림을 보냅니다.

SMS 알림

SMS로 알림을 수신할 수 있습니다.

> SMS 알림 기능은 유료 프로젝트에 한정됩니다.

1. 화면 오른쪽 위의 **자신의 프로필 아이콘** > **계정 관리**로 이동하세요.
2. **계정 정보**의 **사용자 전화번호** 섹션에서 [일반 휴대전화] 버튼을 클릭하세요.
3. **전화번호**에 인증번호를 수신을 전화번호를 입력하세요.
    - SMS를 알림으로 수신할 수 있는 전화번호는 **한국의 휴대전화 번호만 등록**할 수 있습니다.
4. [저장] 버튼을 클릭하세요.

> **등록한 전화번호 변경 및 삭제**
>
> - 전화번호를 변경할 경우, [변경] 버튼을 클릭하세요. SMS 인증이 필요합니다.
> - 전화번호를 삭제할 경우, [삭제] 버튼을 클릭하세요. 전화번호 삭제 시 SMS로 수신받는 알림은 수신할 수 없습니다.

WhatsApp 알림

WhatsApp을 통해 알림을 수신할 수 있습니다.

> WhatsApp 알림 기능은 유료 프로젝트에 한정됩니다.

1. 화면 오른쪽 위의 **자신의 프로필 아이콘** > **계정 관리**로 이동하세요.
2. **계정 정보**의 **사용자 전화번호** 섹션에서 [WhatsApp] 버튼을 클릭하세요.
3. 인증번호를 받을 **전화번호**를 입력하세요.
4. [인증번호 전송] 버튼을 클릭하세요.
5. WhatsApp 애플리케이션으로 전송된 인증번호 6자리를 입력하세요.
    - 10분 내로 인증번호를 받지 못했다면, [인증번호 재전송] 버튼을 클릭하세요.
6. [인증하기] 버튼을 클릭하세요.

모바일 알림

모바일 알림 수신이 필요한 사용자는 모바일 기기에 와탭 애플리케이션을 설치한 후 로그인해야 합니다.

- 와탭 애플리케이션 다운로드 - **안드로이드**
- 와탭 애플리케이션 다운로드 - **iOS**

반복 알림(에스컬레이션) 설정

경고 알림 발생 시간으로부터 알림 발생 상황이 해소되지 않을 경우 최초 알림 발생 시각으로부터의 알림 반복 간격을 설정할 수 있습니다. 위험(Critical) 레벨의 알림만 적용됩니다.

예를 들어, 경고 알림 발생 시간으로부터 0분(즉시), 1시간 후, 1일 후에 경고 알림을 반복하려면 `0,1H,1D` 를 **반복 알림(에스컬레이션)** 컬럼 항목에 입력하세요.

1. **경고 알림** > **이벤트 수신 설정**의 **사용자별 이벤트 수신 설정** 섹션으로 이동하세요.
2. **반복 알림(에스컬레이션)** 항목에서 설정하세요.
   - **M**: 분, **H**: 시간, **D**: 일, 단위를 생략하면 분 단위로 시간을 설정합니다.
   - 숫자 또는 숫자+단위(**M**, **H**, **D**)로 입력하세요. 입력이 올바르지 않으면 메시지가 표시됩니다.
3. [저장] 버튼을 클릭하세요. 저장하지 않으면 설정되지 않습니다.

이벤트 수신 태그

이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. 이벤트 수신 태그를 설정하지 않으면 전체 멤버에게 경고 알림이 전송됩니다.

이벤트 수신 태그 추가/수정/삭제하기

> 이벤트에 적용 중인 **이벤트 수신 태그** 항목은 삭제할 수 없습니다.

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **사용자별 이벤트 수신 설정** 섹션의 사용자 목록에서 **+ 태그 추가** 또는 **+** 를 클릭하세요.
3. **이벤트 수신 태그** 창에서 **+ 새 태그 생성**을 클릭하세요.
4. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
5. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
6. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

이벤트 수신 태그 해제하기

1. **경고 알림** > **이벤트 수신 설정**의 **사용자별 이벤트 수신 설정** 섹션으로 이동하세요.
2. 이벤트 수신 태그를 해제할 사용자의 **이벤트 수신 태그** 항목의 **+** 를 클릭하세요.
3. **이벤트 수신 태그** 창에서 해제할 태그의 X 를 클릭하면 해당 태그가 해제됩니다.

수신 태그 미설정 알림 받기

이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다.

- 이벤트 수신 태그가 설정되지 않은 경고 알림을 받으려면 **수신 태그 미설정 알림 받기** 옵션을 선택하세요.
- 이벤트 수신 태그가 설정된 경고 알림만 받고 싶다면 선택을 해제하세요.

> ⓘ 모든 경고 알림을 받지 않으려면 해당 옵션을 해제하고 선택한 **이벤트 수신 태그**가 없어야 합니다.

이벤트 수신 모드 선택하기

사용자별 이벤트 수신 설정 섹션에서 멤버 또는 3rd 파티 채널별로 이벤트 알림 수신 범위를 다음 네 가지 모드 중에서 선택할 수 있습니다. 수신 모드는 각 멤버 또는 채널 행의 **이벤트 수신 태그** 열에서 라디오 버튼으로 선택하세요.

수신 모드	설명
모든 알림 받기	이벤트 수신 태그와 무관하게 모든 경고 알림 수신
선택한 태그만 받기	선택한 이벤트 수신 태그를 가진 경고 알림만 수신. 태그를 최소 한 개 이상 선택
선택한 태그만 제외하기	선택한 이벤트 수신 태그를 가진 경고 알림만 제외하고 나머지 전부 수신. 제외 태그를 최소 한 개 이상 선택
모든 알림 받지 않기	어떤 이벤트 알림도 받지 않음. 선택해 둔 태그는 삭제되지 않고 그대로 보존

- 이벤트 수신 태그가 지정되지 않은 경고 알림까지 받으려면 태그 선택 영역에서 **미분류** 가상 태그를 선택하세요. **미분류** 태그는 일반 태그와 동일하게 선택하거나 해제할 수 있으며, 태그 자체를 수정하거나 삭제할 수는 없습니다.

> ⓘ **모든 알림 받지 않기** 모드에서는 해당 멤버 또는 채널이 이벤트 알림을 전혀 받지 않습니다. 이때 선택해 둔 태그는 삭제되지 않고 보존되므로, 다시 **선택한 태그만 받기** 모드로 전환하면 이전에 선택한 태그 설정을 그대로 사용할 수 있습니다.

선택한 태그만 받기

1. **사용자별 이벤트 수신 설정** 섹션에서 설정할 멤버 또는 3rd 파티 채널 행의 **이벤트 수신 태그** 열을 클릭하세요.
2. **선택한 태그만 받기** 라디오 버튼을 선택하세요.
3. 태그 선택 영역에서 알림을 받을 태그를 하나 이상 선택하세요.

> ⓘ **선택한 태그만 받기** 모드에서 태그를 하나도 선택하지 않았을 때 나타나는 완전 차단 경고는 **선택한 태그만 제외하기** 모드에는 표시되지 않습니다. 제외 태그가 없는 상태는 알림을 전부 차단하는 상태가 아니라 전부 수신하는 상태이기 때문입니다.

선택한 태그만 제외하기

경고 알림을 대부분 받되 특정 분류만 걸러 내고 싶을 때 사용합니다. 받을 태그를 하나씩 고르는 **선택한 태그만 받기**와 방향이 반대입니다.

1. **사용자별 이벤트 수신 설정** 섹션에서 설정할 멤버 또는 3rd 파티 채널 행의 **이벤트 수신 태그** 열을 클릭하세요.
2. **선택한 태그만 제외하기** 라디오 버튼을 선택하세요.
3. 아래 태그 선택 영역에서 알림을 받지 않을 태그를 하나 이상 선택하세요.
    - 이벤트에 붙은 태그와 제외 태그가 하나라도 겹치면 해당 경고 알림은 전송되지 않습니다. 겹치는 태그가 없는 알림은 모두 수신합니다.
    - 제외 태그는 최소 한 개 이상 선택해야 합니다. 제외 태그가 하나도 없는 상태는 **모든 알림 받기**와 결과가 같아지므로, 마지막 태그는 제거할 수 없고 제외 태그가 비어 있는 동안 **최소한 하나의 태그를 선택해 주세요.** 안내가 표시됩니다.
    - 제외 태그 목록은 **선택한 태그만 받기** 모드의 수신 태그 목록과 별도로 저장됩니다. 두 모드를 번갈아 선택해도 각 모드에서 고른 태그가 그대로 남습니다.
    - **미분류** 가상 태그도 제외 대상으로 선택할 수 있으며, 태그 한 개로 계산됩니다.

알림 수신 언어 선택

사용자의 알림 수신 언어를 선택할 수 있습니다.

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **사용자별 이벤트 수신 설정** 섹션의 사용자 목록에서 언어를 설정할 사용자의 가장 왼쪽에 ▶ 버튼을 클릭하세요.
3. **알림 수신 언어**를 선택하세요.

요일 및 시간별 알림 설정

사용자의 요일별, 시간별 알림 수신 여부를 선택할 수 있습니다.

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **사용자별 이벤트 수신 설정** 섹션의 사용자 목록에서 요일 및 시간별 알림을 설정할 사용자의 가장 왼쪽에 ▶ 버튼을 클릭하세요.
3. 경고 알림 수신을 원하는 요일을 선택하거나 시간을 입력하세요.
    - 알림 수신 수단별로 설정할 수 있습니다.

3rd 파티 플러그인

외부 애플리케이션을 통해 경고 알림을 받을 수 있습니다.

- 지원 애플리케이션: Slack, Telegram, Teams, Teams Workflows, Jandi, AlertNow, Webhook, Webhook JSON, Opsgenie, PagerDuty, LINE, ilert

> ⓘ 와탭랩스의 지원 범위에 포함하지 않는 사내 메신저는 표준 Webhook, Webhook json을 통해 연동할 수 있습니다.

표 | 3rd 파티 플러그인 목록 구성

항목	설명
플러그인 이름	플러그인의 이름

항목	설명
인증 키	인증 키
인증 값	인증값
수신 레벨	경고, 전체, 위험으로 구분
반복 알림(에스컬레이션)	알림 발생 시간으로부터 알림 발생 상황이 해소되지 않았을 경우, 최초 알림 발생 시각으로부터의 알림 재발행 주기 - 위험(Critical) 단계의 알림에만 적용되는 기능
Smart 메시지(AI)	Telegram 3rd 파티 플러그인 추가 시 AI 알림 기능 제공 - 지원 언어: 한국어, 영어, 일본어
이벤트 수신 태그	프로젝트의 멤버 중 특정 멤버 또는 팀을 대상으로 알림 수신 여부
삭제	3rd 파티 플러그인 삭제

3rd 파티 플러그인 추가

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **3rd 파티 플러그인** 섹션의 [추가하기] 버튼을 클릭하세요.
3. **서드 파티 플러그인 추가** 창에서 원하는 서비스 탭으로 선택하세요.
4. 선택한 서비스의 안내(3rd 파티 플러그인 알림 추가 방법)에 따라 설정을 진행하세요.
5. [수신 테스트] 버튼을 클릭해 설정이 잘 되었는지 확인하세요.
6. 모든 과정을 완료하면 [추가하기] 또는 [등록] 버튼을 클릭하세요.

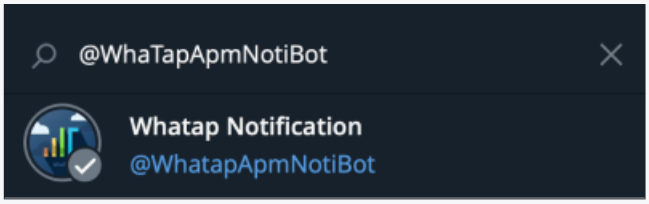
3rd 파티 플러그인 알림 추가 방법

Slack

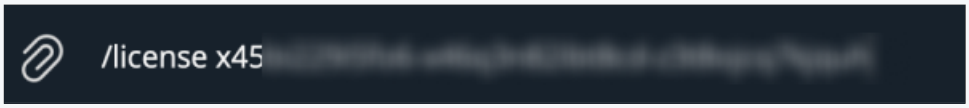
> **서드 파티 플러그인 추가**의 **Slack** 탭 화면의 [**슬랙에 추가하기**] 버튼을 클릭하면 Slack의 팀 및 채널에 **WhaTapNotiBot**을 등록할 수 있습니다.

Telegram

1. **Telegram**에서 **@WhaTapApmNotiBot**을 검색하세요.



2. Bot을 선택하고 [시작] 버튼을 클릭하세요.
3. 채팅창에 `/license` 명령어와 알림을 받을 프로젝트의 액세스 키를 채팅창에 입력하세요.

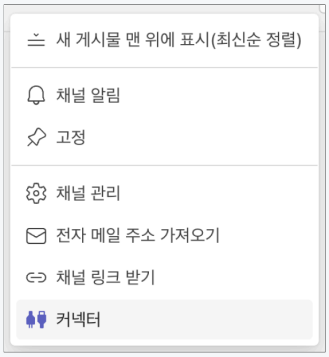


사용할 수 있는 명령어 목록

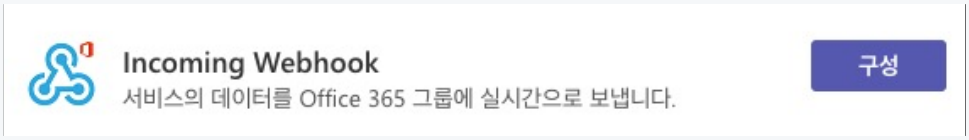
- `/help` : 명령어 목록 보기
- `/license` : 알림 받을 프로젝트의 액세스 키 입력
- `/remove` : 알림을 받지 않을 프로젝트 코드 입력
- `/list` : 현재 알림을 받고 있는 프로젝트 코드 보기

Teams

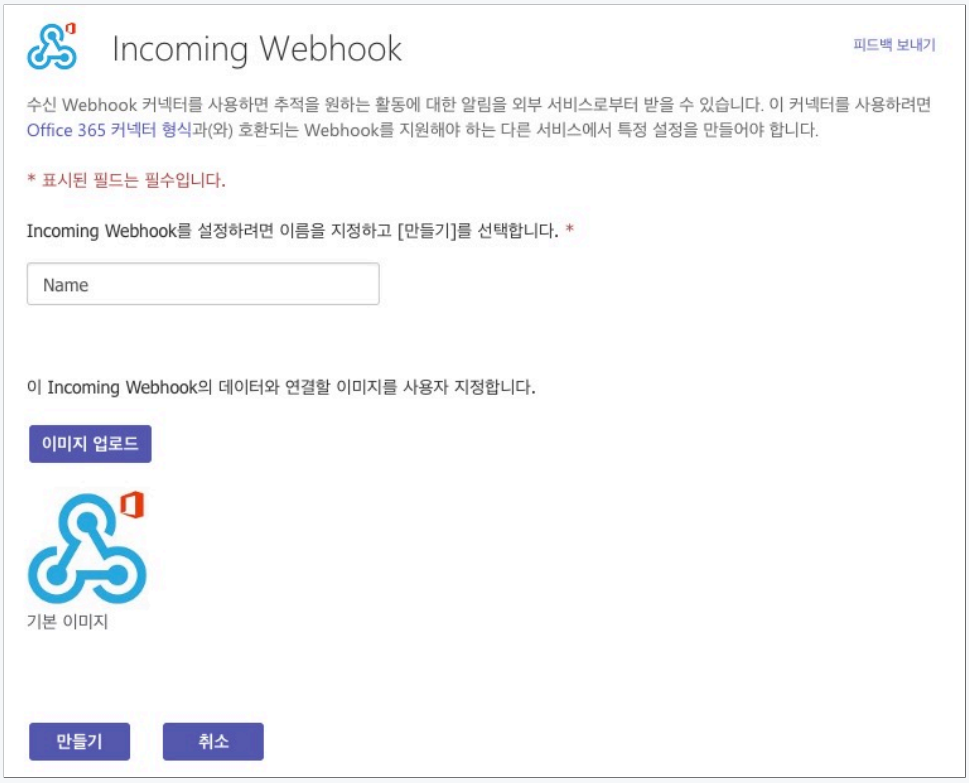
1. 알림을 받을 채널의 **커넥터**로 이동하세요.



2. **Webhook**을 추가하고 [구성] 버튼을 클릭하세요.



3. Webhook 이름을 입력하고 [만들기] 버튼을 클릭하세요.



4. 생성된 Webhhok URL을 복사하세요.

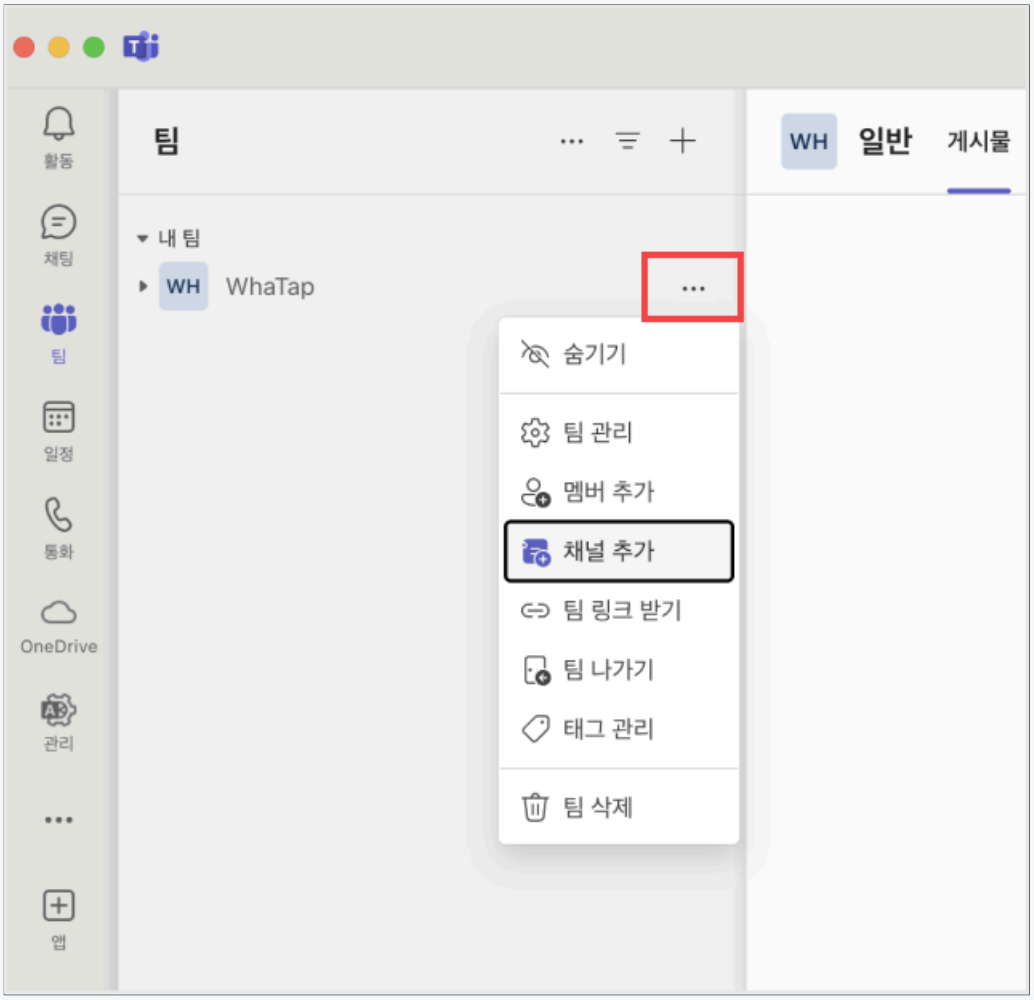


5. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **Teams** 탭 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
6. [**등록**]버튼을 클릭하세요.

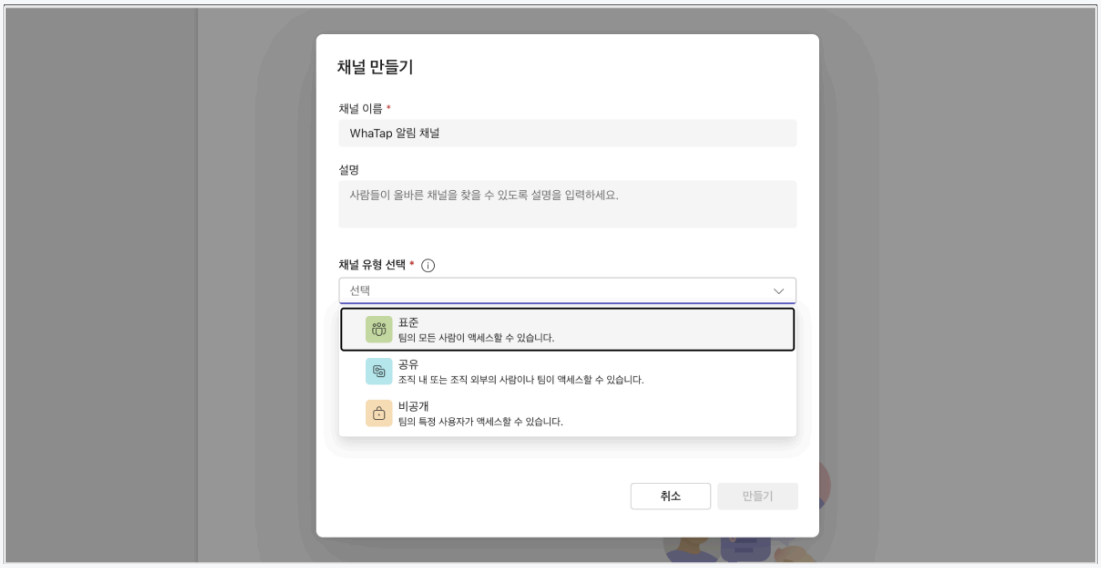
Teams Workflows

워크플로를 사용하려면 **Microsoft Teams**에 액세스할 수 있는 계정과 비즈니스 플랜 구독이 필요합니다.

1. **팀** 메뉴의 더보기(...) > **채널 추가**를 클릭하세요.

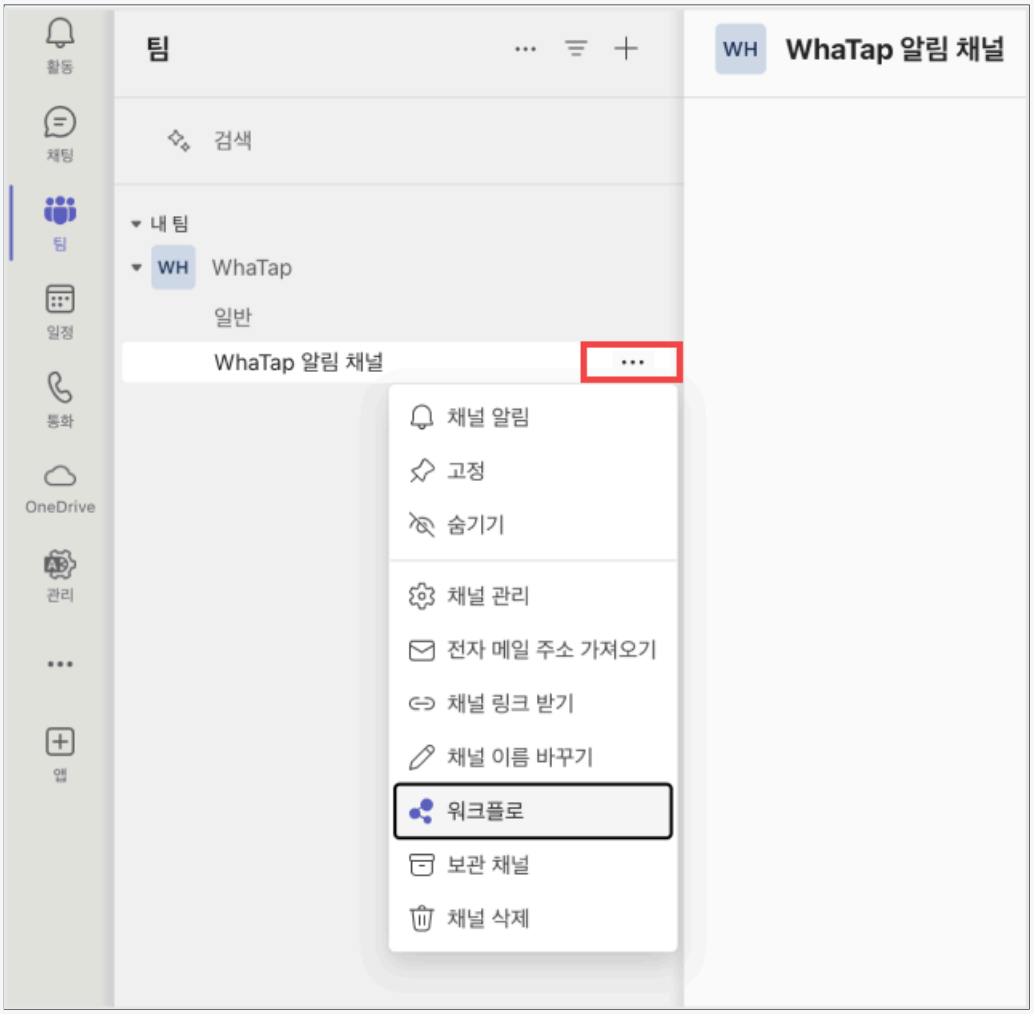


2. **채널 만들기** 창에서 **채널 이름**과 **설명**을 입력하고, **채널 유형 선택** 항목은 **표준**으로 선택한 후, [만들기] 버튼을 클릭하세요.



> 채널 유형을 **공유**로 선택하면 워크플로를 사용할 수 없으며, **비공개**를 선택하면 워크플로를 추가할 수 있지만 알림을 수신할 수 없습니다.

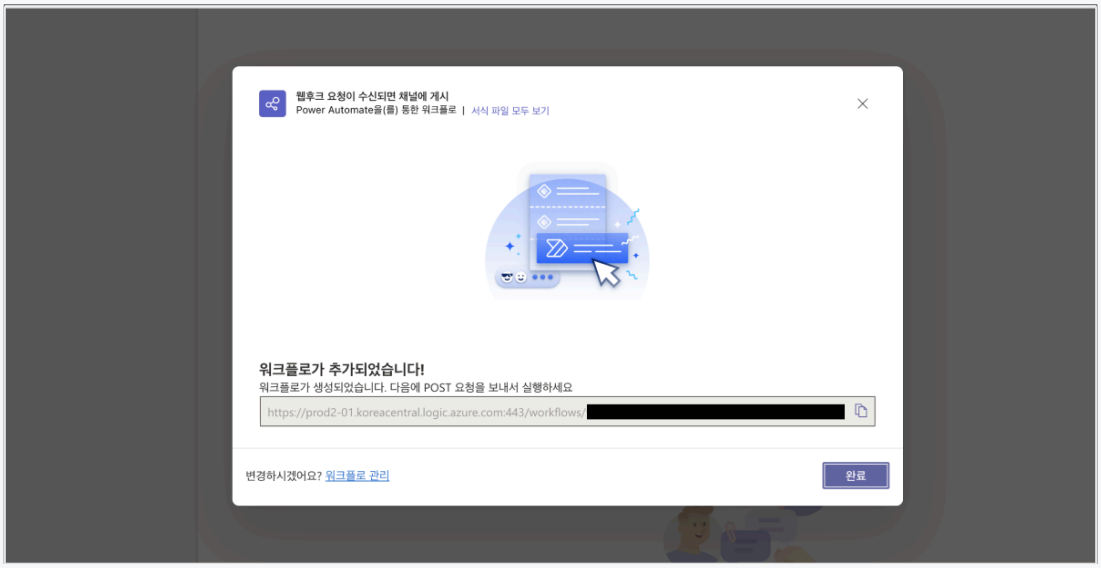
3. 생성한 채널의 더보기(...) > **워크플로**를 클릭하세요.



4. **워크플로** 대화상자의 **워크플로**의 목록에서 **웹후크 요청이 수신되면 채널에 게시**를 선택 후, [추가 워크플로] 버튼을 클릭하세요.



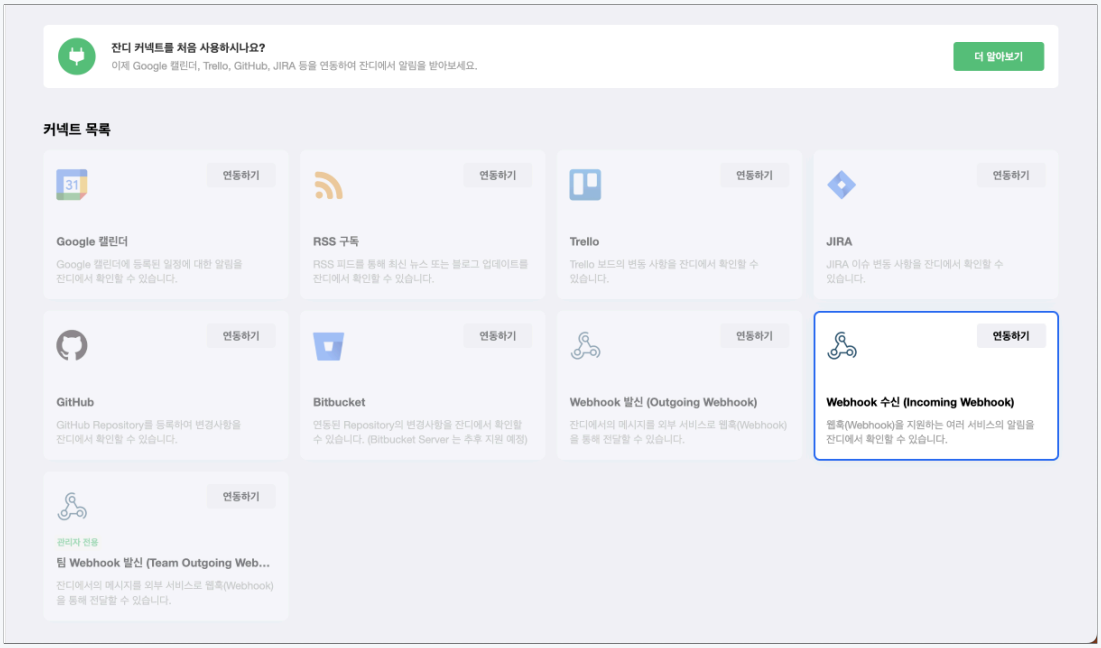
5. **이름**과 **연결** 상태를 확인한 후, [다음] 버튼을 클릭하세요.
6. **세부 정보**를 확인한 후, [워크플로 추가] 버튼을 클릭하세요.
7. 워크플로가 추가되었습니다. 생성된 주소를 복사하세요.



8. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **Teams Workflows** 탭 안내 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
9. [**등록**] 버튼을 선택하세요.

잔디(Jandi)

1. 알림을 받을 토픽 또는 채팅에서 잔디 커넥트 설정으로 이동하세요.
2. **커넥트 목록**에서 **Webhook 수신 (Incoming Webhook)** 항목에서 [연동하기] 버튼을 클릭하세요.



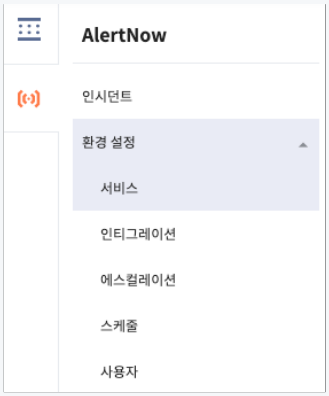
3. [Webhook 수신 추가] 버튼을 클릭하세요.
4. **Webhook URL 등록** 항목에 Webhook URL을 복사하세요.



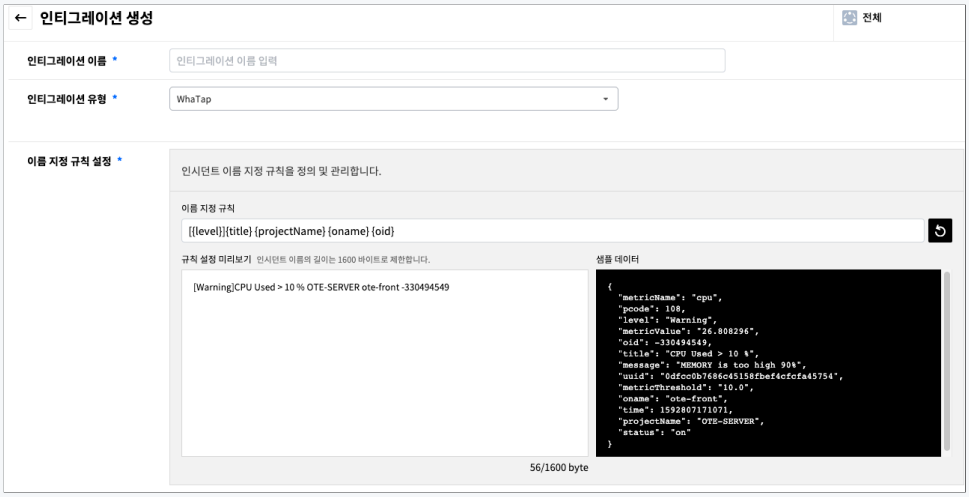
5. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **Jandi** 탭 안내 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
6. [**등록**] 버튼을 클릭하세요.

AlertNow

1. **AlerNow** 서비스를 접속한 후, **환경 설정** > **Integration** 메뉴로 이동하세요.



2. **인티그레이션** 화면에서 [인티그레이션 생성] 버튼을 클릭하세요.
3. Integration 유형 중 **WhaTap**을 선택하세요.
4. **인티그레이션 생성** 화면에서 이름 및 이름 지정 규칙을 설정하세요.



5. [확인] 버튼을 클릭하면 Integration이 생성됩니다.
6. 생성된 인티그레이션 목록을 확인하고 생성된 Integration 항목을 선택하세요.



7. 상세 화면에서 URL 오른쪽의 복사 아이콘을 클릭해 URL을 복사하세요.



8. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **AlertNow** 탭 안내 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
9. [**등록**] 버튼을 클릭하세요.

Webhook

1. 알림을 받을 Webhook을 등록하세요. 등록한 URL에 POST Method로 알림의 `title` , `text` 가 전송됩니다.

```
{
    "title":"Alert Title",
    "text": "Sample warning message"
}
```

1. **Webhook Name**과 **Webhook URL** 항목을 입력하세요.
2. [**등록**] 버튼을 클릭하세요.

Webhook JSON

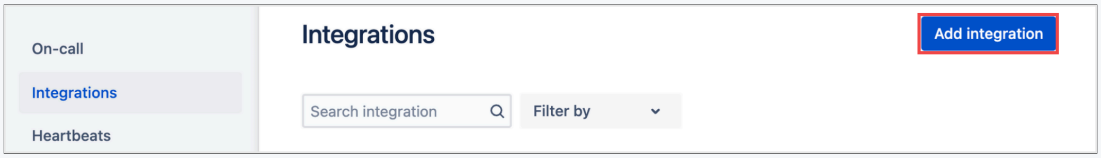
1. 알림을 받을 Webhook을 등록하세요. 등록한 URL에 POST Method로 알림 데이터가 JSON 형태로 전송됩니다. JSON 내에 포함 시킬 수 있는 대체 변수에 대해선 **사용 가능 변수** 부분을 참고하세요.
   - `${something}` 은 와탭에서 제공하는 대체될 값입니다. 지원하는 전체 목록과 의미는 사용 가능 변수를 참고하세요.
   - 이벤트 상태(`status`)는 이벤트 발생(`on`), 이벤트 해소(`off`)로 구분할 수 있습니다.
   - `level` 은 `None` , `Info` , `Warning` , `Critical` 로 구분합니다.

```
{
  "pcode": "${pcode}",
  "projectName": "${projectName}",
  "time": "${time}",
  "oid": "${oid}",
  "oname": "${oname}",
  "title": "${title}",
  "message": "${message}",
  "level": "${level}",
  "status": "${status}",
  "metricName": "${metricName}",
  "metricValue": "${metricValue}",
  "metricThreshold": "${metricThreshold}",
  "uuid": "${uuid}",
  "okind": "${okind}",
  "onode": "${onode}"
}
```

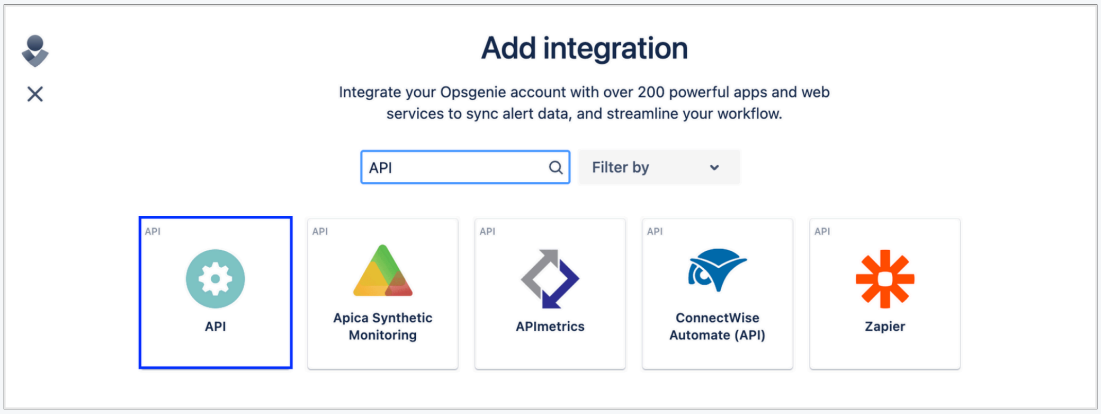
2. **Webhook Name**과 **Webhook URL** 항목을 입력하세요.
3. Header 값을 추가하려면 **사용자 Header 값 추가**의 **+ 추가**를 클릭해 key, value을 추가하세요.
4. [**등록**] 버튼을 클릭하세요.

Opsgenie

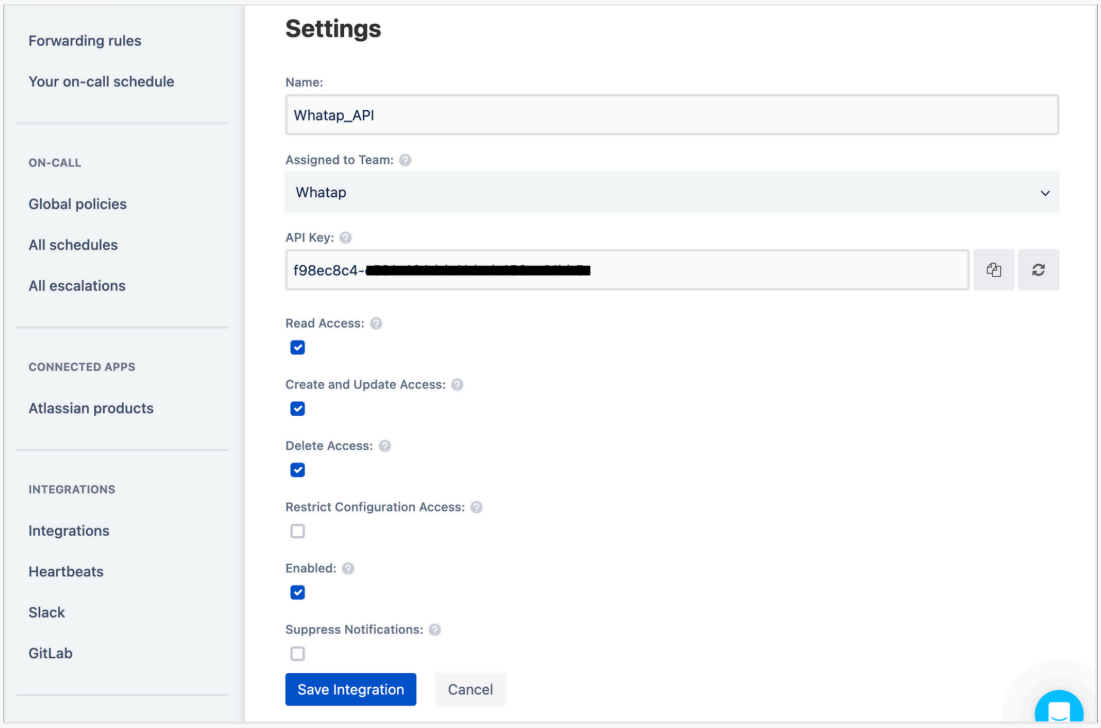
1. **Teams** 메뉴에서 알림을 수신할 팀을 선택하세요.
2. **Integration** 메뉴로 이동해 [Add Integration] 버튼을 클릭하세요.



3. Integration 목록에서 'API' 검색 후 선택하세요.



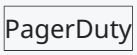
4. API 명과 Team을 할당하고 각 권한을 설정한 후, [Save Integration] 버튼을 클릭하세요.



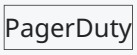
5. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와, **Opsgenie** 탭 안내 하단의 **Name**과 **API Key** 항목을 입력하세요.
6. [**등록**] 버튼을 선택하세요.

PagerDuty

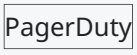
1. PagerDuty 계정에 등록하고 로그인하세요.
2. PagerDuty에서 **Services** > **Service Directory** 메뉴에서 [+ New Service] 버튼을 클릭하세요.

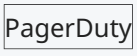
3. Service 정보(**Name**)를 입력한 후 [Next] 버튼을 클릭하세요.

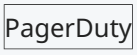
4. Escalation Policy를 설정 후 [Next] 버튼을 클릭하세요.

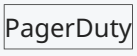
5. **Alert Grouping**과 **Transient Alerts** 항목을 설정한 후 [Next] 버튼을 클릭하세요.

6. **Integrations**에서 **Event API V2**를 선택한 후 [Create Service] 버튼을 클릭하세요.

7. 생성한 Service 화면에서 **Integration Name**과 **Integration Key** 값을 복사하세요.

8. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 화면으로 돌아와 **Integration Name**과 **Integration Key** 항목을 입력하세요.
9. [등록] 버튼을 클릭하세요.

> 메시지를 수신하지 못했거나 플러그인이 등록되지 않았다면 **Integration Name** 또는 **Integration Key**값이 올바른지 확인하세요.

LINE

1. LINE 앱을 설치합니다.
2. 아래 QR 코드를 스캔하여 **WhaTapLabs** 공식 채널을 친구로 추가합니다.



3. 다음 명령어를 사용하여 프로젝트를 라인과 연동하고 알림을 설정하세요.

명령어

```
/help # 사용 가능한 명령어 안내
/list # 연동된 프로젝트 목록 조회
/license [액세스키] # 새 프로젝트 연동 추가
/remove [프로젝트코드] # 프로젝트 연동 해제
/language [ko|ja|en] # 알림 언어 설정 (한국어/일본어/영어)
```

- 프로젝트 액세스 키는 **관리 > 프로젝트 관리**에서 확인하실 수 있습니다.
- 라인 내에서 `/language` 명령어를 사용하는 것은 라인 내 채팅방에만 적용됩니다. 다른 3rd 플러그인이나 프로젝트의 알림 수신 언어가 변경되는 것은 아닙니다.

사용 예시

```
/license XXXXXXXXXX # 프로젝트 연동
/language ko # 한국어로 언어 설정
```

```
/list # 연동된 프로젝트 확인
```

- 프로젝트 액세스 키: WhaTap 프로젝트 내 **관리 > 프로젝트 관리**에서 확인할 수 있음
- 언어 설정: `/language` 명령어는 LINE 채팅방 내에서만 적용되며, 다른 알림 채널의 언어는 변경되지 않음

ilert

1. ilert 콘솔에서 **Alert sources** 화면의 오른쪽 위의 [+ Create alert source]를 클릭하세요.
2. **Integration type 단계**의 하단에 있는 **Filter integrations 검색창**에 `WhaTap` 을 입력해 검색합니다.
3. 검색 결과에서 **WhaTap 타일**을 선택한 후 [Next] 버튼을 클릭합니다.
4. 알림 소스 이름을 입력하고 팀을 지정한 후 [Next] 버튼을 클릭합니다.
   - **에스컬레이션 정책은** 새 정책을 생성하거나 기존 정책을 선택할 수 있습니다.
5. 알림 그룹화 방식을 선택한 후 [Continue setup] 버튼을 클릭합니다.
   - 우선 선택한 **Do not group alerts는** 이후 변경할 수 있습니다.
6. 다음 화면에서 **알림 템플릿, 우선순위 등 추가 설정**을 할 수 있습니다. 우선 [Finish setup] 버튼을 클릭해 설정을 완료합니다.
7. 설정이 완료되면 완료 화면에서 **API 키** 또는 **Webhook URL**이 생성됩니다.
8. **Webhook JSON** 탭에서 생성된 **Webhook URL**을 복사해 아래의 **Webhook URL 입력란**에 붙여넣습니다.
9. [등록] 버튼을 클릭하세요.

대량 알림 발생 방지

알림이 대량으로 발생하면 지정한 시간 동안 알림이 일시적으로 중지됩니다. 예를 들어, 5분 동안 20회의 이벤트가 발생하면 5분

동안 경고 알림을 중지합니다. 설정한 **정지 시간** 시간이 지나면 대량 알림 발생 방지 기능은 해제됩니다.

1. **경고 알림** > **이벤트 수신 설정**의 **대량 알림 발생 방지** 섹션으로 이동하세요.
2. 토글 버튼을 클릭해 **활성화** 또는 **비활성화** 할 수 있습니다.
   - **활성화**: 대량 알림 일시적 중지
   - **비활성화**: 대량 알림 재개
3. **탐지 시간, 탐지 횟수, 정지 시간**을 설정하세요.
   - **탐지 시간** 동안 **탐지 횟수** 이상의 이벤트가 발생하면 **정지 시간** 동안 경고 알림을 중지합니다.

> ⓘ 대량 알림으로 차단되는 경우 차단 안내 박스가 표시됩니다. 대량 알림 차단 기능을 해제하려면 박스의 [대량 알림 발생 중지] 버튼을 클릭하세요.

이벤트 알림 포맷

공용 알림 템플릿

공통 템플릿은 대부분의 경우 적용 가능합니다. 다양한 환경에서 동일한 포멧으로 알림을 받을 수 있습니다.

- Product Type: 애플리케이션, 데이터베이스, 쿠버네티스
- Event Type: 애플리케이션 알림, 데이터베이스 알림, 메트릭스 알림
- Event Channel: sms, mobile, 3rd party plugin, plugin

이벤트 제목

Event title format

```
[Level][Platform][ProjectName][ApplicationName][EventTitle]
```

Event title example

```
[Info][JAVA][애플리케이션 프로젝트][TC-0-1-8081][CRITICAL_HIGH_MEMORY]
```

> ⓘ ApplicationName을 설정하지 않은 경우 이벤트 제목에서 생략합니다.
>
> - Platform은 다음 중 한 가지로 표시합니다.
>   - JAVA
>   - NODEJS
>   - PYTHON
>   - PHP
>   - DOTNET
>   - GO
>   - POSTGRESQL
>   - ORACLE

- MYSQL
- MSSQL
- BSM_JAVA
- CLOUDWATCH
- TIBERO
- KUBERNETES
- KUBE_NS
- URLCHECK
- URLCHECK_ADMIN
- CUBRID
- ALTIBASE
- CLUSTER
- REDIS
- MONGODB
- VR
- RUM

이벤트 메시지

이벤트 메시지에 포함할 수 있는 정보입니다. `optional` 이 `false` 인 경우 항상 메시지에 포함합니다. `optional` 이 `true` 인 경우 해당 데이터를 확인할 수 있으면 표시합니다.

Event message example

```
Project Name : 애플리케이션 프로젝트
Project Code : 3
Agent Name : TC-0-1-8081
Message : RECOVERED: Memory is too high. less than 10%
Event Time : 2022-04-12 18:53:24 +0900
Event OFF Time : 2022-04-12 18:53:24 +0900
Alert Type : APPLICATION_MEMORY
```

```
Metric Name : memory
Metric Value : 20
Metric Threshold : 10
Stateful : true
```

표 | 이벤트 메시지 구성 요소

En	Ko	지원되는 알림 타입	설명
Project Name	프로젝트 이름	전체	-
Project Code	프로젝트 Code	전체	-
Agent Name	에이전트 이름	전체(optional)	oname
Message	이벤트 메시지	전체	-
Alert Type	이벤트 종류	전체	아래의 AlertType 종류 표 참고
Event Time	이벤트 발생 시간	전체	2022-04-13 10:40:49 +0900에서 +0900는 GMT를 의미합니다.
Event Off Time	이벤트 해제 시간	전체(optional)	2022-04-13 10:40:49 +0900에서 +0900는 GMT를 의미합니다.
Metric Name	메트릭스 이름	전체(optional)	이벤트 조건 판단에서 사용하는 메트릭스의 이름
Metric Value	메트릭스 값	전체(optional)	메트릭스 값이 메트릭스 임계치를 넘으면 이벤트 발생 조건이 만족한 경우입니다.
Metric Threshold	메트릭스 임계치	전체(optional)	메트릭스 값이 메트릭스 임계치를 넘으면 이벤트 발생 조건이 만족한 경우입니다.
Stateful	해소된 이벤트 알림	전체(optional)	해소된 이벤트 알림 기능 사용 중이면 true, 아니면 false
Event Rule	이벤트 발생 조건	메트릭스 알림	-

En	Ko	지원되는 알림 타입	설명
Target Filter	이벤트 대상 선택	메트릭스 알림	특정 대상에서 수집된 메트릭스에 대해서만 이벤트 조건을 확인합니다.
Repeat Count	이벤트 반복 횟수	메트릭스 알림	이벤트 조건이 이벤트 반복 시간동안 이벤트 반복 횟수만큼 만족해야 이벤트가 발생됩니다.
Repeat Duration	이벤트 반복 시간	메트릭스 알림	이벤트 조건이 이벤트 반복 시간동안 이벤트 반복 횟수만큼 만족해야 이벤트가 발생됩니다.
Receiver	수신자	메트릭스 알림	-
Query	MXQL 쿼리	복합 메트릭스 알림	-
Rule	이벤트 발생 조건	복합 메트릭스 알림	-
Query Period	쿼리 기간	복합 메트릭스 알림	-
Query Interval	쿼리 간격	복합 메트릭스 알림	-
Silent Time	무음 시간	복합 메트릭스 알림	-
Query	URL	Exception 알림	Exception을 발생시킨 요청의 URL
TXID	트랜잭션 ID	Exception 알림	-
Class	에러 클래스 이름	Exception 알림	-

En	Ko	지원되는 알림 타입	설명
Log Message	로그 메시지	서버 - 파일 로그 알림	-
Log File	로그 파일 경로	서버 - 파일 로그 알림	-
IP	IP	서버 알림 전체	-
CPU	CPU	서버 알림 전체	이벤트 발생 당시의 Snapshot
CPU_load1	CPU_load1	서버 알림 전체	이벤트 발생 당시의 Snapshot
CPU_loadPerCore	CPU_loadPerCore	서버 알림 전체	이벤트 발생 당시의 Snapshot
Memory	Memory	서버 알림 전체	이벤트 발생 당시의 Snapshot
Swap	Swap	서버 알림 전체	이벤트 발생 당시의 Snapshot
Disk Name | Used Percent | Free Size | IO Percent	디스크 퍼포먼스	서버 알림 전체	이벤트 발생 당시의 Snapshot
Name | Bps | Pps	트래픽 퍼포먼스	서버 알림 전체	이벤트 발생 당시의 Snapshot
Message | Time | Name	처리내역 메시지	서버 알림 전체	-

ⓘ 해당 이벤트에서 제공할 수 있는 최대한 많은 정보를 보여줍니다.

- AlertType은 다음 중 한 가지로 표시합니다.

AlertType	설명
APPLICATION_CPU	애플리케이션 CPU 알림
APPLICATION_MEMORY	애플리케이션 MEMORY 알림

AlertType	설명
APPLICATION_DISK	애플리케이션 DISK 알림
APPLICATION_ACTIVE_TRANSACTION	애플리케이션 액티브 트랜잭션 알림
APPLICATION_ERROR_TRANSACTION	애플리케이션 에러 트랜잭션 알림
APPLICATION_SLOW_TRANSACTION	애플리케이션 트랜잭션 응답시간 알림
METRICS	메트릭스 알림
COMPOSITE_METRICS	복합 메트릭스 알림
ANOMALY	이상치 탐지 알림
LOG_REALTIME	로그 실시간 알림
COMPOSITE_LOG	복합 로그 알림
SERVER_REBOOT	서버 - 재시작 알림
SERVER_NO_DATA	서버 - 미수신 알림
SERVER_PORT	서버 - 포트 알림
SERVER_NETWORK_IOPS	서버 - 네트워크 IOPS 알림
SERVER_NETWORK_BPS	서버 - 네트워크 BPS 알림
SERVER_DISK_IO	서버 - 디스크 I/O 알림
SERVER_DISK_QUOTA	서버 - 디스크 사용량 알림
SERVER_DISK_INODE	서버 - inode 알림

AlertType	설명
SERVER_CPU	서버 - CPU 알림
SERVER_MEMORY	서버 - 메모리 알림
SERVER_CPU_STEAL	서버 - steal 알림
SERVER_MEMORY_SWAP	서버 - 스왑 알림
SERVER_LOG_FILE	서버 - 로그 파일 알림
SERVER_WINDOW_EVENT	서버 - 윈도우 이벤트 알림
SERVER_OFF	서버 - 알림 OFF 알림
SERVER_ACKNOWLEDGE	서버 - 처리내역 알림
SERVER_PROCESS_COUNT	서버 - 프로세스 수 알림
SERVER_PROCESS_CPU	서버 - 프로세스 CPU 알림
SERVER_PROCESS_MEMORY	서버 - 프로세스 메모리 알림
SERVER_PROCESS_OFF	서버 - 프로세스 알림 OFF 알림
AGENT_ACTIVE	에이전트 활성화 알림
AGENT_INACTIVE	에이전트 비활성화 알림
AGENT_REACTIVATED	에이전트 재활성화 알림
URL	URL 알림
TOO_MANY_EVENT	너무 많은 이벤트 발생 알림

AlertType	설명
CLOUD_WATCH	Cloud Watch 알림
EXCEPTION	Exception 알림

애플리케이션 경고 알림

애플리케이션 알림은 Event Title, Event Message 모두 제공합니다. **이벤트 상태가 해소되면 추가 알림** 기능을 사용하는 경우 이벤트 발생 조건이 해제되면 Evnet Off Message가 전송됩니다.

Event Type	Event Level	Event Title	Event Message	Event Off Message
서버 CPU	Warning	HIGH_CPU	CPU is high. `${value}% (>= ${threshold}%)`	RECOVERED: CPU is high. less than `${threshold}%`
서버 CPU	Critical	CRITICAL_HIGH_CPU	CPU is too high. `${value}% (>= ${threshold}%)`	RECOVERED: CPU is too high. less than `${threshold}%`
서버 메모리	Warning	HIGH_MEMORY	Memory is high. `${value}% (>= ${threshold}%)`	RECOVERED: Memory is high. less than `${threshold}%`
서버 메모리	Critical	CRITICAL_HIGH_MEMORY	Memory is too high `${value}% (>= ${threshold}%)`	RECOVERED: Memory is too high. less than `${threshold}%`
서버 디스크	Warning	HIGH_DISK	Disk is high `${value}% (>= ${threshold}%)`	RECOVERED: Disk id high. less than `${threshold}%`
서버	Critical	CRITICAL_HIGH_DISK	Disk is too high	RECOVERED: Disk is too highf.

Event Type	Event Level	Event Title	Event Message	Event Off Message
디스크			`${value}% (>= ${threshold}%)`	less than `${threshold}%`
정상 트랜잭션	Warning	HIGH_ACTIVE_TRANSACTION	Active Transaction Count is over `${value} (>= ${threshold})`	RECOVERED: Active Transaction Count is less than `${threshold}`
에러 트랜잭션	Warning	HIGH_ERROR_TRANSACTION	Error Transaction Count is over `${threshold} (${value})`	RECOVERED: Error Transaction Count is less than `${threshold}`
느린 트랜잭션	Warning	TOO_MANY_SLOW_TX	Too many delayed transactions `(${value}, above ${time} ms)`	RECOVERED: Too many delayed transactions. less than `${threshold}`

이벤트 기록 및 모니터링

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 기록**

이벤트 기록은 프로젝트에서 발생한 여러 이벤트를 조회하고 상세 정보를 분석할 수 있습니다. Elasticsearch DSL 쿼리를 지원하며, 원하는 조건에 맞는 이벤트를 정확하고 빠르게 찾을 수 있습니다.

이벤트 기록 조회하기

검색 결과에서 **이벤트 제목을 클릭**하면 해당 이벤트가 발생한 시간대의 상세 데이터를 분석할 수 있는 화면으로 이동합니다. 이를 통해 이벤트 발생 전후 상황을 종합적으로 파악할 수 있습니다.

1. **경고 알림** > **이벤트 기록**으로 이동하세요.
2. **이벤트 기록** 화면의 **시간 선택**에서 이벤트 기록을 조회할 기간를 설정하세요.
3. 필터 입력 창에 쿼리을 입력하고 [돋보기] 버튼을 클릭하세요.
    - 쿼리를 입력하지 않으면 전체 이벤트가 조회됩니다.

컬럼 설정

원하는 컬럼을 표시하거나 숨길 수 있습니다.

1. **이벤트 기록** 화면의 왼쪽 위 [컬럼 설정] 버튼을 클릭하세요.
2. **컬럼 설정** 창에서 표시하거나 숨길 컬럼을 선택하세요.
    - **표시 항목**에서 컬럼의 순서를 변경할 수 있습니다.
    - **컬럼명 검색**창에 원하는 컬럼 항목을 검색할 수 있습니다.

컬럼 항목	설명
이벤트 이름	이벤트의 제목
메시지	이벤트의 메시지 또는 스냅샷 정보

컬럼 항목	설명
이벤트 상태	이벤트의 현재 상태
에이전트 명	이벤트가 발생한 대상 에이전트의 이름(Oname) - 특정 에이전트에 대해 발생한 이벤트가 아닌 경우 빈칸으로 표시됨
에이전트 그룹	이벤트가 발생한 대상 에이전트의 종류(OkindName) - 특정 에이전트에 대해 발생한 이벤트가 아닌 경우 빈칸으로 표시됨
에이전트 노드	이벤트가 발생한 대상 에이전트의 서버(OnodeName) - 특정 에이전트에 대해 발생한 이벤트가 아닌 경우 빈칸으로 표시됨
이벤트 발생 시각	이벤트가 발생한 시간
이벤트 해소 시각	이벤트가 해소된 시간 상태가 없는 이벤트의 경우 `-`로 표시됨

3. 설정을 완료한 후, [확인] 버튼을 클릭해 컬럼 설정을 반영합니다.

CSV 다운로드

조회 결과를 CSV 파일 형식으로 다운로드할 수 있습니다.

1. **경고 알림** > **이벤트 기록**에서 [CSV] 버튼을 클릭하세요.
2. 현재 필터 조건에 맞는 이벤트 목록이 CSV 파일로 저장됩니다.

진행 중인 이벤트만 보기

현재 진행 중인 상태의 이벤트만 조회할 수 있습니다.

1. **경고 알림** > **이벤트 기록**에서 [진행 중인 이벤트만 보기] 버튼을 클릭하세요.
2. 필터 쿼리에 다음 조건이 자동으로 추가되어 검색됩니다.

```
Stateful: true and Status : "ON"
```

이벤트 검색

- **정확한 검색**: 원하는 조건을 정확히 지정하여 필요한 이벤트만 조회
- **복합 조건**: 여러 필드를 동시에 검색하여 복잡한 조건도 한 번에 처리
- **유연한 패턴**: 와일드카드, 부분 검색 등 다양한 검색 방식 지원
- **빠른 성능**: 인덱싱된 데이터를 활용한 고속 검색

검색 가능한 필드

필드와 값에 대해 대소문자를 구분하지 않습니다.

- **필드명 태그**: 필드가 지원되는 이벤트 타입
- **필드 타입**: 필드의 데이터의 타입
  - `#`: Number
  - `T`: String
  - : Boolean
  - : Date

이벤트 정보 필드

필드명	타입	설명	예시 값
Title	String	이벤트 제목	"Database Connection Error"
OffTitle	String	이벤트 해소 제목	"RECOVERED: Database Connection Error"
Message	String	이벤트 메시지	"Connection timeout occurred"
OffMessage	String	이벤트 해소 메시지	"RECOVERED: Connection timeout occurred"
Level	String	현재 이벤트 레벨	`Critical`, `Warning`, `Info`

필드명	타입	설명	예시 값
OriginLevel	String	원본 이벤트 레벨	`Critical`, `Warning`, `Info`
Status	String	이벤트 상태	`ON`, `OFF`, `CANCEL`, `ACKNOWLEDGE`, `MAINTENANCE`, `DISABLED`
ActCount	Number	처리내역 개수	2
MetricName	String	지표 이름	"memory"
MetricValue	String	지표 값	"85.5"
OffValue	String	해소 값	"72.8"
MetricThreshold	String	임계치	"80"
alertType	String	이벤트 종류	"BASIC", "METRICS", "TRANSACTION" 등
alertId	String	이벤트 규칙의 고유식별자	"zf3uojer0fv4v7"

이벤트 타입별 지원 필드

> ⓘ 특정 이벤트 타입에서만 지원되는 필드입니다. 해당 이벤트 타입이 아닌 경우 값이 없을 수 있습니다.

필드명	타입	이벤트 종류	설명	예시 값
eventRule	String	기본, 메트릭스	이벤트 발생 규칙	"memory ≥ 80"
field	String	실시간 로그	로그 검색 키	"content"

필드명	타입	이벤트 종류	설명	예시 값
keyword	String	실시간 로그	로그 검색 값	"Error"
logCategory	String	실시간 로그	로그 카테고리	"AppLog"
logContent	String	실시간 로그	로그 내용	"00:00000:00009:2025/04/06 23:03:52.55 server Error: 1601, Severity: 17, State: 3\n00:00000:00009:2025/04/06 23:03:52.55 server There are not enough 'user connections' available to start a new process."

대상 필드

필드명	타입	설명	예시 값
Oid	Number	에이전트 고유식별자	-98765432
Oname	String	에이전트 이름	"web-server-01"
Okind	Number	에이전트 종류 고유식별자	867318026
OkindName	String	에이전트 종류명	"web-server"
Onode	Number	에이전트 서버 고유식별자	334634079
OnodeName	String	에이전트 서버명	"production-node-1"

이벤트 고유값 필드

필드명	타입	설명	예시 값
Id	Number	이벤트의 고유식별자	5768121

필드명	타입	설명	예시 값
UUID	String	이벤트 고유식별자	"550e8400-e29b-41d4-a716"

상태/플래그 필드

필드명	타입	설명	예시 값
Stateful	Boolean	상태 기반 이벤트 여부	true, false
Disabled	Boolean	종료된 이벤트 여부	true, false
Escalation	Boolean	에스컬레이션 여부	true, false
SystemEvent	Boolean	시스템 이벤트 여부	true, false

검색 유형별 문법

> ⓘ 검색 기본 문법은 WhaTap log search query 문법 문서를 참고하시기 바랍니다.

1. 키워드 검색

Database Connection 문자열이 포함된 이벤트

```
title: "Database Connection"
```

2. 여러 값 검색

여러 옵션 중 하나라도 포함되는 이벤트를 조회합니다.

```
title: Database Connection
```

현재 이벤트 레벨이 Info이거나 Warning인 이벤트

```
level: info warning
```

3. 패턴 검색

와일드카드를 사용하여 패턴을 검색할 수 있습니다.

이름 패턴으로 특정 에이전트에서 발생된 이벤트 찾기

```
oname: web-*-prod
```

제목이 Connection으로 끝나는 이벤트 찾기

```
title: *Connection
```

4. 복합 조건 검색

여러 조건을 조합하여 정확히 검색할 수 있습니다.

Critical 레벨이면서 제목에 Database가 포함된 이벤트

```
level: critical and title: Database
```

특정 에이전트의 Warning 또는 Critical 이벤트

```
oname: web-server-01 and level: Warning Critical
```

5. OR 조건 검색

여러 조건 중 하나라도 만족하는 이벤트를 조회합니다.

제목 또는 메시지에 특정 키워드가 포함된 이벤트

```
title: Connection or message: Connection
```

6. 제외 조건

특정 조건을 제외해 검색할 수 있습니다.

Info 레벨을 제외한 모든 이벤트

```
not level: info
```

실시간 알림

실시간 알림 기능은 **이벤트 설정**을 통해 설정한 이벤트가 발생할 때 팝업 형태 표시됩니다. 이벤트 발생 즉시 알림을 받아 문제를 빠르게 인지하고 대응할 수 있습니다. 이미 발생했거나 놓친 알림 내역을 확인하여 실시간 대응이 어려웠던 상황을 파악하고 해결할 수 있습니다.

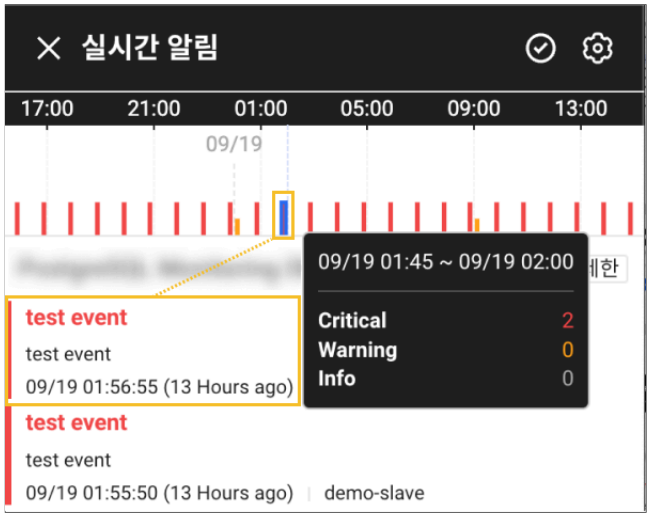
실시간 알림 확인하기

실시간 알림은 화면 오른쪽 위 팝업 형태로 표시됩니다. 알림이 발생한 시간, 제목, 메시지를 확인할 수 있습니다.

- 개별 알림을 닫으려면 메시지 오른쪽 상단의 ✕ 버튼을 클릭하세요.
- 화면에 표시된 모든 팝업 알림을 닫으려면 **전체 알림 닫기**을 클릭하세요.

알림 내역 확인하기

1. 프로젝트를 선택해 진입한 화면의 가장 위에 🔔 아이콘을 클릭하세요.
2. **실시간 알림** 창이 나타납니다.
    - 현재 프로젝트의 최근 24시간 동안의 알림을 최대 100개까지 확인할 수 있습니다.
    - 개별 이벤트 항목을 선택하면 관련 메뉴로 이동합니다.
    - **실시간 알림** 창의 시간별 차트에서 원하는 시간대를 선택하면 해당 이벤트 항목으로 이동할 수 있습니다.



- 전체 알림 내역을 확인하려면 **전체 알림 기록** 버튼을 선택하세요. **경고 알림** > **이벤트 기록** 메뉴로 이동합니다.

> **실시간 알림** 목록이 비어 있을 경우, [전체 알림 기록] 버튼이 표시됩니다. **수신 태그 미설정 알림 받기** 옵션을 해제하세요.

이벤트 일괄 해소하기

실시간 알림 목록에서 진행 중인 모든 이벤트를 일괄 해소합니다. 이벤트 해소 작업 완료까지 약 30초에서 1분 정도 소요됩니다.

- **이벤트 일괄 처리** 기능을 사용하면 현재 진행 중인 이벤트에 대한 해소 알림이 다량 발생할 수 있습니다.
- **이벤트 일괄 처리** 기능을 사용한 후에도 이벤트 규칙이 활성화되어 있고, 데이터가 임계치를 초과했다면 이벤트는 다시 발생할 수 있습니다.
- 이벤트 해소 알림은 **경고 알림** > **이벤트 설정** 메뉴에서 이벤트 수정을 통해 비활성화할 수 있습니다. **이벤트 상태가 해소되면 추가 알림** 또는 **해소된 알림** 옵션을 비활성화하세요.

사용 방법:

1. 프로젝트를 선택해 진입한 화면 상단의 아이콘을 클릭하세요.
2. **실시간 알림** 창이 나타나면 오른쪽 상단의 아이콘을 클릭하세요.

3. `실시간 알림 목록에서 진행중인 이벤트를 일괄 해소합니다.` 메시지 창이 나타나면 [해소] 버튼을 클릭하세요.

실시간 알림 설정하기

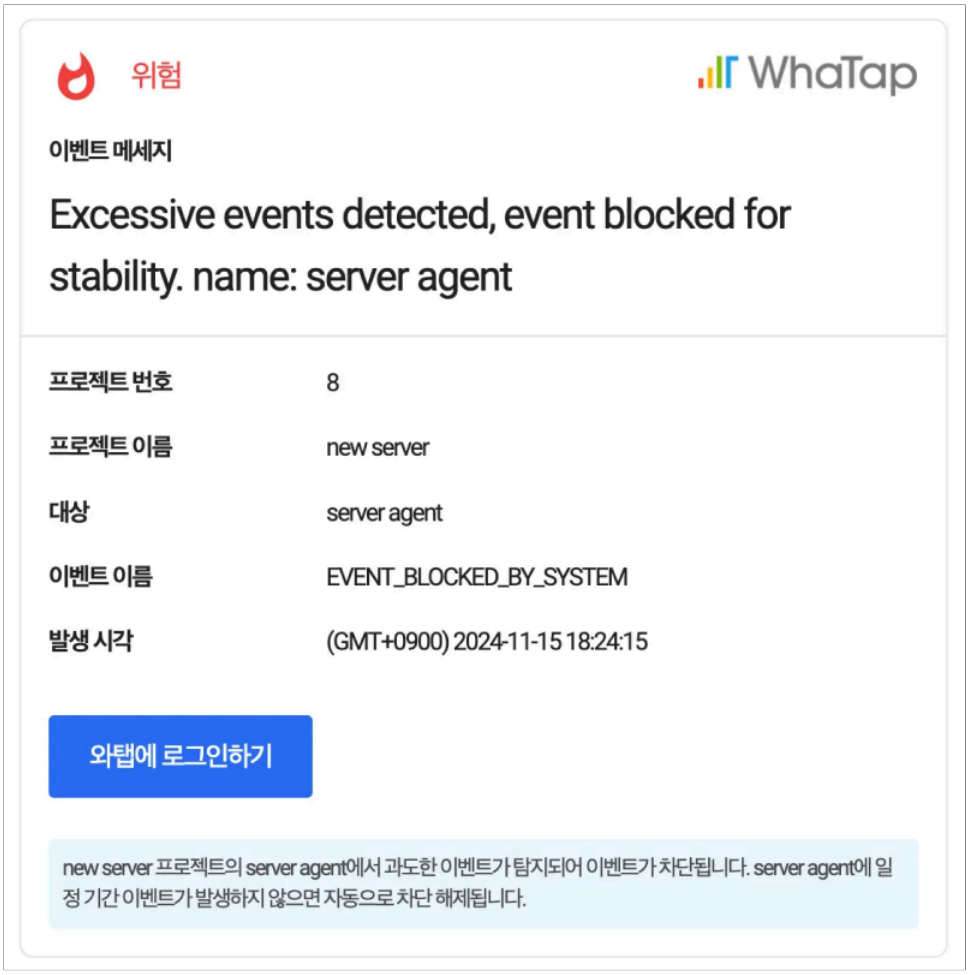
1. 프로젝트를 선택해 진입한 화면 상단의 아이콘을 클릭하세요.
2. **실시간 알림** 창이 나타나면 오른쪽 상단의 아이콘을 클릭하세요.
3. 실시간 알림 설정 창에서 각 항목을 설정하세요.
   - **팝업 알림**: **실시간 알림** 창이 닫힌 상태에서도 알림을 팝업 형태로 제공합니다.
   - **해소된 이벤트 보기**: **실시간 알림** 목록에서 해소된 이벤트를 포함해 확인할 수 있습니다.
   - **알림음 미리 듣기**: 브라우저에서 발생하는 알림음을 미리 들을 수 있습니다. Critical 또는 Warning 옆의 버튼을 클릭하세요. 알림음이 들리지 않으면 브라우저 설정을 확인하시기 바랍니다.

시스템 이벤트

와탭 모니터링은 플랫폼 전반에서 자동으로 탐지 및 생성되는 공용 이벤트를 제공합니다. 이 문서는 시스템에 의해 발생하는 이벤트의 유형 및 발생 조건, 대응 방법에 대해 안내합니다.

EVENT_BLOCKED_BY_SYSTEM

프로젝트에서 짧은 시간에 과도한 이벤트가 발생하면, 와탭 시스템이 이를 탐지하여 이벤트를 차단하고 다음 메시지를 사용자에게 전달합니다. 이벤트가 차단되었다면 사용자의 이벤트 설정을 점검하세요.



차단 기준

이벤트를 차단하는 기준은 다음과 같습니다.

1. **에이전트 단위 차단**

   한 개의 에이전트에서 **1분** 내에 **10개**의 이벤트가 발생하면 해당 에이전트에서 발생하는 이벤트를 차단합니다.

2. **다중 에이전트 차단 시 전체 차단**

   1의 경우로 이벤트가 차단된 에이전트가 **1분** 내에 **3개** 이상이 되는 경우 프로젝트 전체에서 발생하는 이벤트를 차단합니다.

3. **프로젝트 단위 차단**

   한 개의 프로젝트에서 **1분** 내에 **100개**의 이벤트가 발생하면 프로젝트 전체에서 발생하는 이벤트를 차단합니다.

해제 방법

시스템에 의해 이벤트가 차단된 경우, 1분 동안 대상 에이전트 또는 프로젝트에서 이벤트가 발생하지 않아야 합니다.

- **경고 알림** > **이벤트 설정** 메뉴에서 설정한 이벤트를 모두 비활성화하세요. **이벤트 설정을 점검하여 과도한 이벤트가 발생하지 않도록 임계치를 조정해야 합니다**.
- 이벤트를 설정하기 전 **시뮬레이션** 기능을 활용해 이벤트 발생 횟수를 예측해볼 수 있습니다.

인스턴스 성능 분석

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리**

애플리케이션의 환경을 확인하고 성능과 관련한 설정을 확인할 수 있습니다. 에이전트 및 애플리케이션과 관련한 상세 정보를 확인할 수 있습니다.

- 오른쪽 위에 텍스트 입력란을 통해 원하는 항목을 필터링할 수 있습니다.
- 모니터링 대상 서버에 위치한 에이전트 및 애플리케이션의 변경 사항이 자동 반영되지 않는다면 **새로고침** 버튼을 선택하세요.

에이전트 목록

화면 왼쪽에 애플리케이션 목록에서는 프로젝트에 할당된 에이전트 목록을 확인할 수 있습니다. 개별 에이전트 항목을 선택하면 오른쪽 화면에 설정된 환경 변수 및 에이전트 설정, 성능과 관련한 정보를 조회할 수 있습니다.

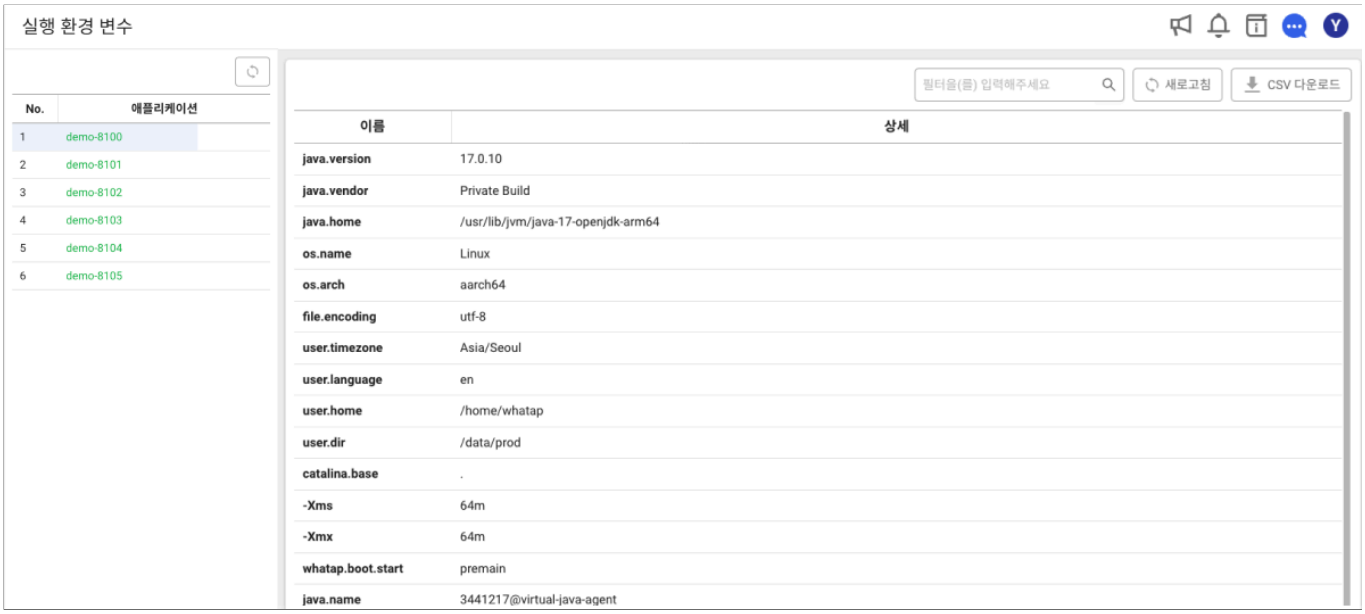


No.	애플리케이션
1	demo-8100
2	demo-8101
3	demo-8102
4	demo-8103
5	demo-8104
6	demo-8105

에이전트 목록을 갱신하거나 비활성화된 에이전트를 재기동한 다음 목록에 자동으로 표시되지 않는다면 버튼을 선택하세요.

실행 환경 변수

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **실행 환경 변수**

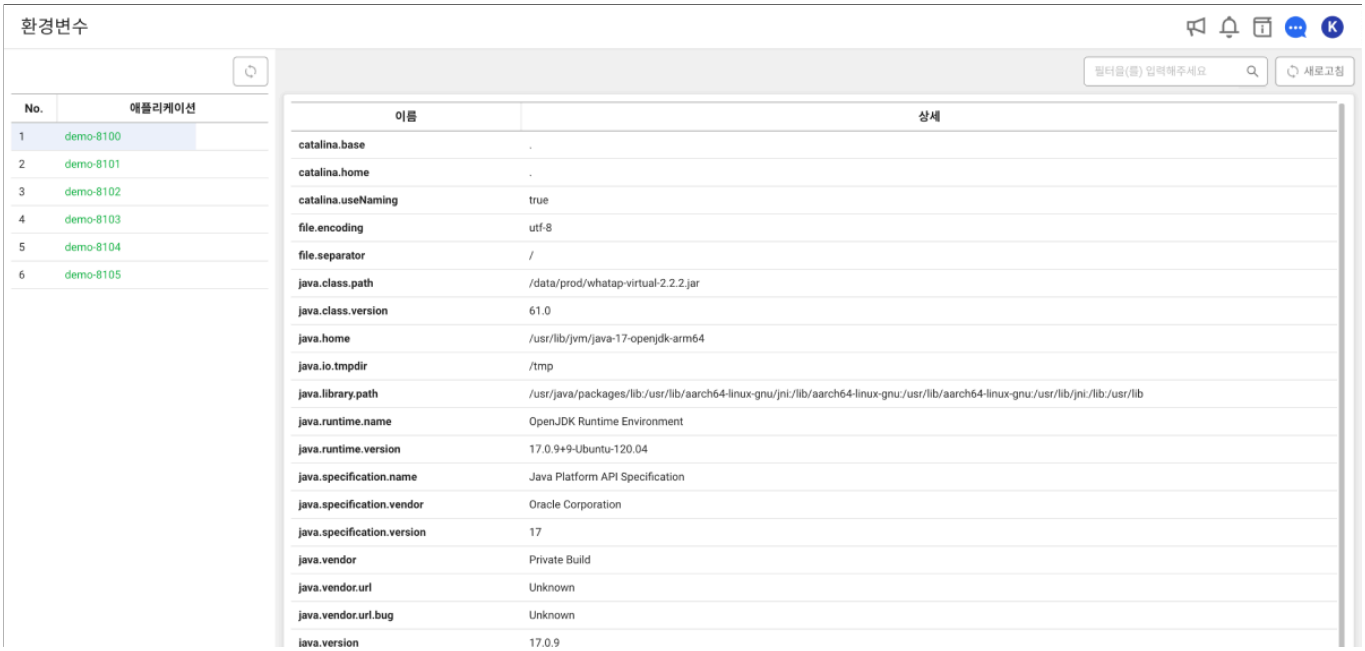


에이전트 실행과 관련한 환경 변수를 조회할 수 있습니다. 환경 변수 중 중요한 정보 또는 애플리케이션의 상태 정보를 와탭 서버에 저장해 에이전트가 다운되더라도 조회가 가능합니다. 에이전트 버전 및 설치 경로, 이름, IP 주소 등을 확인할 수 있습니다. **CSV 다운로드** 버튼 클릭 시 환경 변수 정보를 다운로드할 수 있습니다. 주요 확인 사항은 다음과 같습니다.

- `(애플리케이션).version` : 애플리케이션 버전을 확인할 수 있습니다.
- `file.encoding` : `utf-8` 과 같은 파일 인코딩 형식을 확인할 수 있습니다.
- `os.name` : OS를 확인할 수 있습니다.
- `user.language` : 사용자 언어를 확인할 수 있습니다.
- `-Xms` , `-Xms` : 힙 메모리 설정 관련 환경 변수를 확인할 수 있습니다.
- `(애플리케이션).start` : 애플리케이션 시작 시간을 확인할 수 있습니다.

환경 변수

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **환경변수**

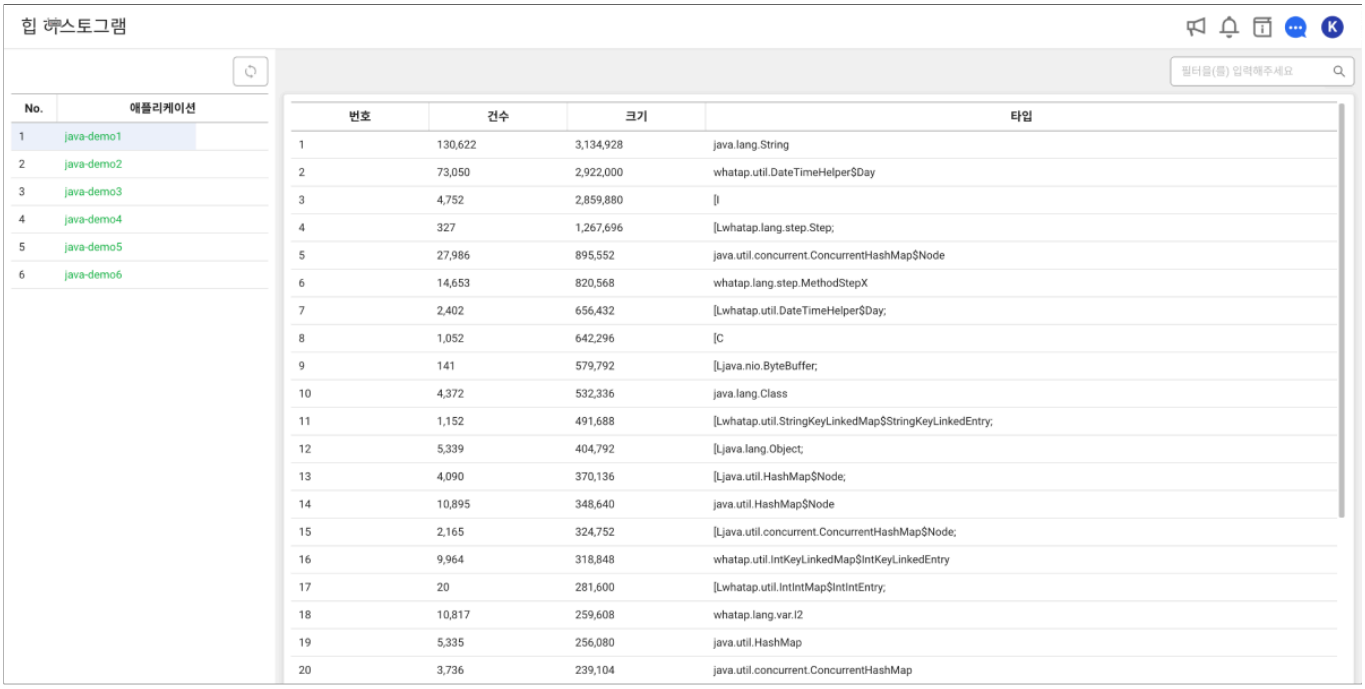


애플리케이션 실행과 관련한 환경 변수 정보를 조회할 수 있습니다. **환경 변수** 메뉴는 조회 시점의 환경 변수 정보를 수집해 제공합니다. `System.getProperty()` 를 사용해 매개변수로 전달된 키의 값을 조회합니다. 주요 확인 사항은 다음과 같습니다.

- `(애플리케이션).version` : 애플리케이션 버전을 확인할 수 있습니다.
- `file.encoding` : `utf-8` 과 같은 파일 인코딩 형식을 확인할 수 있습니다.
- 애플리케이션 상태 정보: 예를 들어 `catalina.home` 환경 변수가 조회될 경우 tomcat 환경임을 파악할 수 있습니다.
- OS 환경 변수: 예를 들어 `(env) LANG` 환경 변수와 같이 접미사 `(env)` 형식으로 제공됩니다.

힙 히스토그램

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **힙 히스토그램**



JVM(자바 가상 머신)의 메모리에 올라가 있는 Heap 점유 객체 현황(힙 메모리상의 객체별 사이즈)을 조회할 수 있습니다. 즉 객체의 순수 크기를 조회합니다. **건수** 컬럼을 통해 주요 객체들의 개수 확인이 용이합니다.

JVM 옵션

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

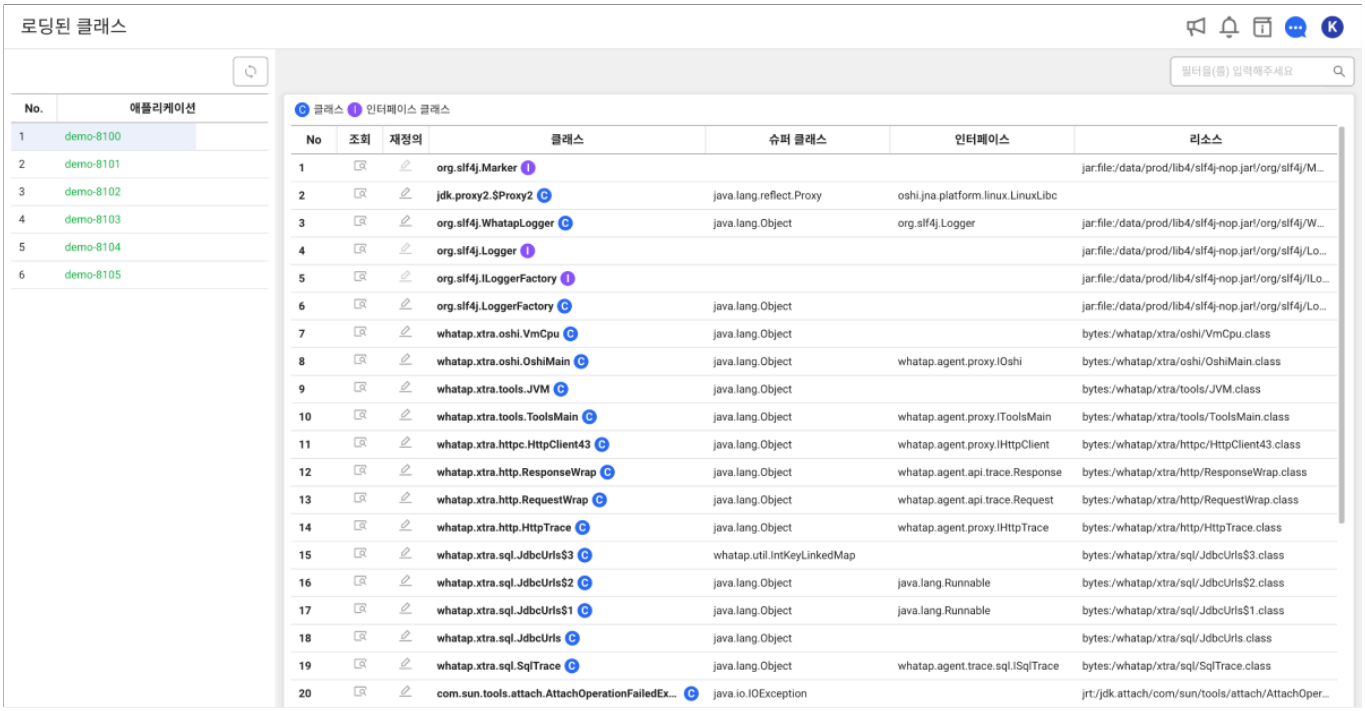
```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

- Java 5 버전 이하, IBM Java는 지원하지 않습니다.

> ⓘ GC 전까지 유지되는 객체의 개수를 조회하기 때문에 실제로 사용되고 있지 않더라도 해당 목록에는 표시됩니다. 정확한 조회를 위해 **시스템 GC** 실행 후 해당 목록을 확인하는 것을 권장합니다. **시스템 GC**에 대한 자세한 내용은 다음 문서를 참조하세요.

로딩된 클래스

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **로딩된 클래스**



Java 애플리케이션에 로딩된 클래스 정보를 확인할 수 있습니다. **로딩된 클래스** 메뉴를 통해 애플리케이션에서 로딩된 클래스 목록과 관련 상세 데이터를 조회하고, 중단없이 클래스의 동작을 변경할 수 있습니다.

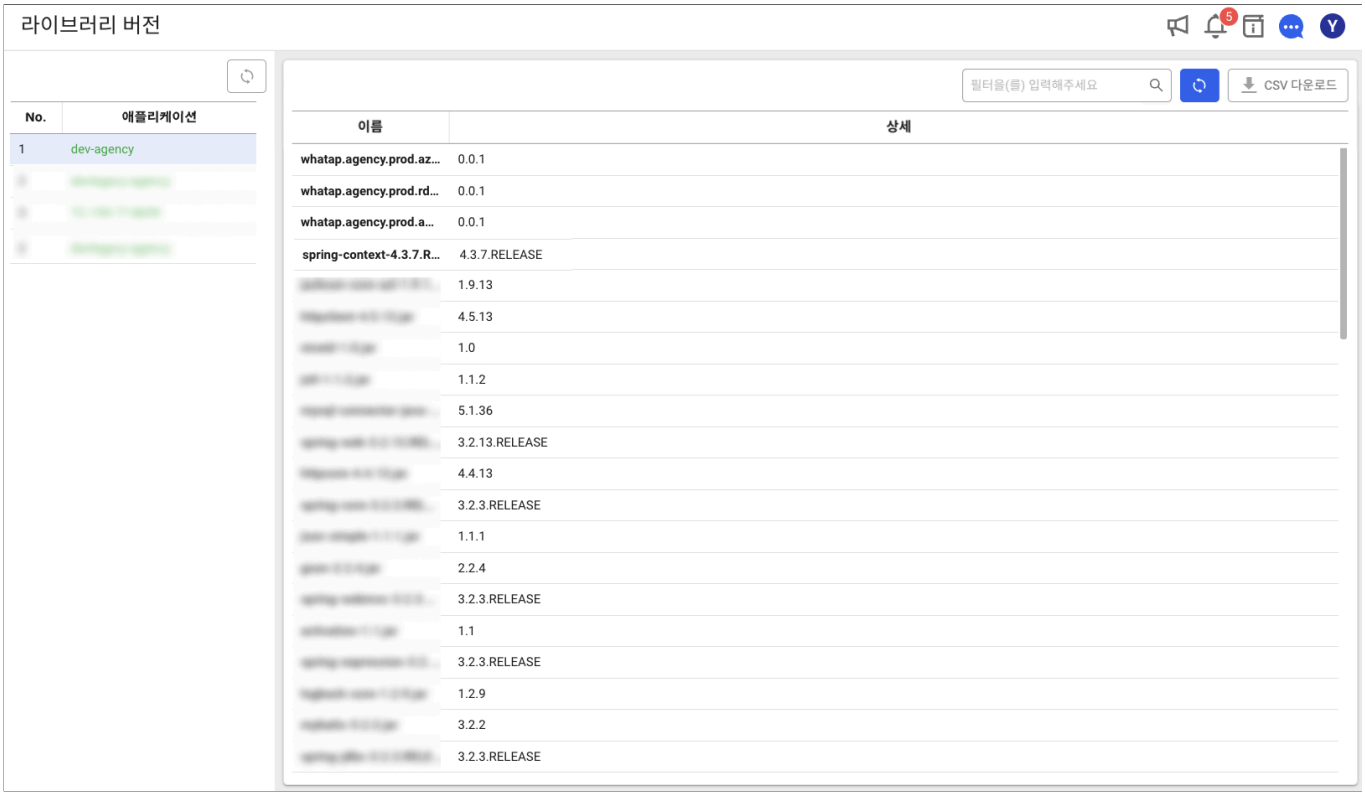
- **클래스 상세**: **조회** 컬럼에서 아이콘 클릭 시 **클래스 상세**를 확인할 수 있습니다. **클래스 상세**는 클래스 내 메소드 및 상속 관계 정보를 제공합니다.
- **재정의**: **재정의** 컬럼에서 아이콘 클릭 시 해당 클래스를 재정의할 수 있습니다. 런타임에 이미 로딩된 클래스를 다시 로딩 후 바이트 코드 삽입(BCI)을 통해 애플리케이션 중단없이 해당 클래스의 동작 변경이 가능합니다.
- **리소스**: 해당 클래스가 물리적으로 어느 파일에서 로딩되었는지 확인할 수 있습니다. 애플리케이션의 복잡한 다계층 구조 특성상 클래스가 정확히 어느 jar 파일에서 로딩되었는지 파악하는 **리소스** 정보 확인이 필요합니다.

> **재정의** 기능과 **리소스** 정보 확인 활용에 대한 자세한 내용은 다음 문서를 참조하세요.

> **BCI(ByteCode Instrumentation)**
>
> 자바 바이트 코드는 플랫폼 독립적인 중간 코드로 JVM에서 실행됩니다. **BCI** 즉 바이트 코드 삽입은 소스 파일 수정 없이 런타임 또는 빌드 타임에 바이트 코드를 동적으로 변경하여 추가 기능을 구현하는 기술입니다. 자세한 내용은 다음 문서를 참조하세요.
>
> - Java SE Specifications
> - Java Package Summary

라이브러리 버전

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **라이브러리 버전**

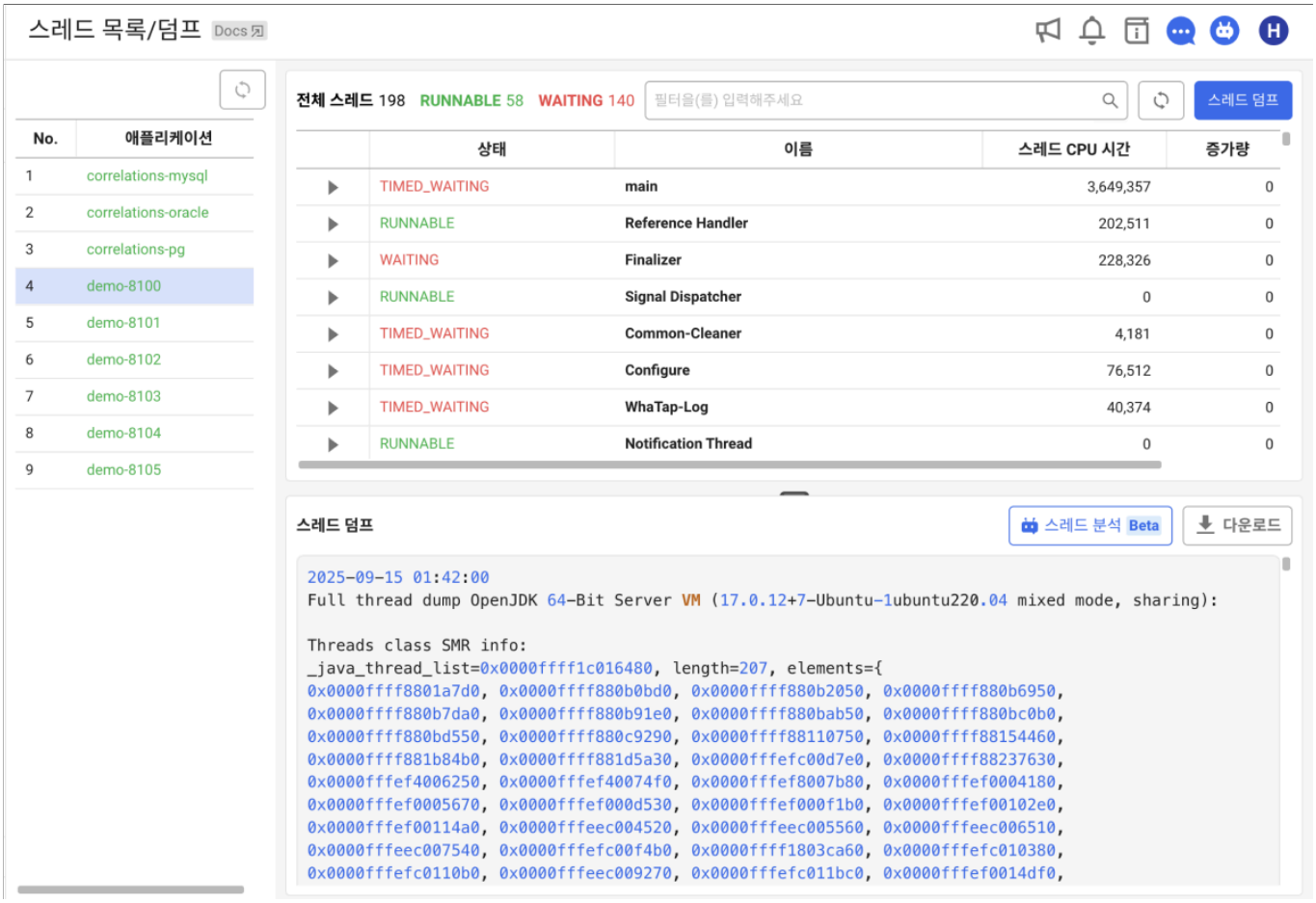


Java 애플리케이션에 배포된 컴포넌트 목록을 조회할 수 있습니다. 오픈 소스 라이브러리를 많이 사용하는 환경에서는 jar 파일의 버전 정보를 확인하여 호환성 문제를 신속히 식별하고 해결할 수 있습니다. 또한 버전 정보 확인은 보안 취약점이 있는 버전을 찾아 패치하는 등 보안 관리에도 용이합니다.

- 컴포넌트 목록 정보: 해당 jar 파일의 **이름** 및 버전 정보를 **상세**에서 제공합니다.
- **CSV 다운로드**: 다운로드 버튼을 클릭해 버전 정보를 다운로드할 수 있습니다.

스레드 목록/덤프

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **스레드 목록/덤프**



Java 프로세스의 모든 스레드 상태에 대한 스냅샷 정보를 제공합니다. **스레드 목록/덤프**는 CPU 최적화 작업에 핵심적인 메뉴입니다. 프로세스가 아닌 스레드 단위로 상태를 확인함으로써 CPU 자원을 더욱 세밀하게 관리할 수 있습니다. 예를 들어 조회 시점에 CPU를 가장 많이 사용하고 있는 스레드 특정 후 해당 스레드의 스택 정보를 확인해 성능 문제를 식별할 수 있습니다.

- 스레드 현황: **전체 스레드** 개수 및 RUNNABLE, WAITING 상태의 스레드 개수를 목록 상단 현황 요약을 통해 확인할 수 있습니다.
- **스레드 덤프**: 상단 오른쪽의 스레드 덤프 버튼을 클릭해 전체 스레드 덤프를 생성할 수 있습니다. 전체 스레드 덤프 생성 후 스레드 목록 하단에서 해당 정보를 확인할 수 있습니다.
- 개별 스레드 덤프: 각 스레드마다 ▶ 아이콘 선택 시 개별 스레드 덤프 정보를 확인할 수 있습니다. **스택** 및 **Stack 추적**에서 스레드 상태와 실행 경로에 대한 세부 정보 및 스택 트레이스를 조회할 수 있습니다. 새로고침 해 확인할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- **스레드 CPU 시간**: 해당 스레드가 CPU를 점유한 누적 시간을 의미합니다. 밀리초 단위 시간을 사용합니다.
- **증가량**: 스레드 목록 상단에서 새로고침 아이콘을 클릭하면 기존 스레드 CPU 시간과 현재 스레드의 CPU 시간의 차잇값이

추가됩니다.

스레드 CPU 시간은 해당 스레드가 CPU를 점유한 총 누적 시간이기 때문에 조회 시점에 가장 많이 CPU를 사용하고 있는 스레드를 특정하기 위해서는 **증가량** 값 확인이 필요합니다. 자세한 내용은 다음 문서를 참조하세요.

CPU 사용량 급증 유형

- 힙 메모리가 가득 찬 경우: 잦은 GC로 CPU 사용량 급증
- 실제 요청이 많은 경우: 대량의 클라이언트 요청 처리로 CPU 사용량 급증
- 잘못된 코드 로직: 비효율적인 알고리즘이나 무한 루프로 CPU 사용량 급증

스레드 덤프 상세보기

스레드 목록/덤프 메뉴에서 확인할 수 있는 개별 스레드 덤프 정보에 대해 안내합니다.

JVM 옵션

스레드 덤프 기능을 각 자바 버전별로 사용하기 위해서는 추가적인 JVM 옵션 설정이 필요합니다.

Java 6 ~ 8 버전에서는 JVM 옵션 없이 기본 지원하지만, 일부 Java 버전에 따라 다음과 같이 JVM 옵션을 적용해야 합니다.

- Java 9 ~ Java 15 버전

```
-Djdk.attach.allowAttachSelf=true
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true -jar {application.jar}
```

- Java 16 버전 이상

```
-Djdk.attach.allowAttachSelf=true
--add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED
```

example

```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

- Java 5 버전 이하, IBM Java는 지원하지 않습니다.

스레드 상태

스레드의 상태는 `java.lang.Thread` 클래스 내부에 `State` 라는 이름의 Enumerated Types(열거형)으로 선언된 항목들입니다.

- `NEW` : 스레드가 생성되었지만 아직 실행되지 않은 상태입니다.
- `RUNNABLE` : 현재 CPU를 점유하고 작업을 수행 중인 상태입니다. 운영체제의 자원 분배로 `WAITING` 상태가 될 수도 있습니다.
- `WAITING` : `wait()` , `join()` , `park()` 메소드 등를 이용해 대기하고 있는 상태입니다.
- `TIMED_WAITING` : `sleep()` , `wait()` , `join()` , `park()` 메소드 등을 이용해 대기하고 있는 상태, `WAITING` 상태와 다른 점은 주어진 시간 동안 대기하고 있는 상태입니다. 외부적인 변화 뿐만 아니라 시간에 의해서도 대기 상태가 해제될 수 있습니다.
- `BLOCKED` : 사용하려는 개체의 락(Lock)이 풀릴 때까지 대기하고 있는 상태입니다.
- `TERMINATED` : 실행을 마친 상태입니다.

스레드 덤프 정보

스레드 목록에서 각 스레드의 ▶ 버튼을 선택하면 개별 스레드 덤프 정보를 확인할 수 있습니다.



스택

항목	속성	설명
스레드 ID	threadId	스레드에 할당된 고유 ID
Lock 소유주 ID	lockOwnerId	스레드가 차단된 개체의 모니터(Monitor) 잠금을 유지하는 스레드 ID
대기수	waitedCount	스레드가 WAITING 또는 TIMED_WAITING 상태가 된 총 횟수
블록 개수	blockedCount	스레드가 BLOCKED 상태가 된 총 횟수
대기 시간	waitedTime	스레드가 WAITING 상태를 지속한 경과 시간(밀리초), 스레드 경합 모니터링이 비활성화된 경우 -1 반환
스레드 CPU 시간	threadCpuTime	스레드의 CPU 시간 합계(나노초)를 밀리초로 계산하여 표시, CPU 시간 측정을 비활성화한 경우 -1 반환
상태	threadstate	스레드 상태
블록 시간	blockedTime	스레드가 BLOCKED 상태가 된 후 경과 시간(밀리초), 스레드 경합 모니터링이 비활성화된 경우 -1 반환
스레드 명	threadName	스레드 고유 이름
Lock 이름	lockName	스레드의 입력이 차단되거나 Object.wait 메소드를 통해 통지를 기다리는 모니터 잠금을 표시한 문자열
Lock 소유주 이름	lockOwnerName	스레드가 차단되는 객체의 모니터 잠금을 수용하는 스레드 이름
스레드 사용자 시간	threadUserTime	스레드가 사용자 모드에서 실행한 CPU 시간(나노초)을 밀리초로 계산하여 표시

ⓘ 스레드 정보에 대한 자세한 내용은 다음 링크를 참조하세요.

Stack 추적

예외가 발생했을 때 스택(Stack) 추적을 표시합니다. 스택 프레임 목록으로 코드가 호출한 메소드 정보가 포함되어 있습니다.

AI 스레드 분석

스레드 덤프 상세 화면에서 **스레드 분석** 버튼을 클릭합니다.

AI 기반 분석 결과는 드로워 형태로 제공됩니다.

분석 결과 항목

- 요약
- 총 스레드 수
- 스레드 상태별 분포
- 이슈
- 권장 사항

스레드 분석 재시도

드로워 상단 버튼을 클릭하여 동일한 스레드 덤프를 다시 분석할 수 있습니다.

분석 결과는 와탭 페이지에 설정된 언어로 제공됩니다. 일부 경우 영어로 표시될 수 있습니다.

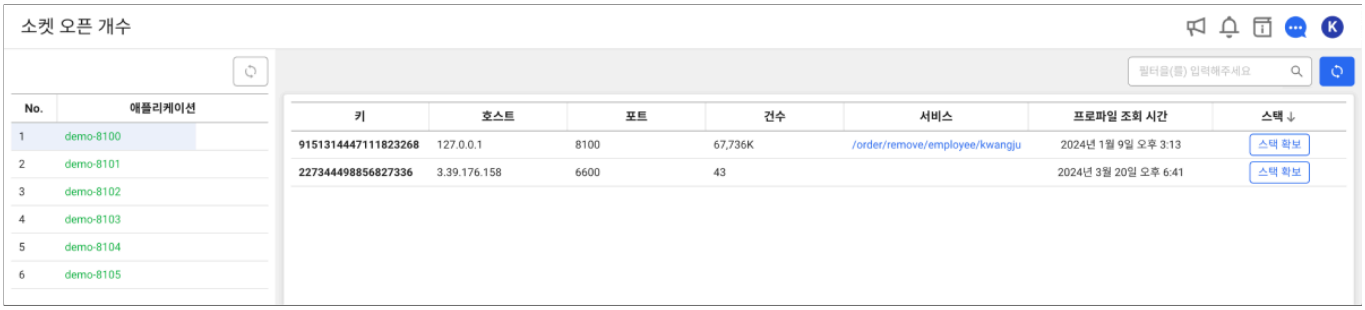
프로젝트당 하루 20회까지 분석할 수 있으며, 프로젝트 타임존 기준 매일 00:00:00에 초기화됩니다.

- 분석 회수 차감 기준
  - 새로운 스레드 덤프 분석 시 1회 차감
    - 애플리케이션 선택 상태 변경
      - 애플리케이션 리스트에서 다른 애플리케이션 선택
    - 시간값 변경
      - 다른 애플리케이션 선택 시 (선택 상태와 시간 변경)
      - 동일한 애플리케이션 재선택 시 (선택 상태 유지, 시점 변경으로 덤프 변경)

- 동일 스레드 덤프에 대해 드로워 내에서 재분석 시도 시에도 1회 차감

소켓 오픈 개수

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **소켓 오픈 개수**



Java 애플리케이션이 TCP 기능을 수행하기 위해 오픈한 소켓(Socket) 정보를 확인할 수 있습니다 이를 통해 애플리케이션 인스턴스가 데이터베이스와 같은 어떤 외부 시스템에 의존하는지 네트워크 의존성(Network Dependency)을 조회할 수 있습니다. 또한 누구와 아웃바운드 호출을 하고 있는지 추적할 수 있습니다. 이러한 정보는 TCP 통신의 효율성, 안정성 및 성능 최적화를 위해 중요합니다.

- **건수**: 외부 시스템에 대한 연결 시도 횟수를 의미합니다.
- **스택**: **스택 확보**를 선택하여 다음 연결 시 확보한 스택을 조회 및 삭제할 수 있습니다.

건수 기준으로 정렬 후 **호스트** 및 **포트**를 확인하여 어떤 서비스 또는 시스템과 연결이 많은지 확인할 수 있습니다. 이후 **스택**을 확보해 상세 내용을 조회해 보세요.

> ⓘ **데이터베이스와 커넥션 풀**
>
> 예를 들어 3306 **포트**의 **건수**가 증가 시 커넥션 풀을 사용하지 않거나 또는 커넥션 풀을 빈번하게 끊고 다시 만드는 등 MySQL의 커넥션 관리가 올바르게 이루어지지 않는 상황일 가능성이 있습니다. 해당 3306 **포트**가 어떤 코드 경로에서 사용되고 있는지 파악하기 위해 **스택**을 확보하여 추적할 수 있습니다.

메소드 성능 상태

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **메소드 성능 상태**

Java 애플리케이션에서 수행 된 메소드(Method)의 성능 통계 정보를 제공합니다.

- **새로고침**: 데이터를 재호출합니다.
- **리셋**: 등록된 메소드 통계 데이터를 초기화합니다.

> ⓘ **메소드 성능 상태** 메뉴를 사용하려면 **에이전트 설정** 메뉴에서 다음 사용자 정의 옵션을 추가하세요. 그리고 재기동 또는 **로딩된 클래스** 메뉴에서 **재정의**하세요.
>
> `hook_method_stat_patterns=io.home.test.baseapp.app.post.mapper.PostMapperImpl.*`

데이터소스 상태

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **데이터소스 상태**

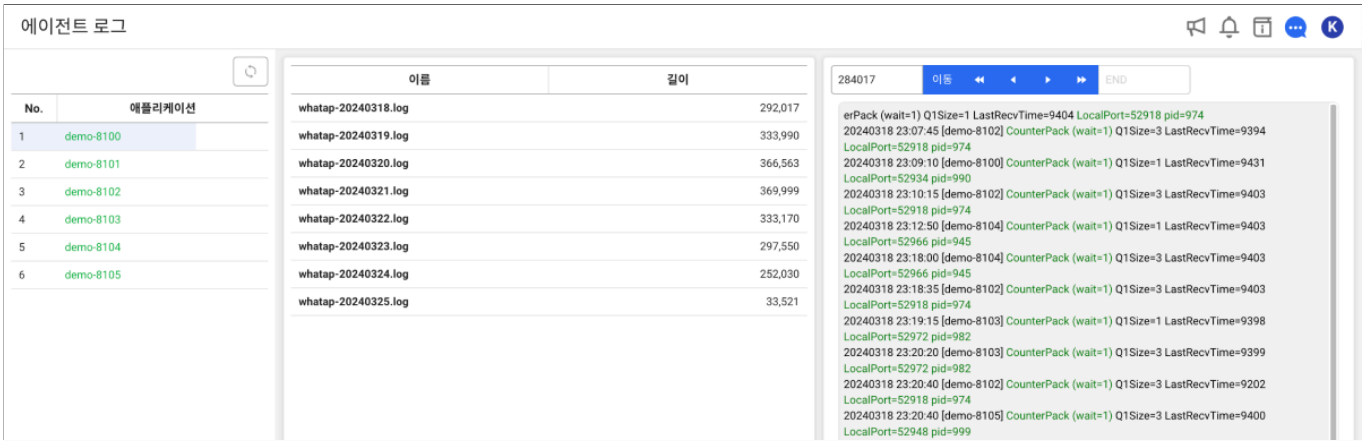


데이터소스(DataSource)의 상태를 확인할 수 있습니다. 애플리케이션이 사용하는 데이터베이스 커넥션 풀의 현재 상태와 구성, 연결 설정 정보, 연결 상태 및 설정 불일치 여부 등의 정보를 제공합니다.

> ⓘ 커넥션 풀 객체의 생성과 동시에 모니터링이 시작되기 때문에 해당 객체가 사용되고 있지 않더라도 데이터소스 상태 정보에 나타날 수 있습니다.

에이전트 로그

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **에이전트 로그**



모니터링 대상 서버에 저장된 에이전트 로그를 조회할 수 있습니다. 로그 파일의 이름은 whatap-`YYYYMMDD`.log 형식입니다. 각 로그를 선택해 로그에 캡쳐되는 오류 및 이벤트에 대한 정보를 액세스할 수 있습니다.

> ⓘ 로그와 관련한 에이전트 설정은 다음 문서를 참조하세요.

시스템 GC

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **시스템 GC**



`System.gc()`를 호출해 JVM(자바 가상 머신)의 Heap 영역에서 동적으로 할당했던 메모리 중 필요없게 된 메모리 객체(garbage)를 모아 제거할 수 있습니다. **GC 실행** 버튼을 선택하면 GC 프로세스를 실행하게 되며, 실행 전과 실행 후의 메모리 용량을 확인할 수 있습니다.

> ⓘ `-XX:+DisableExplicitGC` 와 같이 명시적인 GC 요청을 비활성화하는 옵션이 설정된 경우 **시스템 GC**가 동작하지

ⓘ 않습니다.

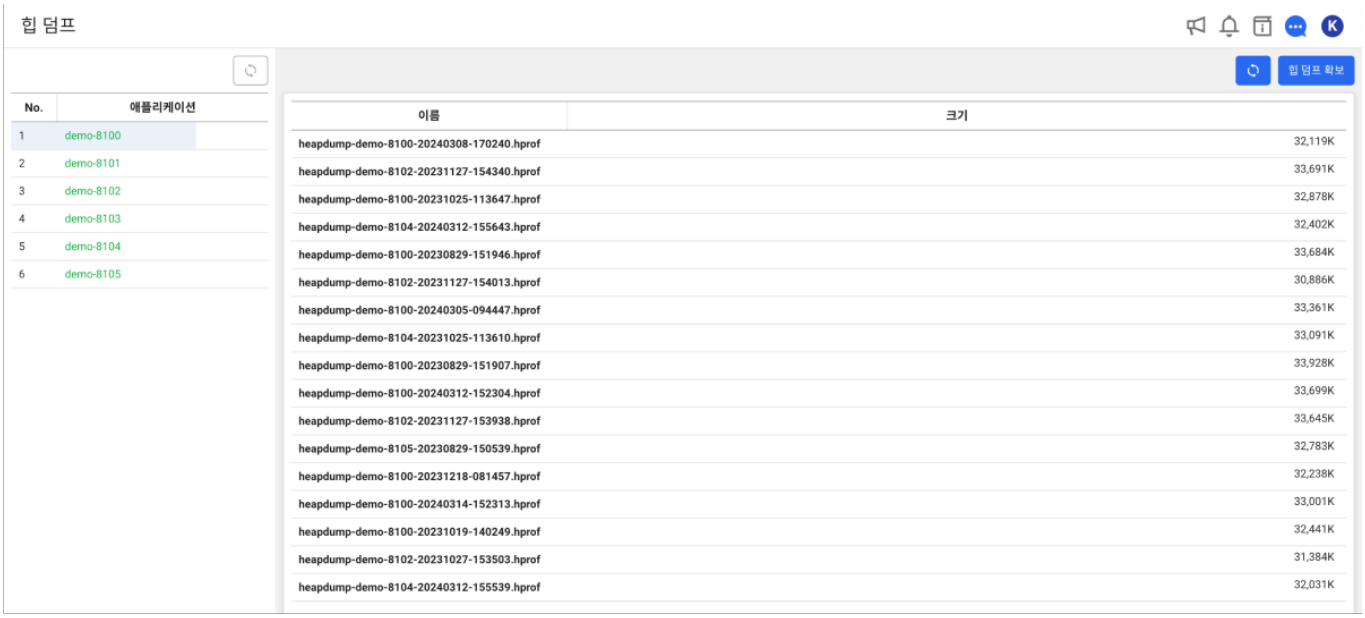
ⓘ **시스템 GC와 메모리 누수**

Heap 메모리 사용량이 지속적으로 상승할 때 메모리 누수의 가능성을 확인하고 정확하게 진단하기 위해 **시스템 GC**가 필요합니다.

- **메모리 요청 유무**: 애플리케이션이 지속적으로 메모리를 요청하고 있는 상태인지 확인합니다. 메모리 요청 증가가 예상치 이상인지 또는 메모리 해제가 정확히 이루어지고 있는지 살펴봅니다.
- **메모리 사용량 확인**: 실제로 메모리가 과다하게 사용되고 있는 상태인지 확인합니다. GC 수행으로 필요하지 않은 객체를 해제하고 메모리 사용량 변화를 살펴봅니다.
- **GC 주기**: 메모리가 충분히 비어있는 상황임에도 GC가 수행되지 않는 상태인지 확인합니다. GC 주기 또는 설정을 살펴봅니다.

힙 덤프

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **힙 덤프**



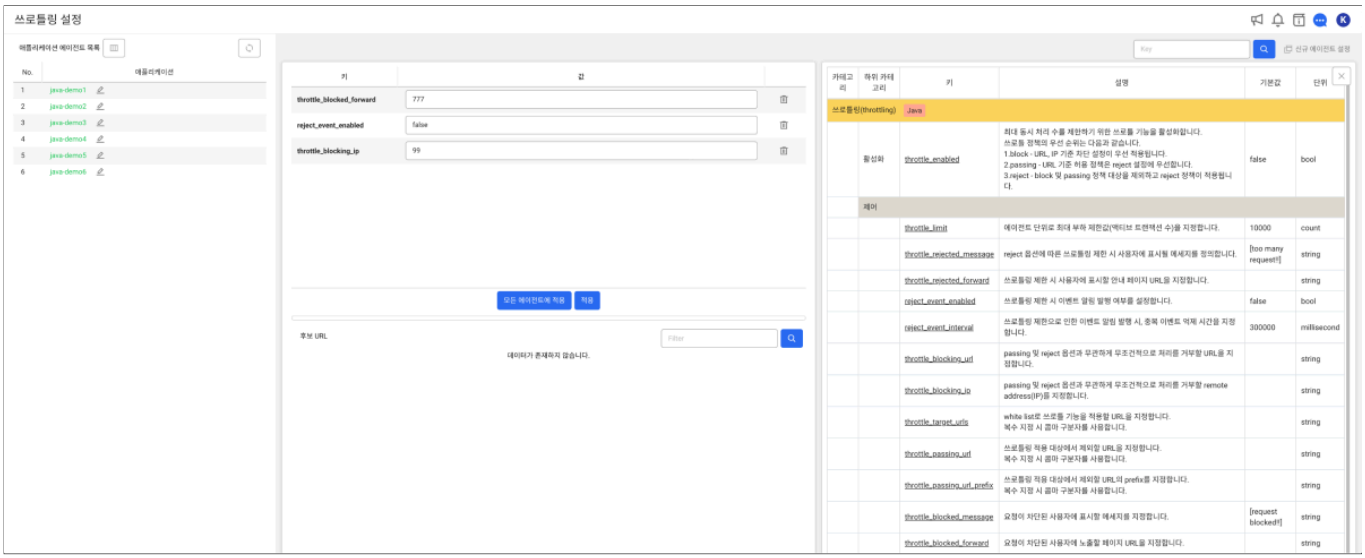
Java 애플리케이션 실행 중 메모리 누수 등의 문제가 발생하면 관련된 문제를 정리하여 덤프 파일을 생성할 수 있습니다. **힙 덤프**

확보 버튼을 선택하세요. 생성된 덤프 파일은 애플리케이션 서버에 저장됩니다. 에이전트를 설치한 경로 하위의 dump 디렉터리에서 확인할 수 있습니다.

> ⓘ 이 기능은 Java 에이전트 1.5.2 버전 이상에서 지원합니다. 에이전트 설정에서 `heapdump_enabled` 옵션을 `true` 로 설정하세요. **힙 덤프 확보** 버튼을 선택해 힙 덤프 파일을 만드는 기능을 활성화합니다. 매번 힙 덤프를 자동 생성하지 않으며 해당 옵션이 성능에 영향을 주진 않습니다. 그러나 덤프 확보 작업은 성능에 영향을 미칠 수 있습니다.
>
> **힙 덤프 확보** 작업은 **부하**가 큰 작업에 속합니다. 부하와 별개로 힙 덤프가 모두 추출될 때까지 애플리케이션의 코드는 실행이 중지(stop the world)되기 때문에 사용자가 느끼는 성능 차이는 더 심할 수 있습니다.

쓰로틀링 설정

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **쓰로틀링 설정**



Java 애플리케이션의 부하량 제어와 관련한 에이전트 설정을 확인하고 옵션을 설정할 수 있습니다. 부하량 제어와 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

에이전트 설정 *Old*

홈 화면 > 프로젝트 선택 > **인스턴스 성능 관리** > **에이전트 설정** *Old*

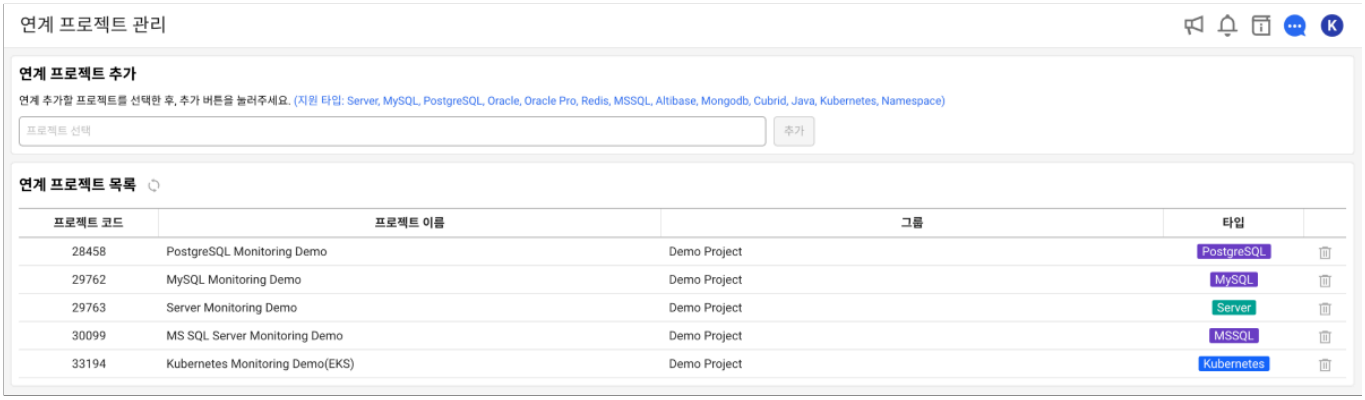
모니터링 대상 서버에 위치한 whatap.conf 파일을 직접 수정하지 않고 에이전트 설정 옵션을 추가하거나 수정, 삭제할 수 있습니다. **설정 파일 다운로드** 버튼 클릭 시 선택한 에이전트의 whatap.conf 파일을 다운로드합니다.

> ⓘ 에이전트 설정 기능은 사용성과 기능을 개선한 **관리** > **에이전트 설정** 메뉴를 이용할 것을 권장합니다. 화면 오른쪽 위에 **신규 에이전트 설정** 버튼을 선택하세요. 자세한 내용은 다음 문서를 참조하세요.

연결된 프로젝트

홈 화면 > 프로젝트 선택 > **관리** > **연계 프로젝트 관리**

데이터베이스 모니터링(DPM) 프로젝트와 애플리케이션 모니터링(APM) 프로젝트를 생성한 경우, 두 프로젝트를 연계하여 DPM에서 수집한 모니터링 데이터를 APM 프로젝트에서 확인할 수 있습니다.



연계 프로젝트 관리 기능을 사용하면 제품 또는 장비 중심의 모니터링을 넘어 여러 프로젝트 간의 모니터링 데이터를 통합해 확인할 수 있습니다. 이는 애플리케이션, 데이터베이스 등 다양한 시스템 구성 요소들이 어떻게 상호 작용하는지 분석하는 데 유용합니다. 특히, 성능 저하가 발생했을 때 장비 문제인지 개별 애플리케이션 문제인지 신속하게 파악할 수 있어, 문제 해결 시간을 크게 단축시킬 수 있습니다. **연계 프로젝트 관리**를 통해 사용자 시스템 단위에서 성능 데이터를 종합적으로 분석하고, 보다 가시적인 인사이트를 확보할 수 있습니다.

다음 동영상을 통해 자세한 내용을 확인할 수 있습니다.

시작하기 전 참고 사항

- **연계 프로젝트** 기능을 지원하는 데이터베이스 플랫폼은 다음과 같습니다.

  PostgreSQL, Oracle, MySQL, SQL Server, CUBRID, Altibase, Redis, MongoDB

- **연계 DB 세션** 기능의 경우 Java 에이전트 2.2.33 버전 이상이 필요합니다. 해당 기능을 지원하는 데이터베이스 플랫폼은 다음과 같습니다.

  PostgreSQL, Oracle, Oracle Pro, MySQL

권한 설정

Oracle 및 Oralce Pro 플랫폼의 경우 연계 프로젝트를 추가하기 위해서는 다음과 같은 권한 설정이 필요합니다.

```
grant select on sys.v_$session to DB접속계정;
# ex) grant select on sys.v_$session to petapp;
```

연계 프로젝트 추가하기

1. **관리** > **연계 프로젝트 관리** 메뉴로 이동하세요.
2. **연계 프로젝트 추가** 섹션에서 **프로젝트 선택** 입력란을 선택하세요.



3. 선택할 수 있는 프로젝트 목록이 표시되면 연계할 프로젝트를 선택하세요. 문자열을 입력해 일치하는 프로젝트를 검색할 수도 있습니다.
4. 프로젝트를 하나 이상 선택한 다음 활성화된 **추가** 버튼을 선택하세요.

연계 프로젝트 목록에 선택한 프로젝트가 추가됩니다.

연계 프로젝트를 추가하기 위해서는 프로젝트 **수정** 권한이 필요합니다. **수정** 권한이 있는 경우 **연계 프로젝트 관리** 메뉴에서 **추가** 버튼이 활성화됩니다.

에이전트 옵션 설정하기

데이터베이스 프로젝트와 연계 분석을 위해 다음과 같이 에이전트 설정 파일(whatap.conf)에 옵션을 설정해야 합니다. 또는 **관리** > **에이전트 설정** 메뉴에서 설정할 수 있습니다.

whatap.conf

```
dbc_sid_enabled=true
hook_db_native_enabled=true
debug_dbc_sid_enabled=true
```

- **dbc_sid_enabled** bool

  기본값 `false`

  SID 발급을 위한 옵션입니다. 연계 분석을 위해 값을 `true` 로 설정하세요. **필수** 옵션입니다.

- **hook_db_native_enabled** bool

  기본값 `false`

  트랜잭션에서 사용하는 DB 종류를 추출하기 위한 옵션입니다. 연계 분석을 위해 값을 `true` 로 설정하세요. **필수** 옵션입니다.

- **debug_dbc_sid_enabled** bool

  기본값 `false`

  Java 에이전트 로그에 SID 관련 정보를 로깅하는 옵션으로 선택 옵션입니다.

> ⓘ **관리** > **에이전트 설정**



연계 프로젝트 지표 확인하기

추가한 연계 프로젝트와 관련한 지표는 다음 메뉴 경로에서 확인할 수 있습니다.

- **대시보드** > **애플리케이션 대시보드**
- **대시보드** > **액티브 트랜잭션**
- **분석** > **히트맵**

- **애플리케이션 대시보드** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
- **액티브 트랜잭션** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.
- **히트맵** 메뉴에 대한 자세한 내용은 다음 문서를 참조하세요.

애플리케이션 대시보드

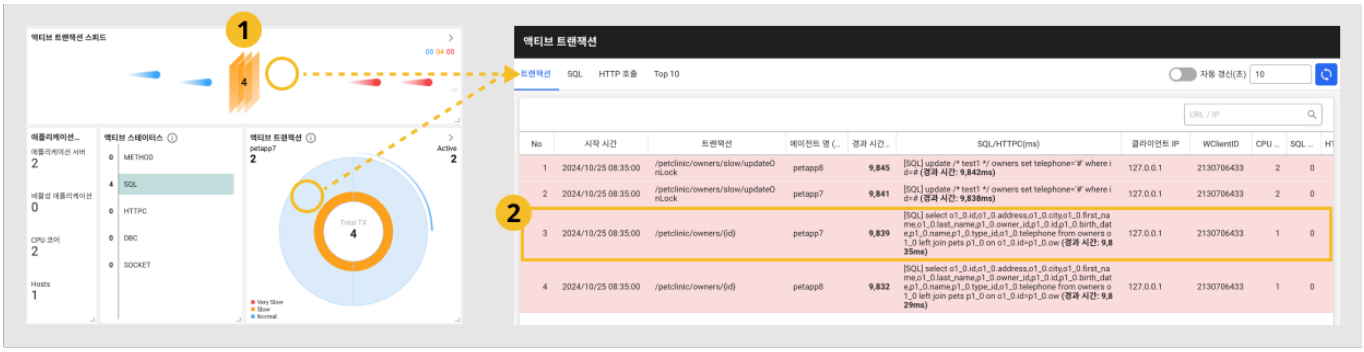
애플리케이션 대시보드 메뉴에서는 다음 위젯을 통해서 연계 프로젝트를 확인할 수 있습니다.

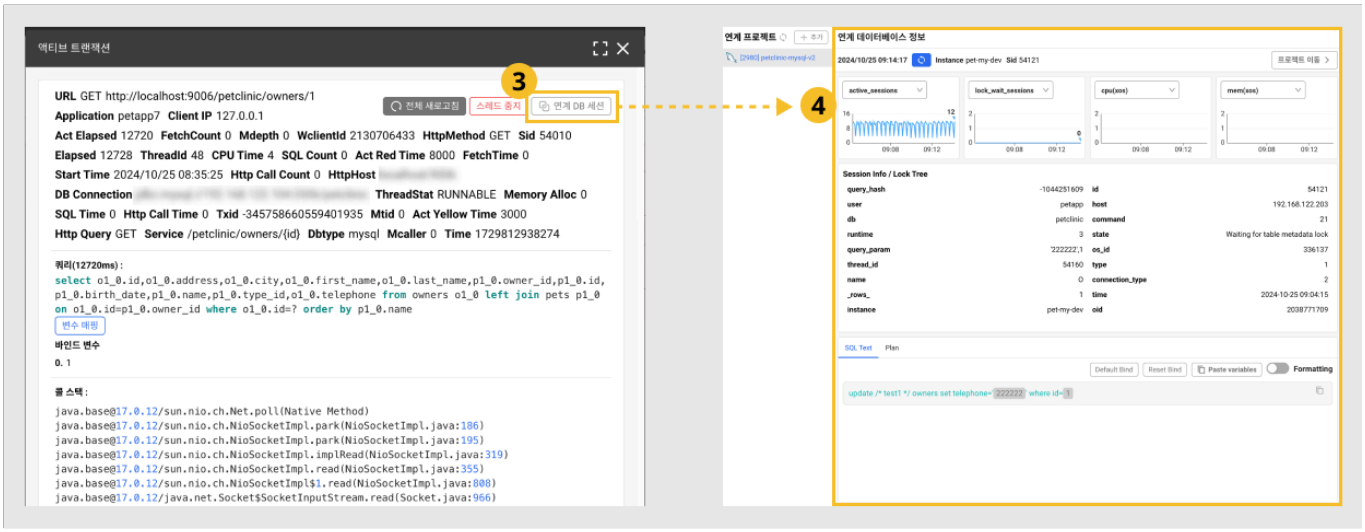
- **액티브 트랜잭션 스피드**
- **액티브 트랜잭션**

- **히트맵**

액티브 트랜잭션 스피드 / 액티브 트랜잭션 위젯

현재 진행 중인 트랜잭션과 연계된 데이터베이스의 SQL 성능을 조회할 수 있습니다. **대시보드** > **애플리케이션 대시보드** 메뉴로 이동하세요.





1. 대시보드에서 **액티브 트랜잭션 스피드** 또는 **액티브 트랜잭션** 위젯의 차트 영역을 클릭하세요.
2. **액티브 트랜잭션** 창이 나타나면 트랜잭션 항목을 선택하세요.
3. 트랜잭션 상세 정보 창이 나타나면 오른쪽 상단의 **연계 DB 세션** 버튼을 선택하세요.
4. **연계 데이터베이스 정보** 창이 나타나면 관련 지표를 확인하세요.

> • 연계분석을 위해 등록한 데이터베이스 프로젝트의 DB가 현재 선택한 트랜잭션의 SQL을 실행한 경우에만, 트랜잭션

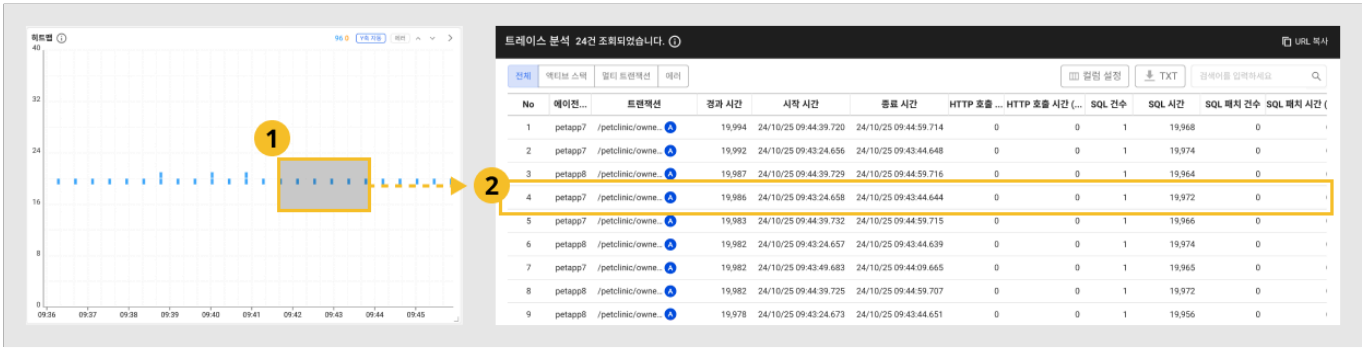
상세 정보 창에 **연계 DB 세션** 버튼이 표시됩니다.

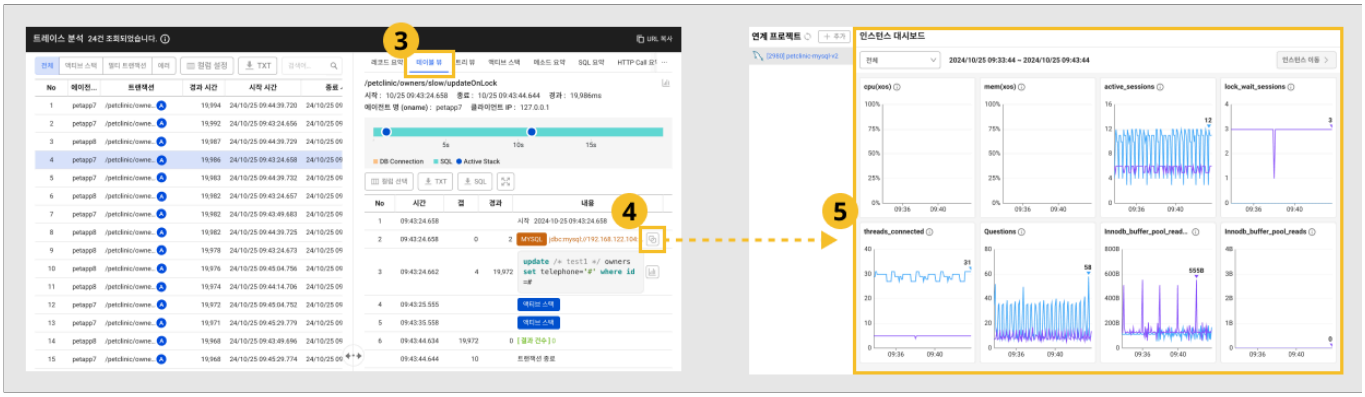
- **연계 데이터베이스 정보** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
- **Java 에이전트 2.2.40** 이전 버전에서는 **액티브 트랜잭션** 창에서 **ThreadStat** 값이 숫자 형식으로 표시됩니다. 문자 형식으로 표시하려면 Java 에이전트 2.2.40 버전 이상을 설치하세요.
- 이 기능을 지원하는 데이터베이스는 **PostgreSQL**, **Oracle**, **Oracle Pro**, **MySQL** 플랫폼입니다.

히트맵 위젯

과거 시점의 트랜잭션과 연계된 데이터베이스의 성능 지표를 조회할 수 있습니다. **대시보드** > **애플리케이션 대시보드** 메뉴로 이동하세요.

DB Connection 스텝에서 확인하기



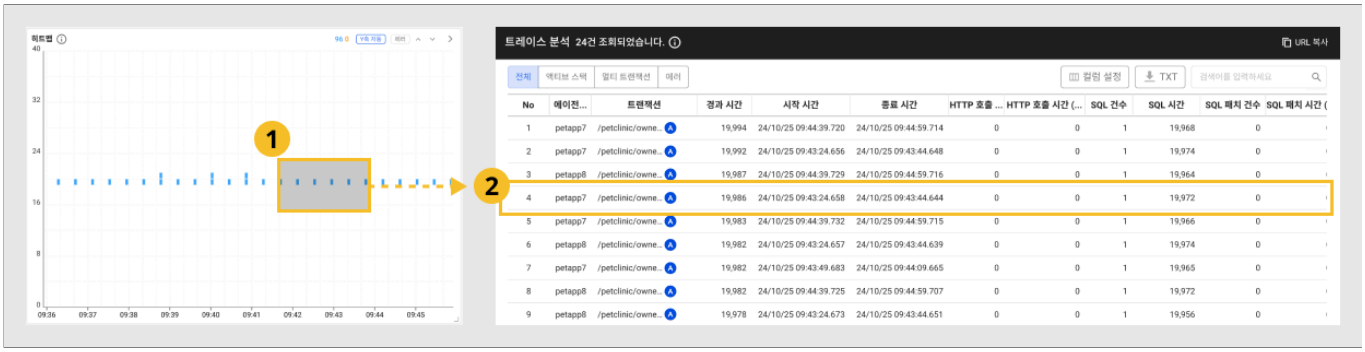


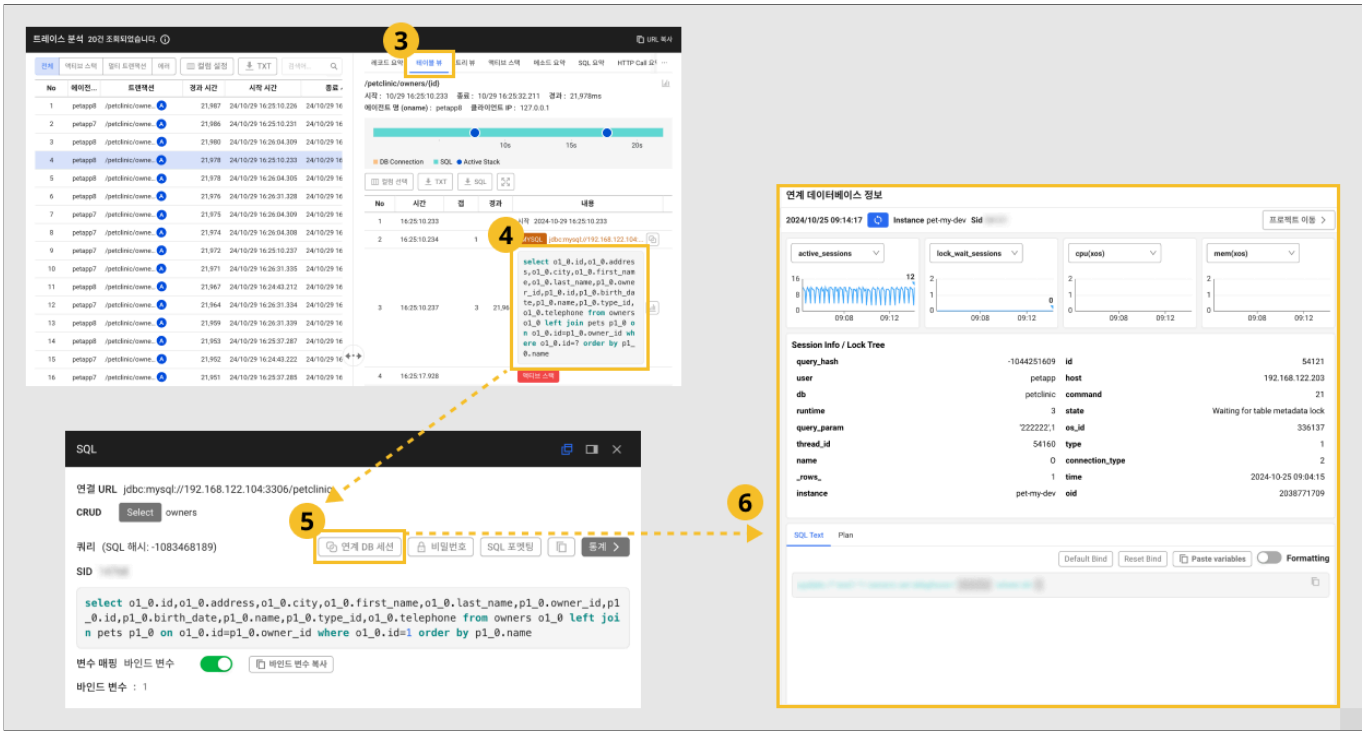
1. 대시보드에서 **히트맵** 위젯의 차트 영역을 드래그하세요.
2. **트레이스 분석** 창이 나타나면 조회하길 원하는 트랜잭션 항목을 선택하세요.

3. 트랜잭션 상세 정보가 오른쪽에 나타나면 **테이블 뷰** 탭을 선택하세요.
4. DB Connection 스텝에서 버튼을 선택하세요.
5. 연계 프로젝트의 **인스턴스 대시보드** 창이 나타나면 관련 지표를 확인하세요.

- 연계분석을 위해 등록한 데이터베이스 프로젝트의 DB가 현재 선택한 트랜잭션의 SQL을 실행한 경우에만, **테이블 뷰** 탭에 버튼이 표시됩니다.
- **인스턴스 대시보드** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
- **트레이스 분석** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
- 이 기능을 지원하는 데이터베이스는 **PostgreSQL**, **Oracle**, **Oracle Pro**, **MySQL** 플랫폼입니다.

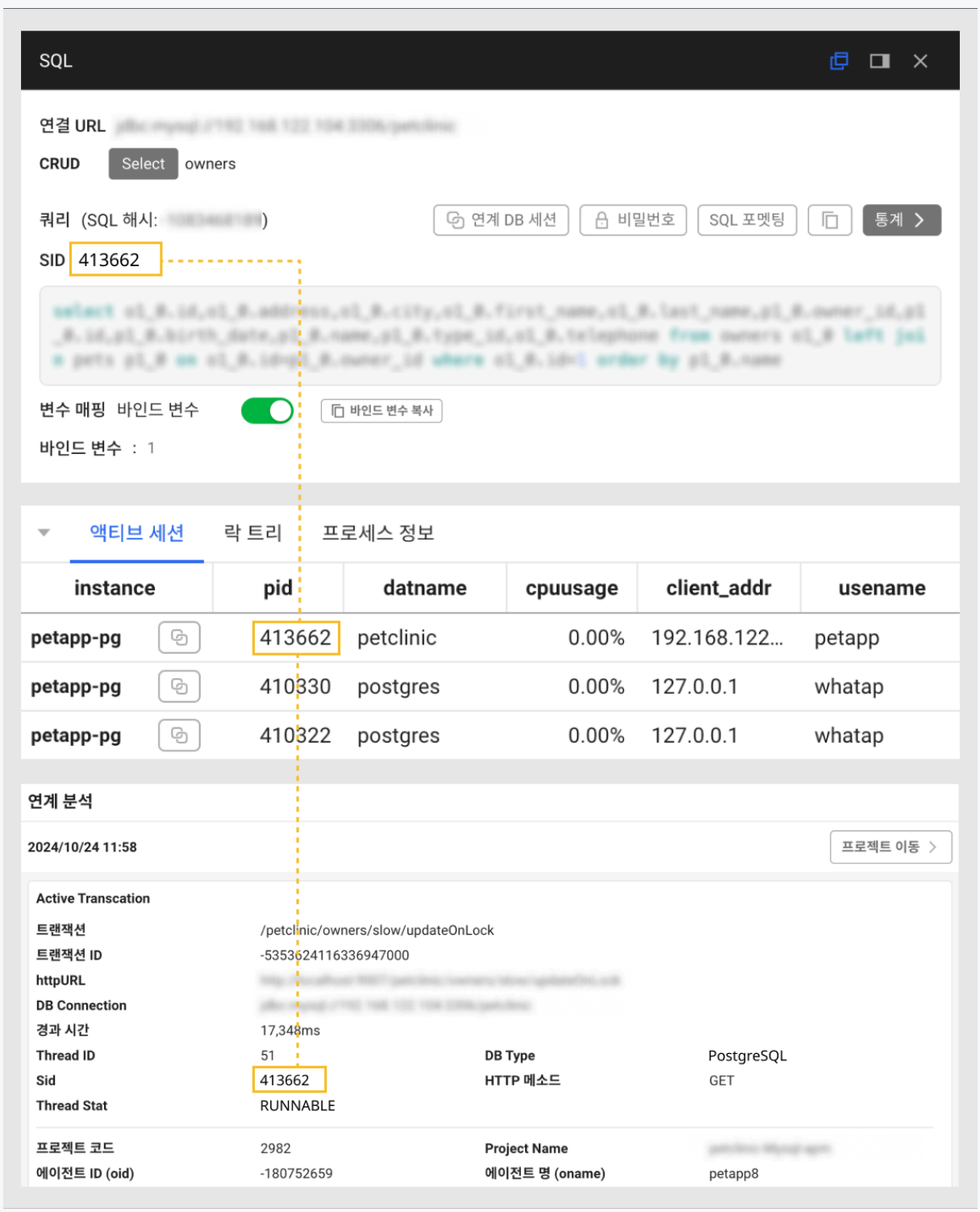
SQL 스텝에서 확인하기





1. 대시보드에서 **히트맵** 위젯의 차트 영역을 드래그하세요.
2. **트레이스 분석** 창이 나타나면 조회하길 원하는 트랜잭션 항목을 선택하세요.
3. 트랜잭션 상세 정보가 오른쪽에 나타나면 **테이블 뷰** 탭을 선택하세요.
4. SQL 스텝에서 SQL 문을 선택하세요.
5. **SQL** 창이 나타나면 **연계 DB 세션** 버튼을 선택하세요.
6. **연계 데이터베이스 정보** 창이 나타나면 관련 지표를 확인하세요.

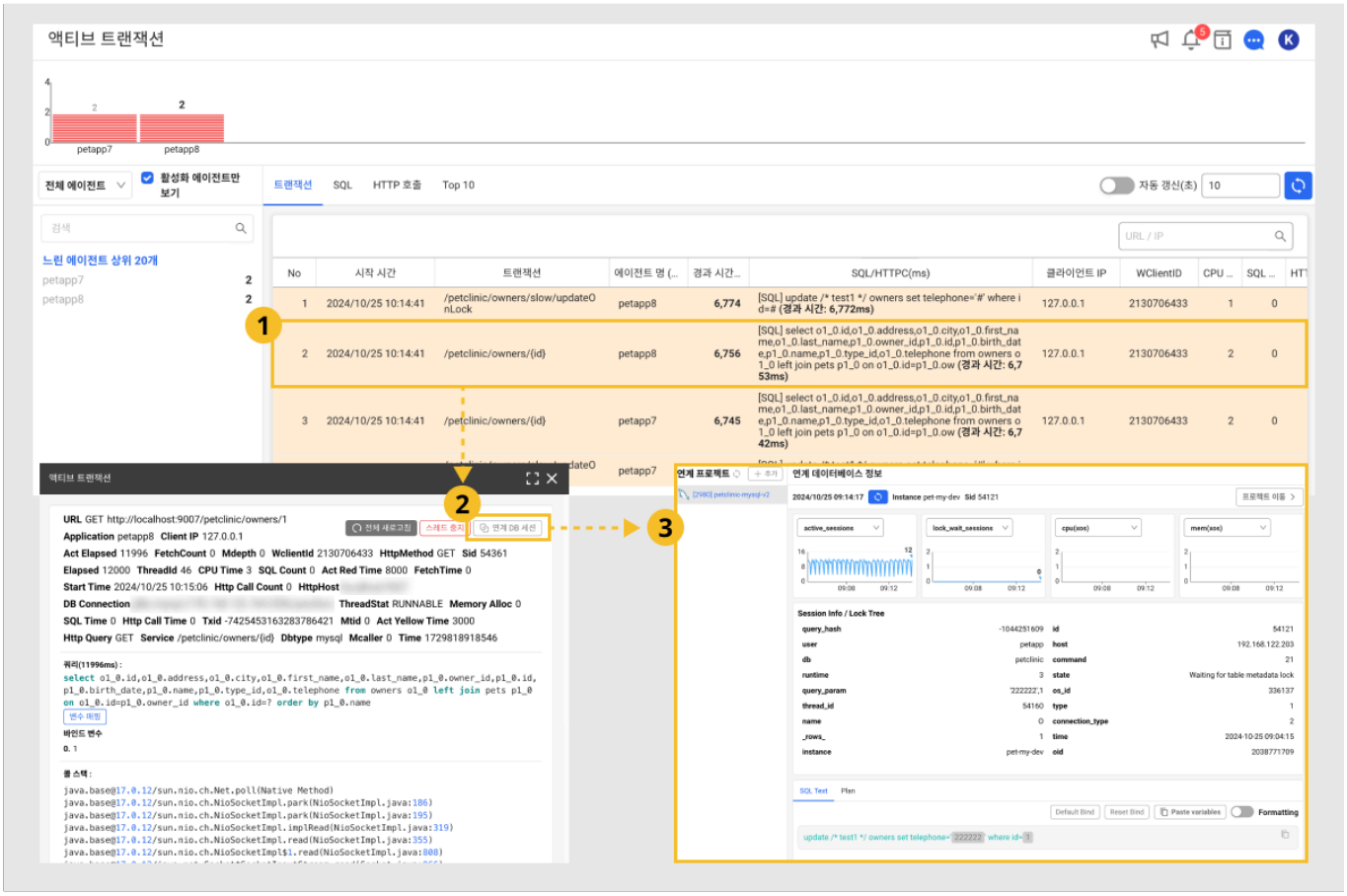
- 연계분석을 위해 등록한 데이터베이스 프로젝트의 DB가 현재 선택한 트랜잭션의 SQL을 실행한 경우에만, **SQL** 창에 **연계 DB 세션** 버튼이 표시됩니다.
- **바인드 변수 복사** 기능을 이용하면 변수를 복사해 **연계 데이터베이스 정보** 창의 **SQL Text** 탭에 변수를 붙여넣기할 수 있습니다. 이 기능은 **비밀번호** 버튼을 선택해 보안키를 입력 후 바인드 변수 값이 표시될 때 이용할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
- **연계 데이터베이스 정보** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
- 이 기능을 지원하는 데이터베이스는 **PostgreSQL**, **Oracle**, **Oracle Pro**, **MySQL** 플랫폼입니다.
- **PostgreSQL** 플랫폼에서 **액티브 세션** 테이블의 **PID** 항목은 **SID** 항목과 같습니다.


SQL
연결 URL
CRUD
Select
owners
쿼리 (SQL 해시:
연계 DB 세션
비밀번호
SQL 포멧팅
통계
SID
413662
변수 매핑 바인드 변수
바인드 변수 복사
바인드 변수 : 1
액티브 세션
락 트리
프로세스 정보
instance
pid
datname
cpuusage
client_addr
usename
petapp-pg
413662
petclinic
0.00%
192.168.122...
petapp
petapp-pg
410330
postgres
0.00%
127.0.0.1
whatap
petapp-pg
410322
postgres
0.00%
127.0.0.1
whatap
연계 분석
2024/10/24 11:58
프로젝트 이동
Active Transcation
트랜잭션
/petclinic/owners/slow/updateOnLock
트랜잭션 ID
-5353624116336947000
httpURL
DB Connection
경과 시간
17,348ms
Thread ID
51
DB Type
PostgreSQL
Sid
413662
HTTP 메소드
GET
Thread Stat
RUNNABLE
프로젝트 코드
2982
Project Name
에이전트 ID (oid)
-180752659
에이전트 명 (oname)
petapp8

> **Java 에이전트 2.2.40** 이전 버전에서는 **액티브 트랜잭션** 창에서 **ThreadStat** 값이 숫자 형식으로 표시됩니다. 문자 형식으로 표시하려면 Java 에이전트 2.2.40 버전 이상을 설치하세요.

액티브 트랜잭션

현재 진행 중인 트랜잭션과 연계된 데이터베이스의 SQL 성능을 조회할 수 있습니다. **대시보드** > **액티브 트랜잭션** 메뉴로 이동하세요.

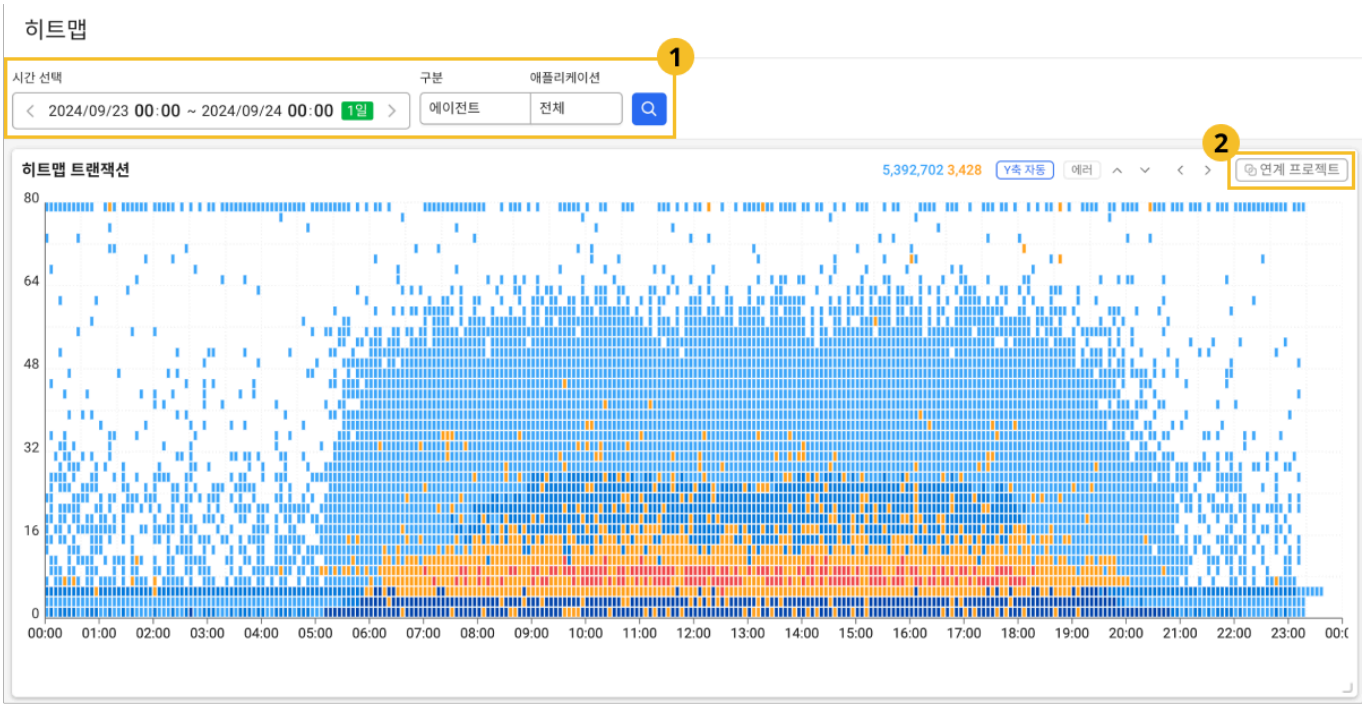


1. 트랜잭션 목록에서 조회하길 원하는 트랜잭션 항목을 선택하세요.
2. **액티브 트랜잭션** 창이 나타나면 오른쪽 상단의 **연계 DB 세션** 버튼을 선택하세요.
3. **연계 데이터베이스 정보** 창이 나타나면 관련 지표를 확인하세요.

- 연계분석을 위해 등록한 데이터베이스 프로젝트의 DB가 현재 선택한 트랜잭션의 SQL을 실행한 경우에만, **액티브 트랜잭션** 창에 **연계 DB 세션** 버튼이 표시됩니다.
- **연계 데이터베이스 정보** 창에 대한 자세한 내용은 다음 문서를 참조하세요.
- **Java 에이전트 2.2.40** 이전 버전에서는 **액티브 트랜잭션** 창에서 **ThreadStat** 값이 숫자 형식으로 표시됩니다. 문자 형식으로 표시하려면 Java 에이전트 2.2.40 버전 이상을 설치하세요.
- 이 기능을 지원하는 데이터베이스는 **PostgreSQL**, **Oracle**, **Oracle Pro**, **MySQL** 플랫폼입니다.

히트맵

트랜잭션을 조회하는 동안 연계된 프로젝트의 성능 지표를 함께 확인할 수 있습니다. 이 기능은 APM 프로젝트에서 트랜잭션을 분석할 때, 느린 SQL을 발견했을 경우 비슷한 여러 DB를 사용하는 환경에서 정확히 어느 DB에서 문제가 발생했는지 확인하기 어려운 상황에 유용합니다.



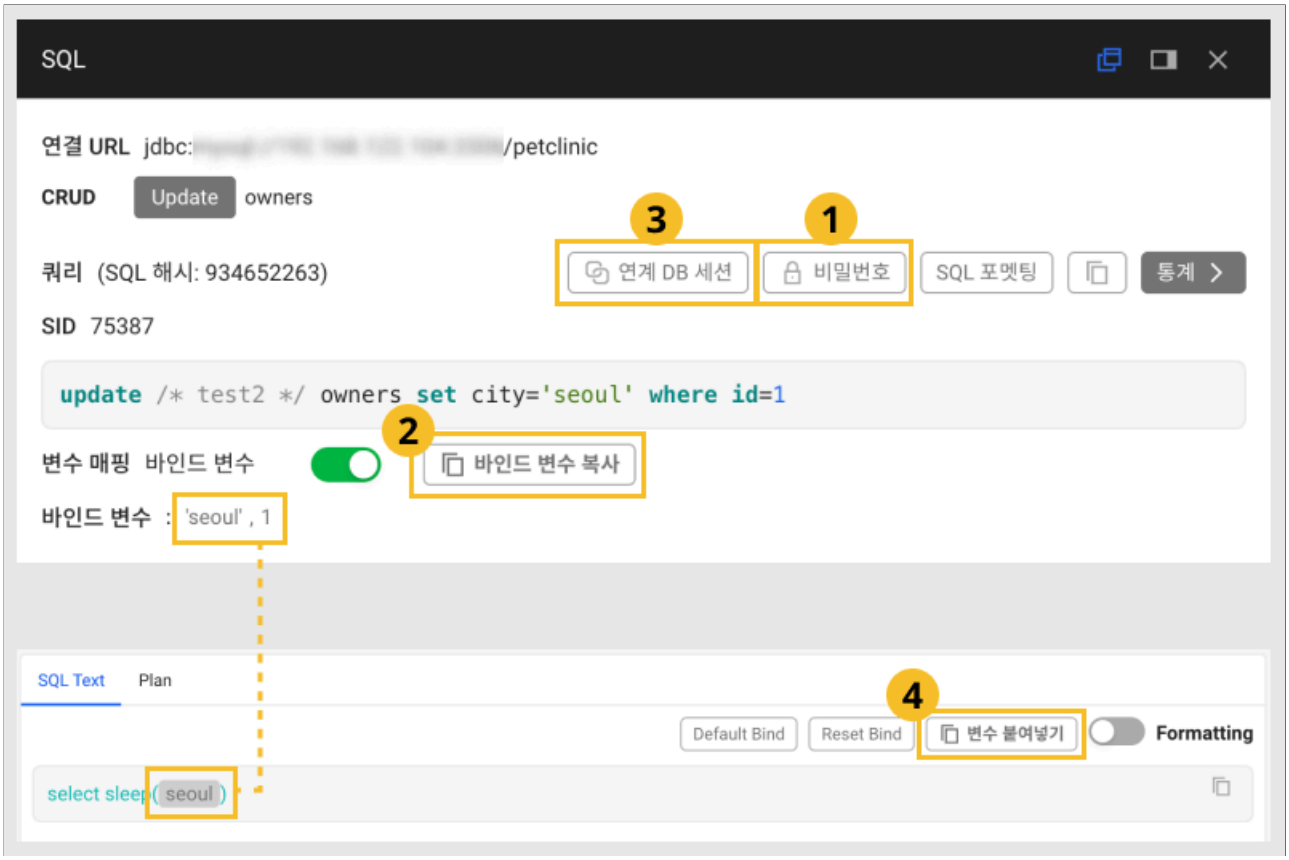
1. 트랜잭션을 조회할 시간과 **애플리케이션**을 설정한 후 버튼을 선택하세요.
2. **히트맵 트랜잭션** 섹션에서 오른쪽 상단의 **연계 프로젝트** 버튼을 선택하세요.
3. 연계 프로젝트의 지표를 확인할 수 있는 새창이 나타납니다.

왼쪽 목록에서 연계 프로젝트를 선택하고 **인스턴스 대시보드** 섹션의 지표를 확인하세요.

> ⓘ **인스턴스 대시보드** 창에 대한 자세한 내용은 다음 문서를 참조하세요.

바인드 변수 복사/붙여넣기

SQL 스텝에서 확인한 바인드 변수 값을 복사해 **연계 데이터베이스 정보** 창의 **SQL Text** 탭에서 붙여넣기할 수 있습니다. **트레이스 분석** 창에서 SQL 스텝을 확인하세요.

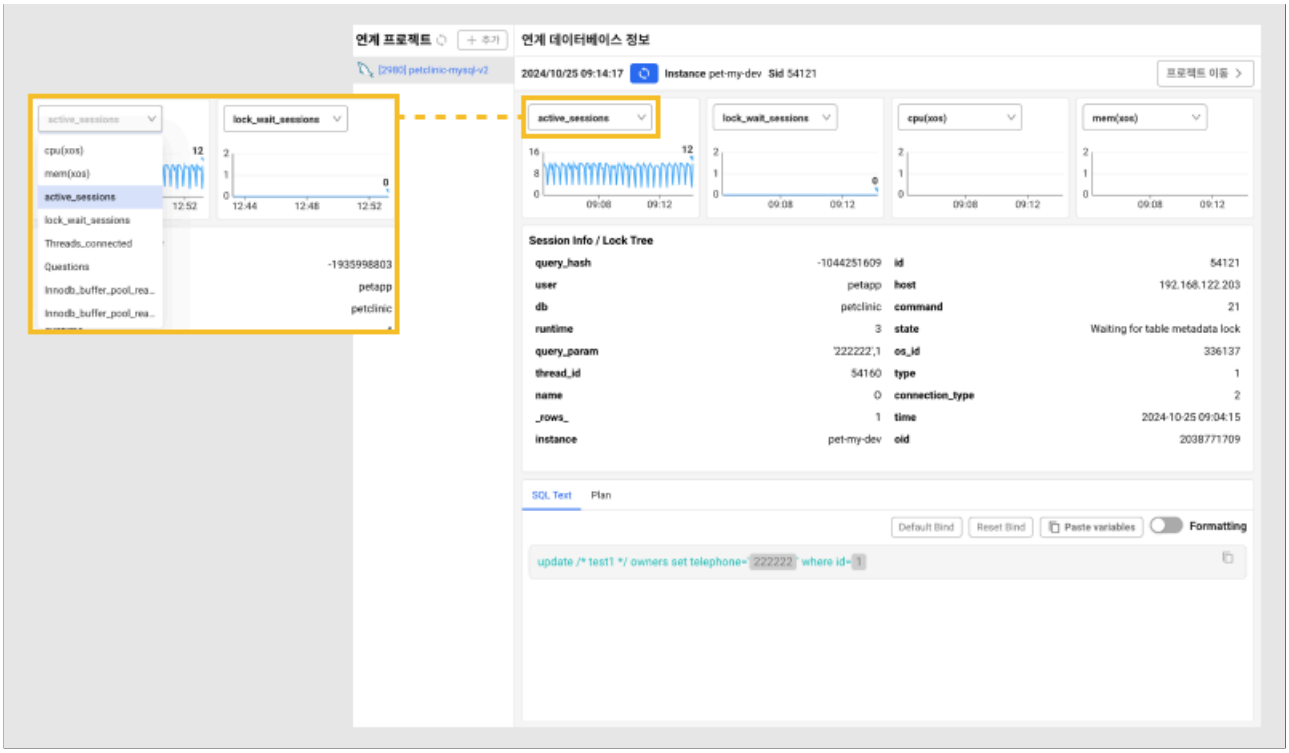


1. **SQL** 창에서 **비밀번호** 버튼을 선택하고 보안키를 입력하세요.
2. 바인드 변수를 확인하고 **바인드 변수 복사** 버튼을 선택하세요.
3. **연계 DB 세션** 버튼을 선택하세요.
4. **연계 데이터베이스 정보** 창이 나타나면 **SQL Text** 탭에서 **변수 붙여넣기** 버튼을 선택하세요.

> - **연계 데이터베이스 정보** 창에서 SQL 스텝을 확인하는 방법에 대한 자세한 내용은 다음 문서를 참조하세요.
> - 보안키와 관련한 에이전트 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

연계 데이터베이스 정보

연계 데이터베이스 정보에서는 애플리케이션에서 현재 진행 중인 트랜잭션과 연계된 데이터베이스의 SQL 성능을 확인할 수 있습니다.



- 화면 상단의 차트에서는 데이터베이스의 주요 지표에 대한 차트를 제공합니다. 선택 상자를 선택해 다른 지표를 조회할 수도 있습니다.
- **Session Info / Lock Tree** 섹션에서는 데이터베이스 세션과 관련한 다양한 세부 정보를 제공합니다.

  Session Info / Lock Tree

  > - **query_hash**: 실행 중인 SQL 쿼리의 고유 해시 값입니다.

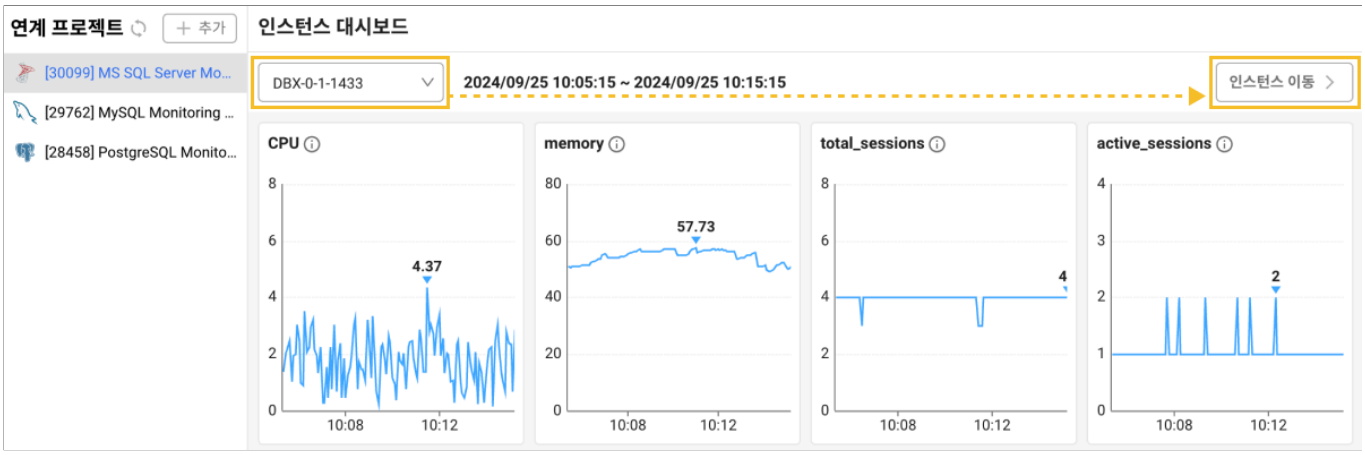
- **user**: 데이터베이스에 연결된 사용자 이름입니다.
- **db**: 연결된 데이터베이스 이름입니다.
- **runtime**: 해당 세션이 실행된 시간(밀리초 단위)입니다.
- **query_param**: 쿼리에서 사용된 매개변수입니다.
- **thread_id**: 이 세션과 연결된 스레드의 고유 식별자입니다.
- **name**: 해당 세션과 관련된 이름입니다.
- **rows**: 실행된 쿼리에서 처리된 행(row)의 수입니다.
- **state**: 해당 세션의 상태입니다.
- **id**: 현재 세션의 고유 식별자입니다.
- **host**: 데이터베이스가 실행 중인 서버의 IP 주소입니다.
- **command**: 현재 실행 중인 명령어 또는 작업을 나타냅니다.
- **os_id**: 운영 체제에서 사용되는 해당 세션의 ID입니다.
- **type**: 해당 세션의 유형입니다.
- **connection_type**: 데이터베이스 연결의 유형입니다.
- **time**: 해당 세션이 시작된 시간입니다.
- **oid**: 해당 세션이 속한 인스턴스(에이전트)의 고유 ID입니다.

- **SQL Text** 탭에서는 실행된 SQL 쿼리의 실제 내용을 확인할 수 있습니다.
  - **Default Bind**: 쿼리 문장에서 변수로 설정된 부분을 기본값으로 바인딩할 수 있습니다.
  - **Reset Bind**: 바인딩된 기본값을 해제하고 변수를 확인할 수 있습니다.
  - **변수 붙여넣기**: 복사한 바인드 변수 값을 붙여넣기할 수 있습니다. 자세한 내용은 다음 문서를 참조하세요.
  - **Formatting**: SQL 쿼리문에 들여쓰기 및 포맷팅을 적용해 가독성을 높일 수 있습니다.
- **Plan**: Plan 정보를 확인할 수 있습니다. **DB Name**, **User Name**, **Password**를 입력한 다음 버튼을 선택하세요.

인스턴스 대시보드

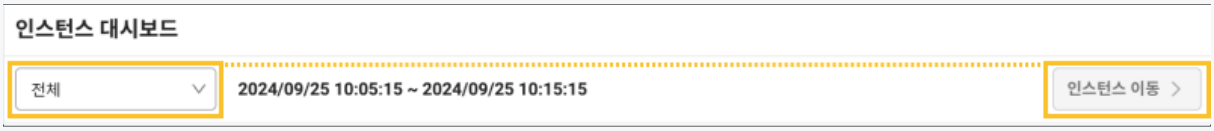
인스턴스 대시보드에서는 애플리케이션에서 실행되었던 트랜잭션과 연계된 데이터베이스의 주요 성능 지표를 확인할 수

있습니다.



선택한 프로젝트의 대시보드 메뉴로 이동하려면 인스턴스를 선택한 후, 오른쪽 상단의 **인스턴스 이동** 버튼을 클릭하세요. **대시보드** > **인스턴스 모니터링** 메뉴로 이동합니다.

- **히트맵 트랜잭션** 섹션의 **연계 프로젝트** 버튼은 연계된 프로젝트가 있을 때만 표시됩니다. 연계 프로젝트 추가에 대한 자세한 내용은 다음 문서를 참조하세요.
- 왼쪽 **연계 프로젝트** 목록 상단의 **추가** 버튼을 선택해 프로젝트를 추가할 수 있습니다.
- **인스턴스 이동** 버튼은 **전체**를 선택한 상태에서는 비활성화됩니다.



- 데이터베이스 제품 종류에 따라 **분석** > **카운트 추이** 메뉴로 이동할 수 있습니다.

연계 프로젝트 삭제하기

1. **관리** > **연계 프로젝트 관리** 메뉴로 이동하세요.
2. **연계 프로젝트 목록** 섹션에서 연계를 해제하려는 프로젝트를 삭제하려면 목록 가장 오른쪽의 버튼을 선택하세요.
3. 확인 메시지가 나타나면 **삭제** 버튼을 선택하세요.

연계 프로젝트 목록 섹션에서 삭제한 프로젝트가 제외됩니다.

Log 모니터링

로그는 애플리케이션 및 시스템에서 발생하는 이벤트와 메시지 등을 기록한 파일입니다. 이상 징후를 파악해 시스템 악화를 막거나 발생한 장애의 원인을 이해하고자 한다면 로그를 확인하는 것이 중요합니다.

최근 IT 서비스 구축 환경은 MSA(Microservice Architecture) 또는 Kubernetes 기반으로 전환되면서 관리 대상이 증가하고 있습니다. 일반적인 로그 모니터링은 서버에 접속해 `tail` 명령어나 편집기를 통해 확인합니다. 하지만 개별 서버에 일일이 접속하는 등의 단순한 방법을 사용하기 어렵습니다. 경우에 따라 수백 또는 수천 대의 서버에서 발생하는 로그를 어떻게 확인할 수 있을까요?

와탭 로그 모니터링 서비스를 통해 수많은 로그를 쉽게 관리할 수 있습니다.

> ⓘ 리눅스 `tail` 명령어는 시간에 따라 내용이 추가되는 로그 등을 확인하기 위한 용도로 많이 사용됩니다.

주요 특장점

- **중앙 통합 관리**

  와탭은 대량의 로그를 중앙에서 통합 관리합니다. 개별 서버에 접근하지 않아도 중앙에서 로그를 확인할 수 있어 편리합니다.

- **모든 로그 수집**

  로그를 선별해서 수집하는 경우 중요한 데이터가 누락될 수 있습니다. 와탭은 모든 로그를 수집하며, 수집된 로그들은 **라이브 테일**에서 실시간으로 확인할 수 있습니다.

- **가시성 확보**

  와탭의 다양한 차트에서 로그를 확인할 수 있습니다. 이를 통해 에러 및 이슈 정보에 대한 접근성을 높이고 장애 상황을 조기에 감지할 수 있게 합니다.

- **유연한 용량 관리**

  개별 서버에 로그 적재 시 로그로 인해 파일 시스템 용량이 과도하게 점유되는 문제가 발생할 수 있습니다. 와탭을 통해 로그를 중앙에 모은다면, 개별 서버에서 발생하는 로그 파일을 유지할 필요없이 중앙에 적재된 로그 데이터의 유지 기간만 관리하면 됩니다.

- **다양한 분석 관점**

장애 상황을 파악하고 예측하기 위해 다양한 관점으로 로그를 분석할 수 있어야 합니다. 와탭은 특정 태그를 포함한 로그 건수 추이를 확인하거나 필터링하여 분석할 수 있으며, 자주 사용하는 패턴은 차트로 저장해 언제든지 조회할 수 있습니다.

- **패턴 알림**

  장애 패턴을 알림으로 설정하여 문제를 예방하거나 빠르게 인지할 수 있습니다. 와탭은 개별 로그를 기준으로 특정 키워드가 포함되면 알림을 받는 **실시간 로그 알림**과, 특정 태그가 포함된 로그 건수 추이를 기준으로 알림을 받는 **복합 로그 알림**을 제공합니다.

로그 분석

라이브 테일

라이브 테일 메뉴에서 실시간으로 수집된 로그를 `tail` 명령어처럼 화면을 통해 흘러가는 로그를 조회할 수 있습니다.

> ⓘ 자세한 내용은 라이브 테일 문서를 참고하세요.

로그 트렌드

로그 트렌드 메뉴에서 수집되는 전체 로그 또는 특정 태그가 포함된 로그의 건수 추이를 확인할 수 있습니다. 로그 발생 건수는 장애 발생 및 해소 시점과 밀접하게 연관되므로, 로그 발생 건수 추이를 통해 장애 원인을 분석하고 대응 속도를 높일 수 있습니다.

> ⓘ 자세한 내용은 로그 트렌드 문서를 참고하세요.

로그 검색

로그 검색 메뉴에서 수집되는 전체 로그 또는 특정 태그가 포함된 로그를 조회할 수 있습니다. 특정 시간대나 서버에서 발생한 로그를 태그를 기준으로 조회하고 확인할 수 있습니다. 인접 로그 기능을 사용하면 특정 Error 또는 Exception 발생 전후의 로그를 함께 확인하여 원인 분석에 활용할 수 있습니다.

> ⓘ 자세한 내용은 로그 검색 문서를 참고하세요.

로그 탐색

로그 탐색 메뉴를 통해 로그 데이터를 빠르게 검색, 필터링, 분석할 수 있습니다. 필드 구조를 파악하고, 쿼리 기반 필터링을 통해 필요한 로그를 효율적으로 검색해, 결과를 데이터 시각화할 수 있습니다. 로그 탐색 기능은 Lucene 인덱스 기반으로 동작합니다.

> ⓘ 자세한 내용은 로그 탐색 문서를 참고하세요.

로그 모니터링 적용하기

와탭 로그 모니터링 서비스 이용을 위한 기본 적용 방법을 안내합니다. 와탭 로그 모니터링은 추가적인 에이전트를 구성하거나 로그에 맞추어 parser를 적용할 필요가 없습니다. 간단한 설정으로 빠르게 시작할 수 있습니다.

로그 모니터링의 작동 원리



일반적인 로그 통합 서비스는 수집기, 처리기, 저장소 그리고 UI 모듈로 이루어져 있습니다. 단계별 설정과 구성 작업을 필요로 하기에 각각의 모듈을 구축하는 과정이 번거로우며 추가 비용이 발생합니다.

와탭 로그 모니터링은 적용이 간단합니다. 기존의 모니터링 에이전트가 수집기 역할을 하기에 에이전트 옵션을 켜는 것만으로 로그 모니터링을 시작할 수 있습니다.

- Java PHP Python Go Server Kubernetes

- 출력된 파일에서 로그를 읽지 않고 Java 애플리케이션의 로그 라이브러리로 전달되는 로그를 직접 수집합니다.
- 로그를 직접 수집하기 때문에 파일 I/O를 유발하지 않아 시스템에 미치는 성능 영향이 매우 낮습니다.
- 트랜잭션 트레이스와 로그의 연결 추적성을 확보하여 트레이스에서 로그를 확인할 수 있습니다.

> **Java 로그 라이브러리**
>
> 대표적인 Java 로그 라이브러리는 Apache Log4j, Logback 입니다.

Java Agent 2.1.1 버전부터 사용할 수 있습니다.

- 기존의 에이전트에 로그 수집 기능을 추가했습니다. 모니터링 에이전트가 로그 파일에 추가로 출력된 로그를 읽어 수집하는 방식을 활용합니다.

PHP Agent 2.3.2 버전부터 사용할 수 있습니다.

- 기존의 에이전트에 로그 수집 기능을 추가했습니다. 모니터링 에이전트가 로그 파일에 추가로 출력된 로그를 읽어 수집하는 방식을 활용합니다.
- 로그에 트랜잭션 ID를 출력하면, 트랜잭션 트레이스와 로그의 연결 추적성을 확보하여 트레이스에서 로그를 확인할 수 있습니다.

Python Agent 1.2.2 버전부터 사용할 수 있습니다.

- 기존의 에이전트에 로그 수집 기능을 추가했습니다. 모니터링 에이전트가 로그 파일에 추가로 출력된 로그를 읽어 수집하는 방식을 활용합니다.
- 기존의 에이전트에 로그 수집 기능을 추가했습니다. 모니터링 에이전트가 로그 파일에 추가로 출력된 로그를 읽어 수집하는 방식을 활용합니다.

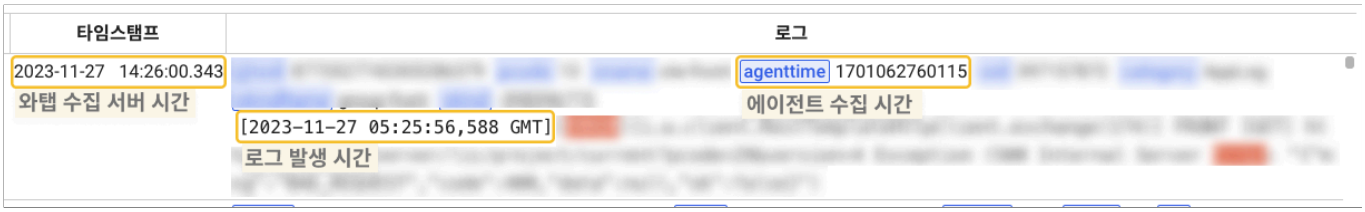
Server Agent 2.1.2 버전부터 사용할 수 있습니다.

- 쿠버네티스 컨테이너에 로그를 수집할 수 있습니다.

- 쿠버네티스 컨테이너 내부 애플리케이션의 로그를 수집할 수 있습니다.

> Kubernetes Agent 1.1.35 버전부터 사용할 수 있습니다.

로그 수집 시간 기준



로그 **수집 시간** 기준 시간이 **와탭 에이전트가 로그를 수집한 시간**에서 **와탭 수집 서버의 로그 처리 시간**으로 변경되었습니다.

일반적인 상황에서 변경 전과 큰 차이가 없으며, 기존 방식과 동일하게 로그를 검색할 수 있습니다. 다음과 같은 경우도 사용자의 추가적인 수정없이 로그 모니터링을 일관적으로 사용할 수 있습니다.

- NTP 사용 시, 모니터링 대상의 서버 시간이 표준 시간보다 과거 또는 미래 시간으로 설정되어있는 경우
- NTP 미사용 시, 2개 이상인 모니터링 대상의 서버 시간이 서로 다른 경우

타임셀렉터 조회 범위 지정

수집 시간 기준 시간 변경으로 타임셀렉터 역시 **와탭 수집 서버 시간**을 기준으로 동작합니다. 이에 따라 **에이전트 수집 시간**과 **와탭 수집 서버 시간**에 차이가 생겨 조회 범위에 포함되지 않는 로그가 발생할 수 있습니다. 이 경우 조회 범위를 넓게 지정 시 조회가 가능합니다.

로그 트렌드

로그 트렌드 추이 차트의 X축은 최소 1분 기준입니다. 로그 기준 시간이 변경된 후에도 차트를 통한 전체적인 추이 파악에 영향을 주지 않습니다.

로그 검색

로그 메시지에 로그 생성 시간을 나타내는 인덱스 `agenttime`이 추가되었습니다. `agenttime` 값을 통해 에이전트 수집 시간을

확인하세요.

> ⓘ UTC를 따르는 **와탭 수집 서버 시간**은 사용자 브라우저 시간에 따라 변환해 표기합니다.
>
> 예, 한국의 경우 UTC+9 기준으로 시간을 표기합니다.

단계별 기준 시간

로그는 3단계를 거쳐 수집됩니다. 단계마다 서로 다른 기준 시간이 사용될 수 있습니다.

와탭 에이전트 수집 시간과 **와탭 수집 서버 시간**은 큰 차이가 있습니다. 모니터링 대상의 서버 시간을 확인하세요. 모니터링 대상의 서버 시간이 과거 또는 미래 시간으로 설정된 경우 에이전트 수집 시간에 영향을 끼칩니다.

1. **로그 발생 시간**

   모니터링 대상의 시간 또는 Logging 정책에 따라 편차가 발생할 수 있습니다.

2. **에이전트 수집 시간**

   사용 중인 제품 또는 에이전트가 로그를 수집하는 방법 및 생성되는 로그에 따라 편차가 발생할 수 있습니다.

   - **Application**

     설정에 따라 로그 라이브러리 또는 로그 파일에서 실시간에 가깝게 로그를 수집합니다.

     > ⓘ 로그 라이브러리는 Java 제품만 지원합니다.

   - **Server**

     로그 파일에서 실시간에 가깝게 로그를 수집합니다.

   - **AWS Log**

     AWS Resource 정책에 따라 준 실시간 또는 수 분마다 로그를 수집합니다.

3. **와탭 수집 서버 시간**

   모니터링 대상 또는 로그 생성 방법과 상관없이 수집 서버에 저장되는 시간을 사용합니다.

로그 모니터링 적용하기

사용하는 애플리케이션에 따른 적용 방법은 다음과 같습니다. 로그 모니터링을 적용하기 전, 지원 버전을 먼저 확인하세요.

1. 지원하는 에이전트 버전을 확인하고 **업데이트**하세요.
2. 로그 모니터링 **옵션**을 설정하세요.
3. 로그 모니터링을 **활성화**하세요.

Java

Java 애플리케이션에서 로그를 수집하는 방법을 안내합니다.

PHP

PHP 애플리케이션에서 로그를 수집하는 방법을 안내합니다.

Node.js

Node.js 애플리케이션에서 로그를 수집하는 방법을 안내합니다.

Python

Python 애플리케이션에서 로그를 수집하는 방법을 안내합니다.

Go

Go 애플리케이션에서 로그를 수집하는 방법을 안내합니다.

Server

Server 애플리케이션에서 로그를 수집하는 방법을 안내합니다.

Kubernetes

쿠버네티스 컨테이너와 컨테이너 내부 애플리케이션의 로그를 수집하는 방법을 안내합니다.

Java

Java 애플리케이션에서 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 업데이트

자바 에이전트 2.1.1 버전부터 가능합니다. 업데이트 방법은 다음 문서를 참고하세요.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기** 탭의 **1. 에이전트 설정 확인**의 안내를 참고해 에이전트 설정을 확인하세요.

에이전트 설정하기

1. 프로젝트의 **관리** > **에이전트 설정** 메뉴를 클릭하세요.
2. **옵션 작성**의 **옵션 선택** 드롭 다운 목록에서 **직접 입력**을 클릭한 후, `logsink_enabled=true` 옵션을 작성하고 **적용** 버튼을 클릭해 추가하세요.
   - 키 입력창에 `logsink_enabled`을 작성하세요.
   - 값 입력창에 `true`을 작성하세요.
3. 로그 모니터링을 적용하기 위해 애플리케이션을 다시 시작하세요.

주요 옵션

- **hooklog_enabled** Boolean

  기본값 `false`

  Log 라이브러리를 후킹(Hooking)하여 로그 모니터링을 활성화합니다.

> 애플리케이션 실행 전, whatap.conf에 본 옵션이 활성화되어 있어야 이후 로그 모니터링의 On/Off를 `logsink_enabled` 설정을 통해 동적으로 제어할 수 있습니다. 애플리케이션 실행 전에 `logsink_enabled` 옵션이 `true` 로 설정된 경우 본 옵션을 별도로 설정하지 않아도 로그 모니터링이 가능합니다.
>
> 앞으로 로그 모니터링을 활용할 가능성이 있다면, 사전에 본 옵션을 꼭 설정할 것을 권장합니다.

- **hooklog_custom_methods**

  사용자 정의 로그를 등록합니다. 임의의 로그 프레임워크 내용을 전달합니다. 사이트에서 개별로 만든 로그 모듈의 로그를 추적할 때 사용합니다.

  Java

```java
package io.home.test;

public class MyLog {
  public void customLog(String log) { ... }
}
```

  whatap.conf

```
hooklog_custom_methods=io.home.test.MyLog.customLog
```

- **logsink_enabled** Boolean

  기본값 `false`

  Log 모니터링 기능을 On/Off 합니다.

> 애플리케이션 실행 전, whatap.conf에 `hooklog_enabled` 옵션이 설정되어 있으면 본 옵션을 통해 로그 모니터링의 On/Off를 동적으로 제어할 수 있습니다.

- **logsink_trace_enabled** Boolean

  기본값 `false`

  Log에 트랜잭션 ID를 삽입하여, 트랜잭션 트레이스의 로그 탭을 노출할지 여부를 지정합니다.

로그 모니터링 활성화

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

로그 모니터링 시작하기 탭의 **2. 로그 모니터링 활성화**에서 와탭 로그 모니터링을 활성화 또는 비활성화 할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

PHP

PHP 애플리케이션에서 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 업데이트

PHP 에이전트 2.3.2 버전부터 가능합니다. 업데이트 방법은 다음 문서를 참조하세요.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기** 탭의 **1. 에이전트 설정 확인**의 안내를 참고해 에이전트 설정을 확인하세요.

설정 파일 경로 확인

로그 모니터링을 원하는 파일의 **설정 파일 경로**를 확인하세요. CLI 환경에서 확인하거나 웹에서 확인할 수 있습니다.

CLI 환경에서 확인하기

```
$ php -i | grep 'Scan'
Scan this dir for additional .ini files => /etc/php8.0.d
```

> ⓘ 해당 경로가 "(None)"인 경우 설정 파일의 경로는 /usr/whatap/php 입니다.

웹에서 확인하기

CLI 환경과 Apache 환경의 PHP 설정이 다르면 웹에서 `phpinfo()` 함수의 결과를 확인하세요.

명령어 입력

설정 파일 경로를 포함한 명령어를 입력하면 로그 모니터링이 시작됩니다.

```
export LOGFILES=/some/path/file1,/some/other/file2
echo "whatap.logsink_enabled=true" | sudo tee -a [설정파일경로]/whatap.ini
echo "whatap.logsink.files=$LOGFILES" | sudo tee -a [설정파일경로]/whatap.ini
```

로그 모니터링 활성화

홈 화면 > 프로젝트 선택 > 로그 > 로그 설정

로그 모니터링 시작하기 탭의 **2. 로그 모니터링 활성화**에서 와탭 로그 모니터링을 활성화 또는 비활성화할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

Node.js

Node.js 애플리케이션에서 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 업데이트

Node.js 에이전트 0.5.1 버전부터 가능합니다. 업데이트 방법은 다음 문서를 참조하세요.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기** 탭의 **1. 에이전트 설정 확인**의 안내를 참고해 에이전트 설정을 확인하세요.

에이전트 설정하기

1. 프로젝트의 **관리** > **에이전트 설정** 메뉴를 클릭하세요.
2. **옵션 작성**의 **옵션 선택** 드롭 다운 목록에서 **직접 입력**을 클릭한 후, `logsink_enabled=true` 옵션을 작성하고 **적용** 버튼을 클릭해 추가하세요.
   - 키 입력창에 `logsink_enabled` 을 작성하세요.
   - 값 입력창에 `true` 을 작성하세요.
3. 로그 모니터링을 적용하기 위해 애플리케이션을 다시 시작하세요.

주요 옵션

- **logsink_enabled** Boolean

  기본값 `false`

  로그 모니터링 활성화 여부를 설정합니다.

- **logsink_trace_txid_enabled** Boolean

  기본값 `true`

  로그에 트랜잭션 ID를 삽입하여 트랜잭션 트레이스의 로그 탭 노출 여부를 설정합니다.

- **logsink_limit_content_enabled** Boolean

  기본값 `true`

  로그 메시지의 최대 크기 제한 여부를 설정합니다.

- **logsink_limit_content_length** Number

  기본값 `10,000`

  로그 메시지의 최대 길이를 설정합니다. 이 값은 `logsink_limit_content_enabled` 옵션이 `true`로 설정된 경우에만 적용됩니다.

로그 모니터링 활성화

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

로그 모니터링 시작하기 섹션의 **로그 모니터링 활성화** 탭에서 토글 버튼으로 와탭 로그 모니터링을 활성화 또는 비활성화할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

Python

Python 애플리케이션에서 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 업데이트

Python 에이전트 1.2.2 버전부터 가능합니다. 업데이트 방법은 다음 문서를 참조하세요.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기** 탭의 **1. 에이전트 설정 확인**의 안내를 참고해 에이전트 설정을 확인하세요.

로그 수집 활성화

설정 파일 경로(WHATAP_HOME)를 포함한 명령어를 입력하면 로그 수집이 바로 시작됩니다.

```
export LOGFILES={로그파일전체경로},...
echo "logsink_enabled=true" | sudo tee -a {설정파일경로}/whatap.conf
echo "logsink.files=$LOGFILES" | sudo tee -a {설정파일경로}/whatap.conf
```

로그와 웹 트랜잭션 연동

1.3.6 이후 버전

1.3.6 버전부터 다음의 방법으로 트랜잭션과 로그 연동을 설정할 수 있습니다. 현재 와탭은 Python의 logging, loguru 라이브러리를 지원하고 있습니다. 사용하는 Python Log 라이브러리에 따라 whatap.conf를 구성하세요.

- logging 모듈

logging

```
echo "trace_logging_enabled=true" | sudo tee -a {설정파일경로}/whatap.conf
```

- loguru 모듈

loguru

```
echo "trace_loguru_enabled=true" | sudo tee -a {설정파일경로}/whatap.conf
```

1.3.6 미만 버전

트랜잭션 별로 발생한 로그를 별도로 조회할 수 있도록 트랜잭션 아이디 `{txid}` 를 로그에 출력합니다. 와탭 모니터링에서 Python LogRecord에 `{txid}` 를 자동 주입하여 포매터 설정 시 로그 파일에 `{txid}` 를 출력할 수 있도록 합니다.

```
settings.py
...
LOGGING = {
...
    'formatters': {
        ...
    },
    'handlers': {
        ...
    },
    'loggers': {
        ...
        '{로거이름}': {
            'handlers': [...],
        ...
        },
    },
}
```

```
try:
    import whatap.trace.mod.logging as whatap_logging
    if whatap_logging.logging_injection_processed:
        LOGGING['formatters']['whatap.formatter']={
            '()': 'django.utils.log.ServerFormatter',
            'format': '[{server_time}] -- {{ "@txid" : "{txid}" }} -- {message}',
            'style': '{',
        }
        LOGGING['handlers']['whatap']={
            'level': 'DEBUG',
            'class': 'logging.handlers.RotatingFileHandler',
            'filename': os.path.join(BASE_DIR, 'logs','whatap_log.log'),
            'formatter': 'whatap.formatter',
        }
        LOGGING['loggers'][{로거이름}]['handlers'].append('whatap')
except:
    pass
...
```

로그 모니터링 활성화

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

로그 모니터링 시작하기 탭의 **2. 로그 모니터링 활성화**에서 와탭 로그 모니터링을 활성화 또는 비활성화할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

Go

Go 애플리케이션에서 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기** 탭의 **1. 에이전트 설정 확인**의 안내를 참고해 에이전트 설정을 확인하세요.

로그 수집 활성화

아래 명령어를 입력하면 로그 수집이 바로 시작됩니다.

```
export LOGFILES=/some/path/file1,/some/other/file2
echo "logsink_enabled=true" | sudo tee -a /usr/whatap/agent/whatap.conf
echo "logsink.files=$LOGFILES" | sudo tee -a /usr/whatap/agent/whatap.conf
```

로그 모니터링 활성화

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

로그 모니터링 시작하기 탭의 **2. 로그 모니터링 활성화**에서 와탭 로그 모니터링을 활성화 또는 비활성화할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

Server

서버 애플리케이션에서 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 업데이트

서버 에이전트 2.1.2 버전부터 가능합니다. 업데이트 방법은 다음 문서를 참고하세요.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기**의 **1. 에이전트 설정 확인**의 **OS 선택** 탭에서 **리눅스(shell)** 또는 **윈도우(Powersehll, 관리자 권한)** 를 선택하세요.
3. **기본 설치** 또는 **카테고리와 함께 설치**를 참고해 에이전트 설정을 확인하세요.

> ⓘ 다음 설정은 에이전트 재시작이 필요합니다.

Linux Shell

- **기본 설치**

  Linux Shell

```
export LOGFILES=/some/path/file1,/some/other/file2
echo "logsink.files=$LOGFILES" | sudo tee -a /usr/whatap/infra/conf/whatap.conf
```

- **카테고리와 함께 설치**

  Linux Shell

```
cd /usr/whatap/infra
sudo mkdir extension

cat <<EOL | sudo tee extension/logsink.conf > /dev/null
[[inputs.logsink]]
 category = "serverlog"
 ## 로그 발생량 통계 별도 데이터로 전송 여부
 stats_enabled = true
 ## 로그 발생량 통계 카테고리
 stats_category = "logsink_stats"
 ## 로그 파일 경로(path)에 별표(*)가 포함되어 제외할 로그 비대상 파일명 설정
 excludeNames = [ ".gz",".zip" ]
 [[inputs.logsink.file]]
  ## 로그 파일 지정 시, 날짜 패턴(strftime.org) 지정 가능
  path = "/some/path/%Y-%m-%d/.log"
  disabled = false
  encoding = "euc-kr"

 [[inputs.logsink.file]]
  path = "/some/other/log"
  disabled = false
  encoding = "utf-8"

 [[inputs.logsink.file]]
  ## 줄 단위 로그에서 해당 키워드 검색 시, 이전 로그에 병합
  nowrap_keywords = ["Caused by:", "Test"]

 [[inputs.logsink.file]]
 ## 프로젝트 코드 및 프로젝트 라이선스 입력 시, 로그를 같은 리전의 원하는 프로젝트로 전송 가능
 # pcode = 프로젝트 코드
 # license = "프로젝트 라이선스"

 [[inputs.logsink.file]]
 # whatap_host_ip = "수집 서버 아이피1/수집 서버 아이피2"
 # whatap_host_port = 6600

 [[inputs.logsink.file]]
 path = "/tmp/mixed.log"
 disabled = false
```

```
  encoding = "utf-8"
  ## [{"A":1,"B":2},{"C":3,"D":4},...] 형태를 각각의 로그로 분리할 수 있다.{"A":1,"B":2} {"C":3,"D":4}
  multiline_pattern = "simple_json_array"
EOL

sudo service whatap-infra restart
```

Windows Powershell

- **기본 설치**

  Windows Powershell

```
$LOGFILES="c:\\whatap\\logs\\%Y-%m-%d\\*.log,c:\\whatap\\logs\\*.log"
Add-Content "c:\Program Files\WhatapInfra\whatap.conf" -Value "logsink.files=$LOGFILES"
```

- **카테고리와 함께 설치**

  Windows Powershell

```
# 관리자 권한 필요
New-Item -type "Directory" -Path "C:\Program Files\WhatapInfra\extension"

$contentToAdd = @"
[[inputs.logsink]]
category = "serverlog"
## 로그 발생량 통계 별도 데이터로 전송 여부
stats_enabled = true
## 로그 발생량 통계 카테고리
stats_category = "logsink_stats"
## 로그 파일 경로(path)에 별표(*)가 포함되어 제외할 로그 비대상 파일명 설정
excludeNames = [ ".gz",".zip" ]
[[inputs.logsink.file]]
## 로그 파일 지정 시, 날짜 패턴(strftime.org) 지정 가능
path = "c:\\whatap\\logs%Y-%m-%d_.log"
disabled = false
encoding = "euc-kr"

```

```
[[inputs.logsink.file]]
## 줄 단위 로그에서 해당 키워드 검색 시, 이전 로그에 병합
nowrap_keywords = ["Caused by:", "Test"]

[[inputs.logsink.file]]
## 프로젝트 코드 및 프로젝트 라이선스 입력 시, 로그를 같은 리전의 원하는 프로젝트로 전송 가능
# pcode = 프로젝트 코드
# license = "프로젝트 라이선스"

[[inputs.logsink.file]]
# whatap_host_ip = "수집 서버 아이피1/수집 서버 아이피2"
# whatap_host_port = 6600

[[inputs.logsink.file]]
path = "/tmp/mixed.log"
disabled = false
encoding = "utf-8"
## [{"A":1,"B":2},{"C":3,"D":4},...] 형태를 각각의 로그로 분리할 수 있다.{"A":1,"B":2} {"C":3,"D":4}
multiline_pattern = "simple_json_array"

"@

New-Item -path "C:\Program Files\WhatapInfra\extension" -name "logsink.conf" -type "file" -value $contentToAdd
-Force

Restart-Service "Whatap Infra"
```

옵션 설정

- **stats_enabled** : 수집 현황 데이터의 수집 여부를 설정합니다. (기본값: false)
  - true 로 설정해야 하며, stats_category 에서 설정한 카테고리(logsink_stats)로 통계 데이터가 발생합니다.
- **stats_category** : 수집 현황 데이터를 저장할 메트릭스 카테고리를 설정합니다. logsink_stats 로 값을 설정해야 합니다.
  통계 데이터 필드는 다음과 같습니다.

  저장되는 통계 데이터 필드: file checkInterval encoding filepos checkedLocalTime lastupdatedLocalTime fileSize error firstCheck transferBytes

- `excludeNames` : 로그 파일 경로(path)에 별표(`*`)를 포함한 경우 로그 비대상 파일을 제외하도록 파일명을 설정할 수 있습니다. 쉼표(`,`)를 구분자로 이용해 복수 설정할 수 있습니다.

  Example

  ```
  excludeNames = [ ".gz",".zip" ]
  ```

- `nowrap_keywords` : 줄 단위 로그 검색 시 해당 옵션값으로 지정한 키워드가 검색될 경우 이전 로그에 병합합니다.
- `pcode` : 로그를 전송할 같은 리전의 프로젝트 코드를 입력합니다.
- `license` : 로그를 전송할 같은 리전의 프로젝트의 라이선스 코드를 입력합니다.

> ⓘ `pcode` , `license` 옵션의 경우 서버 에이전트 2.7.4 버전 이상부터 지원합니다.

윈도우 이벤트 로그 옵션 설정

윈도우 이벤트 로그 수집 시 다음과 같이 옵션을 설정할 수 있습니다.

```
# 관리자 권한 필요
New-Item -type "Directory" -Path "C:\Program Files\WhatapInfra\extension"

$contentToAdd = @"
[[inputs.win_eventlog]]
    category = "win_event_log"
    stats_category = "win_event_log_stats"
    stats_enabled = true
    enabled = true
    [[inputs.win_eventlog.file]]
        #true | false
        enabled = true
        # Application, Security, Setup, System, ForwardedEvents
        file = "Application"
        #1: Information, 2: Warning 3: Error 4: Audit Success 5 Audit Fail
        #event_type =
        #event id
        #event_id =
        #event source name
        #source_name = ""
```

```
21      #프로젝트 코드 및 프로젝트 라이선스 입력 시, 로그를 같은 리전의 원하는 프로젝트로 전송 가능
22      #pcode = 프로젝트 코드
23      #license = "프로젝트 라이선스"
24      #whatap_host_ip = "수집 서버 아이피1/수집 서버 아이피2"
25      #whatap_host_port = 6600
26
27  "@
28
29  New-Item -path "C:\Program Files\WhatapInfra\extension" -name "win_eventlog.conf" -type "file" -value $contentToAdd
    -Force
30
31  Restart-Service "Whatap Infra"
```

- 카테고리 지정(`category`) 필수

  예, `win_event_log`

- 통계 카테고리 지정(`stats_category`) 필수

  예, `win_event_log_stats`

- 통계 카테고리 On/Off(`stats_enabled`) 필수

  예, `true` 혹은 `false`

- 수집 기능 On/Off(`enabled`) 필수

  예, `true` 혹은 `false`

- 파일별 수집 기능 On/Off(`enabled`) 필수

  예, `true` 혹은 `false`

- 파일(`file`) 필수

  예, `Application` , `Security` , `Setup` , `System` , `Forwarded`

- 이벤트 타입(`event_type`) 비필수

  예, `1` , `2` , `3` , `4` , `5`

> ⓘ **이벤트 타입**
>
> 1. Information
> 2. Warning
> 3. Error
> 4. Audit Success
> 5. Audit Fail

- 이벤트 아이디(`event_id`) 비필수
- 이벤트 소스 이름(`source_name`) 비필수
- 프로젝트 코드(`pcode`) 비필수
- 프로젝트 라이선스 (`license`) 비필수

> ⓘ **에이전트 지원 버전**
>
> - 윈도우 이벤트 로그 수집의 경우 서버 에이전트 2.5.2 버전 이상부터 지원합니다.
> - 윈도우 이벤트 로그 옵션 중 `pcode` , `license` 옵션의 경우 서버 에이전트 2.7.4 버전 이상부터 지원합니다.

다중 파일 지정 및 분리

다중 파일 지정

여러 로그 파일을 동일한 카테고리로 설정하려면, 각 로그 파일 경로를 `[[inputs.logsink.file]]` 항목에 추가하고, `path` 설정을 통해 파일 경로를 지정합니다. 또한, `category` , `stats_enabled` , `stats_category` 등의 옵션을 사용하여 로그 발생량 통계 설정과 같은 부가적인 설정을 할 수 있습니다.

/root/test1 및 /root/test2 파일을 동일한 카테고리(`serverlog`)로 지정한 예시입니다.

```
[[inputs.logsink]]
category = "serverlog"
stats_enabled = true
```

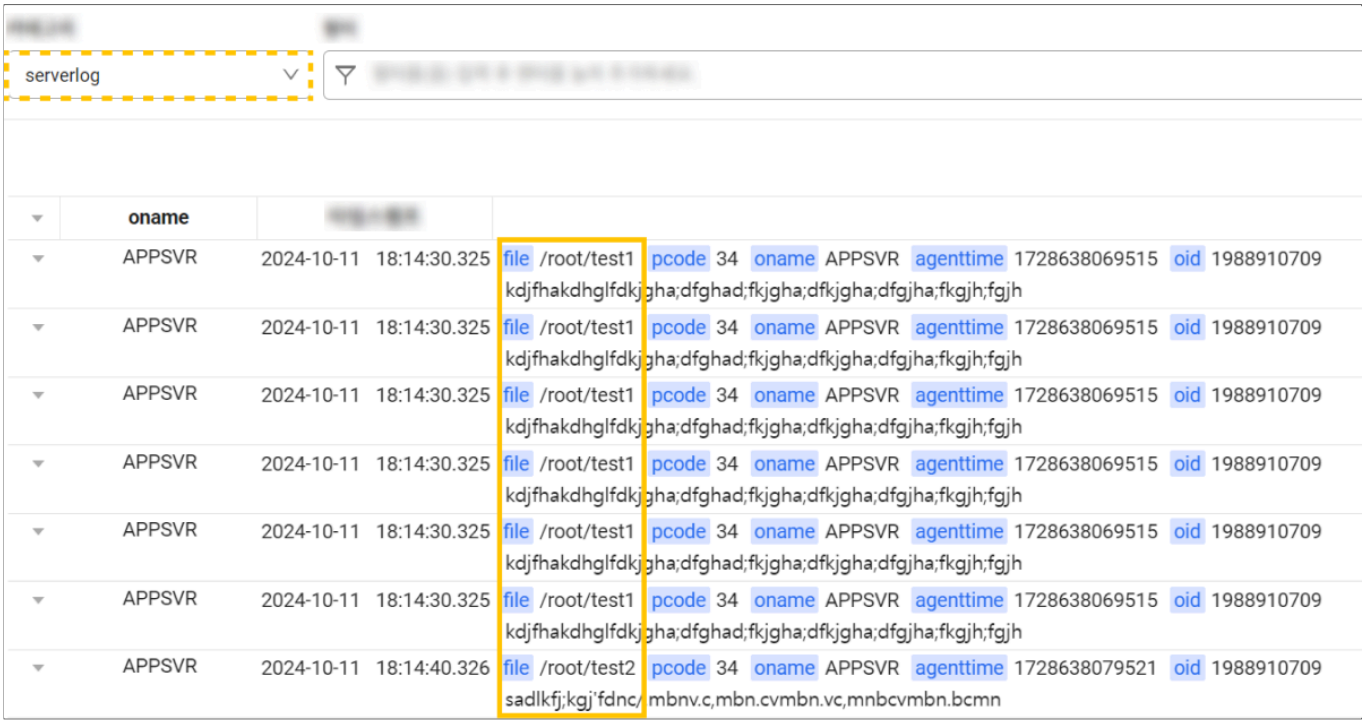
```
  stats_category = "logsink_stats"
  excludeNames = [ ".gz",".zip" ]

[[inputs.logsink.file]]
  path = "/root/test1"
  disabled = false
  encoding = "utf-8"

[[inputs.logsink.file]]
  path = "/root/test2"
  disabled = false
  encoding = "utf-8"
```

ⓘ category , stats_enabled , stats_category , nowrap_keywords 등의 옵션에 대한 내용은 옵션 설정를 참고하세요.

해당 예시의 경우 **라이브 테일** 메뉴에서 다음과 같이 확인할 수 있습니다.



파일 분리

서로 다른 카테고리로 로그 파일을 지정하려면, logsink{숫자}.conf 파일을 각각 생성한 후, 각 파일의 `category` 항목에 서로 다른값을 설정하세요.

logsink.conf와 logsink**1**.conf를 통해 서로 다른 로그 카테고리를 설정한 예시입니다.

logsink.conf

```
# cat logsink.conf
[[inputs.logsink]]
 category = "serverlog1"
 stats_enabled = true
 stats_category = "logsink_stats"
 excludeNames = [ ".gz",".zip" ]

[[inputs.logsink.file]]
 path = "/root/test1"
 disabled = false
 encoding = "utf-8"
```

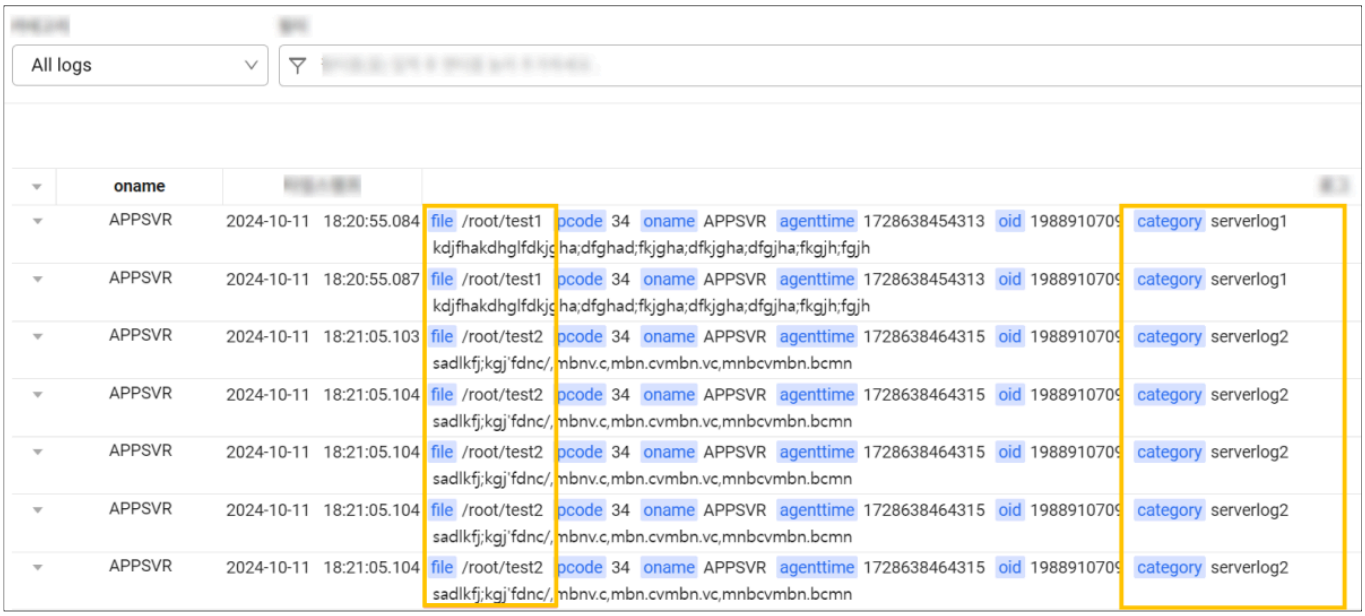
logsink1.conf

```
#cat logsink1.conf
[[inputs.logsink]]
 category = "serverlog2"
 stats_enabled = true
 stats_category = "logsink_stats"
 excludeNames = [ ".gz",".zip" ]

[[inputs.logsink.file]]
 path = "/root/test2"
 disabled = false
 encoding = "utf-8"
```

ⓘ `category` , `stats_enabled` , `stats_category` , `nowrap_keywords` 등의 옵션에 대한 내용은 옵션 설정를 참고하세요.

해당 예시의 경우 **라이브 테일** 메뉴에서 다음과 같이 확인할 수 있습니다.



로그 모니터링 활성화

홈 화면 > 프로젝트 선택 > **로그** > **로그 설정**

로그 모니터링 시작하기 섹션의 **로그 모니터링 활성화** 탭에서 토글 버튼으로 와탭 로그 모니터링을 활성화 또는 비활성화할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

Kubernetes

쿠버네티스 컨테이너와 컨테이너 내부 애플리케이션의 로그 수집을 위한 설정 방법을 안내합니다.

에이전트 업데이트

쿠버네티스 에이전트 1.1.35 버전부터 가능합니다. 업데이트 방법은 다음 문서를 참조하세요.

에이전트 설정 확인

홈 화면 > **프로젝트 선택** > **로그** > **로그 설정**

1. 와탭 모니터링 서비스 초기 화면에서 프로젝트를 선택한 후, **로그** > **로그 설정** 메뉴로 이동하세요.
2. **로그 모니터링 시작하기** 탭에서 **에이전트 설정 및 로그 모니터링 활성화**의 **1. 에이전트 설정 확인**의 안내를 참고해 에이전트 설정을 확인하세요.

컨테이너 로그 수집 활성화

쿠버네티스 컨테이너에 로그를 수집하려면 **1. 에이전트 설정 확인**에서 **로그 설정 적용하기** 버튼을 클릭하세요.

컨테이너 내부 애플리케이션 로그 수집 활성화

Java 2.1.1, Python 1.2.2 버전부터 가능합니다. 쿠버네티스 컨테이너 상에서 실행되는 애플리케이션의 로그를 수집할 수 있도록 다음을 참고하세요.

1. **로그 > 로그 설정 > 로그 모니터링 시작하기** 탭으로 이동하세요.
2. **1. 에이전트 설정 확인** 하단의 **에이전트 설정** 버튼을 클릭하세요.
3. **에이전트 설정**에서 **옵션 작성**의 옵션 선택 드롭 다운 목록에서 **직접 입력**을 클릭하세요.
4. 에이전트 설정 명령어의 키 `logsink_enabled` 와 값 `true` 를 입력창에 입력하세요.

```
logsink_enabled=true
```

5. **적용** 버튼을 선택하세요. 쿠버네티스 컨테이너 내 애플리케이션의 로그를 수집할 수 있습니다.

> - Java 애플리케이션 로그 수집에 대한 자세한 내용은 다음 문서를 참고하세요.
> - Python 애플리케이션 로그 수집에 대한 자세한 내용은 다음 문서를 참고하세요.
> - Go 애플리케이션 로그 수집에 대한 자세한 내용은 다음 문서를 참고하세요.

로그 모니터링 활성화

홈 화면 > 프로젝트 선택 > **로그** > **로그 설정**

로그 모니터링 시작하기 섹션의 **로그 모니터링 활성화** 탭에서 토글 버튼으로 와탭 로그 모니터링을 활성화 또는 비활성화할 수 있습니다.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

> **로그 모니터링 활성화 권한**
>
> 에이전트 설치 후 프로젝트에 대한 **수정 권한**이 있는 사용자만 로그 모니터링을 활성화할 수 있습니다. 권한에 대한 자세한 내용은 다음 문서를 참고하세요.

쿠버네티스 로그 모니터링 카테고리 안내

쿠버네티스 관련 다양한 로그를 확인할 수 있습니다. 와탭 쿠버네티스에서 모니터링 카테고리를 제공합니다.

> 설정에 따라 중복된 로그 내용이 저장될 수 있으니 반드시 중복 여부를 확인하시기 바랍니다.

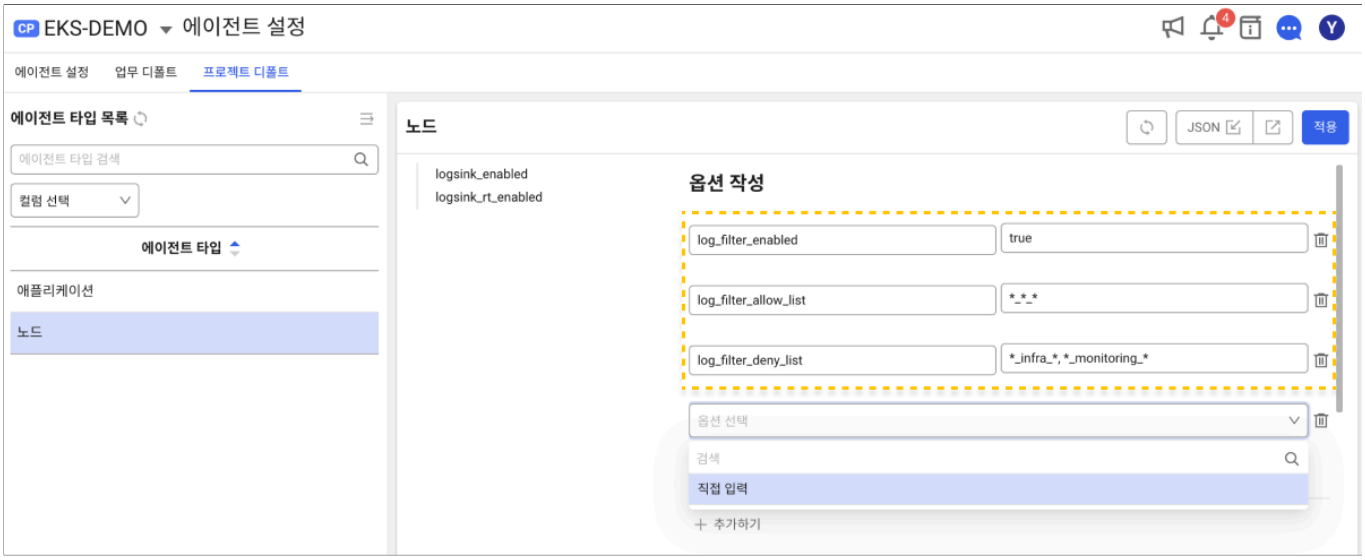
카테고리	설명
`#K8sEvent`	• 쿠버네티스에서 발생하는 이벤트가 저장된 로그 • 사용자 설정과 무관하게 기본 생성
`#WhatapEvent`	• 와탭 이벤트 설정에 의해 발생한 이벤트가 저장된 로그 • 사용자 설정과 무관하게 기본 생성
`containerStdout`	• 컨테이너 Standard Out 로그 • 사용자 설정 시 생성 • 노드 에이전트에 `logsink_enabled=true` 설정 추가 시
`AppLog`	• 컨테이너 내 애플리케이션 로그 • 사용자 설정 시 생성 • 애플리케이션 에이전트에 `logsink_enabled=true` 설정 추가 시

containerStdOut 로그 필터 설정

홈 화면 > 프로젝트 선택 > **관리** > **에이전트 설정**

쿠버네티스 로그에서 필요한 정보를 선별하기 위한 필터링 옵션을 안내합니다. 필터링을 통해 허용하거나 제외할 로그를 설정할 수 있습니다.

1. **관리** > **에이전트 설정**으로 이동하세요.
2. **프로젝트 디폴트** 탭에서 **옵션 작성** 방식으로 **직접 입력**을 클릭하세요.



로그 네이밍 규칙

쿠버네티스 containerStdOut 로그는 노드의 /var/log/containers/ 경로 하위에 생성됩니다. 다음과 같은 네이밍 규칙을 따르고 있습니다.

```
<podName>_<namespace>_<containerName>-<containerId>.log
```

로그 필터 옵션

- **log_filter_enabled** bool

  기본값 `false`

  로그 필터 사용 여부를 설정합니다.

- **log_filter_allow_list** list

  허용할 로그의 리스트를 설정합니다. 필터링 적용 시 포함할 항목을 의미합니다.

- **log_filter_deny_list** list

  제외할 로그의 리스트를 설정합니다. 필터링 적용 시 제외할 항목을 의미합니다.

로그 필터 동작

`log_filter_enabled` 옵션이 활성화된 경우만 로그 필터가 동작합니다. 해당 옵션의 값이 `true`로 설정된 상태에서 `log_filter_allow_list` 및 `log_filter_deny_list`를 통해 로그를 선택적으로 수집할 수 있습니다.

로그 필터는 블랙리스트 기반으로 작동합니다. 예를 들어 `log_filter_enabled` 활성화 후 `log_filter_allow_list` 옵션에 추가할 항목을 설정하지 않을 경우 기본적으로 모든 컨테이너의 로그 수집이 차단됩니다. 또한 로그 필터는 allow 규칙 보다 deny 규칙의 우선 순위가 높습니다. 동일한 항목이 allow와 deny에 모두 설정된 경우, deny 규칙이 우선 적용되어 해당 로그는 수집되지 않습니다.

로그 필터 예시

단일 네임스페이스 로그 수집

- 허용 **log_filter_allow_list** `*_infra_*`

`infra` 네임스페이스에 존재하는 모든 컨테이너가 수집 대상으로 특정됩니다.

다중 네임스페이스 로그 수집

- 허용 **log_filter_allow_list** `*_infra_*, *_monitoring_*`

`infra`, `monitoring` 네임스페이스에 존재하는 모든 컨테이너가 수집 대상으로 특정됩니다.

특정 단어를 포함하는 Pod 제외

- 허용 **log_filter_allow_list** `*_*_*`
- 제외 **log_filter_deny_list** `*prod*_*_*`

모든 컨테이너가 수집 대상으로 특정되지만, `podName`에 `prod`가 포함된 로그는 제외됩니다.

특정 단어를 포함하는 Pod 로그 수집

- 허용 **log_filter_allow_list** `*prod*_*_*`

`podName`에 `prod` 가 포함된 파드의 로그만 수집됩니다.

다중 네임스페이스 로그 수집 및 특정 단어 포함하는 Pod 제외

- 허용 **log_filter_allow_list** `*_infra_*, *_monitoring_*`
- 제외 **log_filter_deny_list** `*prod*_*_*`

`infra`, `monitoring` 네임스페이스의 로그가 수집되고, `podName`에 `prod`가 포함된 로그는 제외됩니다.

컨테이너 이름 내 특정 단어로 시작하는 컨테이너 로그 제외

- 허용 **log_filter_allow_list** `*_*_db-*`

모든 네임스페이스에서 `containerName`이 `db`로 시작하는 컨테이너의 로그는 제외됩니다.

라이브 테일

> ⓘ **로그 조회** 권한이 없을 경우 해당 메뉴에 진입할 수 없습니다.

홈 화면 > 프로젝트 선택 > **로그** > **라이브 테일**

라이브 테일 메뉴에서 서버 콘솔에 접속하지 않고, 로그 데이터 스트림을 모니터링 화면상에서 바로 확인할 수 있습니다. 복잡한 대량의 로그도 **필터**와 **하이라이트** 기능을 활용해 선별하고 빠르게 인지할 수 있습니다. 로그 데이터 조회 주기는 2초입니다.

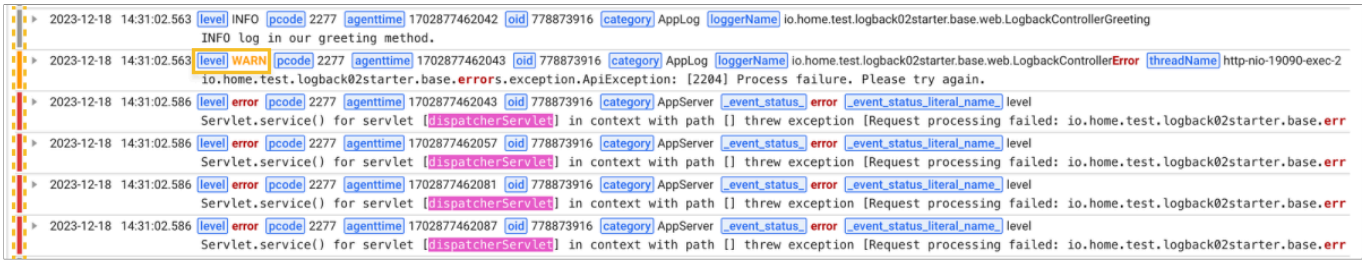
주요 용어는 다음과 같습니다.

- **Category**: 로그의 수집 및 조회 단위
- **Content**: 로그 메시지
- **Search Key**: 로그 파서 설정을 통해 생성되는 검색 키
- **Tag**: 수집된 로그를 검색할 수 있는 태그

> ⓘ 로그 테이블 컬럼 가장자리를 드래그해 컬럼 너비를 수정할 수 있습니다.

에이전트 옵션

에이전트 옵션이 설정된 경우 로그 레벨을 수집해 로그 레벨 기준 색상이 다음과 같이 표시됩니다.



에이전트 옵션 설정

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- Java 에이전트 2.2.22 버전 이후부터 위빙 설정에 log4j-2.17 또는 logback-1.2.8 설정 시 사용할 수 있습니다. 에이전트 재시작이 필요합니다.
- 로그 레벨은 파싱된 키워드 중 `level`, `type` 기준으로 판별합니다. `level`, `type`으로 파싱된 키가 존재하고 파싱 값이 FATAL, CRITICAL, ERROR, WARN, WARNING, INFO를 포함할 경우 로그 레벨 색상을 표시합니다.

필터 적용하기

필터를 적용하면 입력한 조건에 맞는 로그를 필터링합니다. 복수의 필터를 입력할 수 있습니다. 필터의 태그가 같은 경우 OR(`||`)로, 그렇지 않은 경우는 AND(`&&`)로 적용됩니다.

입력 창에 값을 직접 입력하거나 **필터** 입력 창을 클릭해 필터를 지정할 수 있습니다. 필터 태그는 `검색 키`, `연산자`, `검색 값`의 순서로 입력합니다. **검색** 버튼을 선택하면 필터가 적용된 데이터를 로그 목록에서 조회할 수 있습니다.

> **가이드 UI**
>
> 다음과 같이 입력 창 아래 **가이드 UI**를 제공합니다. 입력 창 또는 필터 태그에 마우스 커서가 있을 때 ESC 키를 통해 **가이드 UI**를 닫을 수 있습니다.



검색 키, 연산자, 검색 값 입력

- `검색 키` 입력 시 일반 인덱스, 예약어 인덱스, 숫자만 입력할 수 있는 인덱스를 구분해 추천 값을 제공합니다
- `연산자` 입력 시 일반 인덱스 검색 키의 경우 `==`, `!=` 옵션을 하단에 안내합니다. 숫자만 입력할 수 있는 인덱스의 경우 `>`, `<`, `<=`, `>=`, `==`, `!=` 옵션을 제공합니다.
- `검색 값` 입력 시 일치 검색(`>`, `<`, `<=`, `>=`, `==`)일 때 파란색으로, 제외 검색(`!=`)일 때 붉은색으로 하이라이팅합니다. 대소문자 구분 옵션을 활용해 검색할 수 있습니다.

- 필터 태그가 2줄 이상 길어지는 경우 ∧ **접기** 아이콘을 선택해 접어둘 수 있습니다.
- 필터 태그 입력 후 입력 창에서 Shift와 Tab 키를 동시에 사용하여 이전 필터 태그로 이동할 수 있습니다.
- 필터 태그 입력 후 입력 창에서 Tab 키를 통해 다음 필터 태그로 이동할 수 있습니다.

필터 태그 추가

입력 창에 텍스트 입력 후 Enter 또는 Tab 키를 누르거나, 입력창 하단의 가이드 UI에서 추천 값을 클릭하거나 방향키로 선택해 추가할 수 있습니다.

필터 태그 제거

Backspace 키 또는 태그의 **X** 아이콘을 눌러 태그를 삭제할 수 있습니다. 입력창 오른쪽의 **X** 아이콘을 클릭해 모든 태그를 한 번에 삭제할 수 있습니다.

필터 즐겨찾기

필터 **즐겨찾기** 기능을 제공합니다. 입력 창 오른쪽의 ☆ **즐겨찾기** 아이콘을 선택하여 원하는 필터 검색 조건을 즐겨찾기에 **추가**, **제거** 및 기존 즐겨찾기를 선택할 수 있습니다. **즐겨찾기**는 최대 50개까지 저장됩니다.



필터 적용 예외 상황

- 숫자만 입력할 수 있는 인덱스(`.n` 으로 끝나는 `검색 키`)를 입력한 태그에서 `검색 값` 은 숫자만 입력할 수 있습니다.
- 중복된 `검색 키` , `검색 값` 은 입력할 수 없습니다.
- `검색 키` , `검색 값` 중 하나라도 없는 태그가 존재할 때 검색할 수 없습니다. 유효하지 않는 태그의 경우 회색으로 표시합니다.

> ⓘ **라이브 테일** `검색 키` 로 `category` 를 입력할 수 없습니다.
> 입력된 필터 값 아래에 있는 수식(expression)은 로그 데이터 조회 시 적용될 필터 수식 미리 보기입니다.

미파싱 키워드 필터 적용

로그에서 파싱되지 않은 즉 인덱스가 생성되지 않은 키워드를 포함한 로그를 조회할 수 있습니다. 이 경우 지정 범위 내 모든 로그를 Full Scan합니다. 그렇기 때문에 인덱스가 생성된 키와 비교해 검색 속도가 다소 떨어질 수 있습니다. 정형화된 로그 데이터의 경우 **로그 파서 설정**을 통해 인덱스 키 값을 활용해 검색하는 것을 권장합니다.



1. **카테고리**를 선택하세요.
2. **필터** 입력창에 `content` 기준 띄어쓰기 후 검색을 원하는 키워드를 입력하세요.
   예시, `content *select*`
3. **검색** 버튼을 클릭해 로그를 조회하세요.

> **라이브 테일**의 경우 모든 로그 검색이 가능해 카테고리를 지정할 필요가 없습니다.
> 파서 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

필터 수정

필터에 값을 입력한 뒤 입력한 값을 클릭하면 해당 값을 수정할 수 있습니다.



- 입력 창에 텍스트 재입력해 수정할 수 있습니다.
- 입력 창 아래 **가이드 UI**를 통해 추천 값을 선택해 수정할 수 있습니다.

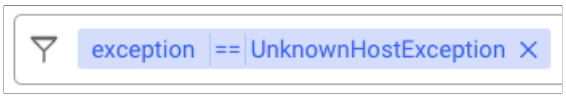
검색 키(Search Key)

다음 이미지에서 파란색 박스 부분은 파싱(parsing)된 검색 키입니다. 검색 키는 **로그 설정**의 **로그 파서 설정** 탭에서 파싱 로직을 등록해 설정할 수 있습니다.



필터 입력 문법

태그는 검색 키와 검색 값으로 구성되어있습니다. 다음의 예시에서 검색 키는 `exception` , 검색 값은 `UnknownHostException` 입니다. 해당 예시는 수집한 로그 데이터 중 IP 주소와 도메인 주소가 매칭되지 않아 서버를 호스트에 연결할 수 없을 경우 발생하는 예외(`UnknownHostException`)가 포함된 로그 데이터를 조회합니다.



검색 키 종류

검색 키 종류	검색 키 포멧	의미	검색 키와 검색 값 예시	검색 예시
문자열 키워드	keyword	파일 이름	- 키: fileName - 값: /data/whatap/logs/yard.log	fileName:/data/whatap/logs/yard.log
숫자 키워드	keyword.n	응답시간	- 키: response_time.n - 값: 2945	response_time.n>=2945
예약어 키워드 (사전 정의 키워드)	@keyword	트랜잭션 ID	- 키: @txid - 값: 85459614215434144	-
로그 본문 키워드	content	로그 본문	- 키: content - 값: 사용자 입력값	content: `*ERROR*`

Content 검색 키

- Content 검색 키는 인덱싱되지 않은 로그의 본문을 대상으로 검색합니다. 예를 들어 `content *ERROR*` 와 같이 입력하는 경우 로그 본문 중 `ERROR` 를 포함한 로그를 검색합니다.
- 어떤 키워드로 인덱싱을 걸어야하는지 모르는 경우 Content 검색 키를 활용해 문제가 되는 키워드를 포함한 로그를 식별합니다. 이후 **로그 설정** 메뉴의 로그 파서 설정을 통해 해당 키워드로 파서를 설정해 인덱스를 생성하는 방식으로 검색 속도를 향상시킬 수 있습니다.

공통 문법

문법 종류	설명	예시
`==searchValue`	검색 값과 일치하는 로그를 검색합니다.	`exception==RuntimeExceptionexception`
`!=searchValue`	검색 값을 제외한 로그를 검색합니다.	`exception!=RuntimeException`

문법 종류	설명	예시
*searchValue	검색 값으로 끝나는 로그를 검색합니다.	word==*hello
searchValue*	검색 값으로 시작하는 로그를 검색합니다.	word==hello*
searchValue	검색 값으로 중간에 포함된 로그를 검색합니다.	word==*hello*
*search*Value*	검색 값으로 포함된 로그를 검색합니다.	word==*he*llo*
re:{regex}	정규표현식에 매칭되는 로그를 검색합니다.	caller==re:^i\.w\.a\.w\.s\.v\.r\.
**	검색 키에 해당하는 모든 로그를 검색합니다.	

검색 키가 숫자 키워드(keyword.n)인 경우 문법

다음의 문법은 검색 키가 keyword.n 형식인 경우에만 지원합니다.

- 검색 값으로는 숫자만 올 수 있습니다.
- .n 키워드의 값에는 prefix를 붙이지 않습니다. .n 이 아닌 키워드는 모두 prefix를 붙여야합니다.

  예, +>searchValue 는 유효하지 않습니다.

문법 종류	설명	예시
>searchValue	검색 값보다 큰 값이 포함된 로그를 조회합니다.	response_time.n>3000
>=searchValue	검색 값보다 크거나 같은 값이 포함된 로그를 조회합니다.	response_time.n>=3000
==searchValue	검색 값보다 같은 값이 포함된 로그를 조회합니다.	response_time.n==3000
!=searchValue	검색 값보다 다른 값이 포함된 로그를 조회합니다.	response_time.n!=3000
<searchValue	검색 값보다 작은 값이 포함된 로그를 조회합니다.	response_time.n<3000
<=searchValue	검색 값보다 작거나 같은 값이 포함된 로그를 조회합니다.	response_time.n<=3000

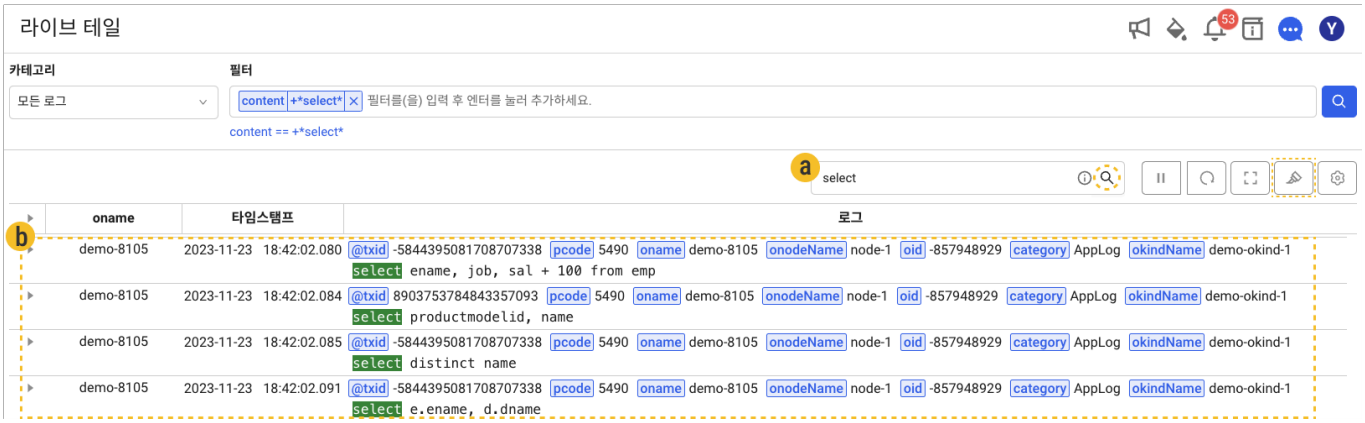
개인정보 조회

로그 개인정보 조회 권한이 있을 경우, **개인정보 표시** 토글 버튼 활성화를 통해 비식별화 처리한 개인정보를 표시할 수 있습니다.

> ⓘ
> - **로그 개인정보 조회** 권한에 대한 자세한 내용은 다음 문서를 참고하세요.
> - **로그 개인정보 비식별화** 기능에 대한 자세한 내용은 다음 문서를 참고하세요.

콘텐츠 하이라이트

로그의 콘텐츠 중 원하는 키워드를 손쉽게 식별하기 위해 하이라이트 기능을 제공합니다. 단일 또는 복수 키워드로 필터링할 수 있습니다.



1. 키워드 입력창에 하이라이트를 원하는 키워드를 입력하세요.
   - 예시, `select`
2. 로그 목록의 Content 내 검색한 키워드가 하이라이팅 됩니다.
   - **로그 전체 화면** 아이콘을 선택하면 **로그**와 **타임스탬프**를 전체 화면에서 확인할 수 있습니다.
   - 하이라이트된 키워드 간 이동은 단축키로 조작할 수 있습니다.
     - `Enter` : 다음 row로 이동
     - `Shift+Enter` : 이전 row로 이동

- ∧ / ∨ 아이콘: row 이동
- `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): 키워드 검색

복수 키워드 조건

복수 키워드로 하이라이팅을 할 경우 다음과 같이 작성합니다.

입력 문자열	설명	결과
`a b c`	띄어쓰기로 각 키워드를 구분합니다.	a, b, c
`"Whatap is good."`	띄어쓰기를 키워드에 포함하고 싶은 경우 `''` 또는 `""`로 감쌉니다.	Whatap is good.
`"Whatap\\ is good."`	""로 감싸진 키워드에서 `\`를 포함할 경우, `\\`로 입력해야합니다.	Whatap\ is good.

하이라이트 색상 설정

하이라이팅할 키워드 및 색상을 설정할 수 있습니다.

- 기본적으로 로그 레벨에 따른 하이라이팅(FATAL, ERROR, WARN, SEVERE, CRITICAL)이 적용되어 있습니다.
- 설정한 내용은 **프로젝트 단위**로 저장됩니다.

1. 아이콘을 클릭하세요.
2. **하이라이트** 창에서 추가적으로 색상 설정을 원하는 키워드를 입력창에 입력하세요.
3. 입력창 왼쪽 색상 클릭 시 색을 선택할 수 있습니다.

테이블 설정

컬럼 설정

컬럼을 추가하거나 순서를 설정할 수 있습니다. 화면 오른쪽에서 **컬럼 설정** 버튼을 클릭하세요.

컬럼 추가

태그를 선택하여 테이블에 컬럼을 추가할 수 있습니다. **log** 컬럼 선택을 해제할 경우 **로그 표시 상세 설정**을 확인할 수 없습니다. 반드시 한 개 이상의 컬럼을 선택하세요.

- **제공되는 컬럼**: 로그, 수집 시간, content(로그 메시지)

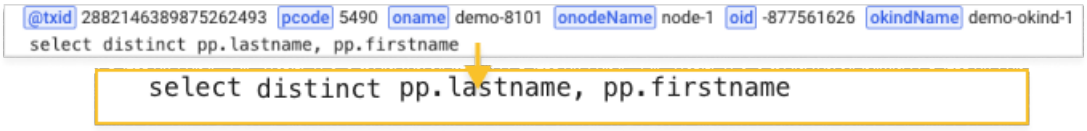
컬럼 순서 변경

컬럼을 추가하면 **표시 항목**에 해당 컬럼이 추가됩니다. 원하는 컬럼을 드래그하여 컬럼의 순서를 변경하세요.

로그 표시 상세 설정

1. 화면의 오른쪽에서 ⚙ **로그 표시 상세 설정** 버튼을 클릭하세요.
2. **로그 표시 상세 설정** 창에서 **content**와 **태그** 중 하나는 반드시 선택하세요.
    - 기본으로 **content**, **태그** 모두 선택되어있으며 두 가지 항목 모두 표시합니다.

체크가 해제된 항목은 테이블에 표시되지 않습니다. 다음과 같이 **태그**를 해제할 경우 테이블에서 로그의 **태그**는 표시되지 않습니다.



태그 관리 목록에 태그를 추가하면 추가한 순서대로 로그의 태그가 나열됩니다. 태그의 순서는 드래그하여 변경할 수 있습니다. 추가한 태그를 비활성화하면 비활성화한 태그는 로그의 태그에 노출되지 않습니다.

> ⓘ **컬럼 및 로그 표시 설정 적용 범위**
>
> - **컬럼 설정**과 **로그 표시 상세 설정**은 **라이브 테일**, **로그 검색**, **로그 트렌드**에서 사용할 수 있습니다.
> - 동일한 프로젝트 내 **라이브 테일**, **로그 검색**, **로그 트렌드**는 **컬럼 설정**과 **로그 표시 상세 설정**을 공유합니다.

로그 파일 다운로드

검색한 로그의 결과값을 CSV, TXT 파일 형식으로 수집된 로그를 다운로드 받을 수 있습니다.

1. 화면 중앙의 오른쪽의 **다운로드** 버튼을 클릭합니다.
2. **CSV 다운로드** 또는 **Log 다운로드**를 클릭해 원하는 형태의 파일을 다운로드 받습니다.

로그 탐색

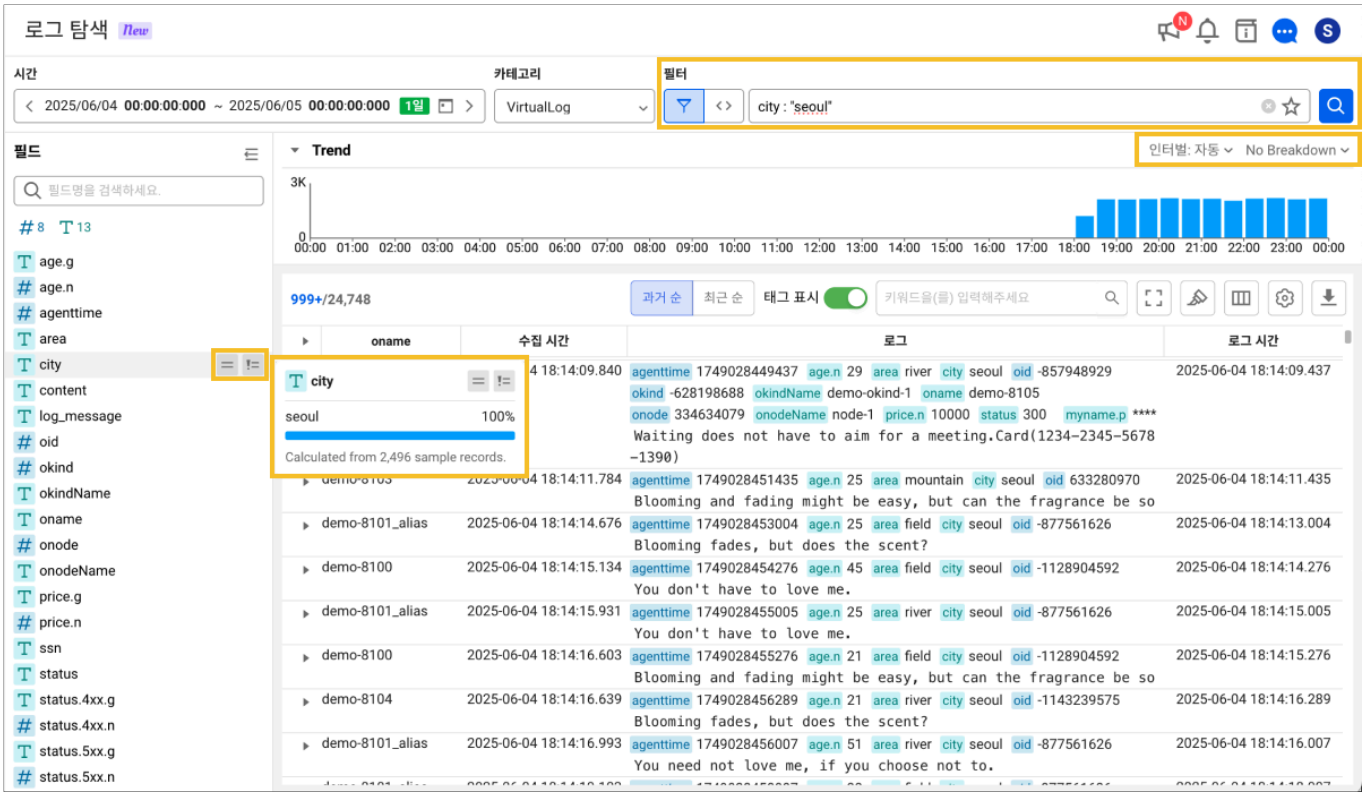
> **로그 조회** 권한이 없을 경우 해당 메뉴에 진입할 수 없습니다.

홈 화면 > 프로젝트 선택 > **로그** > **로그 탐색** *New*

로그 탐색(Log Explorer) 기능은 로그 데이터를 빠르게 검색, 필터링, 분석합니다. 필드 구조에 대한 정보를 파악하고, 쿼리 기반 필터링을 통해 필요한 로그를 효율적으로 검색해, 결과를 데이터 시각화하여 확인할 수 있습니다. 본 기능은 Lucene 인덱스 기반으로 동작하며, 주요 특징은 다음과 같습니다.

1. **로그 트렌드 분석 및 검색**: 로그 트렌드와 검색 기능을 동시에 지원하여, 시간에 따른 로그 변화 추이를 파악할 수 있습니다.
2. **빠른 검색 및 필터링**: 사용자는 간단한 단어 입력만으로도 문제의 로그를 신속하게 검색할 수 있습니다. WhaTap log search query와 Lucene 쿼리 언어를 지원하여, 다양한 검색 조건을 설정할 수 있습니다.
3. **자동완성 기능**: 쿼리 입력 시 자동완성 기능을 통해 사용자가 쉽게 쿼리를 작성하고 학습할 수 있도록 돕습니다.
4. **데이터 시각화**: 스택 그래프를 활용하여 시계열 로그 건수 데이터를 시각적으로 표현하며, 드릴다운 기능을 통해 특정 로그를 심층 분석할 수 있습니다.
5. **즐겨찾기 및 공유**: 자주 사용하는 필터를 즐겨찾기로 저장하고, URL을 통해 다른 사용자와 쉽게 공유할 수 있습니다.

기본 화면 안내



시간 설정

상단 옵션바의 **타임셀렉터**를 통해 로그 검색 시간 범위를 설정할 수 있습니다.

카테고리

상단 옵션바의 **카테고리 선택**에서 탐색하고 싶은 로그 종류 선택합니다. (예: AppLog 등)

필터

상단 옵션 바의 **필터**에서 쿼리 언어를 선택하고 검색 조건을 입력하여 필터를 적용합니다. 검색 쿼리 문법은 WhaTap 로그 검색 쿼리와 Lucene 쿼리를 지원합니다.

- : WhaTap 로그 검색 쿼리
- : Lucene 쿼리

검색 쿼리 문법

WhaTap log search query

WhaTap 로그 검색 쿼리는 자동 완성 기능을 제공하며, 간단한 단어 입력만으로도 문제의 로그를 신속하게 검색할 수 있습니다. and, or 등 다양한 문법을 활용해 다양한 검색 조건을 입력할 수 있습니다.

- 예, 검색 필터에 `error` 만 입력하여 content(로그 본문) 검색할 수 있습니다.

    1. 상단 옵션 바에서 필터 아이콘을 클릭한 후, 쿼리 검색창에서 필드(선택한 시간과 카테고리에 맞는 목록 노출)를 선택하세요.
    2. 연산자를 선택하세요.

        a. 기본: `:` , `:*` , `and` , `or`
        b. 선택한 필드가 `Number` 인 경우(기본 + `<=` , `>=` , `<` , `>`)
        c. 선택한 필드가 `String` 인 경우(기본)

    3. 값(선택한 시간, 카테고리, 필드에 맞는 목록 노출)을 선택하세요.

검색 인덱스 필드 타입

지원 필드 타입은 `Number` , `String` , `Boolean` 세 가지 유형만 존재합니다.

- `Number` : Long, double, float 타입의 숫자 값
- `String` : 이메일 본문과 같은 전체 텍스트
- `Boolean` : True 또는 False 값

> WhaTap log search query 문법에 대한 자세한 내용은 다음 문서를 참고하세요.

Lucene

apache lucene 8.11.1 버전 오픈소스 문법을 그대로 사용하세요. 단, 자동 완성 기능을 제공하지 않습니다.

> Lucene의 QueryParser 클래식 패키지에 대한 자세한 내용은 공식 문서를 참고하세요.

필드 목록

드릴 다운 분석에 활용할 수 있습니다.

필드 정보 필터 반영하기

필드 목록에서 필드명을 확인하고, `=`, `!=` 버튼을 클릭해 조건을 필터에 적용하세요.

- 필드에 마우스 호버 시 `=`, `!=` 버튼이 보이고 클릭 시 필터에 반영됩니다.
- 필드 검색 기능을 통해 원하는 필드를 빠르게 찾을 수 있습니다.

Trend 차트

Trend 차트에서 **인터벌**, **Breakdown**을 적용해 그래프로 로그를 확인할 수 있습니다.

- **그래프 인터랙션**: 마우스 드래그 및 버튼으로 시간 줌인, 바 클릭으로 해당 영역의 로그 확인 가능
- **인터벌**: 자동, 1초, 5초, 30초, 1분, 5분, 10분, 30분, 1시간 중 선택 가능
- **Breakdown**: 특정 필드를 기준으로 Top 3 + Others 값을 분포 시각화하고, 드릴다운 필터 적용 가능

breakdown

breakdown은 로그 건수 추이를 특정 필드로 분석해주는 시각화 기능입니다.

- 특정 필드의 Top 3 값과 Others(나머지 값)를 구분하여 스택 그래프로 분포를 표시합니다.
- 드릴다운 분석에 활용할 수 있으며, 범례와 스택 그래프에서 특정 영역(부분 그래프)를 클릭해 특정 값만 강조하거나 필터 조건을 추가할 수 있습니다.

로그 확인하기

검색 조건에 맞는 로그 목록을 확인할 수 있으며, **필드 목록**에서 필드 정보와 로그 태그 색이 동일하게 표시됩니다.

콘텐츠 하이라이트

로그의 콘텐츠 중 원하는 키워드를 손쉽게 식별하기 위해 하이라이트 기능을 제공합니다. 단일 또는 복수 키워드로 필터링할 수 있습니다.

1. 키워드 입력창에 하이라이트를 원하는 키워드를 입력하세요.
    - 예시, `select`
2. 로그 목록의 Content 내 검색한 키워드가 하이라이팅 됩니다.
    - ⛶ **로그 전체 화면** 아이콘을 선택하면 **로그**와 **타임스탬프**를 전체 화면에서 확인할 수 있습니다.
    - 하이라이트된 키워드 간 이동은 단축키로 조작할 수 있습니다.
        - `Enter` : 다음 row로 이동
        - `Shift+Enter` : 이전 row로 이동
        - ∧ / ∨ 아이콘: row 이동
        - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): 키워드 검색

복수 키워드 조건

복수 키워드로 하이라이팅을 할 경우 다음과 같이 작성합니다.

입력 문자열	설명	결과
`a b c`	띄어쓰기로 각 키워드를 구분합니다.	a, b, c
`"Whatap is good."`	띄어쓰기를 키워드에 포함하고 싶은 경우 `''` 또는 `""`로 감쌉니다.	Whatap is good.
`"Whatap\\ is good."`	""로 감싸진 키워드에서 `\`를 포함할 경우, `\\`로 입력해야합니다.	Whatap\ is good.

하이라이트 색상 설정

하이라이팅할 키워드 및 색상을 설정할 수 있습니다.

- 기본적으로 로그 레벨에 따른 하이라이팅(FATAL, ERROR, WARN, SEVERE, CRITICAL)이 적용되어 있습니다.

- 설정한 내용은 **프로젝트 단위**로 저장됩니다.

1. 아이콘을 클릭하세요.
2. **하이라이트** 창에서 추가적으로 색상 설정을 원하는 키워드를 입력창에 입력하세요.
3. 입력창 왼쪽 색상 클릭 시 색을 선택할 수 있습니다.

컬럼 설정

버튼을 클릭해, 컬럼을 추가하거나 순서를 설정할 수 있습니다.

- **컬럼 추가**

  컬럼 설정에서 태그를 선택하여 테이블에 컬럼을 추가할 수 있습니다.

  > 로그 컬럼 선택을 해제할 경우 **로그 표시 상세 설정**을 확인할 수 없습니다. 반드시 한 개 이상의 컬럼을 선택하세요.

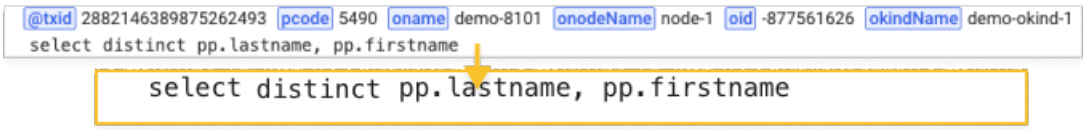
- **컬럼 순서 설정**

  컬럼 설정에서 컬럼을 추가하면 **표시 항목**에 해당 컬럼이 추가됩니다. 원하는 컬럼을 드래그하여 컬럼의 순서를 변경하세요.

로그 표시 상세 설정

버튼을 클릭해, 로그 표시를 상세 설정합니다. 기본으로 **content**, **태그** 모두 체크가 되어있으며 두 가지 항목 모두 표시합니다.

- **content**와 **태그** 중 하나는 반드시 선택하세요.
- **태그 관리** 목록에 태그를 추가하면 추가한 순서대로 로그의 태그가 나열됩니다. 태그의 순서는 드래그하여 변경할 수 있습니다. 추가한 태그를 비활성화하면 비활성화한 태그는 로그의 태그에 노출되지 않습니다.
- 체크가 해제된 항목은 테이블에 표시되지 않습니다. 다음과 같이 **태그**를 해제할 경우 테이블에서 로그의 **태그**는 표시되지 않습니다.



> **컬럼 및 로그 표시 설정 적용 범위**

- **컬럼 설정**과 **로그 표시 상세 설정**은 **라이브 테일**, **로그 검색**, **로그 트렌드**에서 사용할 수 있습니다.
- 동일한 프로젝트 내 **라이브 테일**, **로그 검색**, **로그 트렌드**는 **컬럼 설정**과 **로그 표시 상세 설정**을 공유합니다.

로그 파일 다운로드

검색한 로그의 결과값을 CSV, TXT 파일 형식으로 수집된 로그를 다운로드 받을 수 있습니다.

1. 화면 중앙의 오른쪽의 **다운로드** 버튼을 클릭합니다.
2. **CSV 다운로드** 또는 **Log 다운로드**를 클릭해 원하는 형태의 파일을 다운로드 받습니다.

WhaTap log search query 문법

> WhaTap 로그 탐색에서 사용할 수 있는 WhaTap log search query 문법에 관한 문서입니다.

1. **Equal**
   - `field : value`
2. **Not**
   - `not field : value`
3. **Wildcard** (`String` 타입 필드만 사용 가능, 검색 값 앞뒤에 `"` 사용 금지)
   - `field : val*`
4. **Exist**
   - `field : *`
5. **Range** (`Number` 타입 필드만 사용 가능)
   - `<` , `>` , `<=` , `>=` 사용 가능
   - 예: `field < 30 and field >= 10`
6. **Combining Multiple Queries**
   - AND 예: `field < 30 and field >= 10`
   - OR 예: `field < 30 or field >= 10`
7. **Grouping**
   - 예: `(field < 30 or field >= 10) and other : 70`
8. **Escaping Special Characters**
   - Lucene의 이스케이프 지원 특수 문자 목록: `&&` `||` `!` `(` `)` `{` `}` `^` `"` `~` `*` `?` `:` `\` `/`
   - 문자 앞에 `\` 를 사용해 이스케이프함
   - 예: `(1+1):2` 검색하려면, `\(1\+1\)\:2`

검색 인덱스 필드 타입

지원 필드 타입은 `Number` , `String` 두 가지 유형만 존재합니다.

- `Number` : Long, double, float 타입의 숫자 값
  - `field : "value"` 로 사용
- `String` : 이메일 본문과 같은 전체 텍스트
  - `field : value` 로 사용
- `Boolean` : True 또는 False
  - `field : true` 또는 `field : false` 로 사용

로그 트렌드

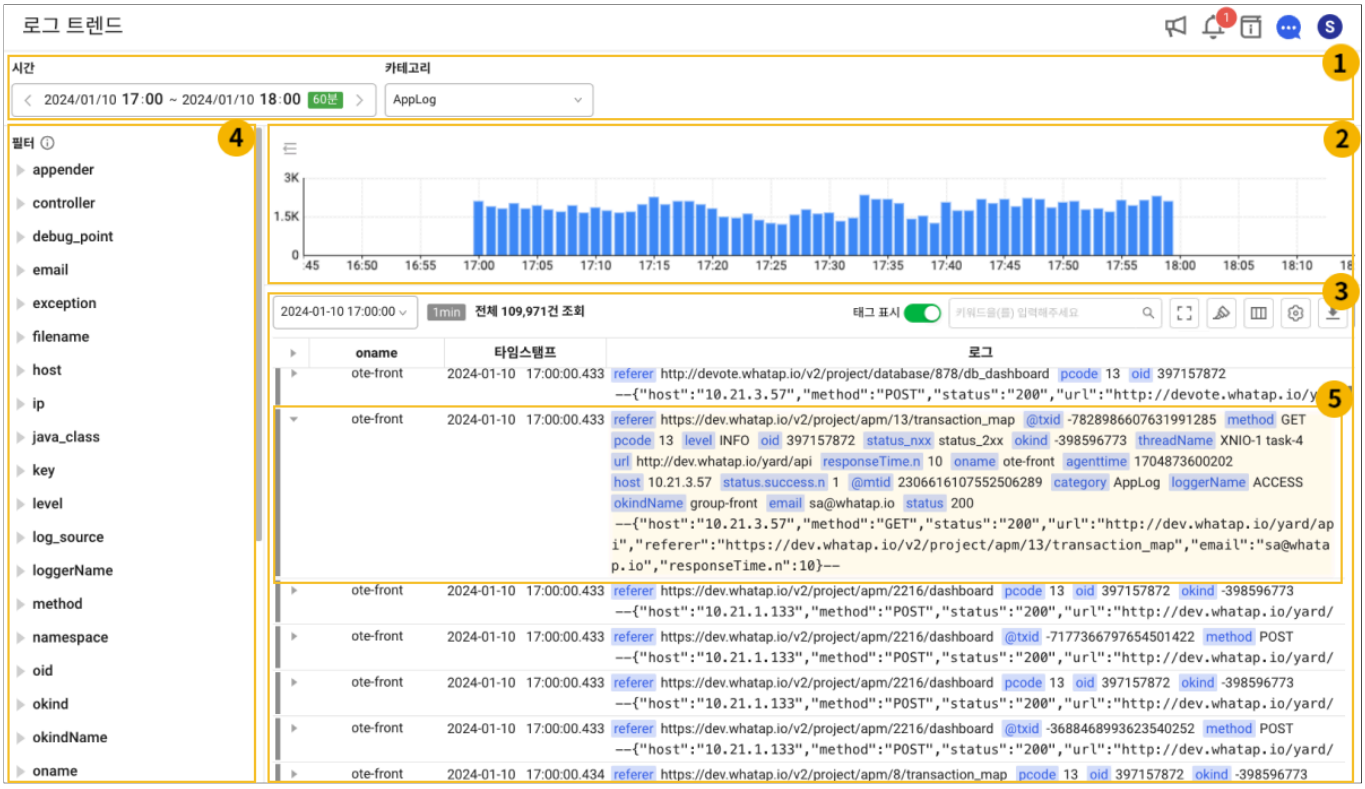
> ⓘ **로그 조회** 권한이 없을 경우 해당 메뉴에 진입할 수 없습니다.

홈 화면 > 프로젝트 선택 > **로그** > **로그 트렌드**

로그 트렌드에서 유형별로 분류된 로그의 발생 건수 추이를 통해 특정 에러 유형의 발생 패턴을 확인하고 시간별 상세 로그 데이터를 확인할 수 있습니다. 하이라이트 기능을 통해 원하는 로그를 빠르게 식별할 수 있습니다. 카테고리별로 수집된 로그의 추이를 조회할 수 있습니다.

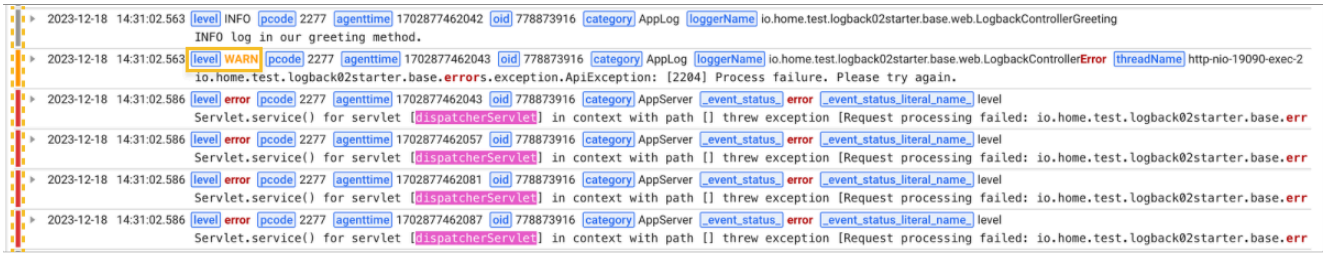
주요 용어는 다음과 같습니다.

- **Category**: 로그의 수집 및 조회 단위
- **Content**: 로그 메시지
- **Search Key**: 로그 파서 설정을 통해 생성되는 검색 키
- **Tag**: 수집된 로그를 검색할 수 있는 태그



데이터 조회하기

- 스크롤이 바닥에 닿으면 다음 데이터를 조회합니다.
- 1에서 탐색할 로그 데이터의 **시간**과 수집 단위인 **카테고리**를 지정합니다.
- **카테고리**를 변경하면 선택된 카테고리에 해당하는 로그를 조회합니다. 2 바 차트와 3 로그 테이블에서 확인할 수 있습니다.
- 2 바 차트의 막대를 클릭하면 막대의 시간 범위에 해당하는 로그를 조회합니다.
- 3 로그 테이블 상단 왼쪽에서 조회한 총 로그 개수를 확인할 수 있습니다.
- 3 로그 테이블 상단 오른쪽 **전체 화면** 아이콘을 선택하면 **로그**와 **수집 시간**을 전체 화면에서 확인할 수 있습니다.
- 4 사이드 메뉴에서 태그로 필터를 걸어서 로그를 확인할 수 있습니다. 검색 키는 2개까지 선택할 수 있고, 검색값은 복수 개 선택이 가능합니다.
- **에이전트 옵션이 설정된 경우** 로그 레벨을 수집해 로그 레벨 기준 색상이 다음과 같이 표시됩니다.



에이전트 옵션 설정

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- Java 에이전트 2.2.22 버전 이후부터 위빙 설정에 log4j-2.17 또는 logback-1.2.8 설정 시 사용할 수 있습니다. 에이전트 재시작이 필요합니다.
- 로그 레벨은 파싱된 키워드 중 `level`, `type` 기준으로 판별합니다. `level`, `type` 으로 파싱된 키가 존재하고 파싱 값이 FATAL, CRITICAL, ERROR, WARN, WARNING, INFO를 포함할 경우 로그 레벨 색상을 표시합니다.

> ⓘ 로그 테이블 컬럼 가장자리를 드래그해 컬럼 너비를 수정할 수 있습니다.

개인정보 조회

로그 개인정보 조회 권한이 있을 경우, **개인정보 표시** 토글 버튼 활성화를 통해 비식별화 처리한 개인정보를 표시할 수 있습니다.

> ⓘ
> - **로그 개인정보 조회** 권한에 대한 자세한 내용은 다음 문서를 참고하세요.
> - **로그 개인정보 비식별화** 기능에 대한 자세한 내용은 다음 문서를 참고하세요.

로그 Content 확인하기

Content는 로그 메시지입니다. 로그 컬럼의 첫 번째 줄은 로그의 파싱(parsing)된 키와 값이고 두 번째 줄은 Content입니다.

- 3 로그 테이블의 행(로그)마다 ▸ **더보기** 버튼이 있습니다. ▸ **더보기** 버튼을 선택하면 5처럼 해당 로그의 전체 content를 확인할 수 있습니다.

하이라이트 설정하기

원하는 키워드를 손쉽게 식별할 수 있도록 하이라이트 기능을 제공합니다. 아이콘을 클릭해 키워드를 입력 후 엔터를 눌러 추가하세요.

콘텐츠 하이라이트

로그의 콘텐츠 중 원하는 키워드를 손쉽게 식별하기 위해 하이라이트 기능을 제공합니다. 단일 또는 복수 키워드로 필터링할 수 있습니다.

1. 키워드 입력창에 하이라이트를 원하는 키워드를 입력하세요.
   - 예시, `select`
2. 로그 목록의 Content 내 검색한 키워드가 하이라이팅 됩니다.
   - **로그 전체 화면** 아이콘을 선택하면 **로그**와 **타임스탬프**를 전체 화면에서 확인할 수 있습니다.
   - 하이라이트된 키워드 간 이동은 단축키로 조작할 수 있습니다.
     - `Enter` : 다음 row로 이동
     - `Shift+Enter` : 이전 row로 이동
     - ∧ / ∨ 아이콘: row 이동
     - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): 키워드 검색

복수 키워드 조건

복수 키워드로 하이라이팅을 할 경우 다음과 같이 작성합니다.

입력 문자열	설명	결과
`a b c`	띄어쓰기로 각 키워드를 구분합니다.	a, b, c
`"Whatap is good."`	띄어쓰기를 키워드에 포함하고 싶은 경우 `''` 또는 `""`로 감쌉니다.	Whatap is good.
`"Whatap\\ is good."`	""로 감싸진 키워드에서 `\`를 포함할 경우, `\\`로 입력해야합니다.	Whatap\ is good.

하이라이트 색상 설정

하이라이팅할 키워드 및 색상을 설정할 수 있습니다.

- 기본적으로 로그 레벨에 따른 하이라이팅(FATAL, ERROR, WARN, SEVERE, CRITICAL)이 적용되어 있습니다.
- 설정한 내용은 **프로젝트 단위**로 저장됩니다.

1. 아이콘을 클릭하세요.
2. **하이라이트** 창에서 추가적으로 색상 설정을 원하는 키워드를 입력창에 입력하세요.
3. 입력창 왼쪽 색상 클릭 시 색을 선택할 수 있습니다.

차트로 로그 조회하기



- 바 차트에서 a 원하는 시간 영역 바를 클릭하거나 드래그하여 해당 시간 범위의 로그를 확인할 수 있습니다.
- 바 차트 아래 로그 테이블 상단 왼쪽의 b 시간 선택 옵션을 이용해 다음과 같이 선택한 시간대에서 더 세분화해 로그를 검색할 수 있습니다.
  - 1min: interval (차트의 막대 사이 간격)
  - 시간 선택 옵션: 선택된 시간 범위를 6개의 구간으로 나눈 시간대

테이블 설정하기

컬럼 설정

컬럼을 추가하거나 순서를 설정할 수 있습니다. 3 영역 오른쪽에서 **컬럼 설정** 버튼을 클릭하세요.

컬럼 추가

태그를 선택하여 테이블에 컬럼을 추가할 수 있습니다. **log** 컬럼 선택을 해제할 경우 **로그 표시 상세 설정**을 확인할 수 없습니다. 반드시 한 개 이상의 컬럼을 선택하세요.

- **제공되는 컬럼**: 로그, 수집 시간, content(로그 메시지)

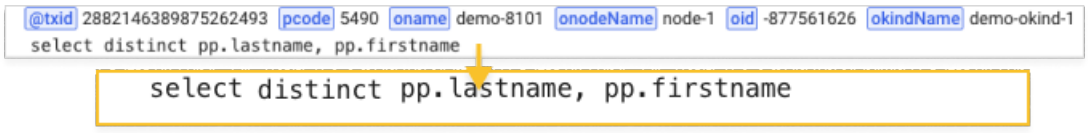
컬럼 순서 변경

컬럼을 추가하면 **표시 항목**에 해당 컬럼이 추가됩니다. 원하는 컬럼을 드래그하여 컬럼의 순서를 변경하세요.

로그 표시 상세 설정

3 영역 오른쪽에서 ⚙ **로그 표시 상세 설정** 버튼을 선택하세요. 기본으로 **content**, **태그** 모두 체크가 되어있으며 두 가지 항목 모두 표시합니다. **content**와 **태그** 중 하나는 반드시 선택하세요.

체크가 해제된 항목은 테이블에 표시되지 않습니다. 다음과 같이 **태그**를 해제할 경우 테이블에서 로그의 **태그**는 표시되지 않습니다.



태그 관리 목록에 태그를 추가하면 추가한 순서대로 로그의 태그가 나열됩니다. 태그의 순서는 드래그하여 변경할 수 있습니다. 추가한 태그를 비활성화하면 비활성화한 태그는 로그의 태그에 노출되지 않습니다.

> ⓘ **컬럼 및 로그 표시 설정 적용 범위**
>
> - **컬럼 설정**과 **로그 표시 상세 설정**은 **라이브 테일**, **로그 검색**, **로그 트렌드**에서 사용할 수 있습니다.
> - 동일한 프로젝트 내 **라이브 테일**, **로그 검색**, **로그 트렌드**는 **컬럼 설정**과 **로그 표시 상세 설정**을 공유합니다.

로그 파일 다운로드

검색한 로그의 결과값을 CSV, TXT 파일 형식으로 수집된 로그를 다운로드 받을 수 있습니다.

1. 화면 중앙의 오른쪽의 **다운로드** 버튼을 클릭합니다.
2. **CSV 다운로드** 또는 **Log 다운로드**를 클릭해 원하는 형태의 파일을 다운로드 받습니다.

로그 검색

> ⓘ **로그 조회** 권한이 없을 경우 해당 메뉴에 진입할 수 없습니다.

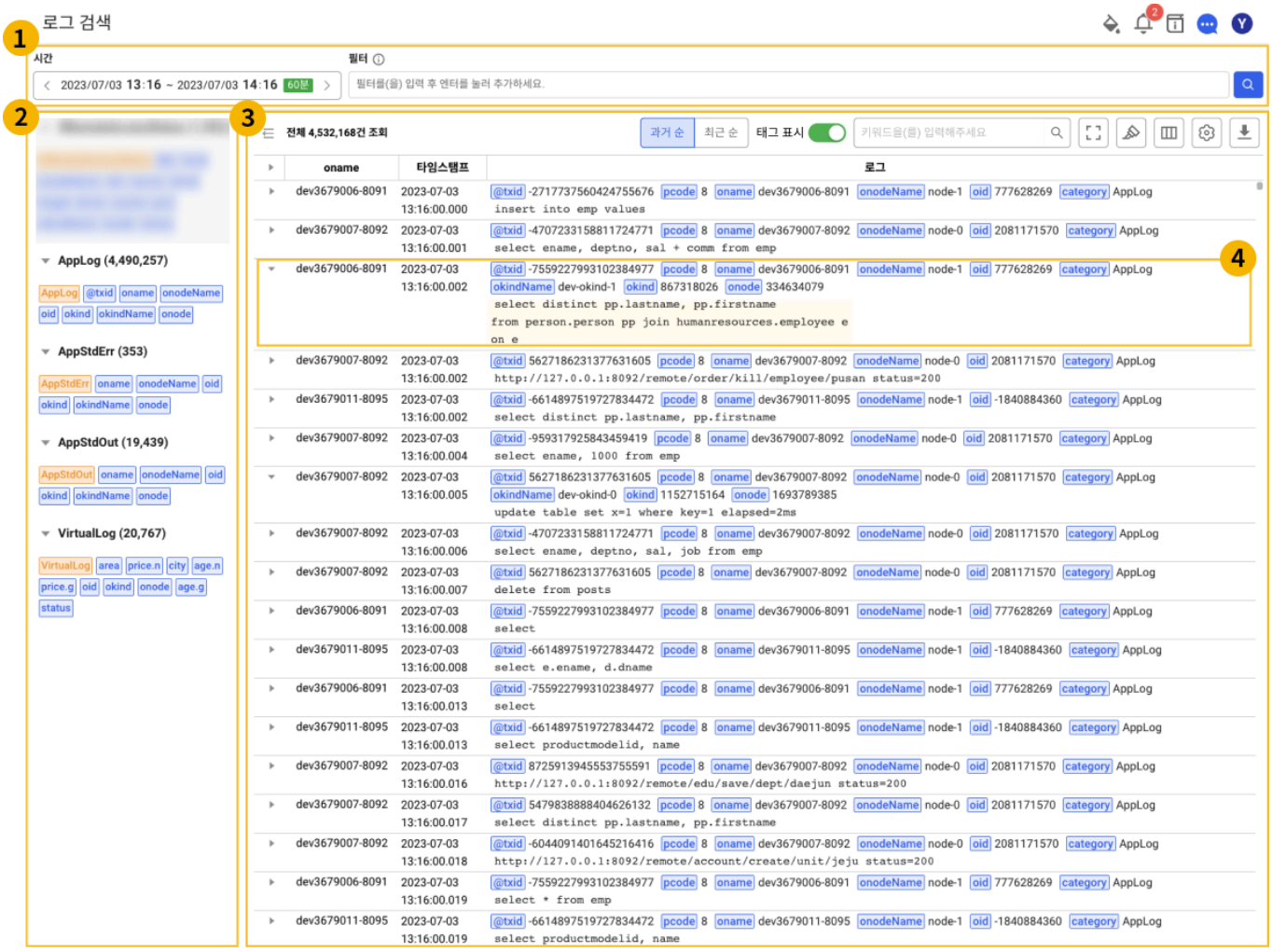
홈 화면 > 프로젝트 선택 > **로그** > **로그 검색**

로그 검색 메뉴에서 통합 수집된 대량의 로그를 다양한 조건으로 검색하고 사용자가 원하는 로그를 특정할 수 있습니다. 복수의 검색 조건을 파싱된 키와 밸류로 지정할 수 있어 원하는 조건에 일치하는 로그 데이터만 추출합니다.

동적 페이지로 검색된 로그 데이터를 정해진 라인 단위로 가져오며, 스크롤 등에 의해 하단에 닿으면 자동으로 다음 데이터를 가져와 표시합니다.

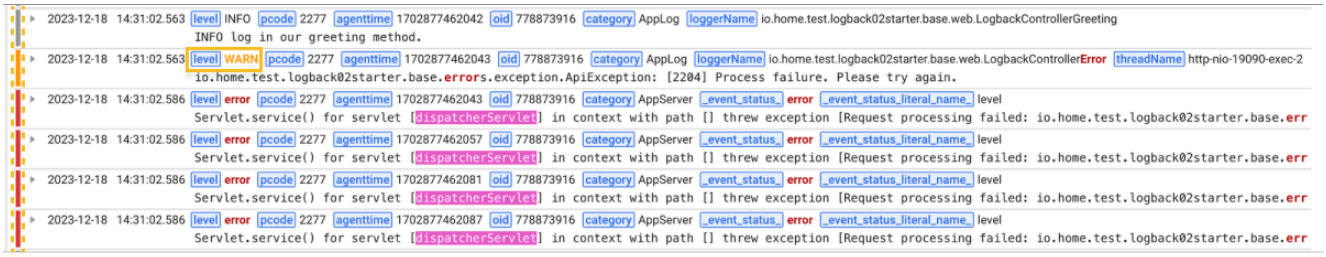
주요 용어는 다음과 같습니다.

- **Category**: 로그의 수집 및 조회 단위
- **Content**: 로그 메시지
- **Search Key**: 로그 파서 설정을 통해 생성되는 검색 키
- **Tag**: 수집된 로그를 검색할 수 있는 태그



데이터 조회하기

- 스크롤이 바닥에 닿으면 다음 데이터를 조회합니다. 한 번에 10,000개의 로그를 조회합니다.
- 3 로그 테이블 상단 왼쪽에서 조회한 총 로그 개수를 확인할 수 있습니다.
- 로그 데이터를 시간 순과 역순으로 조회할 수 있습니다. 3 로그 테이블 상단 오른쪽에서 **과거 순**과 **최근 순** 중 원하는 조회 방식을 선택하세요.
- 시간 범위 지정 후 **적용** 버튼을 선택 해 조회 시간을 설정하고 **검색** 버튼을 선택해 데이터를 조회합니다.
- 3 로그 테이블 상단 오른쪽 **로그 전체 화면** 아이콘을 선택하면 **로그**와 **타임스탬프**를 전체 화면에서 확인할 수 있습니다.
- **에이전트 옵션이 설정된 경우** 로그 레벨을 수집해 로그 레벨 기준 색상이 다음과 같이 표시됩니다.



에이전트 옵션 설정

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- Java 에이전트 2.2.22 버전 이후부터 위빙 설정에 log4j-2.17 또는 logback-1.2.8 설정 시 사용할 수 있습니다. 에이전트 재시작이 필요합니다.
- 로그 레벨은 파싱된 키워드 중 `level`, `type` 기준으로 판별합니다. `level`, `type` 으로 파싱된 키가 존재하고 파싱 값이 FATAL, CRITICAL, ERROR, WARN, WARNING, INFO를 포함할 경우 로그 레벨 색상을 표시합니다.

> ⓘ 로그 테이블 컬럼 가장자리를 드래그해 컬럼 너비를 수정할 수 있습니다.

로그 Content 확인하기

Content는 로그 메시지입니다. 로그 컬럼의 첫 번째 줄은 로그의 파싱(parsing)된 키와 값이고 두 번째 줄은 Content입니다.

- 3 로그 테이블의 행(로그)마다 ▸ **더보기** 버튼이 있습니다. ▸ **더보기** 버튼을 선택하면 4처럼 해당 로그의 전체 Content를 확인할 수 있습니다.
- 로그의 태그를 선택하면 **복사**, **검색**, **제외 검색**, **인접 로그**를 검색 할 수 있는 드롭다운 메뉴가 나타납니다.

필터

필터 적용

1 왼쪽 **시간 선택창**에서 시간 범위를 지정할 수 있습니다. 오른쪽에서 필터를 적용하면 입력한 조건에 맞는 로그를 필터링합니다. 복수의 필터를 입력할 수 있습니다. 필터의 태그가 같은 경우 OR(`||`)로, 그렇지 않은 경우는 AND(`&&`)로 적용됩니다.

입력 창에 값을 직접 입력하거나 **필터** 입력 창을 클릭해 필터를 지정할 수 있습니다. 필터 태그는 `검색 키` , `연산자` , `검색 값` 의 순서로 입력합니다. **검색** 버튼을 선택하면 필터가 적용된 데이터를 로그 목록에서 조회할 수 있습니다.

가이드 UI

다음과 같이 입력 창 아래 **가이드 UI**를 제공합니다. 입력 창 또는 필터 태그에 마우스 커서가 있을 때 ESC 키를 통해 **가이드 UI**를 닫을 수 있습니다.



검색 키, 연산자, 검색 값 입력

- `검색 키` 입력 시 일반 인덱스, 예약어 인덱스, 숫자만 입력할 수 있는 인덱스를 구분해 추천 값을 제공합니다
- `연산자` 입력 시 일반 인덱스 검색 키의 경우 `==` , `!=` 옵션을 하단에 안내합니다. 숫자만 입력할 수 있는 인덱스의 경우 `>` , `<` , `<=` , `>=` , `==` , `!=` 옵션을 제공합니다.
- `검색 값` 입력 시 일치 검색(`>` , `<` , `<=` , `>=` , `==`)일 때 파란색으로, 제외 검색(`!=`)일 때 붉은색으로 하이라이팅합니다. 대소문자 구분 옵션을 활용해 검색할 수 있습니다.

> ⓘ
> - 필터 태그가 2줄 이상 길어지는 경우 ∧ **접기** 아이콘을 선택해 접어둘 수 있습니다.
> - 필터 태그 입력 후 입력 창에서 Shift와 Tab 키를 동시에 사용하여 이전 필터 태그로 이동할 수 있습니다.
> - 필터 태그 입력 후 입력 창에서 Tab 키를 통해 다음 필터 태그로 이동할 수 있습니다.

필터 태그 추가

입력 창에 텍스트 입력 후 Enter 또는 Tab 키를 누르거나, 입력창 하단의 가이드 UI에서 추천 값을 클릭하거나 방향키로 선택해 추가할 수 있습니다.

필터 태그 제거

Backspace 키 또는 태그의 **X** 아이콘을 눌러 태그를 삭제할 수 있습니다. 입력창 오른쪽의 **X** 아이콘을 클릭해 모든 태그를 한 번에 삭제할 수 있습니다.

필터 즐겨찾기

필터 **즐겨찾기** 기능을 제공합니다. 입력 창 오른쪽의 ☆ **즐겨찾기** 아이콘을 선택하여 원하는 필터 검색 조건을 즐겨찾기에 **추가**, **제거** 및 기존 즐겨찾기를 선택할 수 있습니다. **즐겨찾기**는 최대 50개까지 저장됩니다.

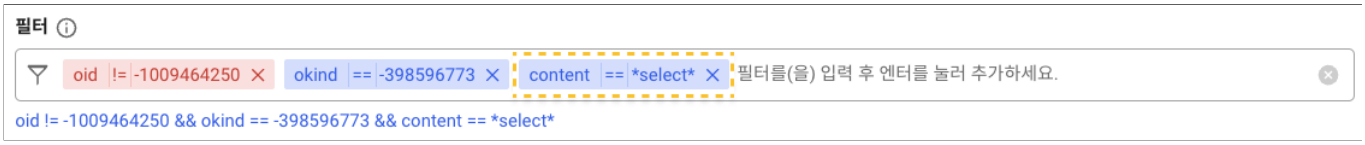


필터 적용 예외 상황

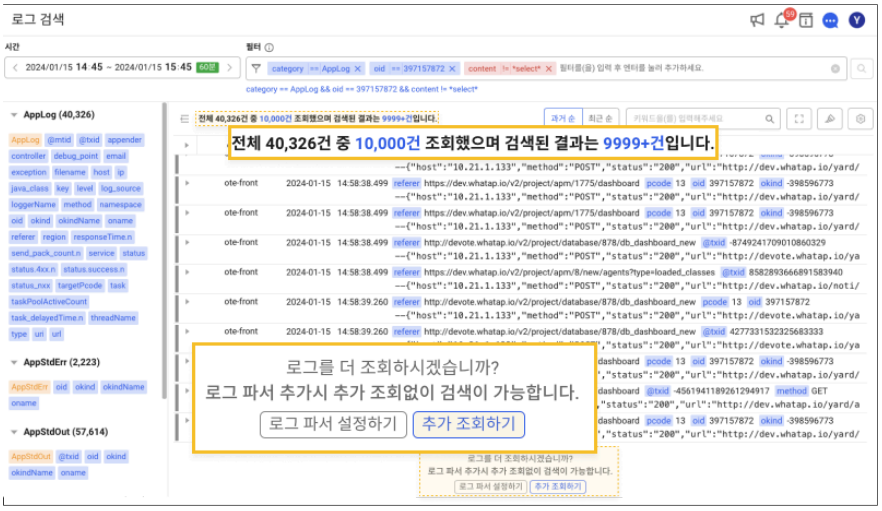
- 숫자만 입력할 수 있는 인덱스(.n 으로 끝나는 검색 키)를 입력한 태그에서 검색 값 은 숫자만 입력할 수 있습니다.
- 중복된 검색 키 , 검색 값 은 입력할 수 없습니다.
- 검색 키 , 검색 값 중 하나라도 없는 태그가 존재할 때 검색할 수 없습니다. 유효하지 않는 태그의 경우 회색으로 표시합니다.

미파싱 키워드 필터 적용

로그에서 파싱되지 않은 즉 인덱스가 생성되지 않은 키워드를 포함한 로그를 조회할 수 있습니다. 이 경우 지정 범위 내 모든 로그를 Full Scan합니다. 그렇기 때문에 인덱스가 생성된 키와 비교해 검색 속도가 다소 떨어질 수 있습니다. 정형화된 로그 데이터의 경우 **로그 파서 설정**을 통해 인덱스 키 값을 활용해 검색하는 것을 권장합니다.



1. **카테고리**를 선택하세요. 카테고리 지정이 필수적입니다.
2. **필터** 입력창에 content 기준 띄어쓰기 후 검색을 원하는 키워드를 입력하세요.
   예시, content *select*
3. **검색** 버튼을 클릭해 로그를 조회하세요. 전체 로그 중 일부 먼저 조회합니다. 1회당 검색 결과는 최대 1만 건입니다.
4. 스크롤을 내려 하단의 **추가 조회하기** 버튼 선택 시 추가 조회할 수 있습니다.



ⓘ 전체 로그 중 서버 조회 범위 당 1만 건씩 조회합니다. 서버 조회 범위의 경우 기본 20만 건이지만 전체 로그 양에 따라 비율이 달라질 수 있습니다.
파서 설정에 대한 자세한 내용은 다음 문서를 참조하세요.

필터 수정

필터에 값을 입력한 뒤 입력한 값을 클릭하면 해당 값을 수정할 수 있습니다.



- 입력 창에 텍스트 재입력해 수정할 수 있습니다.
- 입력 창 아래 **가이드 UI**를 통해 추천 값을 선택해 수정할 수 있습니다.

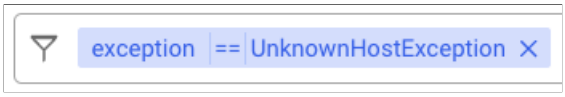
검색 키(Search Key)

검색 키는 로그 데이터 내에서 원하는 특정 값에 접근하기 위한 식별자를 의미합니다. 검색 키에 해당하는 실제 데이터가 검색 값입니다. 왼쪽 2 영역에 있는 태그는 카테고리별로 파싱(parsing) 된 검색 키입니다. 태그를 선택하여 필터를 입력할 수 있습니다. 주황색 태그는 카테고리, 파란색 태그는 검색 키입니다.

예를 들어 2 영역의 AppLog와 AppStdOut은 카테고리, 그 아래 oid와 같은 태그는 파싱(parsing) 된 검색 키입니다. 검색 키는 **로그** > **로그 설정**의 **로그 파서 설정** 탭에서 파싱 로직을 등록해 설정할 수 있습니다.

필터 입력 문법

태그는 검색 키와 검색 값으로 구성되어있습니다. 다음의 예시에서 검색 키는 `exception` , 검색 값은 `UnknownHostException` 입니다. 해당 예시는 수집한 로그 데이터 중 IP 주소와 도메인 주소가 매칭되지 않아 서버를 호스트에 연결할 수 없을 경우 발생하는 예외(`UnknownHostException`)가 포함된 로그 데이터를 조회합니다.



검색 키 종류

검색 키 종류	검색 키 포멧	의미	검색 키와 검색 값 예시	검색 예시
문자열 키워드	keyword	파일 이름	- 키: fileName - 값: /data/whatap/logs/yard.log	fileName:/data/whatap/logs/yard.log
숫자 키워드	keyword.n	응답시간	- 키: response_time.n - 값: 2945	response_time.n>=2945
예약어 키워드 (사전 정의 키워드)	@keyword	트랜잭션 ID	- 키: @txid - 값: 85459614215434144	-

공통 문법

문법 종류	설명	예시
`==searchValue`	검색 값과 일치하는 로그를 검색합니다.	`exception==RuntimeExceptionexception`
`!=searchValue`	검색 값을 제외한 로그를 검색합니다.	`exception!=RuntimeException`
`*searchValue`	검색 값으로 끝나는 로그를 검색합니다.	`word==*hello`
`searchValue*`	검색 값으로 시작하는 로그를 검색합니다.	`word==hello*`
`*searchValue*`	검색 값으로 중간에 포함된 로그를 검색합니다.	`word==*hello*`
`*search*Value*`	검색 값으로 포함된 로그를 검색합니다.	`word==*he*llo*`
`re:{regex}`	정규표현식에 매칭되는 로그를 검색합니다.	`caller==re:^i\.w\.a\.w\.s\.v\.r\.`

문법 종류	설명	예시
**	검색 키에 해당하는 모든 로그를 검색합니다.	

검색 키가 숫자 키워드(keyword.n)인 경우 문법

다음의 문법은 검색 키가 keyword.n 형식인 경우에만 지원합니다.

- 검색 값으로는 숫자만 올 수 있습니다.
- .n 키워드의 값에는 prefix를 붙이지 않습니다. .n 이 아닌 키워드는 모두 prefix를 붙여야합니다.

  예, +>searchValue 는 유효하지 않습니다.

문법 종류	설명	예시
>searchValue	검색 값보다 큰 값이 포함된 로그를 조회합니다.	response_time.n>3000
>=searchValue	검색 값보다 크거나 같은 값이 포함된 로그를 조회합니다.	response_time.n>=3000
==searchValue	검색 값보다 같은 값이 포함된 로그를 조회합니다.	response_time.n==3000
!=searchValue	검색 값보다 다른 값이 포함된 로그를 조회합니다.	response_time.n!=3000
<searchValue	검색 값보다 작은 값이 포함된 로그를 조회합니다.	response_time.n<3000
<=searchValue	검색 값보다 작거나 같은 값이 포함된 로그를 조회합니다.	response_time.n<=3000

개인정보 조회

로그 개인정보 조회 권한이 있을 경우, **개인정보 표시** 토글 버튼 활성화를 통해 비식별화 처리한 개인정보를 표시할 수 있습니다.

- **로그 개인정보 조회** 권한에 대한 자세한 내용은 다음 문서를 참고하세요.
- **로그 개인정보 비식별화** 기능에 대한 자세한 내용은 다음 문서를 참고하세요.

로그 태그 옵션

로그 태그 선택 시 드롭다운 메뉴가 나타납니다. **검색**, **제외 검색**, **인접 로그**, **트랜잭션 정보** 옵션을 확인할 수 있습니다.



로그 태그 옵션 메뉴	설명
복사	태그에 해당하는 로그를 복사합니다.
검색	**필터**에 해당 태그가 포함('==') 조건으로 입력됩니다.
제외 검색	**필터**에 해당 태그가 제외('!=') 조건으로 입력됩니다.
인접 로그	**인접 로그** 상세 창에서 선택한 로그의 서버를 대상으로 선택한 로그와 인접한 시간대의 로그를 조회합니다. 시간을 선택해 인접한 시간대의 로그를 조회할 수 있습니다. 기준 로그는 파란색 바탕으로 표시됩니다.
트랜잭션 정보	로그에 `@txid` 및 `@mtid` 태그가 포함된 경우, **트랜잭션 정보**를 클릭하면 **트랜잭션 정보** 상세창에서 정보를 확인할 수 있습니다.

콘텐츠 하이라이트

로그의 콘텐츠 중 원하는 키워드를 손쉽게 식별하기 위해 하이라이트 기능을 제공합니다. 단일 또는 복수 키워드로 필터링할 수 있습니다.

1. 키워드 입력창에 하이라이트를 원하는 키워드를 입력하세요.

- 예시, `select`

2. 로그 목록의 Content 내 검색한 키워드가 하이라이팅 됩니다.
   - **로그 전체 화면** 아이콘을 선택하면 **로그**와 **타임스탬프**를 전체 화면에서 확인할 수 있습니다.
   - 하이라이트된 키워드 간 이동은 단축키로 조작할 수 있습니다.
     - `Enter` : 다음 row로 이동
     - `Shift+Enter` : 이전 row로 이동
     - ∧ / ∨ 아이콘: row 이동
     - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): 키워드 검색

복수 키워드 조건

복수 키워드로 하이라이팅을 할 경우 다음과 같이 작성합니다.

입력 문자열	설명	결과
`a b c`	띄어쓰기로 각 키워드를 구분합니다.	a, b, c
`"Whatap is good."`	띄어쓰기를 키워드에 포함하고 싶은 경우 `''` 또는 `""`로 감쌉니다.	Whatap is good.
`"Whatap\\ is good."`	""로 감싸진 키워드에서 `\`를 포함할 경우, `\\`로 입력해야합니다.	Whatap\ is good.

하이라이트 색상 설정

하이라이팅할 키워드 및 색상을 설정할 수 있습니다.

- 기본적으로 로그 레벨에 따른 하이라이팅(FATAL, ERROR, WARN, SEVERE, CRITICAL)이 적용되어 있습니다.
- 설정한 내용은 **프로젝트 단위**로 저장됩니다.

1. 아이콘을 클릭하세요.
2. **하이라이트** 창에서 추가적으로 색상 설정을 원하는 키워드를 입력창에 입력하세요.
3. 입력창 왼쪽 색상 클릭 시 색을 선택할 수 있습니다.

테이블 설정하기

컬럼 설정

컬럼을 추가하거나 순서를 설정할 수 있습니다. 3 영역 오른쪽에서 **컬럼 설정** 버튼을 클릭하세요.

컬럼 추가

태그를 선택하여 테이블에 컬럼을 추가할 수 있습니다. **log** 컬럼 선택을 해제할 경우 **로그 표시 상세 설정**을 확인할 수 없습니다. 반드시 한 개 이상의 컬럼을 선택하세요.

- **제공되는 컬럼**: 로그, 수집 시간, content(로그 메시지)

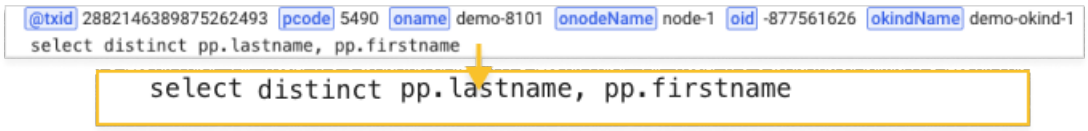
컬럼 순서 변경

컬럼을 추가하면 **표시 항목**에 해당 컬럼이 추가됩니다. 원하는 컬럼을 드래그하여 컬럼의 순서를 변경하세요.

로그 표시 상세 설정

3 영역 오른쪽에서 **로그 표시 상세 설정** 버튼을 선택하세요. 기본으로 **content**, **태그** 모두 체크가 되어있으며 두 가지 항목 모두 표시합니다. **content**와 **태그** 중 하나는 반드시 선택하세요.

체크가 해제된 항목은 테이블에 표시되지 않습니다. 다음과 같이 **태그**를 해제할 경우 테이블에서 로그의 **태그**는 표시되지 않습니다.



태그 관리 목록에 태그를 추가하면 추가한 순서대로 로그의 태그가 나열됩니다. 태그의 순서는 드래그하여 변경할 수 있습니다. 추가한 태그를 비활성화하면 비활성화한 태그는 로그의 태그에 노출되지 않습니다.

> **컬럼 및 로그 표시 설정 적용 범위**
>
> - **컬럼 설정**과 **로그 표시 상세 설정**은 **라이브 테일**, **로그 검색**, **로그 트렌드**에서 사용할 수 있습니다.

- 동일한 프로젝트 내 **라이브 테일**, **로그 검색**, **로그 트렌드**는 **컬럼 설정**과 **로그 표시 상세 설정**을 공유합니다.

로그 파일 다운로드

검색한 로그의 결과값을 CSV, TXT 파일 형식으로 수집된 로그를 다운로드 받을 수 있습니다.

1. 화면 중앙의 오른쪽의 **다운로드** 버튼을 클릭합니다.
2. **CSV 다운로드** 또는 **Log 다운로드**를 클릭해 원하는 형태의 파일을 다운로드 받습니다.

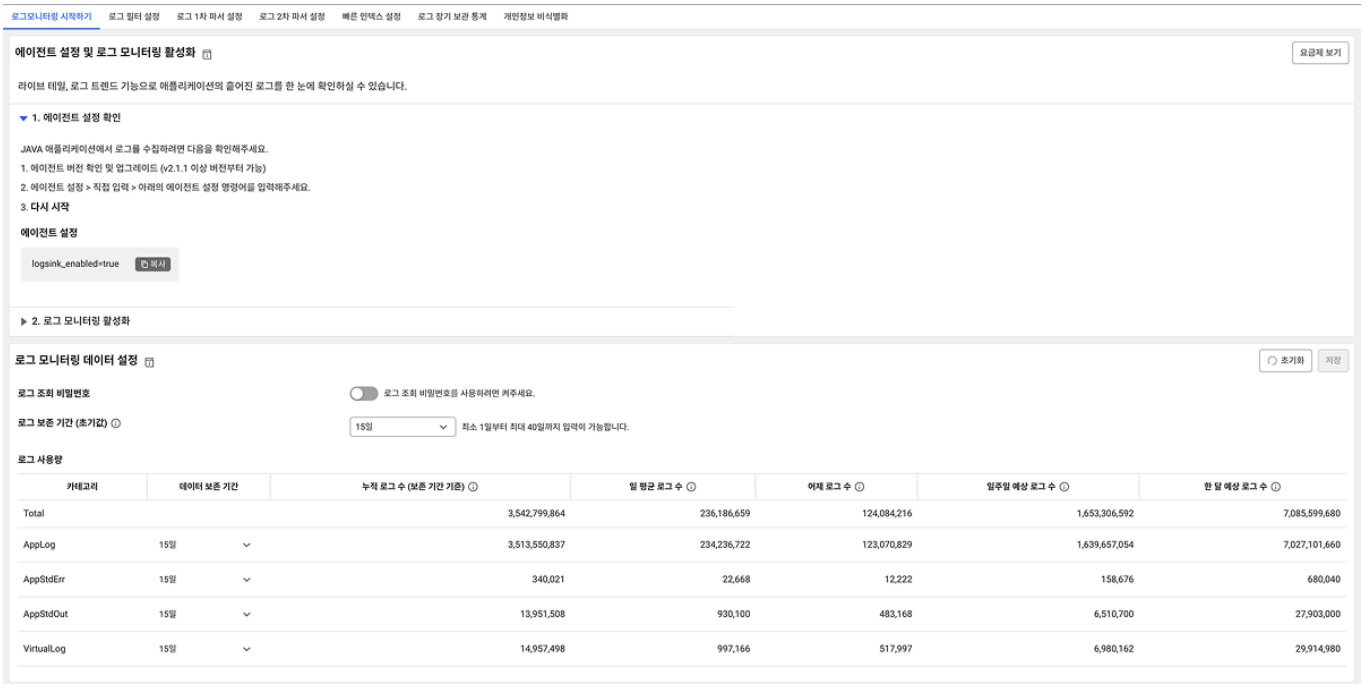
로그 설정

홈 화면 > 프로젝트 선택 > 로그 > 로그 설정

로그 설정의 상단의 탭에서 로그 모니터링을 설정할 수 있습니다. 에이전트 설정 확인, 로그 모니터링의 활성화 여부 결정, 로그 데이터의 유지 기간 및 조회 비밀번호 설정, 로그 파서 등록, 빠른 인덱스 설정, 개인정보 비식별화 관리 등을 할 수 있습니다.

> **권한**
>
> - **로그 모니터링 활성화**을 사용하려면 **프로젝트 수정** 권한이 필요합니다.
> - **로그 편집** 권한이 있으면 **로그 모니터링 활성화** 외 **로그 설정** 메뉴를 수정할 수 있습니다.

로그 모니터링 시작하기



1. **로그 > 로그 설정**의 **로그모니터링 시작하기** 탭을 클릭하세요.
2. **가이드 보기** 아이콘과 **요금제 보기** 버튼을 클릭하면 안내 화면으로 이동합니다.

에이전트 설정 및 로그 모니터링 활성화

1 영역에서 에이전트 설정을 확인하고 로그 모니터링 활성화 및 비활성화 여부를 설정할 수 있습니다.

에이전트 설정 확인

로그 모니터링을 시작하기 위해 에이전트 버전과 설정 정보를 확인하세요. **1. 에이전트 설정 확인**에서 안내에 따라 진행하세요.

- **애플리케이션 모니터링**

  애플리케이션별 적용 방법은 Java, PHP, Node.js, Python, Go 문서를 참고하세요.

- **서버 모니터링**

  서버 모니터링 적용 방법은 Server 문서를 참고하세요. 안내에 따라 whatap.conf 에 로그 감시 대상 파일 설정을 추가하세요.

- **쿠버네티스 모니터링**

  쿠버네티스 모니터링 적용 방법은 Kubernetes 문서를 참고하세요.

로그 모니터링 활성화

로그 모니터링을 활성화 또는 비활성화할 수 있습니다. **2. 로그 모니터링 활성화**에서 로그 모니터링 여부를 설정하세요.

- 로그 모니터링 활성화: 토글 버튼을 켜면 로그 모니터링이 활성화되고, 활성화한 날부터 15일 동안 무료로 체험하실 수 있습니다.
- 로그 모니터링 비활성화: 토글 버튼을 끄면 로그 모니터링이 비활성화되고, 로그를 더 이상 저장하지 않습니다.

로그 모니터링 데이터 설정

로그 사용량을 확인할 수 있습니다. 또한 **로그 보존 기간** 및 **로그 조회 비밀번호** 설정을 변경할 수 있습니다.

로그 조회 비밀번호

보안을 강화하기 위해 **로그 조회 비밀번호**를 설정하세요. 로그 조회 비밀번호 지정은 선택 사항입니다. 로그 조회 비밀번호를 사용 중이라면 로그 화면 진입 시 반드시 비밀번호를 입력해야 합니다.

> **비밀번호 분실**

로그 편집 권한이 있는 사용자는 **로그 설정**에서 새 비밀번호로 변경할 수 있습니다.

로그 보존 기간

공통으로 적용할 기본(default) 데이터 보존 기간입니다.

- 최소 1일부터 최대 40일까지 입력할 수 있습니다. 미지정 시 기본값은 1일입니다. 로그 보존 기간 선택 옵션 외에 사용자가 원하는 기간을 입력할 수 있습니다.
- **로그 사용량** 목록에서 별도로 설정하지 않으면 이 로그 데이터 보존 기간이 기본적으로 적용됩니다. **로그 사용량** 목록에서 카테고리별 데이터 보존 기간을 설정하고 **초기화** 버튼을 클릭하면 기본 데이터 보존 기간으로 초기화됩니다.

로그 사용량

로그 사용량 목록에서 카테고리별 로그 데이터 보존 기간을 지정할 수 있습니다. 로그 수는 해당 기간 동안 쌓인 로그 건수를 의미합니다. 예를 들어 **금일 로그수**는 하루 동안 쌓인 로그 건수, **예상 로그수**는 데이터 보관일에 금일 로그 수를 곱한 로그 건수를 의미합니다.

로그 데이터 보존 기간을 다음과 같이 지정할 수 있습니다. 기간 지정에 따라 오래된 데이터를 삭제해 공간을 확보할 수 있습니다.

- **트라이얼 프로젝트**

  데이터 보존 기간으로 1일, 2일, 3일을 선택할 수 있습니다.

- **유료 프로젝트**

  데이터 보존 기간으로 1일, 2일, 3일, 4일, 5일, 6일, 7일, 10일, 30일, 40일을 선택할 수 있습니다.

- **저장량 기준 과금**

  데이터 보존 기간에 따라 비용이 달라집니다.

  예, 일 평균 200만 로그 건수가 쌓이고 데이터 보존 기간을 3일로 지정한 경우라면 평균 600만 로그 건수가 수집 서버에 유지되고 과금 대상이 됩니다.

로그 필터 설정

로그 필터 설정은 사용자가 지정한 필터 조건에 해당하는 로그만 저장하는 기능입니다. 사용자가 필요한 로그만 저장하여 비용을 절감할 수 있습니다.

> **권한**
>
> 로그 편집 권한이 있는 사용자만 설정할 수 있습니다.

로그 필터 설정 추가하기

새로운 필터 설정을 추가하는 방법입니다. 로그 필터 설정 시 **필터 이름, 라이브 테일 표시 여부, 카테고리, 필터 조건**은 필수 입력 항목입니다.

로그 필터 설정에서 저장하지 않는 로그도 **라이브 테일 표시 여부** 토글을 활성화하면 **라이브 테일**에서 확인할 수 있습니다.

> 카테고리별로 하나의 필터만 설정할 수 있습니다.

1. **로그 > 로그 설정 > 로그 필터 설정** 탭으로 이동하세요.
2. 로그 필터 설정을 위해 **+ 추가하기** 버튼을 클릭하세요.
3. **로그 필터 설정 추가**에서 **필터 이름, 라이브 테일 표시 여부, 카테고리, 필터 조건**을 입력하세요.
    - **필터 이름**: 필터를 식별할 수 있는 이름
    - **라이브 테일 표시 여부**: 저장 여부와 관계없이 라이브 테일에 표시 여부 선택
    - **카테고리**: 필터가 적용될 로그 카테고리 **(복수 선택 가능)**
    - **필터 조건**: 필터 조건에 저장할 로그의 조건

    > 필터 조건 설정 시, 자세한 내용은 WhaTap log search query 문법를 참고하세요.

    - **예시**

- **포함**: `oid : 123`
  - oid가 123인 로그만 수집합니다.
- **불포함**: `not oid : 123`
  - oid가 123이 **아닌** 로그만 수집합니다.
- **전체 포함**: 필터 조건 미입력
- **전체 불포함**: `not content : *`

4. **추가** 버튼을 클릭하면 **등록 결과 미리보기**에서 각 필터 설정을 확인하고 수정할 수 있습니다.
5. 확인이 완료되었다면 **적용** 버튼을 클릭해 로그 필터 설정을 적용합니다.

시뮬레이션

사용자가 입력한 카테고리와 필터 조건으로 로그를 미리 검색하여, 로그 필터 설정을 적용하기 전에 결과를 확인할 수 있습니다.

시뮬레이션 버튼을 클릭하면 로그 필터 설정이 적용된 **로그 검색** 화면이 나타납니다.

로그 필터 설정 수정하기

등록한 로그 필터는 필요에 따라 수정 아이콘을 클릭해 수정할 수 있습니다. **로그 필터 설정 수정**에서 기존에 입력한 값이 표시되며, 원하는 항목을 변경한 뒤 적용하면 됩니다.

> ⚠ 로그 필터 설정을 수정할 때, 하나의 카테고리만 선택할 수 있습니다.

로그 필터 설정 삭제하기

등록한 필터는 삭제 아이콘을 클릭해 삭제할 수 있습니다. 삭제 후 반드시 **저장** 버튼을 클릭하세요.

> ⚠ 변경 사항을 저장하지 않으면 필터가 삭제되지 않습니다.

로그 필터 설정 JSON 가져오기/내보내기

로그 필터 설정은 JSON 형식으로 Import/Export 할 수 있습니다. 다른 프로젝트로 동일한 로그 필터 설정을 쉽게 적용할 수 있습니다. 다른 프로젝트로 로그 필터 설정 내용을 그대로 옮기고 싶은 경우 유용하게 사용할 수 있습니다.

JSON 내보내기

1. 화면 오른쪽 위의 JSON 버튼을 클릭하세요.
2. **JSON 내보내기** 창이 열리면, **내보내기** 버튼을 클릭하세요.
3. 로그 필터 설정 내용이 JSON 형식의 파일로 다운로드됩니다.

JSON 가져오기

1. 화면 오른쪽 위의 버튼을 클릭하세요.
2. 가져올 JSON 형식의 파일을 선택하세요.
3. **JSON 가져오기** 창에서 **목록에 추가하기** 또는 **덮어쓰기** 버튼을 클릭해 로그 필터 설정을 가져옵니다.
4. **저장** 버튼을 클릭해 변경 사항을 반영합니다.

유효성 검사 조건

설정 내용 저장 시 유효성 검사 내용은 다음과 같습니다.

- **type**: `GENERAL` 만 지원
- **category**: 필수값으로 `string` 타입 지원
- **condition**
  - **language**: `jsonDsl` , `lucene` 만 지원
  - **description**: 유효하지 않은 필터 조건이면 저장 불가
- **showLiveTail**: `boolean` 타입만 지원
- **name**: 필수값으로 `string` 타입 지원

로그 파서 설정

로그 1차 파서 설정

로그 설정 상단의 **로그 1차 파서 설정** 탭을 선택해 수집된 로그에 대한 파서를 등록할 수 있습니다.

로그 1차 파서는 **GROK**과 **JSON** 파서를 제공하며 유형별 로그 발생 수 집계나 특정 로그 탐색을 위한 필수 파서입니다. 패턴 조건에 맞는 검색 키와 검색 값을 추출하며, 파싱된 키는 로그 유형 분류 및 검색 인덱싱에 활용합니다.

- **GROK**: 기본은 정규 표현식 기반 파싱에 해당합니다. 예약 키워드 기반의 파싱을 제공합니다.
- **JSON**: 로그 중 JSON으로 출력된 부분에 대해서 일괄 파싱을 제공합니다.

> **파싱 로직 미등록 시 검색 가능한 key**
>
> `category` , `oid` , `oname` , `okind` , `okindName` , `@txid` , `@login` , `httphost`
>
> **파서 등록이 불가한 예약어**
>
> `timestamp` , `message` , `pcode` , `category` , `content` , `logContent`
>
> - 다음 예약어의 경우 파서를 등록하더라도 인덱스가 생성되지 않습니다.

> 로그 파서에 대한 자세한 내용은 다음 문서를 참고하세요.

설정 항목

설정 값	설명	기타
카테고리	패턴을 적용할 카테고리	required
로그 검출 조건	필터로 적용할 검색 키, 검색 값을 입력 - 로그 검출 조건에 맞는 로그 데이터에만 패턴 적용, 로그 검출 조건을 입력하지 않으면 모든 로그를 대상으로 패턴 적용	optional

설정 값	설명	기타
패턴	로그를 파싱(parsing)할 패턴 - 작성한 패턴에 맞추어서 파싱을 하고 인덱스 생성 GROK, 정규 표현식 문법 지원	required

파서 목록

등록한 파서를 조회하고, 추가 및 편집할 수 있습니다. 파서는 등록 순서대로 적용되며, 최초로 일치하는 파서만 적용됩니다.

- 상단 오른쪽 **+ 추가하기** 버튼을 클릭하면 **파서 추가** 창이 나타납니다.
- **적용 순서** 컬럼의 ⦀ 아이콘을 드래그해 파서 설정 순서를 변경할 수 있습니다.
- **활성화** 토글로 파서 활성화 여부를 지정할 수 있습니다.
- **수정** 및 **삭제** 아이콘을 통해 등록한 파서를 수정 및 삭제할 수 있습니다.

파서 등록

다음은 파서를 등록하는 공통 순서입니다.

1. **로그 1차 파서 설정**에서 **+ 추가하기** 버튼을 클릭하세요.
2. **파서 추가** 창에서 **파서**를 선택하세요.

   > **파서 및 패턴 등록 방법**
   >
   > - **GROK** 파서 및 패턴 등록
   > - **JSON** 파서 및 패턴 등록

3. 파서를 선택하면 **패턴 등록** 버튼이 활성화됩니다. **패턴 등록** 버튼을 클릭한 후, **파서 시뮬레이션** 창에서 패턴과 로그를 입력합니다.

   a. **패턴**을 입력하세요.

   b. **로그**의 **카테고리**와 **시뮬레이션 파싱 로그**를 입력하세요. **로그 검색** 버튼을 클릭한 후, 검색된 목록에서 원하는 로그 오른쪽의 **선택** 버튼을 클릭하면 로그가 자동으로 입력됩니다.

> ⓘ **로그 검색**을 통해 수집된 로그 중 예시 로그를 선택할 수 있습니다. 선택한 로그는 시뮬레이션 입력창에 자동 반영되어 복사/붙여넣기 없이 바로 테스트할 수 있습니다.

c. 등록하려는 패턴이 정상인지 **시뮬레이션** 버튼을 클릭해 시뮬레이션 및 패턴의 퍼포먼스를 측정하세요.

> ⓘ 시뮬레이션과 퍼포먼스 측정에 관한 자세한 내용은 다음 문서를 참고하세요.

4. 성공적으로 파싱되는지 시뮬레이션 결과를 확인한 후, **패턴 적용** 버튼을 클릭하세요.
5. **추가** 버튼을 클릭하면 파서가 추가됩니다.

> ⓘ **개인정보 비식별화**
>
> 파서 등록 시 검색 키 이름을 `검색 키.p` 로 지정하면 비식별화 대상이 됩니다(예: `myname.p` `**`). 와탭은 **개인정보 비식별화** 탭에서 비식별화를 관리하는 방식을 권장합니다. 자세한 내용은 개인정보 비식별화를 참고하세요.

GROK 파서 패턴 등록

기본 문법은 `%{SYNTAX:SEMANTIC}` 입니다. GROK 파서에 대한 자세한 내용은 다음 문서를 참조하세요. 패턴 등록 및 시뮬레이션이 필수적입니다.

- **SYNTAX**

  GROK 정의 패턴입니다.

- **SEMANTIC**

  파싱된 데이터에 할당할 키입니다.

> ⓘ SEMANTIC에는 예약어 등이 사용되지 않도록 조합어 사용을 권장합니다.

JSON 포맷 파서 패턴 등록

로그 전체 또는 일부가 JSON 형태로 출력되는 경우 해당 부분을 파싱할 수 있습니다. **Prefix**, **Postfix** 옵션으로 JSON 인식 범위를 지정합니다. JSON 파서에 대한 자세한 내용은 다음 문서를 참조하세요. 시뮬레이션이 필수적입니다.

옵션	설명
Prefix	JSON 문자열의 시작 부분 앞의 문자열을 지정합니다. 미지정 시 로그 출력문의 맨 앞부터 JSON 문자열로 식별합니다.
Postfix	JSON 문자열의 종료 부분 뒤의 문자열을 지정합니다. 미지정 시 로그 출력문의 맨 뒤까지를 JSON 문자열로 식별합니다.
Ignore	JSON 출력부 중 키 추출을 제외할 필드를 지정합니다.

- **등록 예시**

  Log

  ```
  [2022-10-25 10:15:34:145]...(개행)
  Request : {"key1":"value1","key2":"value2",...}(개행)
  Response : {"key3":"value3","key4":"value4",...}
  ```

  예시처럼 유입되는 로그가 Request JSON, Response JSON을 모두 파싱하고자 하는 경우 다음의 2가지 패턴을 등록합니다.

  - Request 파싱용 패턴

    "Request : " 와 "Response" 사이의 문자열 `{"key1":"value1","key2":"value2",...}` 대상

  - Response 파싱용 패턴

    "Response : " 부터 로그의 마지막 까지의 문자열 `{"key3":"value3","key4":"value4",...}` 대상

- **JSON 커스텀 패턴 등록**

  로그 중 일부가 JSON 형태로 출력되는 경우 JSON으로 출력된 부분을 전용 커스텀 파서를 통해 파싱할 수 있습니다. 패턴을 다음과 같이 입력하세요.

  ```
  io.whatap.logsink.parser.JsonFormatParser{}
  ```

  로그 중 JSON 형태로 출력된 부분을 검출하기 위해 **Prefix**, **Postfix** 옵션을 조합해 로그의 어느 부분을 JSON으로 인식하여 파싱할지 지정하세요.

  `JsonFormatParser{}` 의 `{}` 에 옵션을 지정합니다.

  - **등록 예시**

    Log

```
[2022-10-25 10:15:34:145]...(개행)
Request : {"key1":"value1","key2":"value2",...}(개행)
Response : {"key3":"value3","key4":"value4",...}
```

예시처럼 유입되는 로그가 Request JSON, Response JSON을 모두 파싱하고자 하는 경우 다음의 2가지 패턴을 등록합니다.

- Request 파싱용 패턴

  "Request : " 와 "Response" 사이의 문자열 `{"key1":"value1","key2":"value2",...}` 대상

```
io.whatap.logsink.parser.JsonFormatParser {prefix:"Request : ",postfix:"Response"}
```

- Response 파싱용 패턴

  "Response : " 부터 로그의 마지막 까지의 문자열 `{"key3":"value3","key4":"value4",...}` 대상

```
io.whatap.logsink.parser.JsonFormatParser {prefix: "Response : "}
```

파서 시뮬레이션 및 퍼포먼스 측정

파서 시뮬레이션 후 패턴을 등록할 수 있습니다. **퍼포먼스 측정**은 시뮬레이션 수행 대상 문자열에 대하여 파서의 반복 파싱 소요 시간을 측정합니다.

1. **패턴**과 **로그**를 입력하세요. **로그 검색** 버튼을 클릭한 후, 검색된 목록에서 원하는 로그 오른쪽의 **선택** 버튼을 클릭하면 로그가 자동으로 입력됩니다.
2. **시뮬레이션** 버튼을 클릭하여 등록하려는 패턴으로 파싱에 성공하는지 확인하세요.
3. 시뮬레이션 성공 시 **시뮬레이션 결과** 및 **퍼포먼스 측정 결과**를 조회할 수 있습니다.
4. 시뮬레이션 후 **패턴 적용** 버튼 클릭 시 선택한 파서에 입력한 패턴이 적용됩니다.

파싱 성공

파싱 로직을 등록해 **키**(key)가 생성되면 로그 조회 시 해당 키로 파싱된 값이 추가됩니다. 다음 **라이브 테일** 메뉴 예시와 같이 파싱된 키와 값이 추가됩니다.



파싱된 키는 **라이브 테일**, **로그 검색**, **로그 트렌드**에서 확인할 수 있습니다.

로그 2차 파서 설정

로그 설정 메뉴 상단에서 **로그 2차 파서 설정** 탭을 선택해 로그 파서를 등록할 수 있습니다.

로그 2차 파서는 1차 파서(GROK 또는 JSON)로 추출한 값을 가공해 통계 데이터나 새로운 필드를 생성합니다. HTTP 상태 코드 기반 통계 파서와 함께 키워드 필터, 필드 가공, 응답 시간 분류, GeoIP 변환 등 다양한 프로세서를 제공합니다.

> ⓘ 로그 2차 파서는 1차 파싱된 결과에 대하여 특수 목적의 2차 파싱 기능을 제공합니다. 2차 파서를 사용하기 위해서는 **1차 파서**가 **등록**되어 있어야 합니다.

파서 목록



등록한 파서를 조회하고, 추가 및 편집할 수 있습니다.

파서 추가, 순서 변경, 활성화 토글, 수정 및 삭제 등의 조작 방법은 1차 파서 목록과 동일합니다.

파서 등록 순서

다음은 파서를 등록하는 공통 순서입니다.



1. **+ 추가하기** 버튼을 선택하면 **파서 추가** 창이 나타납니다.
2. **파서** 선택 창에서 파서를 선택하세요. 각 파서별 설정 항목은 다음 문서를 참조하세요.
   - **4xx, 5xx 상태 코드 파서** 설정 항목
   - **상태 코드 성공률 파서** 설정 항목
3. **제외할 상태 코드**를 입력하세요.
4. **카테고리** 선택 창에서 카테고리를 선택하거나 직접 입력하세요.
5. **로그 검출 조건**을 선택하거나 직접 입력하세요.
6. **추가** 버튼을 선택해 파서를 등록하세요.

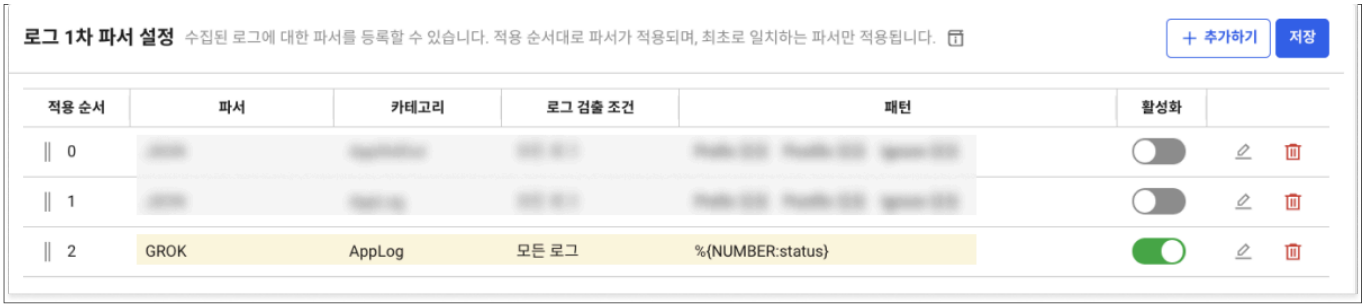
4xx, 5xx 상태 코드 파서 설정 항목

4xx, 5xx 상태 코드 파서는 1차 파서로 파싱된 status 값을 기반으로 4xx, 5xx 건수 데이터를 생성합니다. 특정 상태 코드를 제외하려면 제외할 상태 코드를 입력하세요.

설정 항목

설정 값	설명	기타
카테고리	4xx, 5xx건수 데이터를 생성할 카테고리입니다.	required
로그 검출 조건	필터로 적용할 검색 키, 검색 값을 입력합니다. 로그 검출 조건에 맞는 로그 데이터에 대해서만 4xx, 5xx건수 데이터를 생성합니다. 로그 검출 조건을 입력하지 않으면 모든 로그를 대상으로 데이터를 생성합니다.	optional
제외할 상태 코드	통계 데이터 생성 시 제외할 상태 코드입니다. 입력하지 않으면 4xx~5xx에 해당하는 전체 오류 상태 코드를 대상으로 4xx, 5xx건수 데이터를 생성합니다.	optional

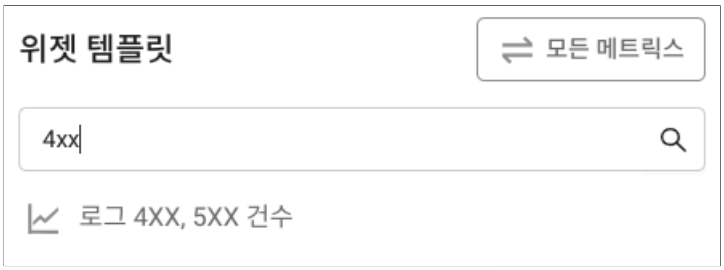
status 파서 등록 예시



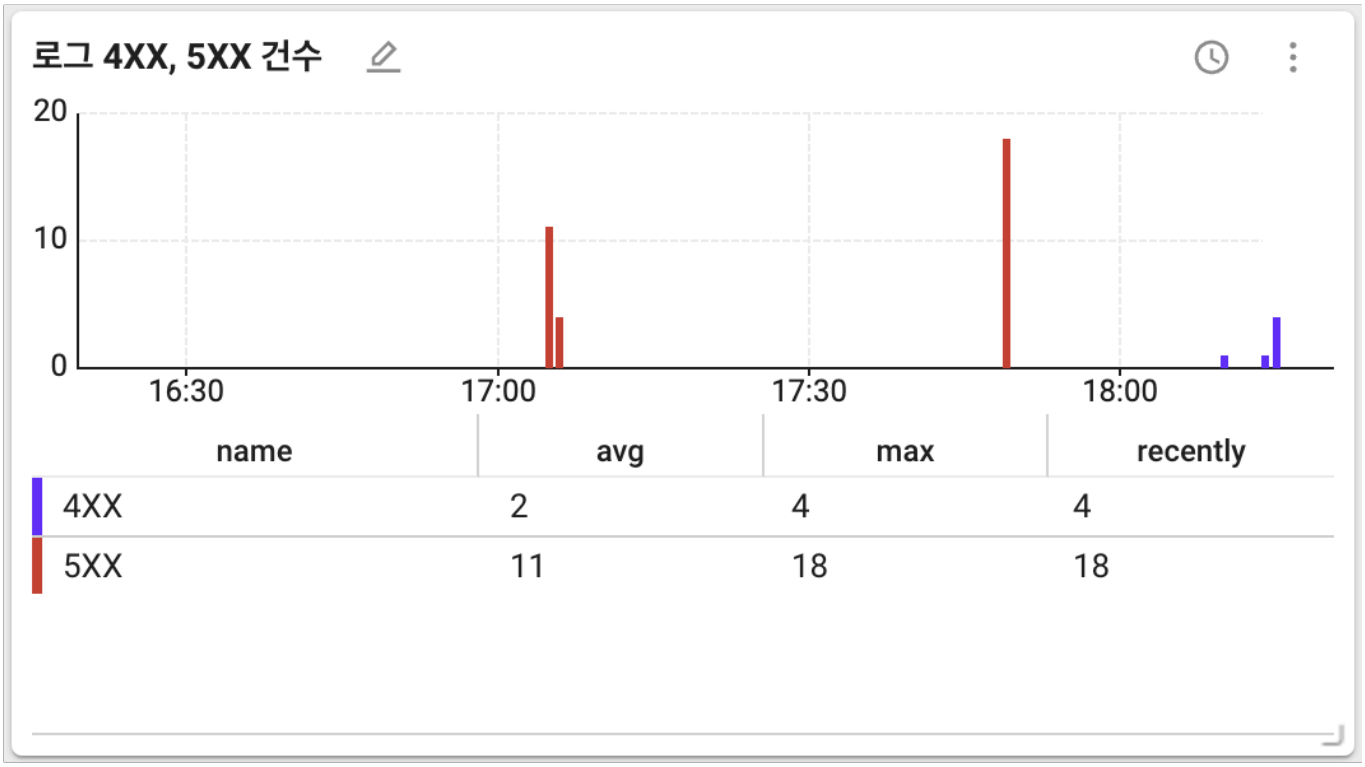
유입되는 로그가 `{"msg":"message","status":404}` 이고 예시처럼 GROK 파서로 status를 파싱한다면, `status: 404` 와 같이 파싱됩니다. status가 정상적으로 파싱되는 것을 확인했다면 4XX,5XX 상태 코드 파서에서 제외할 상태 코드를 등록하세요.

데이터 조회

파서를 모두 등록하면 **통합 Flex 보드**로 이동해 **로그 4XX, 5XX 건수** 위젯을 생성하세요.



위젯을 생성하면 다음과 같이 데이터를 확인할 수 있습니다.



- **avg**: 조회 기간 데이터 평균값입니다.
- **max**: 조회 기간 데이터 중 최댓값입니다.
- **recently**: 조회 기간 데이터 중 마지막 값입니다.

상태 코드 성공률 파서 설정 항목

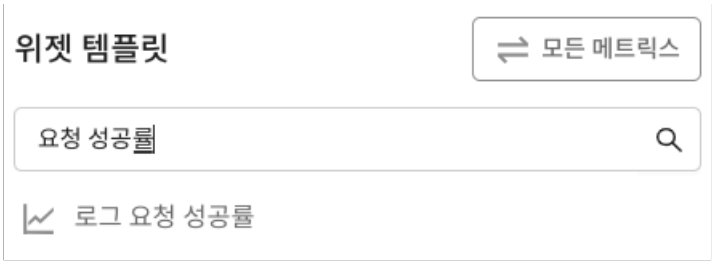
상태 코드 성공률 파서는 1차 파서로 파싱된 status 값을 기반으로 HTTP 요청 성공률 데이터를 생성합니다. 특정 상태 코드를 제외하려면 제외할 상태 코드를 입력하세요. status 파싱에 관한 내용은 4xx, 5xx 상태 코드 파서 설정 항목을 참조하세요.

설정 항목

설정 값	설명	기타
카테고리	요청 성공률 데이터를 생성할 카테고리입니다.	required
로그 검출 조건	필터로 적용할 검색 키, 검색 값을 입력합니다. 로그 검출 조건에 맞는 로그 데이터에 대해서만 요청 성공률 데이터를 생성합니다. 로그 검출 조건을 입력하지 않으면 모든 로그를 대상으로 데이터를 생성합니다.	optional
제외할 상태 코드	요청 성공률 데이터 생성 시 제외할 상태 코드입니다. 입력하지 않으면 2xx~3xx에 해당하는 전체 성공 상태 코드를 대상으로 요청 성공률 데이터를 생성합니다.	optional

데이터 조회

파서를 모두 등록하면 **통합 Flex 보드**로 이동해 로그 요청 성공률 위젯을 생성하세요.



위젯을 생성하면 다음과 같이 데이터를 확인할 수 있습니다.

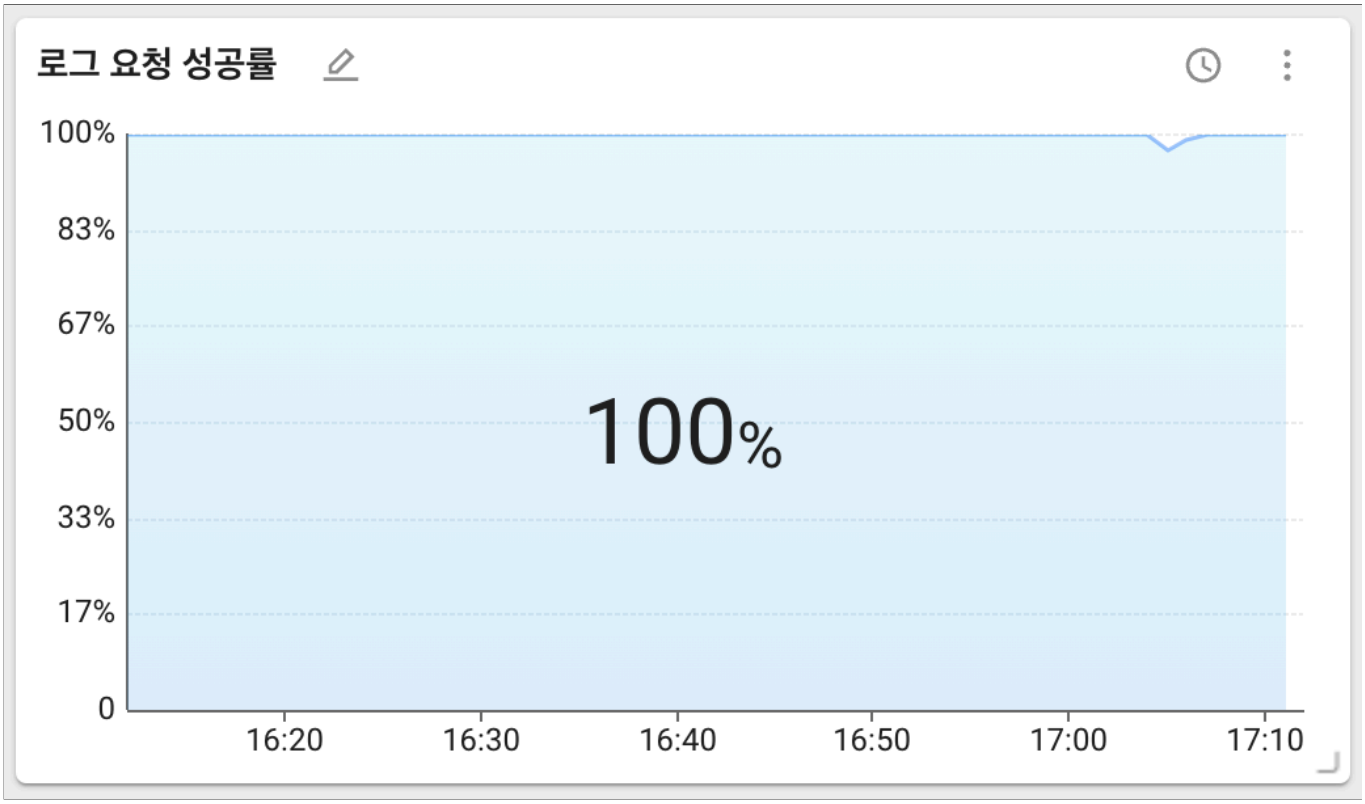


차트 위 데이터는 조회 기간에 대한 통계를 나타냅니다. 통계 방법을 최근값, 최댓값, 평균값 등으로 선택할 수 있습니다. 최근값이 기본으로 선택되어 있습니다.

2차 파서 프로세서

로그 2차 파서는 **4xx, 5xx 상태 코드 파서**와 **상태 코드 성공률 파서** 외에도 다음과 같은 필드 가공 프로세서를 제공합니다.

로그 2차 파서 프로세서 목록

프로세서	설명
모든 키워드 포함 필터	지정한 모든 키워드가 포함된 로그에 플래그 필드를 추가합니다.
일부 키워드 포함 필터	지정한 키워드 중 하나라도 포함된 로그에 플래그 필드를 추가합니다.
필드명 변경	기존 필드명을 다른 이름으로 변경합니다.

프로세서	설명
상태 코드 그룹 분류	HTTP 상태 코드를 2xx/3xx/4xx/5xx 그룹으로 자동 분류합니다.
URL 정적/동적 분류	URL에 확장자(`.`)가 있으면 정적 콘텐츠, 없으면 동적 서블릿으로 분류합니다.
URL 파라미터 제거	URL에서 `?` 이후의 쿼리 파라미터를 제거한 값을 저장합니다.
응답 시간 분류	응답 시간(ms)을 fast/normal/slow/critical 구간으로 자동 분류합니다.
JSON 필드 추출	JSON 문자열 필드에서 특정 키를 추출하여 새 필드로 저장합니다.
GeoIP 변환	IP 주소를 국가/도시 정보로 변환합니다.
직접 입력	FQN 클래스명으로 프로세서를 직접 등록합니다.

모든 키워드 포함 필터

지정한 **모든** 키워드가 포함된 로그에 플래그 필드를 추가합니다.

설정 값	설명	기본값
입력 필드	키워드를 검색할 필드명	`content`
저장 필드	결과를 저장할 필드명	`has_str_all`
keyword	검색할 키워드 목록 (쉼표로 구분해 입력)	required

활용 예시: 입력 필드를 `content`, 키워드를 `error, timeout, database`로 설정하면 세 키워드가 모두 포함된 로그에만 `has_str_all` 필드가 추가됩니다.

일부 키워드 포함 필터

지정한 키워드 중 **하나라도** 포함된 로그에 플래그 필드를 추가합니다.

설정 값	설명	기본값
입력 필드	키워드를 검색할 필드명	`content`
저장 필드	결과를 저장할 필드명	`has_str_any`
keyword	검색할 키워드 목록 (쉼표로 구분해 입력)	required

활용 예시: 입력 필드를 `content`, 키워드를 `reboot, shutdown, restart`로 설정하면 키워드 중 하나라도 포함된 로그에 `has_str_any` 필드가 추가됩니다.

필드명 변경

기존 필드명을 다른 이름으로 변경합니다. 대상 필드명이 이미 존재하면 덮어쓰지 않습니다.

설정 값	설명	기본값
원본 필드	변경할 기존 필드명	required
새 필드명	변경 후 필드명	required

상태 코드 그룹 분류

HTTP 상태 코드를 `status_2xx`, `status_3xx`, `status_4xx`, `status_5xx` 그룹으로 자동 분류합니다.

설정 값	설명	기본값
입력 필드	상태 코드가 담긴 필드명	`status`
저장 필드	그룹 분류 결과를 저장할 필드명	`status_nxx`

활용 예시: `status` 값이 `404`이면 `status_nxx` 필드에 `status_4xx`가 저장됩니다.

URL 정적/동적 분류

URL에 확장자(`.`)가 포함되어 있으면 **정적 콘텐츠**, 포함되지 않으면 **동적 서블릿**으로 분류합니다.

설정 값	설명	기본값
입력 필드	URL이 담긴 필드명	`url`
저장 필드	분류 결과를 저장할 필드명	`url_type`

URL 파라미터 제거

URL에서 `?` 이후의 쿼리 파라미터를 제거한 경로만 저장합니다.

설정 값	설명	기본값
입력 필드	URL이 담긴 필드명	`url`
저장 필드	파라미터 제거 결과를 저장할 필드명	`url_without_params`

활용 예시: `/api/v1/users?page=1&size=20` → `url_without_params` 필드에 `/api/v1/users`가 저장됩니다.

응답 시간 분류

응답 시간(ms)을 구간별로 자동 분류합니다. 구간 임계값을 직접 설정할 수 있습니다.

설정 값	설명	기본값
입력 필드	응답 시간(ms)이 담긴 필드명	`elapsed`
저장 필드	분류 결과를 저장할 필드명	`response_class`
Fast 상한(ms)	fast 구간 상한값	`100`
Normal 상한(ms)	normal 구간 상한값	`500`
Slow 상한(ms)	slow 구간 상한값	`2000`

기본값 기준 분류 결과는 다음과 같습니다.

응답 시간	분류
0 ~ 100ms	`fast`
101 ~ 500ms	`normal`
501 ~ 2000ms	`slow`
2001ms 이상	`critical`

활용 예시: 로그 탐색에서 `response_class: critical`로 필터링하면 응답 시간이 2초를 초과하는 로그만 조회할 수 있습니다.

JSON 필드 추출

1차 파서로 추출한 필드에 JSON 문자열이 포함된 경우 해당 JSON에서 특정 키를 추출하여 새 필드로 저장합니다.

설정 값	설명	기본값
입력 필드	JSON 문자열이 담긴 필드명	`payload`
추출할 키	추출할 JSON 키 목록 (쉼표로 구분해 입력)	required
저장 접두사	추출된 키 이름 앞에 붙일 접두사 (예: `json_`)	optional

- 로그 전체가 JSON인 경우 1차 JSON 파서가 자동으로 파싱합니다. 이 프로세서는 **GROK 1차 파서** 사용 시 텍스트 로그 안에 포함된 JSON 부분을 추가로 파싱할 때 유용합니다.
- 성능 보호를 위해 JSON 필드 크기가 10KB를 초과하면 파싱을 건너뜁니다.

활용 예시

1차 GROK 파싱 결과

```
payload = {"user":"alice","action":"purchase","product":"laptop","amount":45000}
```

입력 필드 `payload`, 추출할 키 `user, action`, 저장 접두사 `pay_` 설정 시 다음과 같이 추출됩니다.

JSON 필드 추출 결과

```
pay_user = alice
pay_action = purchase
```

GeoIP 변환

IP 주소를 국가 및 도시 정보로 변환합니다. IP2Location 데이터베이스를 활용해 접근 로그의 지역별 분석이 가능합니다.

설정 값	설명	기본값
입력 필드	IP 주소가 담긴 필드명	`client_ip`
저장 접두사	결과 필드 이름의 접두사	`geo`

결과로 `{접두사}_country`, `{접두사}_city` 두 개의 필드가 생성됩니다.

활용 예시: 입력 필드 `client_ip` 값이 `223.130.195.200`이고 저장 접두사가 `geo`인 경우:

GeoIP 변환 결과

```
geo_country = KR
geo_city = Seongnam
```

직접 입력

목록에 없는 프로세서를 FQN(Fully Qualified Name) 클래스명으로 직접 등록할 수 있습니다.

직접 입력 형식

```
io.whatap.logsink.processor.ClassName{"param1":"value1","param2":"value2"}
```

ⓘ 클래스명과 파라미터 형식이 정확해야 합니다. 잘못된 입력 시 프로세서가 동작하지 않을 수 있습니다.

2차 파서 체이닝

동일 카테고리에 여러 2차 파서를 등록해 체이닝할 수 있습니다. 각 프로세서는 등록 순서대로 실행됩니다.

원본 로그

```
2026-04-23T10:30:00.123+0900|INFO|api-01|192.168.1.10|/api/v1/
orders?detail=true|200|1234|{"user":"alice","action":"purchase"}|Request processed
```

순서	파서	입력 → 출력
1차	GROK	timestamp, level, host, client_ip, url, status, elapsed, payload, content
2차-1	JSON 필드 추출	payload → user, action
2차-2	응답 시간 분류	elapsed → response_class (`slow`)
2차-3	상태 코드 그룹 분류	status → status_nxx (`status_2xx`)
2차-4	URL 파라미터 제거	url → url_without_params (`/api/v1/orders`)
2차-5	GeoIP 변환	client_ip → geo_country (`KR`), geo_city (`Seoul`)

ⓘ 2차 파서의 **카테고리**는 1차 파서와 동일하게 설정해야 합니다. 다른 카테고리로 설정하면 해당 로그에 2차 파서가 적용되지 않습니다.

로그 파서를 사용하면 불규칙한 형태의 로그를 쿼리가 가능한 구조화된 형태로 변경할 수 있습니다. 와탭 로그 모니터링은 다음과 같이 두 가지 유형의 파서를 제공합니다.

- **GROK 파서**: 임의의 형태로 수집되는 로그를 정규 표현식과 GROK 문법을 활용해 파싱합니다.
- **JSON 파서**: JSON 형태로 수집되는 로그를 파싱합니다.

ⓘ **공통 주의사항**

- 같은 카테고리에 여러 개의 파서가 등록되어 있는 경우 첫 번째로 매칭되는 파서만 적용됩니다.

- **와탭은 와탭 서비스의 안정성에 영향을 줄 수 있는 파서를 비활성화할 수 있는 권한을 가집니다.**

GROK 파서

로그가 불규칙한 형태로 수집되는 경우 GROK 파서를 사용해 로그를 파싱할 수 있습니다. GROK 문법은 named regular expressions를 제공해 정규 표현식을 보다 쉽고 편리하게 사용할 수 있습니다.

GROK 파서 패턴 등록에 관해 다음 동영상 가이드를 참조하세요.

GROK 시작하기

GROK은 두 가지 형태의 문법을 제공합니다.

1. `%{SYNTAX:SEMANTIC}` : GROK 라이브러리에서 제공하는 문법입니다. **named regular expressions**를 활용해 태그를 추출할 수 있습니다. 활용 예시는 다음을 참조하세요.
   - **SYNTAX**: GROK이 제공하는 named regular expressions를 지정합니다.
   - **SEMANTIC**: 매칭되는 값에 부여할 이름을 지정합니다.

   named regular expressions

   GROK에서 제공하는 문법입니다. 복잡한 정규 표현식에 이름을 부여해 사용할 수 있도록 GROK에서 제공하는 기능입니다.

name	regular expression
WORD	`\b\w+\b`
SPACE	`\s*`
NOTSPACE	`\S+`
UUID	`[A-Fa-f0-9]{8}-(?:[A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}`

> ⓘ 와탭에서 제공하는 모든 named regular expressions 확인을 원한다면 다음 링크를 참조하세요.

2. `(?<SEMANTIC>REGX)` : 정규 표현식의 **named capturing group** 문법입니다. 정규 표현식을 활용해서 사용자의 의도에 맞게 태그를 추출할 수 있습니다. 활용 예시는 다음을 참조하세요.
   - **SEMANTIC**: 매칭되는 값에 부여할 이름을 지정합니다.
   - **REGX**: 매칭에 사용할 정규 표현식을 입력합니다.

> ⓘ **named capturing group**
>
> 정규 표현식에서 제공하는 문법입니다.
> - capturing group: 여러 개의 토큰을 하나로 묶어 하나의 매칭 단위로 사용하는 기능을 의미합니다.
> - named capturing group: capturing group에 이름을 부여한 것입니다.
> - 문자열 매칭 예시를 살펴보겠습니다. dev@whatap.io
>   - 예시 1 `(\w+)@(\w+\.\w+)`
>   - 예시 2 이메일 전체 매칭 및 username과 domain 추가 매칭 시 `(?<username>\w+)@(?<domain>\w+\.\w+)`

`%{SYNTAX:SEMANTIC}` 활용 예시

다음은 `%{SYNTAX:SEMANTIC}` 문법을 활용하는 예시입니다.

Sample log

```
[2023-08-08 02:02:30,101 GMT][INFO ][i.w.y.l.c.LogSinkDexScheduleThread.realProcess(159)] 8 VirtualLog 20230808 02:01:00.000 {area=4, city=5} 56ms
```

샘플 로그를 보고 각 단어가 의미하는 내용을 유추할 수 있습니다. 각 부분을 semantic한 단어로 치환 시 다음과 같이 표현할 수 있습니다.

semantic replace

```
[date][logLevel][caller] projectCode logCategory dexBuildStartTime {area=areaEnum, city=cityEnum} dexBuildElapsed
```

semantic한 단어 모두 정규 표현식으로 대체할 수 있습니다. GROK 파서를 사용하면 사전 정의된 named regular expressions를 활용할 수 있습니다. 여기서 사용된 `TIMESTAMP_ISO8601`, `LOGLEVEL`, `DATA` 는 GROK에서 제공하는 named regular expressions입니다. 이 값들은 각각 다음의 정규 표현식으로 대체되어 매칭됩니다.

- name: `TIMESTAMP_ISO8601`
  - regular expression: `%{YEAR}-%{MONTHNUM}-%{MONTHDAY}[T ]%{HOUR}:?%{MINUTE}(?::?%{SECOND})?%{ISO8601_TIMEZONE}?`
- name: `LOGLEVEL`
  - regular expression: `LOGLEVEL ([Aa]lert|ALERT|[Tt]race|TRACE|[Dd]ebug|DEBUG|[Nn]otice|NOTICE|[Ii]nfo|INFO|[Ww]arn?(?:ing)?|WARN?(?:ING)?`
- name: `DATA`
  - regular expression: `.*?`

GROK parsing pattern

```
\[%{TIMESTAMP_ISO8601:date}\sGMT\]\[%{LOGLEVEL:level}\s\]\[%{DATA:caller}\]
```

위와 같은 문법으로 파싱을 하면 다음과 같이 태그를 추출할 수 있습니다. 이렇게 GROK의 `%{SYNTAX:SEMANTIC}` 문법은 복잡하고 긴 정규 표현식을 쉽고 간결하게 적용할 수 있도록 도와주는 역할을 합니다.

Tag extraction

```
- date : 2023-08-08 02:02:30,101
- caller : i.w.y.l.c.LogSinkDexScheduleThread.realProcess(159)
- level : LEVEL
```

`(?<SEMANTIC>REGX)` 활용 예시

named regular expressions로 매칭되지 않는 부분은 `(?<SEMANTIC>REGX)` 패턴을 사용해서 파싱할 수 있습니다. 위의 샘플 로그에서 `%{SYNTAX:SEMANTIC}` 문법만으로 파싱되지 않는 영역은 다음과 같습니다.

Unparsed area

```
8 VirtualLog 20230808 02:01:00.000 {area=4, city=5} 56ms
```

해당 로그의 각 부분을 semantic한 단어로 치환 시 다음과 같이 표현할 수 있습니다.

semantic replace

```
projectCode logCategory dexBuildStartTime {area=areaEnum, city=cityEnum} dexBuildElapsed
```

이렇게 불규칙한 형태의 문자열은 다음과 같은 `(?<SEMANTIC>REGX)` 문법을 사용해 파싱할 수 있습니다.

샘플 로그 파싱 키워드별 매칭되는 정규 표현식

파싱 키워드	(?<SEMANTIC>REGX)
8	(?<projectCode>\d)
VirtualLog	(?<logCategory>\w*)
20230808 02:01:00.000	(?<dexBuildStartTime>\d{8}\s\d{2}:\d{2}:\d{2}\.\d{3})
area=4	area=(?<areaEnum>\d)
city=5	city=(?<cityEnum>\d)
56ms	(?<dexBuildElapsed>\d{2})ms

기본 정규 표현식 문법

문법	의미	별칭
?	0 or 1	-
+	1 or more	-
*	0 or more	-
a{5}	exactly 5	-
\w	word character	[a-zA-Z_0-9]

문법	의미	별칭
\s	white space	-
.	any character except newline	-
[abc]	any of	-
[^abc]	not a,b, or c	-
[a-z]	character between a and z	-
[1-3[7-9]]	union (combining two or more character classes)	-
[1-6&&[3-9]]	intersection (교집합)	-
[0-9&&[^2468]]	subtraction (차집합)	-
a{2,}	2 or more	-
a{1,3}	between 1 and 3	-
a+?	match as few as possible	-
{2,3}?	match as few as possible	-
(abc)	capturing group (여러 개의 문자열을 single unit으로 처리함)	-
\d	digit	[0-9]
\D	non-digit	[^0-9]
\W	non-word character	-
\S	non-white space	-

이렇게 파싱된 키워드를 띄어쓰기(`\s`)와 특수 문자 escape(`\{` , `\,` , `\}`)로 연결하면 다음과 같이 패턴을 적용할 수 있습니다.

GROK parsing pattern

```
(?<projectCode>\d)\s(?<logCategory>\w*)\s(?<dexBuildStartTime>\d{8}\s\
d{2}:\d{2}:\d{2}\.\d{3})\s\{area=(?<areaEnum>\d),\scity=(?<cityEnum>\d)\}\s(?<dexBuildElapsed>\d{2})ms
```

위와 같은 문법으로 파싱을 하면 다음과 같이 태그를 추출할 수 있습니다.

Tag extraction

```
- projectCode : 8
- logCategory : VirtualLog
- dexBuildStartTime : 20230808 02:01:00.000
- areaEnum : 4
- cityEnum : 5
- dexBuildElapsed : 56
```

GROK 적용하기

로그 설정 > 로그 1차 파서 설정

1. GROK 패턴 파서를 적용하려면 **로그 설정** 메뉴의 **로그 1차 파서 설정** 탭으로 이동하세요.
2. **+ 추가하기**를 선택 후 **파서** 입력란에서 **GROK** 파서를 선택하세요.
3. **패턴 등록** 버튼 선택 시 오른쪽에 패턴 등록 및 시뮬레이션 창이 나타납니다.
4. **패턴**과 **로그** 입력 후 **시뮬레이션** 버튼을 클릭하여 적용하려는 패턴으로 파싱에 성공하는지 확인하세요.

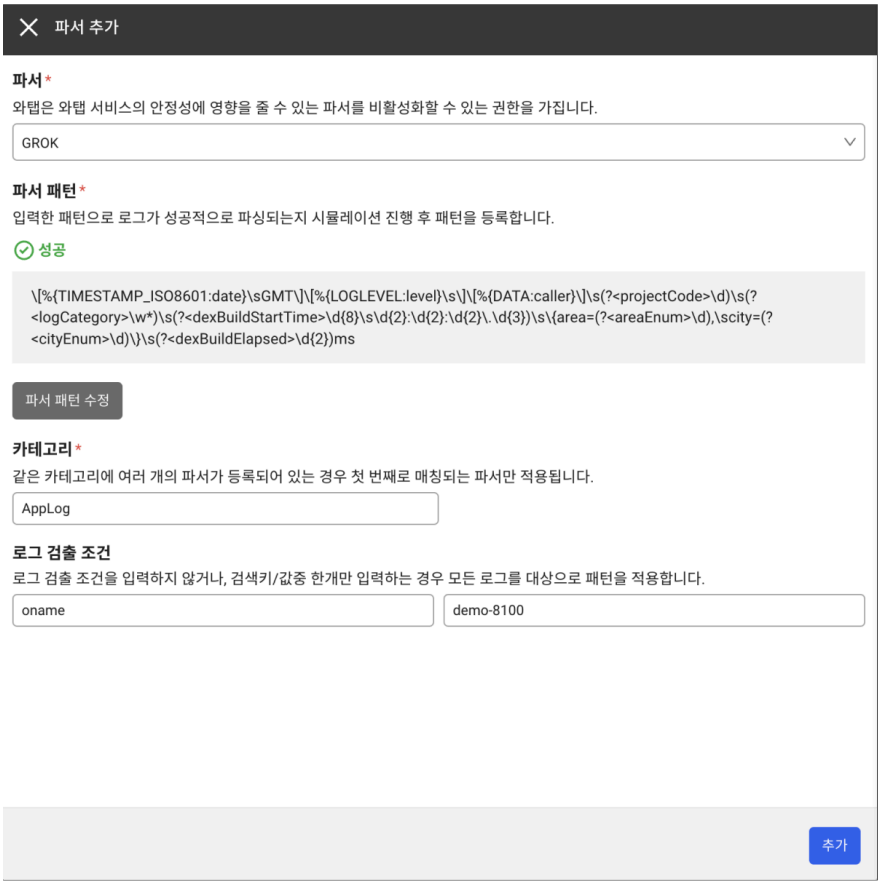
   패턴 예시:

   ```
   [%{TIMESTAMP_ISO8601:date}\sGMT\][%{LOGLEVEL:level}\s\][%{DATA:caller}\]\s(?<projectCode>\d)\s(?<logCateg
   d{2}:\d{2}:\d{2}\.\d{3})\s\{area=(?<areaEnum>\d),\scity=(?<cityEnum>\d)\}\s(?<dexBuildElapsed>\d{2})ms
   ```

   로그 예시:

   ```
   [2023-08-08 02:02:30,101 GMT][INFO ][i.w.y.l.c.LogSinkDexScheduleThread.realProcess(159)] 8
   VirtualLog 20230808 02:01:00.000 {area=4, city=5} 56ms
   ```

5. 시뮬레이션 성공 시 **시뮬레이션 결과** 및 **퍼포먼스 측정 결과**를 조회할 수 있습니다.
6. 시뮬레이션 후 **패턴 적용** 버튼 클릭 시 선택한 파서에 입력한 패턴이 적용됩니다.
7. 패턴 적용 후 **카테고리** 및 **로그 검출 조건**, **패턴**을 입력하세요.



- **카테고리**

  로그 카테고리를 선택하세요. **카테고리**는 필수로 입력해야 합니다.

- **로그 검출 조건**
  - 조건에 만족하는 로그만 파서가 적용됩니다.
  - **검색 키**와 **검색 값**을 선택하거나 직접 입력하세요.
  - **로그 검출 조건**은 모든 파서가 수행되기 전에 적용됩니다. 즉 파서의 결과로 추가되는 **태그**를 사용할 수 없습니다.

8. **추가** 버튼을 선택해 파서를 등록하세요.

- 로그 파서 목록에서 해당 파서의 **적용 순서**를 변경하거나 **활성화** 및 **수정**, **삭제**할 수 있습니다.
- **파서 시뮬레이션** 후 패턴을 등록할 수 있습니다.

GROK 파서 주의사항

- GROK 파서는 `%{SYNTAX:SEMANTIC}` , `%{SYNTAX:SEMANTIC}` 두 가지 패턴을 지원합니다.
- `%{SYNTAX:SEMANTIC}` 패턴 사용 시 `SEMANTIC` 을 반드시 입력해야합니다.
- `%{SYNTAX:SEMANTIC}` 패턴 사용 시 `SEMANTIC` 은 하나의 파서 안에서 unique 해야합니다.
- `(?<SEMANTIC>REGX)` 패턴 사용 시 `SEMANTIC` 은 문자(a-z, A-Z)와 숫자(0-9) 그리고 지정된 특수문자(`.` , `_` , `-`)만 올 수 있습니다.
- `SEMANTIC` 은 문자(a-z, A-Z)로 시작해야 합니다.
- `SEMANTIC` 은 문자(a-z, A-Z) 또는 숫자(0-9)로 끝나야 합니다.

GROK 추가 패턴 선택하기

GROK 파서 시뮬레이션 시 기본 패턴(`WORD` , `NUMBER` , `IP` 등) 외에 특정 로그 형식에 맞는 **추가 패턴**을 선택할 수 있습니다. Java, Nginx, MySQL 등 32개 패턴 타입이 11개 카테고리로 그룹핑되어 제공됩니다.

추가 패턴 카테고리 드롭다운

1. 파서 시뮬레이션 창에서 **추가 패턴 (선택)** 영역의 드롭다운을 클릭하세요.
2. 카테고리별로 그룹핑된 패턴 타입 목록에서 필요한 패턴 타입을 선택하세요. 검색 입력란에 패턴 타입 이름 또는 설명을 입력해 검색할 수도 있습니다.
3. 선택한 패턴 타입은 시뮬레이션 시 기본 패턴과 함께 로드됩니다.

기본 패턴(Core)은 항상 자동으로 포함됩니다. 추가 패턴 선택은 선택 사항입니다.

카테고리별 패턴 타입 목록

카테고리	패턴 타입
Web Servers	haproxy, httpd, iis, nginx, squid
Application Servers & Frameworks	java, log4j, rails, ruby, springboot

카테고리	패턴 타입
Databases	mongodb, mysql, oracle, postgresql, redis
Cloud	aws
System & Infrastructure	docker, kubernetes, linux-syslog
Messaging & Search	elasticsearch, kafka
Network & Security	bind, bro, firewalls, junos, zeek
Mail	exim, postfix
Monitoring & Backup	bacula, mcollective, nagios
Build Tools	maven

활용 예시

Tomcat Catalina 8 로그를 파싱하려면 **java** 패턴 타입을 추가로 선택한 뒤 `%{CATALINA8_LOG}` 패턴을 사용하세요.

Sample log

```
31-Jul-2020 16:40:38 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version name: Apache Tomcat/8.5.57
```

GROK parsing pattern

```
%{CATALINA8_LOG}
```

Tag extraction

```
- log_timestamp : 31-Jul-2020 16:40:38
- log.level : INFO
- java.log.origin.thread.name : main
- java.log.origin.class.name : org.apache.catalina.startup.VersionLoggerListener
- log.origin.function : log
```

```
- log_message : Server version name: Apache Tomcat/8.5.57
```

패턴 상세 보기

선택한 추가 패턴 타입에 어떤 패턴이 포함되어 있는지 상세 창에서 확인할 수 있습니다. 패턴 정의(정규 표현식)와 샘플 로그를 조회하고 시뮬레이션에 바로 적용할 수 있습니다.

패턴 상세 보기

추가 패턴 타입을 선택한 뒤 선택된 패턴 타입 태그를 클릭하면 오른쪽에 상세 창이 열립니다. 상세 창은 시뮬레이션 폼을 가리지 않고 나란히 표시되므로, 패턴 정의를 확인하면서 시뮬레이션을 진행할 수 있습니다.

각 패턴 항목에서 다음 기능을 사용할 수 있습니다.

버튼	기능	설명
샘플	샘플 로그 입력	해당 패턴에 매칭되는 예시 로그를 **시뮬레이션 파싱 로그** 입력란에 자동으로 입력합니다.
선택	패턴 삽입	`%{패턴이름}` 형식으로 **패턴** 입력란에 삽입합니다. 해당 패턴 타입도 자동으로 선택됩니다.
복사	클립보드 복사	`%{패턴이름}` 형식을 클립보드에 복사합니다.

- 패턴 항목을 클릭하면 패턴 정의(정규 표현식)가 펼쳐집니다. 여러 패턴을 동시에 펼쳐 정의를 비교할 수 있습니다.
- 패턴 정의에 포함된 `%{PATTERN:field}` 참조는 패턴 이름, 필드명, 리터럴 문자가 서로 다른 색상으로 구분되어 표시됩니다.
- **샘플** 버튼으로 입력된 로그로 바로 **시뮬레이션**을 실행해 파싱 결과를 확인할 수 있습니다.

시뮬레이션 샘플 로그 저장

시뮬레이션에 사용한 로그는 파서 설정과 함께 자동으로 저장됩니다. 파서를 수정할 때 이전에 사용한 샘플 로그가 시뮬레이션 창에 자동으로 복원되어 매번 로그를 다시 입력할 필요가 없습니다.

JSON 파서

로그가 JSON 포맷으로 수집될 경우 JSON 파서를 사용해 쉽고 편리하게 파싱할 수 있습니다.

JSON 적용하기

로그 설정 > 로그 1차 파서 설정

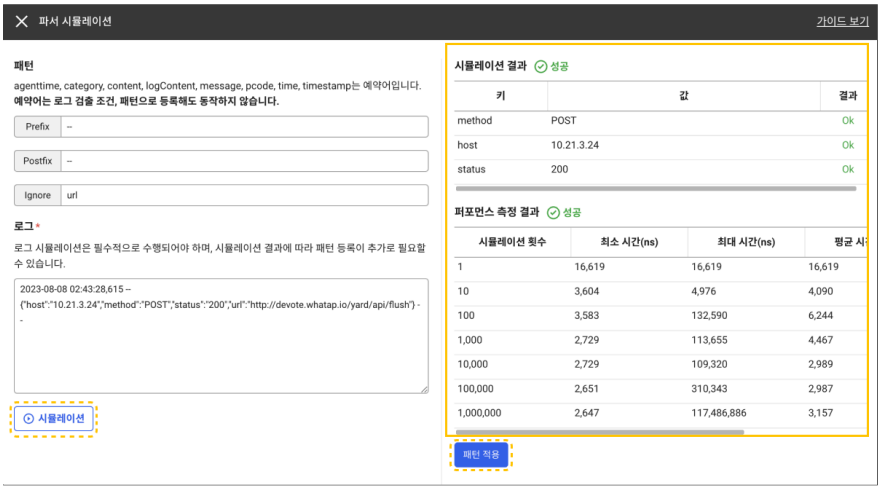
1. JSON 패턴 파서를 적용하려면 **로그 설정** 메뉴의 **로그 1차 파서 설정** 탭으로 이동하세요.
2. **+ 추가하기**를 선택 후 **파서** 입력란에서 **JSON** 파서를 선택하세요.
3. **패턴 등록** 버튼 선택 시 오른쪽에 패턴 등록 및 시뮬레이션 창이 나타납니다.
4. **패턴**과 **로그** 입력 후 **시뮬레이션** 버튼을 클릭하여 적용하려는 패턴으로 파싱에 성공하는지 확인하세요.

   Example

   ```
   2023-08-08 02:43:28,615 -- {"host":"10.21.3.24","method":"POST","status":"200","url":"http://dev.whatap.io/yard/api/flush"} --
   ```

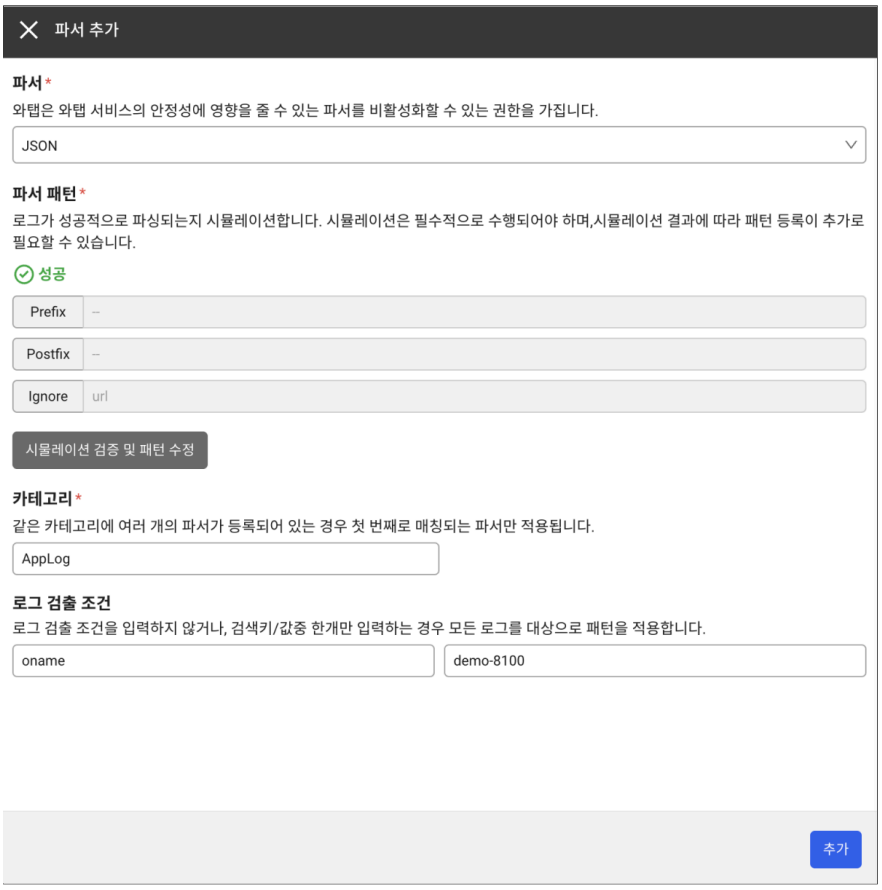
   예시 로그에서 **Prefix**와 **Postfix**를 `--` 로 지정하고 **Ignore**에 `url` 을 지정 시 `host` , `method` , `status` 3개의 **태그**만 생성됩니다.

   - **Prefix**
     로그에서 JSON 포맷이 시작하는 위치를 지정합니다. 로그 전체가 JSON 포맷인 경우 빈 값으로 설정합니다.
   - **Postfix**
     로그에서 JSON 포맷이 끝나는 위치를 지정합니다. 로그 전체가 JSON 포맷인 경우 빈 값으로 설정합니다.
   - **Ignore**
     JSON 포맷 중 태그를 생성하지 않을 키를 지정합니다.
5. 시뮬레이션 성공 시 **시뮬레이션 결과** 및 **퍼포먼스 측정 결과**를 조회할 수 있습니다.



6. 시뮬레이션 후 **패턴 적용** 버튼 클릭 시 선택한 파서에 입력한 패턴이 적용됩니다.

7. 패턴 적용 후 **카테고리** 및 **로그 검출 조건**, **패턴**을 입력하세요.



- **카테고리**

  로그 카테고리를 선택하세요. **카테고리**는 필수로 입력해야 합니다.

- **로그 검출 조건**
  - 조건에 만족하는 로그만 파서가 적용됩니다.
  - **검색 키**와 **검색 값**을 선택하거나 직접 입력하세요.
  - **로그 검출 조건**은 모든 파서가 수행되기 전에 적용됩니다. 즉 파서의 결과로 추가되는 **태그**를 사용할 수 없습니다.

8. **추가** 버튼을 선택해 파서를 등록하세요.

> - 로그 파서 목록에서 해당 파서의 **적용 순서**를 변경하거나 **활성화** 및 **수정**, **삭제**할 수 있습니다.
> - 파서를 등록하기 전에 **시뮬레이션**을 통해 등록하려는 패턴이 정상적인지 확인할 수 있습니다. GROK 파서 등록 시뮬레이션 과정과 동일합니다. 다음 문서를 참조하세요.

JSON 포맷 활용 예시

Sample log

```
{"host":"10.21.3.24","method":"POST","status":"200","url":"http://dev.whatap.io/yard/api/flush"}
```

위와 같은 샘플 로그가 수집된 경우 **파서 추가** 창에서 **JSON** 파서를 선택하세요. 복잡한 파싱 로직을 작성할 필요없이 로그 분석에 필요한 **태그**를 다음과 같이 추출할 수 있습니다.

Tag extraction

```
- host : 10.21.3.24
- method : POST
- status : 200
- url : http://dev.whatap.io/yard/api/flush
```

JSON 포맷 일부 구성 시 활용 예시

Some JSON format sample log

```
2023-08-08 02:43:28,615 -- {"host":"10.21.3.24","method":"POST","status":"200","url":"http://dev.whatap.io/yard/api/flush"} --
```

만약 예시와 같이 로그의 일부만 JSON 포맷으로 구성되어있다면 **Prefix**와 **Postfix**를 지정해 주세요. 와탭 로그 모니터링은 **Prefix**와 **Postfix** 사이의 영역을 JSON 포맷으로 인식 후 파싱합니다.

Tag extraction

```
- host : 10.21.3.24
- method : POST
- status : 200
- url : http://dev.whatap.io/yard/api/flush
```

로그 빠른 인덱스 설정

빠른 인덱스 설정

로그 설정 메뉴 상단에서 **빠른 인덱스 설정** 탭을 선택하세요. 대량의 로그를 수집할 경우 로그 검색 성능이 현저하게 저하될 수 있습니다. 자주 사용하는 검색 조건을 **인덱스**(index)로 생성하면 로그 검색 성능을 개선해 빠른 탐색이 가능합니다. 설정 항목은 다음과 같습니다.

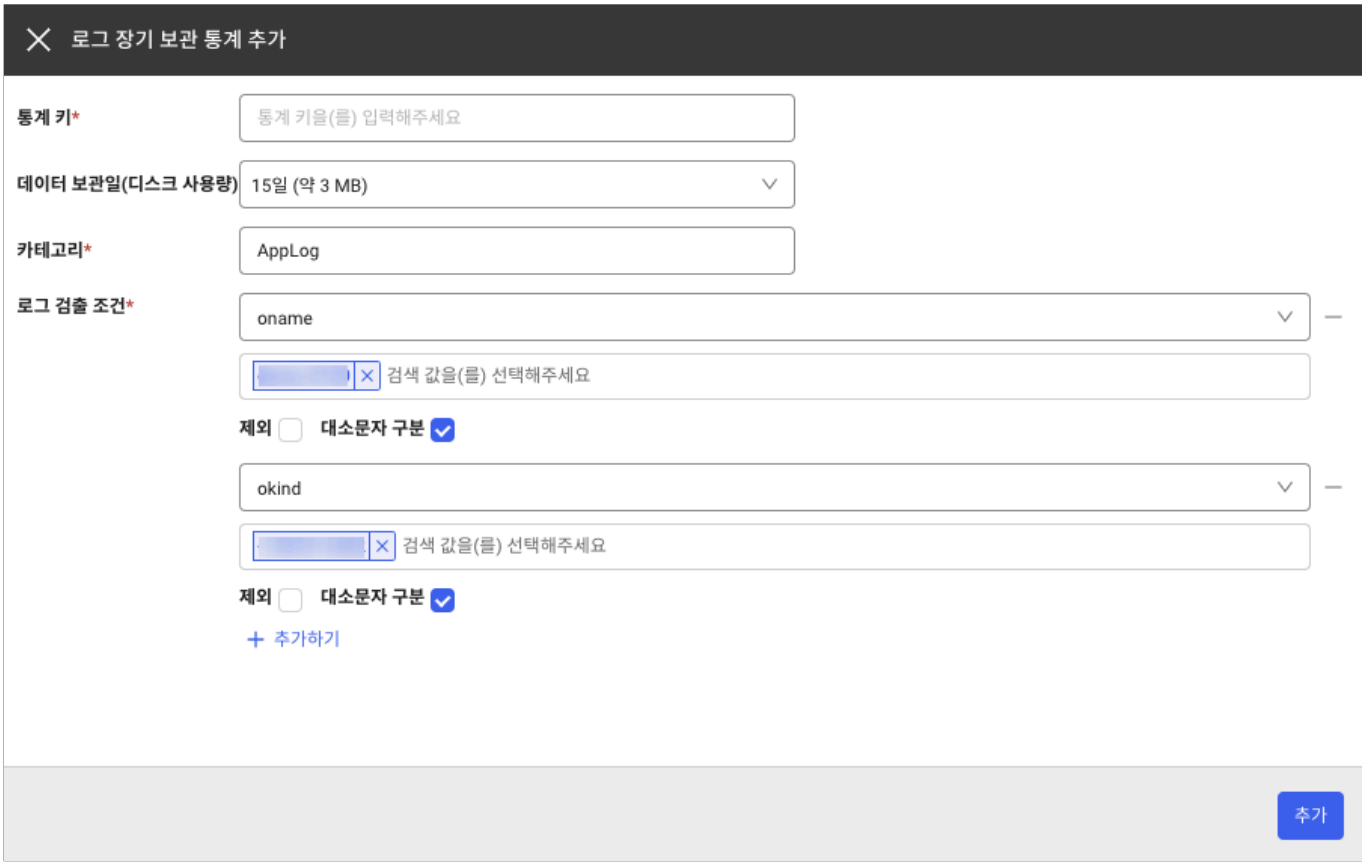
설정 값	필수 여부	설명
카테고리	필수	빠른 인덱스를 설정할 카테고리
검색 키	필수	빠른 인덱스를 설정할 검색 키
대소문자 구분 안 함	옵션	대소문자 구분 여부
규칙	필수	`*` 한 개 이상 포함 필수
활성	필수	활성 또는 비활성 여부(기본값 `true`)

로그 장기 보관 통계 설정

로그 장기 보관 통계 설정

로그 설정 메뉴 상단에서 **로그 장기 보관 통계** 탭을 선택하세요. 로그 데이터는 용량이 매우 커서 장기간 보관하기가 어렵습니다. 따라서 로그 통계 데이터 설정 기능을 사용하여 **특정 조건을 만족하는 로그 데이터가 5분마다 몇 건씩 수집되었는지에 대한 정보를 저장**할 수 있습니다. 장기간 시 실제 로그 데이터는 삭제되어도 해당 조건을 만족하는 로그가 얼마나 수집되었는지 추이를 확인할 수 있습니다.

로그 장기 보관 통계 추가



로그 장기 보관 통계 탭에서 **+ 추가하기** 버튼을 선택하면 **로그 장기 보관 통계 추가** 창이 나타납니다. **+ 추가하기** 버튼을 통해

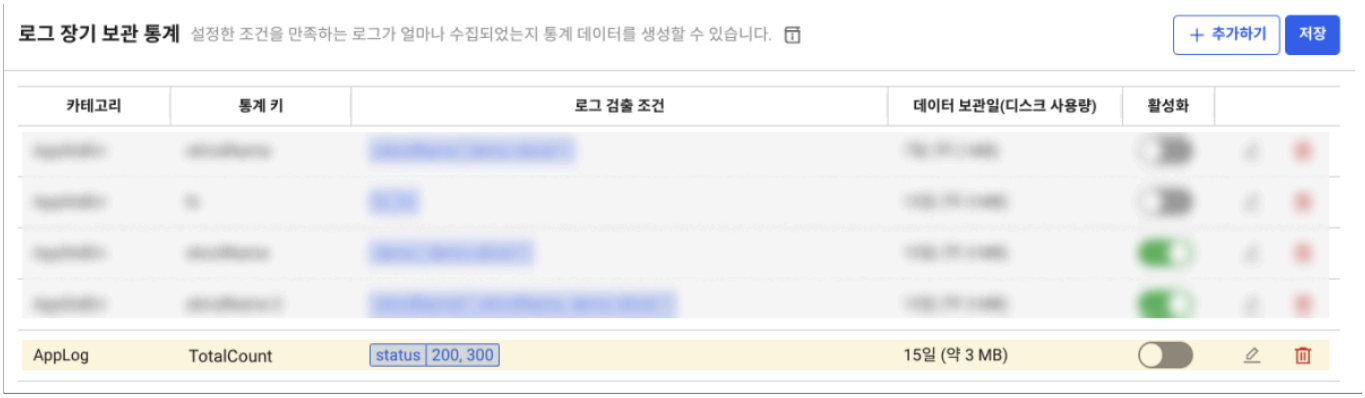
규칙을 추가하거나 생성한 규칙을 - 아이콘을 통해 삭제할 수 있습니다.

설정 항목

필드	설명
카테고리	규칙을 적용할 카테고리입니다.
통계 키(key)	규칙을 만족하는 로그 발생 시 저장할 키 값으로 동일한 키를 중복으로 설정할 수 없습니다.
로그 검출 조건	로그 통계 데이터를 생성할 조건입니다. 이 조건을 만족하는 로그가 얼마나 수집되었는지를 기반으로 통계 데이터를 생성합니다.
제외	제외를 체크하면 입력한 조건에 해당하지 않는 값으로 통계 데이터를 생성합니다.
대소문자 구분	입력한 로그 검출 조건의 값에 대해서 대소문자 구분 여부를 지정합니다.
활성	활성 또는 비활성 여부(기본값 `true`)

예시

다음과 같이 설정을 추가한 경우 수집된 로그 중 status가 `200` , `300` 인 로그를 대상으로 **TotalCount**라는 키값으로 통계 데이터를 생성합니다.



데이터 조회

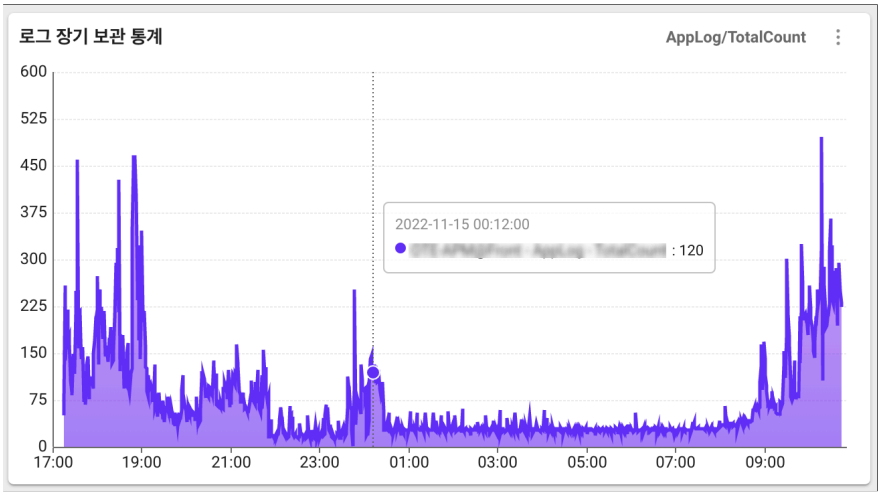
1. **통합 Flex 보드**의 **위젯 템플릿**에서 **로그 장기 보관 통계**를 검색해 위젯을 생성하세요.



2. 데이터를 조회할 **카테고리**와 **통계 키**를 지정한 뒤 **적용** 버튼을 선택합니다.



3. 추가한 설정값으로 **로그 장기 보관 통계** 데이터를 다음처럼 확인할 수 있습니다.


로그 장기 보관 통계
AppLog/TotalCount
600
525
450
375
300
225
150
75
0
17:00
19:00
21:00
23:00
01:00
03:00
05:00
07:00
09:00
2022-11-15 00:12:00
: 120

로그 개인정보 비식별화 설정

개인정보 비식별화

개인정보 비식별화에서 로그 데이터 내 개인정보를 마스킹 처리하거나 안전한 값으로 치환할 수 있습니다.

- 암호화 저장: 지정된 민감 정보는 암호화되어 데이터베이스에 저장됩니다.
- 마스킹 처리: 로그 화면에 표시될 때 민감 정보는 기본적으로 마스킹(`**`) 처리되어 표시됩니다.
- 치환 처리: 민감한 정보를 사용자가 지정한 안전한 값으로 변경합니다.

개인정보 분류

일반적으로 개인정보로 분류하는 항목은 다음과 같습니다. 일반 개인정보 외에도 민감정보로 처리해야 하는 항목이 있다면 **개인정보 비식별화** 기능을 통해 로그 내에서 원하는 항목을 마스킹 처리 또는 치환 처리하세요.

- 주민등록번호
- 여권번호
- 운전면허번호
- 외국인 등록번호
- 신용카드 번호
- 계좌번호
- 생체인식 정보

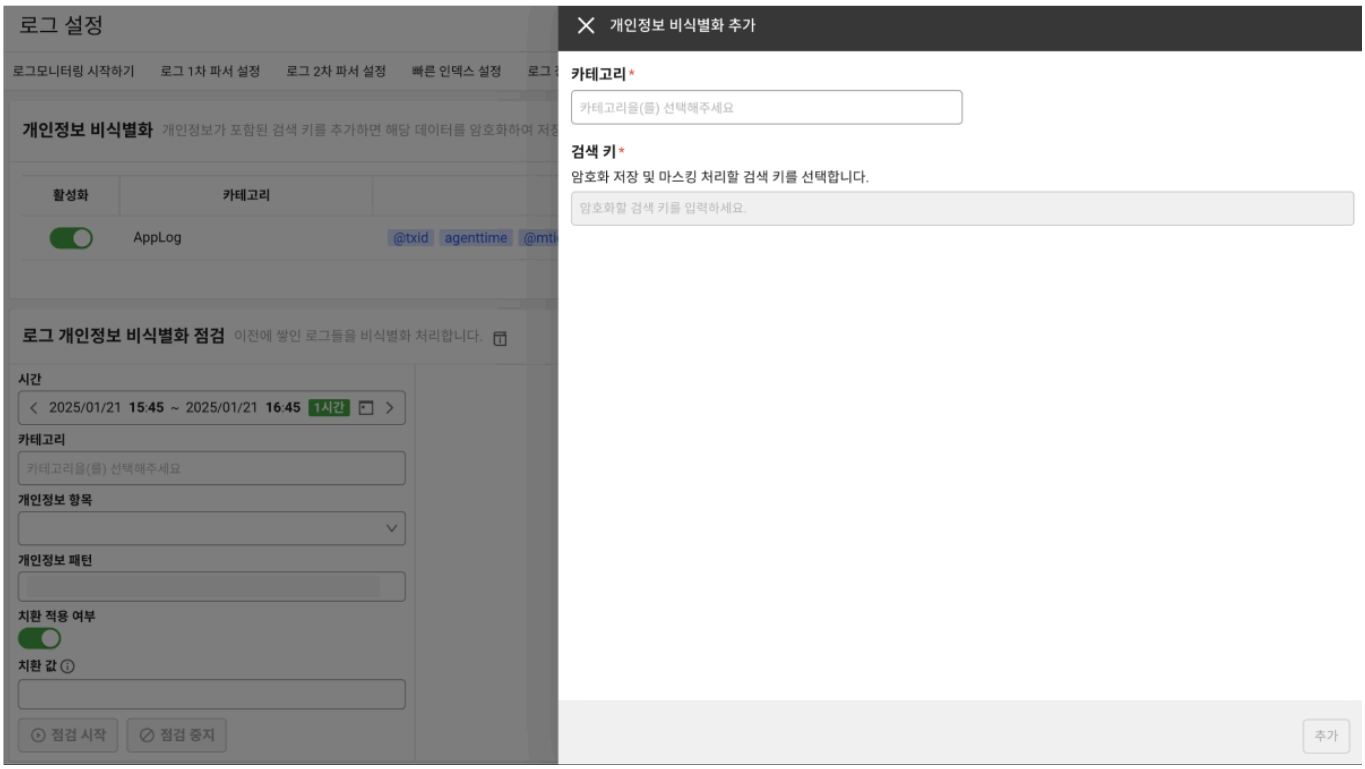
요구 권한

비식별화된 개인정보는 마스킹 처리되어 표시됩니다. **로그 개인정보 조회** 권한이 있는 사용자만 **개인정보 표시** 토글을 통해 원본 데이터를 확인할 수 있습니다. 비식별화 대상 검색 키의 추가 및 수정은 **로그 편집** 권한이 필요합니다.

> ⓘ **로그 개인정보 조회** 권한 및 **로그 편집** 권한에 대한 자세한 내용은 다음 문서를 확인하세요.

개인정보 비식별화 설정

신규 로그 데이터에 마스킹 처리를 적용하여 민감 정보를 보호할 수 있습니다.



1. **개인정보 비식별화** 탭에서 **추가하기** 버튼을 선택하세요.
2. 오른쪽 **개인정보 비식별화 추가** 창에서 **카테고리**를 선택하고, 마스킹 처리할 **검색 키**를 입력하세요.
3. **추가** 버튼을 클릭하여 설정을 추가하세요.
4. **개인정보 비식별화** 목록에서 비식별화 항목 확인 후 오른쪽 상단 **저장** 버튼을 눌러 현재 설정을 저장하세요.



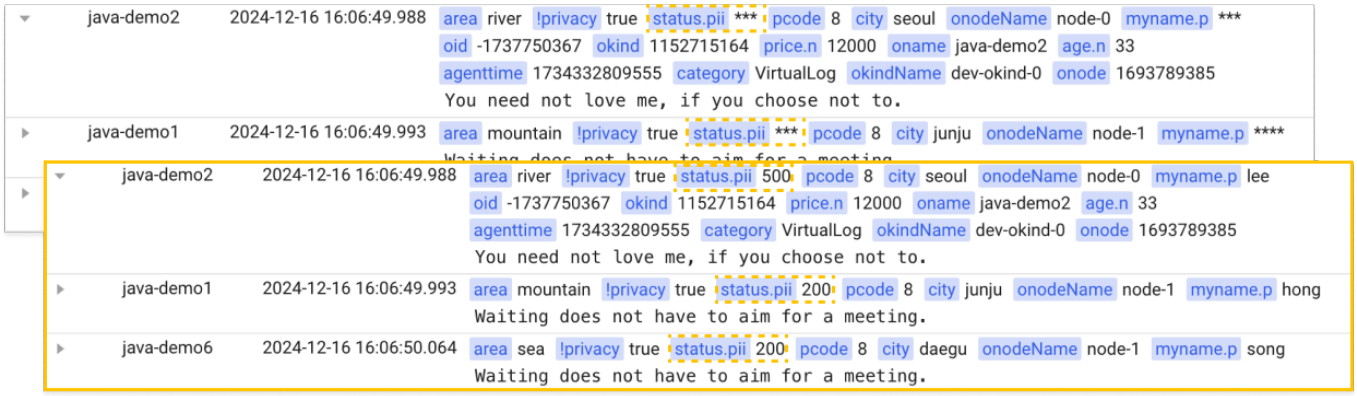
- 비식별화 처리된 검색 키의 경우 로그 콘텐츠 내 `검색 키.pii` 형식으로 출력됩니다.
- **개인정보 비식별화** 목록 내 **활성화** 컬럼에서 비식별화 항목을 **활성화**할 수 있습니다.

- **개인정보 비식별화** 목록 내 **편집** 컬럼에서 **수정** 및 **제거**할 수 있습니다.

> **로그 1차 파서 설정** 탭에서 검색 키 이름을 `검색 키.p` 로 지정하면 비식별화 대상으로 자동 등록됩니다. 자동 등록된 항목은 **개인정보 비식별화** 탭에서 **카테고리** 선택 후 **지정된 검색 키**에서 확인하거나 해제할 수 있습니다.
>
> 파서에서 `.p` 접미사로 지정하는 방식보다 **개인정보 비식별화** 탭에서 직접 설정하는 것을 권장합니다.

개인정보 표시

비식별화된 데이터는 **라이브 테일**, **로그 트렌드**, **로그 검색** 메뉴에서 마스킹 상태로 표시됩니다. **로그 개인정보 조회** 권한이 있는 사용자는 로그 목록의 **개인정보 표시** 버튼을 클릭해 원본 데이터를 확인할 수 있습니다.



> **개인정보 비식별화** 탭에서 지정한 검색 키는 로그 메뉴 내 로그 콘텐츠에서 `검색 키.pii` 형식으로 나타납니다.
>
> 예시, `status.pii` `**`

로그 개인정보 비식별화 점검

> **로그 개인정보 비식별화 점검** 기능을 통해 기존 데이터를 치환할 경우 해당 데이터는 원복이 불가능합니다.

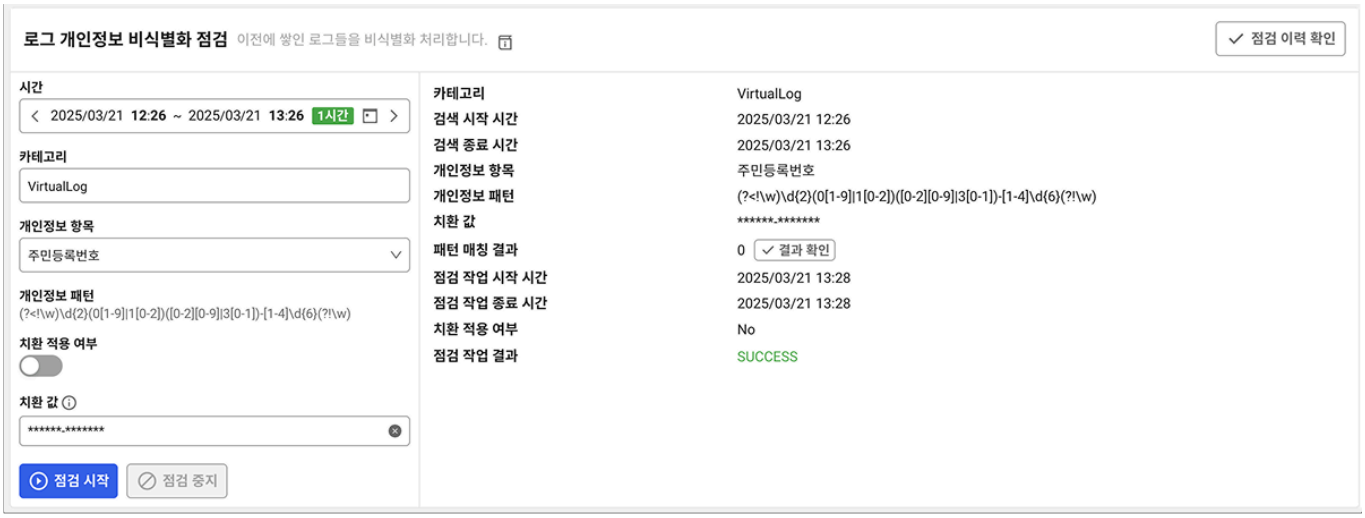
로그 개인정보 비식별화 점검는 기존 로그 데이터 내 개인정보를 식별하여 사용자가 지정한 **치환 값**으로 대체합니다.

대상 키 지정 치환

대상 키를 지정하면 해당 키에 저장된 값만 치환하고 나머지 로그 내용은 그대로 유지합니다. 대상 키는 로그의 `field` 값과 `tag` 값 모두에 적용됩니다. 대상 키를 지정하지 않으면 기존과 동일하게 로그 콘텐츠 전체에서 개인정보를 식별해 치환합니다.

> ⓘ 치환은 저장된 로그 데이터에만 적용됩니다. 검색 인덱스에는 치환 이후에도 원본 값이 남아 있을 수 있습니다.

대상 키는 로그 개인정보 비식별화 점검 폼의 **대상 키** 입력란에 지정하며, 화면에서는 `field` 와 `tag` 를 구분하지 않고 하나의 대상 키로 입력합니다.



1. 시작 시간 및 종료 시간을 설정하세요.
2. **카테고리**를 설정하세요.
3. **개인정보 항목**을 예시 중 선택 또는 **직접 입력**을 선택하세요.

   > ⓘ **직접 입력** 선택 시 개인정보를 식별하기 위한 정규표현식 패턴을 직접 입력합니다.

4. **치환 적용 여부** 토글을 통해 개인정보가 식별된 경우 이를 치환할지 여부를 선택하세요.
5. 기존 로그에서 정규표현식 패턴으로 식별된 문자열을 변경할 **치환 값**을 입력하세요.

> ⓘ
> - **치환 값**은 원본 값보다 길어서는 안됩니다.
> - **치환 적용 여부** 토글 활성화 시 반드시 **치환 값**을 입력해야 합니다.

6. 조건 입력 후 **점검 시작** 버튼을 선택하세요.

> ⓘ 점검 중인 경우 언제든지 **점검 중지** 버튼을 통해 점검 과정을 중지할 수 있습니다.

7. 치환 적용 시 **개인정보 비식별화 점검 시작** 확인 창에서 **치환하기** 입력 후 **확인** 버튼을 클릭하세요.



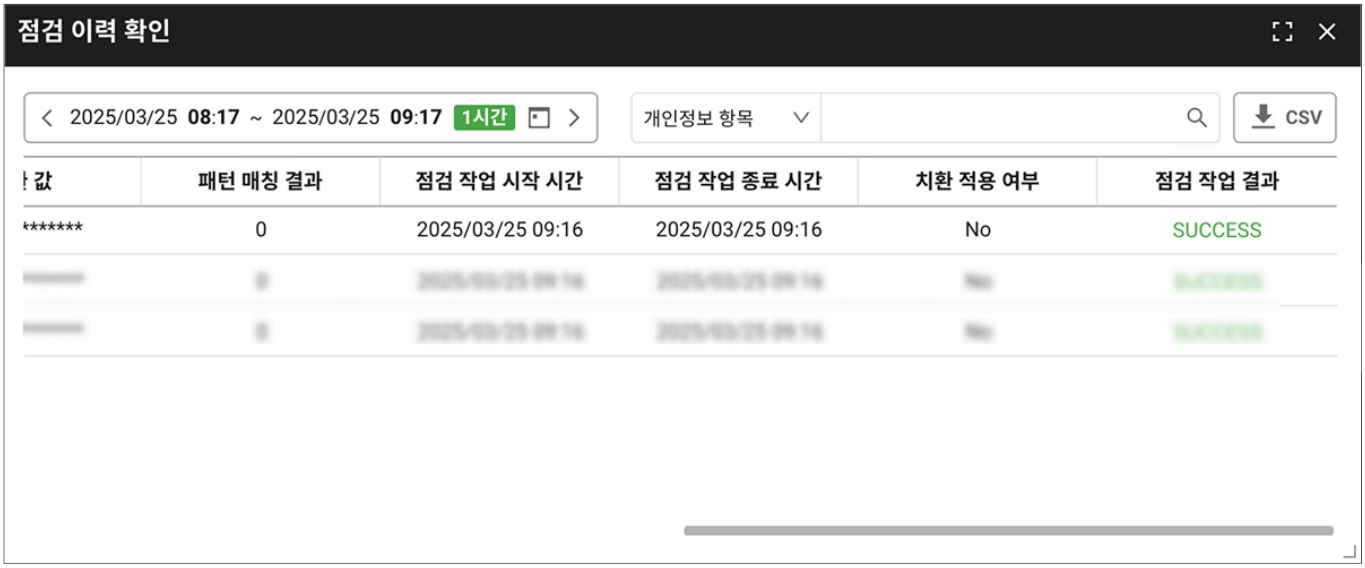
8. 점검 작업 완료 후 예시 이미지와 같이 오른쪽에서 **점검 작업 결과**를 확인할 수 있습니다.
   - **카테고리**: 점검된 카테고리 정보를 표시합니다.
   - **검색 시작 시간** 및 **검색 종료 시간**: 점검이 실행된 시간 범위를 표시합니다.
   - **개인정보 항목**: 점검된 개인정보 항목을 표시합니다.
   - **개인정보 패턴**: 사용된 정규표현식 패턴을 표시합니다.
   - **치환 값**: 적용된 치환 값을 표시합니다.
   - **패턴 매칭 결과**: 식별된 개인정보 개수를 표시합니다. **결과 확인** 클릭 시 샘플 로그를 확인할 수 있습니다.
   - **점검 작업 결과** : 개인정보가 식별된 결과를 표시합니다.

     `SUCCESS` (점검 성공), `FAILED` (점검 실패), `STOP` (점검 중지), `TIMED_OUT` (시간 초과로 점검 종료)

점검 이력 확인

로그 개인정보 비식별화 점검을 진행한 후 과거에 수행된 점검 결과를 조회하려면 상단 오른쪽에서 **점검 이력 확인** 버튼을

클릭하세요. 치환 내역, 점검 시점, 적용된 개인정보 항목 등을 추적할 수 있습니다. 또한 **점검 이력 확인** 창 상단 오른쪽에서 **CSV** 버튼 클릭 시 CSV 파일 형식으로 해당 목록 정보를 다운로드할 수 있습니다.



개인정보 패턴

기존 로그 데이터에서 개인정보를 식별할 때 사용되는 정규표현식 패턴 예시는 다음과 같습니다.

개인정보 패턴	개인정보 예시	정규식 패턴	치환 값
주민등록번호	123456-1234567	(?<!w)d{2}(0[1-9]|1[0-2])([0-2][0-9]|3[0-1])-[1-4]d{6}(?!w)	`****_*****`
여권번호	M1234567	([MSRODTC][0-9]{7}\b)	`******`
운전면허번호	01-23-456789-00	(?<!w)d{2}-d{2}-d{6}-d{2}(?!w)	`**_**_****_**`
외국인등록번호	987654-1234567	(?<!w)d{6}-?[5-8]d{6}(?!w)	`****_*****`
신용카드번호	1234-5678-9012-3456	(?<!w)d{4}-d{4}-d{4}-d{4}(?!w)	`***_***_***_***`

개인정보 패턴	개인정보 예시	정규식 패턴	치환 값
전화번호	010-1234-5678	(?<!w)(+82-?1[016789]-?d{3,4}-?d{4}|01[016789]-?d{3,4}-?d{4})(?!w)	`**-***-***`
이메일 주소	`example@domain.com`	([a-zA-Z0-9][a-zA-Z0-9._%+-]*@[a-zA-Z0-9.-]+.[a-zA-Z]{2,})	`**@**.**`
계좌번호	123456-01-654321	(?<!w)d{6}-d{2}-d{6}(?!w)	`****-**-****`
사업자등록번호	111-11-01111	(?<!w)d{3}-d{2}-d{5}(?!w)	`**-**-****`
법인등록번호	111111-1111111	(?<!w)d{6}-?d{7}(?!w)	`****-*****`

로그 설정 가져오기/내보내기

로그 설정 가져오기/내보내기

파서 설정, 빠른 인덱스 설정, 필터 설정 내용을 JSON 파일로 내보내고, 다른 프로젝트에서 가져와 적용할 수 있습니다.
프로젝트마다 반복해서 설정을 작성하는 번거로움을 줄일 수 있습니다.

1. 하나의 프로젝트에 파서 설정, 빠른 인덱스 설정, 필터 설정을 추가하세요.
2. 각 설정 탭에서 화면 오른쪽 위에 **JSON** 버튼을 선택하세요.
3. **JSON 내보내기** 창이 나타나면 화면 오른쪽 위에 **내보내기** 버튼을 선택하세요.
4. JSON 설정 파일이 사용자 PC에 저장됩니다.
5. 다른 프로젝트로 이동한 다음 **로그** > **로그 설정** 메뉴로 이동하세요.
6. 앞서 JSON 설정 파일을 내보낸 설정 탭을 선택한 다음 버튼을 선택하세요.
7. 파일 선택 창이 나타나면 사용자의 PC에 저장한 JSON 설정 파일을 선택하세요.
8. **JSON 가져오기** 창이 나타나면 설정 내용을 확인한 다음 **목록에 추가하기** 또는 **덮어쓰기** 버튼을 선택하세요.
9. 화면 오른쪽 위에 **저장** 버튼을 선택하세요.

> JSON 설정 파일을 가져온 다음 **저장** 버튼을 선택하지 않으면 가져온 설정 내용을 저장할 수 없습니다.

경고 알림

모니터링 대상 시스템 또는 애플리케이션에서 문제가 발생했을 때, 미리 설정한 조건에 따라 사용자에게 알림을 전달하는 기능입니다. 알림은 이메일, SMS, 메신저, 앱 푸시 등 다양한 채널로 전송됩니다.

예를 들어, CPU 사용률, 메모리 사용량, 디스크 I/O, 네트워크 트래픽, 트랜잭션 수, 오류 발생, 응답 시간 지연 등 주요 성능 지표에 알림 기준을 설정해두면, 해당 기준을 초과하거나 이상 징후가 감지될 때 알림이 전송됩니다.

와탭의 AI 기반 알림 기능을 사용하면 복잡한 조건을 직접 설정하지 않아도 됩니다. AI는 애플리케이션의 정상적인 동작 패턴을 학습해 이상 패턴을 자동으로 감지하여 알려줍니다. 태그 카운트 알림 기능을 활용하면 특정 태그 기반으로 집계된 값에 조건을 설정하여 복잡한 상황도 정밀하게 감지할 수 있습니다.

주요 기능

- **이벤트 설정**: 임계치 기반 알림 조건과 수신 채널 설정
- **이벤트 수신 설정**: 사용자별 수신 채널, 시간대 설정
- **이벤트 기록**: 알림 발생 기록 조회

권한

이벤트를 추가하거나 설정하려면 **알림 설정 권한**이 필요합니다.

- **조회**: 메트릭스 이벤트 설정 내용을 확인하려면 **조회 분석 권한**이 필요합니다.
- **추가/수정/삭제**: 설정을 변경하려면 **알림 설정 권한**이 필요합니다.
- **알림 수신**: 알림을 수신하려면 **알림 수신 권한**이 필요합니다.

> **참고 문서**
>
> 자세한 내용은 멤버 권한을 참고하시기 바랍니다.

이벤트 설정

홈 화면 > 프로젝트 선택 > **경고 알림** > **이벤트 설정**

이벤트 설정에서 모니터링 대상 시스템 또는 애플리케이션에서 발생할 수 있는 이상 징후를 기준으로 이벤트를 설정하고 관리하는 기능을 제공합니다. 설정된 조건이 충족되면 이메일, SMS, 메신저, 앱 푸시 등 원하는 채널로 알림을 받을 수 있습니다. 모니터링 목적에 따라 이벤트를 구성할 수 있습니다.

메트릭스

메트릭스 이벤트는 실시간 데이터를 기반으로 시스템 및 애플리케이션의 성능 지표를 모니터링할 수 있습니다.

실시간 로그

실시간으로 수집한 로그에서 설정한 이벤트 조건이 발견되면 경고 알림을 보냅니다.

복합 로그

로그 내용을 기준으로 경고 알림을 발생시키는 복합 로그 이벤트를 설정합니다.

참고

- 이벤트 설정 시 적정 임계값을 사용하여 불필요한 이벤트 발생을 방지하세요.
- 알림 설정 시 해소 알림(문제 해결 시 알림) 여부도 함께 고려하면 운영 대응에 도움됩니다.

공통 기능

모든 이벤트 설정을 **JSON 또는 Excel 형식으로 다운로드**할 수 있습니다. 같은 유형의 제품 설정을 공유하면 반복되는 이벤트 설정 작업을 줄일 수 있습니다.

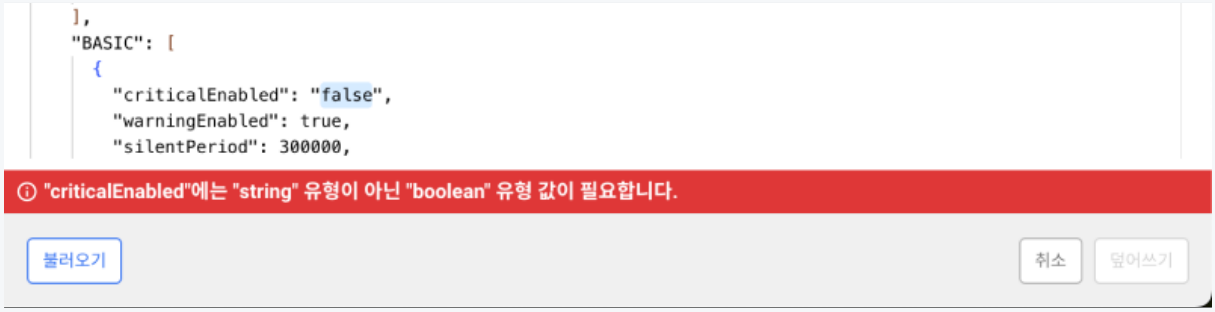
- **JSON 일괄 수정**: 모든 이벤트 조건을 JSON 형식으로 일괄 수정할 수 있습니다.
- **JSON 일괄 다운로드**: 현재 등록된 이벤트 조건을 JSON 파일로 저장할 수 있습니다.
- **Excel 일괄 다운로드**: 이벤트 목록을 Excel 파일로 내려받아 외부 보고나 백업에 활용할 수 있습니다.

JSON 일괄 수정

1. 화면 상단의 [JSON 일괄 수정] 버튼을 클릭하세요.
2. **JSON 일괄 다운로드** 기능을 통해 다운로드 받은 JSON 파일을 가져오거나 직접 수정하세요.
3. [덮어쓰기] 버튼을 클릭해 저장합니다.
4. 확인 메시지가 나타나면 [덮어쓰기] 버튼을 클릭하세요.

덮어쓰기 후 이전의 이벤트 설정으로 복구할 수 없습니다.

- `productType` 또는 `platform` 속성의 값을 임의로 수정하면 다른 프로젝트에서 이벤트가 정상 동작하지 않을 수 있습니다.
- JSON 편집창은 JSON 검증(Validation) 기능이 내장되어 있습니다. 형식이나 값이 올바르지 않은 경우, 편집창 아래에 오류 메시지가 표시되며 [덮어쓰기] 버튼은 비활성화됩니다.



JSON 데이터 구조

```
{
  "productType": "{productType}",
  "platform": "{platform}",
  "events": {
    "HITMAP": {
      ...
    },
    "LOG_REALTIME": [
      {...}
    ],
    "METRICS": [
      {...}
    ],
    "COMPOSITE_LOG": [
      {...}
    ],
    "BASIC": [
      {...}
    ],
    "COMPOSITE_METRICS": [
      {...}
    ],
    "ANOMALY": [
      {...}
    ]
  }
}
```

JSON 필드	설명
productType	제품 타입(예: APM, Infra 등)
platform	플랫폼 정보(예: java, node 등)
events	이벤트 데이터
ㄴ HITMAP	히트맵 패턴

JSON 필드	설명
ㄴ LOG_REALTIME	실시간 로그 이벤트
ㄴ METRICS	메트릭스
ㄴ COMPOSITE_LOG	복합 로그 이벤트
ㄴ BASIC	기본 이벤트
ㄴ COMPOSITE_METRICS	복합 메트릭스
ㄴ ANOMALY	이상치 탐지

JSON 일괄 다운로드

화면 상단의 [JSON 일괄 다운로드] 버튼을 클릭하세요.

- JSON 파일 이름은 integrated-event-`YYYYMMDD`.json 형식입니다.
- JSON 데이터의 구조와 `events` 속성의 하위 속성은 제품(`platform`) 또는 제품 유형(`productType`)에 따라 다를 수 있습니다.

JSON 데이터의 속성 중 `productType` 또는 `platform` 속성의 값을 임의로 수정하면 다른 프로젝트에서 이벤트가 동작하지 않을 수 있습니다.

Excel 일괄 다운로드

화면 상단의 [Excel 일괄 다운로드] 버튼을 클릭하세요.

메트릭스

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **메트릭스** 탭

프로젝트에 설정된 모든 메트릭스 기반 이벤트 목록을 확인할 수 있습니다. 사용자가 설정한 메트릭스 이벤트를 확인, 수정, 삭제하거나, 새로운 이벤트를 추가할 수 있습니다.

기본 옵션

- **이벤트 레벨**: 이벤트의 심각도를 의미하며 정상(Info), 경고(Warning), 위험(Critical) 세 단계로 구분
- **에이전트 중복 알림**: 설정한 시간(30초, 1분, 3분, 5분, 10분)에 같은 이름의 에이전트가 동시에 2개 이상 동작하는 경우, 선택한 시간 동안 이벤트 발생 안함
- **검색**: 이벤트 이름을 기준으로 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **+ 이벤트 추가**: 새로운 메트릭스 이벤트 추가
- **컬럼 크기 조정**: 각 컬럼의 경계를 드래그하여 너비 조정 가능

표 | 메트릭스 이벤트 목록 구성

항목	설명
No.	이벤트의 순번
	이벤트의 활성화 여부 – 활성화: 조건을 충족하면 이벤트 발생 – 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음
	이벤트 설정 편집 및 삭제
이벤트 이름	사용자가 지정한 이벤트명 – 최대 255자까지 입력 가능
규칙	이벤트 발생 조건

항목	설명
대상	이벤트가 적용되는 모니터링 대상
발생 횟수	이벤트 발생 기준이 되는 기간과 횟수 – 선택한 시간 동안 설정한 횟수만큼 조건을 충족하면 이벤트 발생
일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 – 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 – 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가 발생해도 알림이 발생하지 않음
해소된 알림	Critical과 Warning 레벨의 이벤트가 해소되면 RECOVERED(정상) 상태의 알림 수신 여부 – 활성화: 진행 중 → 정상 상태로 전환되면 RECOVERED 알림을 보내고, 이벤트 기록에서 진행 중/해소 상태를 구분해 볼 수 있음 – 비활성화: 임곗값을 초과할 때마다 단발성 이벤트 발생
이벤트 수신	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)

이벤트 설정

메트릭스 이벤트는 모니터링 대상에서 발생하는 이벤트를 조건에 따라 정의하고, 효율적으로 알림을 받을 수 있도록 설정할 수 있는 기능을 제공합니다.

이벤트 추가

메트릭스 이벤트를 추가하려면 **이벤트 설정** > **메트릭스** 탭으로 이동한 후, 오른쪽 상단의 [+ 이벤트 추가] 버튼을 클릭하세요.

메트릭스 이벤트 추가 방법
Step 1. 템플릿 선택: 이벤트의 기본 구조를 선택합니다.
Step 2. 이벤트 조건 정의: 이벤트를 어떤 기준으로 발생시킬지 조건을 설정합니다.

메트릭스 이벤트 추가 방법
Step 3. 이벤트 대상 선택: 이벤트가 발생하는 모니터링 대상을 선택합니다.
Step 4. 기본 정보 및 수신 설정: 이벤트 이름 및 메시지, 수신 대상을 설정합니다.

템플릿 선택

1. 이벤트를 어떻게 구성할지에 따라 화면 왼쪽의 3가지 템플릿 중 하나를 선택하세요.
    - **새로 만들기**: 카테고리와 지표를 처음부터 자유롭게 구성할 수 있으며, 복잡한 조건이나 세밀한 모니터링 규칙이 필요한 고급 사용자에게 적합
    - **빠른 설정**: 사전 정의된 카테고리, 지표, 기본 임계치가 제공되며, 임곗값만 조정하여 빠르게 적용할 수 있어 초급 사용자에게 적합
    - **고급 설정**: 사전 정의된 카테고리와 지표를 기반으로 조건을 추가하거나 자유롭게 변경

> ⓘ 메트릭스 이벤트에서는 다양한 설정 방식을 지원합니다. 모니터링 플랫폼에 따라 제공하는 템플릿은 다를 수 있습니다.

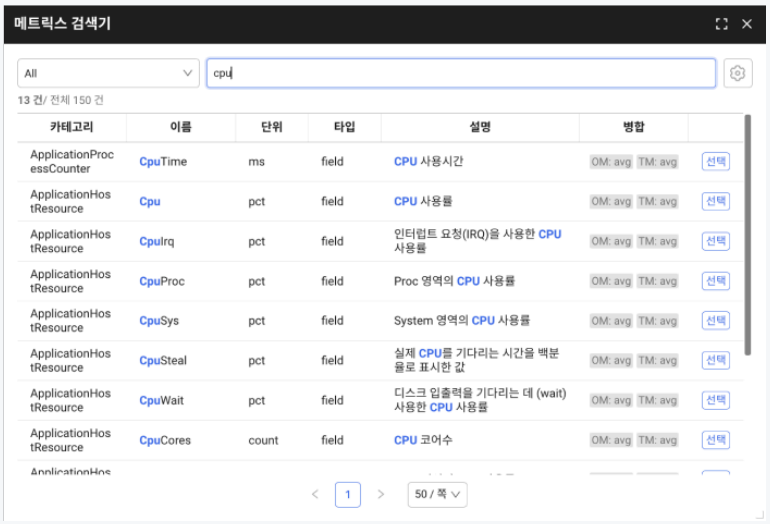
이벤트 조건 정의

이벤트 조건 정의 단계에서 이벤트가 언제 발생할지 결정하는 지표 설정, 발생 횟수, 일시 중지, 해소된 알림, 이벤트 동작 시간을 설정합니다.

지표 설정

1. 검색창에서 카테고리를 입력하거나, 카테고리 목록에서 카테고리(지표)를 선택하세요.
   카테고리를 선택해야 이후 지표 설정을 진행할 수 있습니다.
    - 카테고리 목록은 카테고리 이름, 데이터 수집 주기, 키를 확인할 수 있습니다. 최근 3시간 동안 프로젝트에서 수집된 메트릭스 데이터가 표시됩니다.
2. 목록에 없는 지표를 사용하려면 **직접 입력하기**를 선택하여 카테고리를 직접 입력하세요.

> **카테고리와 지표를 모르는 경우 메트릭스 검색기를 활용하세요.**
>
> 처음 지표를 설정하는 경우 메트릭스 검색기를 활용하면 더 빠르게 지표를 찾을 수 있습니다.
>
> - 메트릭스 검색기에서 원하는 지표를 검색하면 관련 카테고리와 상세 정보를 확인할 수 있습니다.
> - 검색 결과에서 필요한 메트릭스의 [선택] 버튼을 클릭하면, 선택한 값이 이벤트 설정 화면에 자동 반영됩니다.



3. 이벤트 레벨(Critical, Warning, Info)을 선택하세요. 하나의 지표에 여러 레벨을 설정할 수 있습니다.

> **이벤트 레벨 동작**
>
> 여러 조건을 만족하더라도 가장 높은 우선 순위(Critical → Warning → Info)의 레벨만 이벤트가 발생합니다. 레벨 동작 방식은 레벨 동작 가이드를 참고하세요.
>
> - 이벤트 레벨 수준은 위험(Critical)과 경고(Warning), 정상(Info)으로 구분합니다.
> - 같은 지표로 레벨을 설정하고 단계별 임계치를 설정하는 것을 권장합니다.

4. 선택한 이벤트 레벨의 필드, 연산자, 값을 입력하세요.

   a. **+ 추가**를 클릭해 조건을 추가할 수 있습니다.

   - `&&` : **모든 조건을 동시에 만족**해야 이벤트 발생

- `||`: **하나 이상의 조건만 만족**하면 이벤트 발생

b. — 를 클릭하면 해당 조건이 삭제됩니다.

> ⓘ 조건식 작성 방법과 지원되는 연산자는 조건 설정 가이드를 참고하세요.

발생 횟수

조건이 몇 번 충족되었을 때 이벤트를 발생시킬지 설정합니다. 지속적인 문제만 감지하고 싶을 때 유용합니다.

1. 발생 빈도를 선택하세요.
   - **연속**: 설정한 횟수만큼 이벤트가 연속으로 발생하는 경우 이벤트 발생
   - **최근**: 선택한 시간 내에 설정한 횟수만큼 이벤트가 발생하는 경우 이벤트 발생
     - 선택 가능 시간: `5초` , `10초` , `15초` , `30초` , `1분` , `2분` , `3분` , `4분` , `5분` , `10분` , `30분` , `1시간`
2. 발생 횟수를 입력하세요.
   - **Interval** 값은 선택한 카테고리의 데이터 수집 간격입니다. (Interval: 10s)

일시 중지

과도한 이벤트 발생을 방지하기 위해, 이벤트 동작을 일시 중지할 시간을 선택하세요. 이벤트 발생 후 설정한 시간 동안 동일한 이벤트가 발생하지 않습니다.

- 선택 가능 시간: `사용 안 함` , `5분` , `10분` , `15분` , `20분` , `30분` , `1시간` , `2시간` , `3시간` , `6시간` , `12시간` , `1일`

> ⓘ **해소된 알림 활성화 시**
>
> 알림은 정상 상태 알림(**RECOVERED**)을 받은 시점을 기준으로 설정한 시간이 지나기 전까지 동일한 이벤트가 발생해도 알림이 발생하지 않습니다.

해소된 알림

해소된 알림 기능을 활성화하면, 이벤트 발생 이후 상태 변화(진행 중 → 해소됨)를 추적할 수 있습니다. **이벤트 기록** 메뉴에서 진행

중인 이벤트로 표시됩니다.

- 해소된 알림은 Critical과 Warning 레벨에만 적용되며, Info 레벨은 항상 단발성 이벤트로 동작합니다.
- Critical과 Warning 레벨의 이벤트가 해소되면, 정상(RECOVERED) 상태의 알림이 전송됩니다.
- 토글 버튼을 클릭해 **활성화** 또는 **비활성화**하세요.
  - **활성화**: 이벤트는 상태 기반으로 동작하며, 임계치 초과 시 진행 중, 임계치 이하로 복구 시 해소 상태로 전환됨
  - **비활성화**: 임곗값을 초과할 때마다 단발성 이벤트 발생

이벤트 동작 시간

이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. 이벤트 동작 시간에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 삭제됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

시뮬레이션

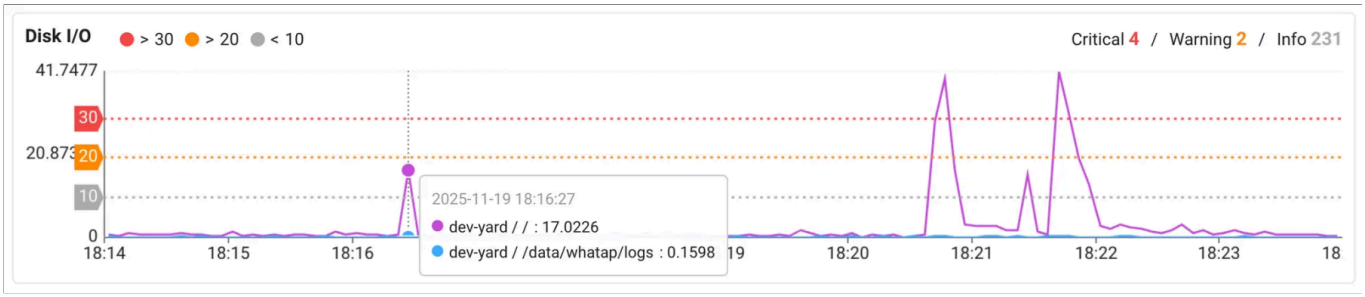
시뮬레이션 기능은 설정한 이벤트 조건을 과거 데이터에 기반으로 사전 테스트 할 수 있습니다. 예상 이벤트 수가 지나치게 많거나 부족하다면 조건을 조정해 최적의 임곗값을 찾을 수 있습니다.

- **규칙 검증**: 새로운 이벤트 규칙을 만들었을 때, 과거 데이터에 적용하면 어떤 결과가 나오는지 확인 가능
- **임곗값 튜닝**: 이벤트가 자주 발생하거나 전혀 발생하지 않을 때 적절한 수준으로 임곗값 조정

시뮬레이션 시작하기

1. 이벤트 조건 설정이 완료되면 [시뮬레이션] 버튼을 클릭하세요.
2. 예상 이벤트 건수와 시각화된 결과를 확인합니다.

시뮬레이션 결과 이해하기



1. **시계열 데이터**

   지표의 시간대별 변화와 각 레벨의 임계선이 그래프로 표시되어 이벤트 발생 시점을 직관적으로 확인할 수 있습니다.

2. **레벨별 발생 현황**

   화면 우측에 Critical, Warning, Info 레벨의 이벤트 발생 횟수가 표시됩니다. 전체 이벤트 분포와 빈도를 파악할 수 있습니다.

3. **대상별 상세 정보**

   차트의 특정 지점에 마우스를 올리면 해당 시점의 정확한 지표 값과 모니터링 대상이 표시됩니다. 여러 대상이 있는 경우 색상으로 구분되며, 대상별 패턴이 다른 경우 필터링 기능을 사용해 개별적으로 분석할 수 있습니다.

4. **피크 구간 식별**

   그래프에서 지표가 급격히 상승하는 시간을 확인하여 문제가 자주 발생하는 시간대나 패턴을 파악할 수 있습니다.

시뮬레이션 활용 팁

- **다양한 시간대 시뮬레이션**

  평일/주말, 피크/비피크 시간대를 각각 시뮬레이션하여 시간대별 최적 임곗값을 설정하세요.

- **실제 장애 기간 포함**

  과거 장애가 발생했던 기간을 포함하여 시뮬레이션을 실행하면 현재 규칙이 장애를 정확히 감지하는지 확인하세요.

- **대상 필터링 추가**

모니터링 대상이 많으면 시뮬레이션 결과가 복잡해집니다. 필터링 기능으로 특정 대상만 선택하여 명확한 결과를 확인할 수 있습니다.

- **임곗값 단계적 조정**

  임곗값을 한 번에 크게 변경하기보다 10% 단위로 점진적으로 조정하면서 최적값을 탐색하는 것이 좋습니다.

이벤트 대상 선택

이벤트 대상을 지정하지 않으면, 프로젝트에 포함된 전체 대상으로 이벤트가 발생되어 많은 이벤트가 발생할 수 있습니다. 따라서 이벤트가 적용될 대상을 명확히 설정하는 것을 권장합니다.

이벤트 대상을 변경한 경우, 이벤트 건수가 달라질 수 있습니다. **시뮬레이션** 버튼을 클릭한 후, 다시 실행해 예상 이벤트 수를 확인하세요.

대상 선택

이벤트가 적용될 대상을 **선택 입력** 또는 **직접 입력** 방식으로 지정할 수 있습니다.

- 선택 입력 **직접 입력**

1. 태그, 연산자, 값을 선택 또는 입력하세요. a. 태그를 찾기 어렵다면, **메트릭스 검색기**에서 태그를 검색할 수 있습니다.
2. **+ 추가**를 클릭해 조건(`&&` , `||`)으로 이벤트 대상을 추가하세요.

모니터링할 대상의 태그, 연산자, 값을 직접 입력하세요.

- 직접 입력을 선택하면 입력란 아래에 사용 가능한 태그 목록(**태그 선택 입력**)이 함께 제공됩니다.

```
ex. endsWith(okindName, 'example_name') && container == 'prod.billing'
ex. ${4xxErrorType} == '401'
```

입력 방법을 변경하면, 작성한 내용이 초기화됩니다.

대상 선택 연산자 목록

선택 입력 방식에서 문자열 태그를 선택하면 연산자 목록에 다음 두 항목이 함께 표시됩니다. 목록에 나타나는 이름은 함수 표기 그대로이며, `startsWith(key, 'match_text')` 와 같은 기존 항목과 형태가 같습니다.

- `equalsIgnoreCase(key, 'match_text')`
- `notEqualsIgnoreCase(key, 'match_text')`

두 연산자는 대소문자를 구분하지 않고 값을 비교합니다. **직접 입력** 방식에서는 같은 함수를 조건식에 직접 작성해 사용할 수 있습니다. 조건식 문법과 지원되는 함수 전체 목록은 조건 설정 가이드를 참고하세요.

> ⓘ 이벤트 발생 조건과 이벤트 대상 선택은 같은 문법을 사용하므로 두 함수를 양쪽 조건식에 모두 작성할 수 있습니다.

기본 정보 및 수신 설정

이벤트의 이름, 메시지, 수신 대상을 설정합니다. 설정한 내용은 이벤트 기록과 실제 알림 메시지에 그대로 표시되므로, 식별하기 쉽고 명확하게 작성하는 것이 중요합니다.

이벤트 활성화

설정한 조건을 충족했을 때 이벤트 발생 여부를 설정합니다.

- **활성화**: 설정한 조건을 충족하면 이벤트가 발생됨
- **비활성화**: 조건을 검사하지 않음

이벤트 이름

이벤트 제목으로 사용될 이벤트 이름을 입력하세요. 설정한 이벤트 이름은 **이벤트 목록** 및 **이벤트 기록** 메뉴에 표시되며, 해당 이름으로 검색할 수 있습니다.

메시지

이벤트 메시지를 입력하세요. 입력한 메시지는 이벤트 기록 및 알림에 사용됩니다. 을 클릭하면 이전에 작성한 메시지 기록을 확인할 수 있습니다. 메시지에 변수를 사용하여 이벤트 발생 시점의 실제 값을 포함할 수 있습니다.

- **변수 사용 규칙**
  - 변수 형식: `${지표명}`

- 설정한 **카테고리와 동일한 카테고리** 내 지표만 변수로 사용할 수 있습니다.
- 여러 변수를 조합해 상세한 메시지를 구성할 수 있습니다.
- 사용 가능한 변수 목록은 **메트릭스 조회** 메뉴에서 확인할 수 있습니다.

- **변수 활용 예시**

	이벤트 설정	이벤트 발생 시
제목	Disk 사용량 증가	Disk 사용량 증가
메시지	`Disk = ${disk}%`	`Disk = 89.9%`

이벤트 수신

이벤트 알림을 받을 멤버를 지정합니다.

- **전체 수신** **태그 선택 수신**

프로젝트에서 이벤트 수신 설정이 활성화된 모든 대상(멤버, 채널)에게 알림을 전송합니다.

해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다.

1. **태그 선택 수신**을 선택하세요.
2. **수신 태그** 항목에서 [+ 태그 추가] 또는 [+] 버튼을 클릭하세요.
3. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
4. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
5. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
6. **이벤트 수신 태그** 창에서 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> **수신 태그**
>
> 수신자 태그는 알림을 받을 사용자 또는 채널을 **그룹 단위로 관리하기 위한 기능**입니다. 이벤트에 수신자 태그를 설정하면, 해당 태그가 포함된 사용자에게만 알림이 전달됩니다. 수신 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 설정**의 **메트릭스** 탭으로 이동하세요.
2. 이벤트 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 규칙 수정** 창에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 규칙 수정** 창의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules- `YYYYMMDD` .json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.
3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

> ⓘ 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.
>
> Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)

JSON 데이터 구조


```
{
  "version": 2,
  "eventId": "zcef7s7f27rum1",
  "enabled": true,
  "stateful": false,
  "title": "Active Transaction",
  "message": "Active Transaction = ${active_tx_count}",
  "category": "app_counter",
  "alertLabel": [
    "oid"
  ],
  "repeatCount": 1,
  "repeatDuration": 0,
  "silent": 300000,
  "receiver": [],
  "timeTag": [],
  "selectString": "",
  "conditions": [
    {
      "level": 20,
      "enabled": true,
      "rule": "active_tx_count > 100"
```

```
    }
  ],
  "createTime": 1763632674345,
  "lastModifiedTime": 1763632674347,
  "lastModifiedUser": "support@whatap.io",
  "basic": true,
  "metaId": "java004",
  "selectCondition": {
        "oid": [
          "-1010758404",
          "-250906941"
        ]
  }
}
```


표 | JSON 데이터 구조

JSON 필드	타입	설명	비고
version	Integer	이벤트 규칙의 버전	변경 제한
eventId	String	이벤트 규칙의 고유 식별자	
enabled	boolean	이벤트 활성화 여부	
stateful	boolean	상태 기반 이벤트 여부	
title	String	사용자가 설정한 이벤트 규칙 이름	
message	String	사용자가 설정한 이벤트 규칙 메시지	
category	String	데이터 카테고리	
alertLabel	`List<String>`	이벤트의 상태 관리를 위해 카테고리마다 지정되는 기본 식별자(primary key) 값	변경 제한
repeatCount	Integer	반복 횟수	
repeatDuration	Long	반복 기간	

JSON 필드	타입	설명	비고
silent	Integer	일시 중지 시간	
receiver	List<String>	수신자 태그의 키값 목록	
timeTag	List<String>	동작시간 태그의 키값 목록	
selectString	String	대상 선택	
conditions	List<Object>	레벨별 발생 조건 목록	
ㄴ level	byte	레벨 값	Critical : 30 Warning : 20 Info : 10
ㄴ enabled	boolean	레벨 활성화 여부	
ㄴ rule	String	레벨 발생 조건	
createTime	Long	이벤트 규칙 최초 생성 시간	
lastModifiedTime	Long	이벤트 규칙 마지막 수정 시간	
lastModifiedUser	String	이벤트 규칙 마지막 수정 계정	
basic	boolean	빠른 설정으로 생성한 이벤트 여부	변경 제한
metaId	String	빠른 설정 템플릿 고유 식별자	변경 제한: 이 값은 빠른 설정으로 생성한 이벤트에만 존재합니다. (basic=true)
selectCondition	Object	빠른 설정 대상 선택값	이 값은 빠른 설정으로 생성한 이벤트에만 존재합니다. (basic=true)

‘변경 제한’으로 표시된 필드는 이벤트 동작에 영향을 줄 수 있으므로 값을 변경하지 않는 것을 권장합니다.

부록

발생 조건, 대상 선택 가이드

메트릭스 경고 알림의 이벤트 발생 조건과 이벤트 대상 선택은 동일한 문법을 사용합니다. 단, 이벤트 발생 조건은 필드(Field)의 Key를 변수로 사용하고, 이벤트 대상 선택은 태그(Tag)의 Key를 변수로 사용합니다.

레벨 동작 가이드

이벤트 레벨은 위험(Critical)과 경고(Warning), 정상(Info)으로 구분합니다.

1. **우선순위 기반 이벤트 발생**

   여러 레벨의 조건을 동시에 충족할 경우, 가장 높은 우선순위의 이벤트만 발생합니다.

   예시

   ```
   설정:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   현재 상태: CPU 95%
   → 결과: Critical 이벤트만 발생 (Warning은 억제됨)
   ```

2. **레벨 상승**

   진행중 상태를 갖는 이벤트에서 낮은 레벨 이벤트가 진행 중일 때 높은 레벨 조건을 충족하면, 두 레벨 모두 진행 중 상태가 됩니다.

   예시

   ```
   설정:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   시나리오:
   1) CPU 80% → Warning 발생 (진행 중)
   ```

```
2) CPU 95% → Warning (진행 중), Critical (진행 중)
```

3. **레벨 하강**

   진행중 상태를 갖는 이벤트에서 높은 레벨에서 낮은 레벨로 전환되면 높은 레벨은 해소되고 낮은 레벨이 유지됩니다.

   예시

```
설정:
- Warning: CPU > 70%
- Critical: CPU > 90%

시나리오:
1) CPU 97% → Critical 발생
2) CPU 85% → Critical 해소, Warning 유지
3) CPU 65% → Warning 해소
```

4. **동작 흐름도**

```
- Warning: CPU > 70%
- Critical: CPU > 90%

메트릭 값: 60%  →  75%  →  92%  →  85%  →  60%
상태:      정상    Warning   Critical  Warning   정상
                   (발생)    (발생)    (유지)    (해소)
                             Warning
                             (유지)
```

> 진행 중 상태를 갖는 이벤트는 조건이 충족되는 동안 계속 활성화 상태를 유지하며, 조건이 해제될 때까지 지속됩니다.

조건 설정 가이드

> 지표명이 숫자로 시작하거나 특수문자가 포함된 경우 `${지표명}` 형태로 작성해야 합니다.
>
> - 괄호 `()`, `[]`

- 연산자 `+`, `-`, `*`, `/`, `%`
- 구분자 `:`, `@`, `#`, `,`, (공백)
- 기타 특수문자 `!`, `^`, `&`, `|`, `~`, ```, `=`

`${}` 사용 예시:

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` 없이 사용 가능: 영문자, 언더스코어(`_`), 점(`.`)만 포함된 경우

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. 비교 연산자

숫자 비교

```
크다:          cpu > 80
크거나 같다:    cpu >= 80
작다:          memory < 1000
작거나 같다:    memory <= 1000
같다:          status == 200
같지 않다:      error != 0
```

문자열 비교

```
status == 'OK'
region == "us-east-1"
```

2. 산술 연산자

- 기본 산술 연산

```
덧셈:   cpu + 10 >= 90
뺄셈:   memory - 100 >= 500
곱셈:   cpu * 2 >= 100
나눗셈:  disk / 1024 >= 100
나머지:  value % 10 == 0
```

- 괄호를 이용한 우선순위 제어

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 음수 표현

```
cpu > -100
```

3. 논리 연산자

- AND 연산 (&&)

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 연산 (||)

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 복합 논리 연산

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. 패턴 매칭 연산자

- LIKE 연산자

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 연산자

```
oname not like 'test-*'
url not like '*debug*'
```

5. 내장 함수

- Null 체크 함수

```
isNull(value)          # null인지 확인
isNotNull(value)       # null이 아닌지 확인
nvl(value, 0)          # null이면 기본값 반환
isEmpty(str)           # 빈 문자열인지 확인
isNotEmpty(str)        # 빈 문자열이 아닌지 확인
```

- 집계 함수

```
sum(cpu, memory, disk)          # 합계
avg(cpu, memory)                # 평균
max(cpu, memory, disk)          # 최대값
min(cpu, memory, disk)          # 최소값
count(value1, value2, value3)   # 개수
```

- 수학 함수

```
round(cpu, 2)          # 반올림 (소수점 2자리)
```

- 문자열 함수

```
length(str)            # 문자열 길이
startsWith(str, 'prefix')  # 접두사 확인
```

```
endsWith(str, 'suffix')    # 접미사 확인
indexOf(str, 'search')     # 문자열 위치
substring(str, 0, 10)      # 부분 문자열
trim(str)                  # 공백 제거
replace(str, 'old', 'new') # 문자열 치환
hasStr(str, 'search')      # 문자열 포함 여부
```

문자열 함수

```
equalsIgnoreCase(str, 'match_text')     # 대소문자를 구분하지 않는 일치 비교
notEqualsIgnoreCase(str, 'match_text')  # 대소문자를 구분하지 않는 불일치 비교
```

`==` 와 `!=` 는 대소문자를 구분하므로 `WebServer` 와 `webserver` 를 다른 값으로 판단합니다. 표기가 일정하지 않은 태그나 필드를 비교할 때 위 두 함수를 사용하면 대소문자 차이를 무시하고 값을 맞출 수 있습니다.

예시

```
equalsIgnoreCase(okindName, 'WebServer')
notEqualsIgnoreCase(container, 'PROD.billing')
```

이벤트 발생 조건과 이벤트 대상 선택은 같은 문법을 사용하므로 두 함수를 양쪽 조건식에 모두 작성할 수 있습니다.

- 조건 함수
  - **if:** 조건식이 참(`true`)이면 `true값` 을 반환하고, 거짓(`false`)이면 `false값` 을 반환하는 조건 함수

    형식

    ```
    if(조건, true값, false값)
    ```

    예시

    ```
    if(value >= 90, 'High', 'Medium') == 'High'
    ```

  - **decode**: 값과 조건을 비교하여 일치하는 조건의 결과를 반환, 없으면 기본값 반환

    형식

```
decode(값, 조건1, 결과1, 조건2, 결과2, ..., 조건N, 결과N, 기본값)
```

예시

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

- **in**: 값이 지정한 후보값 중 하나와 일치하면 `true`, 아니면 `false`

형식

```
in(값, 후보값1, 후보값2, ..., 후보값N)
```

예시

```
in(status, 200, 201, 204) == false
```

작성 시 주의사항

- 권장사항
  - 간단한 변수명은 `${}` 없이 사용 (예: cpu, memory)
  - 특수문자 포함 시 반드시 `${}` 사용
  - 복잡한 연산은 괄호로 명확하게 구분
  - 함수명은 정확히 입력 (대소문자 구분)
- 피해야 할 사항
  - 불완전한 표현식 작성 (예: `cpu >`, `memory &&`)
  - 괄호 미스매칭 (예: `(cpu > 80 && cpu > 70)`)
  - 연속된 연산자 사용 (예: `cpu >> 80`, `cpu >< 80`)

실시간 로그 이벤트

홈 > 프로젝트 선택 > 경고 알림 > **이벤트 설정** > **실시간 로그** 탭

수집한 로그 데이터를 조건에 맞춰 필터링하려 경고 알림을 설정할 수 있습니다. 로그 이벤트를 사용하려면 로그 모니터링이 활성화되어야 합니다. **로그 모니터링 활성화**에 대한 자세한 내용은 각 제품 로그 문서를 참고하세요.

- **실시간 로그 이벤트**: 실시간으로 수집한 로그에서 설정한 이벤트 조건이 충족되면 이벤트가 발생합니다.

기본 옵션

- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **알림 메시지 설정**: 수신할 메시지의 상세 항목 선택 및 항목 별 내용 편집
- **+ 이벤트 추가**: 새로운 로그 이벤트 추가

표 | 실시간 로그 이벤트 목록 구성

항목	설명
이벤트 이름	알림 제목으로 사용되는 이벤트 이름 - 최대 255byte까지 입력 가능
카테고리	로그 구분 명칭(로그 폴더명)
검색 키	로그 데이터에서 찾고 싶은 항목 - 예: HTTP 응답 상태 코드를 나타내는 값에 접근하고자 할 경우 검색 키 `status`
검색 값	검색 키에 해당하는 실제 데이터로 로그에서 입력한 단어를 포함할 경우 경고 알림을 보냄 - 예: 검색 키 `status` 검색 값 `200`을 설정한 경우 HTTP 응답 상태 코드 200을 포함하는 로그 데이터 수집 시 경고 알림 발생
이벤트 발생 일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 - 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 - 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가

항목	설명
	발생해도 알림이 발생하지 않음
이벤트 대상 필터링	이벤트가 적용될 대상의 조건 - 대상 조건을 설정하지 않으면 수집되는 모든 데이터에 대해 알림 여부 판단 - 선택 입력: 태그, 연산자, 값을 선택해 조건 구성 - 직접 입력: 조건을 직접 입력해 대상 지정
이벤트 수신 태그	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)
	이벤트 수정 및 삭제
	이벤트의 활성화 여부 - 활성화: 조건을 충족하면 이벤트 발생 - 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음

이벤트 설정

로그 이벤트는 실시간 로그 또는 일정 조건을 만족하는 로그가 누적될 때 경고 알림을 설정할 수 있는 기능입니다.

실시간 로그 이벤트 추가

1. **경고 알림** > **이벤트 설정**의 **실시간 로그** 탭을 클릭하세요.
2. 화면 오른쪽의 [+ 이벤트 추가] 버튼을 클릭하세요.
3. **이벤트 규칙 추가** 창에서 로그 이벤트 조건을 설정하세요.
    - **이벤트 이름**: 이벤트 제목으로 사용할 이름을 입력하세요.
    - **이벤트 활성화**: 토글 버튼 활성화 시, 토글 버튼 활성화 시, 조건을 충족하면 이벤트 발생합니다.
    - **레벨**: 이벤트 레벨(Critical, Warning, Info)을 선택하세요.
    - **메시지**: 이벤트 발생 시 전달받을 알림 메시지를 입력하세요. 메시지에 변수를 사용하여 이벤트 발생 시점의 실제 값을 포함할 수 있습니다.

> **변수 사용 규칙**
>
> - 변수 형식: `${지표명}`
> - 설정한 **카테고리와 동일한 카테고리** 내 지표만 변수로 사용할 수 있습니다.
> - 여러 변수를 조합해 상세한 메시지를 구성할 수 있습니다.
> - 사용 가능한 변수 목록은 **메트릭스 조회**에서 확인할 수 있습니다.

- **카테고리**: 로그 구분 명칭(로그 폴더명)을 목록에서 선택하거나 직접 입력하세요. 카테고리를 선택해야 검색 키와 검색 값을 입력할 수 있습니다.
- **검색 키**: 로그 데이터에서 찾고 싶은 항목을 입력하세요.
  - 예: HTTP 응답 상태 코드를 나타내는 값에 접근하고자 할 경우 **검색 키** `status`
- **검색 값**: **검색 키**에 해당하는 실제 데이터를 입력하세요. 로그에서 입력한 단어를 포함할 경우 경고 알림을 보냅니다.
  - 예: **검색 키** `status` **검색 값** `200` 을 설정한 경우 HTTP 응답 상태 코드 200을 포함하는 로그 데이터 수집 시 경고 알림 발생
- **이벤트 대상 필터링**: 이벤트가 적용될 대상의 조건을 입력하세요. 입력값이 없으면, 수집되는 모든 데이터에 대해 알림 여부 판단합니다.
  - **선택 입력**: 태그, 연산자, 값을 선택해 대상 조건을 설정하세요.

    > 조건식 작성 방법과 지원되는 연산자는 조건 설정 가이드를 참고하세요.

  - **직접 입력**: 조건을 직접 입력하세요.
- **이벤트 발생 일시 중지**: 알림을 일시 중지할 시간을 선택하세요. 이벤트 발생 후 설정한 시간 동안 동일한 이벤트가 발생하지 않습니다.
  - 선택 가능 시간: `사용 안 함` , `1분` , `2분` , `3분` , `5분` , `10분` , `20분` , `30분` , `1시간` , `3시간` , `6시간` , `12시간`
- **이벤트 동작 시간**: 이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.

1. **+ 추가**를 클릭하세요.
2. **이벤트 동작 시간**에서 **+새로운 태그 생성**을 클릭하세요.
3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
5. 수정 또는 삭제할 태그의 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 제거됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

- **이벤트 수신 태그**: 이벤트에 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신자 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.

  1. **+ 태그 추가** 또는 +를 클릭하세요.
  2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
  3. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
  4. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
  5. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

  > 이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. **경고 알림 > 이벤트 수신 설정**에서 프로젝트 멤버와 3rd-party 플러그인에 각각 태그를 지정할 수 있습니다.

4. **이벤트 규칙 추가** 창에서 설정이 완료되면 [저장] 버튼을 클릭하세요.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 설정**의 **실시간 로그** 탭으로 이동하세요.

2. 로그 이벤트 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 설정** 화면에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 설정** 화면의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules-`YYYYMMDD`.json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.

3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

ⓘ 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.

```
Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)
```

부록

발생 조건, 대상 선택 가이드

메트릭스 경고 알림의 이벤트 발생 조건과 이벤트 대상 선택은 동일한 문법을 사용합니다. 단, 이벤트 발생 조건은 필드(Field)의 Key를 변수로 사용하고, 이벤트 대상 선택은 태그(Tag)의 Key를 변수로 사용합니다.

레벨 동작 가이드

이벤트 레벨은 위험(Critical)과 경고(Warning), 정상(Info)으로 구분합니다.

1. **우선순위 기반 이벤트 발생**

   여러 레벨의 조건을 동시에 충족할 경우, 가장 높은 우선순위의 이벤트만 발생합니다.

   예시

   ```
   설정:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   현재 상태: CPU 95%
   → 결과: Critical 이벤트만 발생 (Warning은 억제됨)
   ```

2. **레벨 상승**

   진행중 상태를 갖는 이벤트에서 낮은 레벨 이벤트가 진행 중일 때 높은 레벨 조건을 충족하면, 두 레벨 모두 진행 중

상태가 됩니다.

예시

```
설정:
- Warning: CPU > 70%
- Critical: CPU > 90%

시나리오:
1) CPU 80% → Warning 발생 (진행 중)
2) CPU 95% → Warning (진행 중), Critical (진행 중)
```

3. 레벨 하강

진행중 상태를 갖는 이벤트에서 높은 레벨에서 낮은 레벨로 전환되면 높은 레벨은 해소되고 낮은 레벨이 유지됩니다.

예시

```
설정:
- Warning: CPU > 70%
- Critical: CPU > 90%

시나리오:
1) CPU 97% → Critical 발생
2) CPU 85% → Critical 해소, Warning 유지
3) CPU 65% → Warning 해소
```

4. 동작 흐름도

```
- Warning: CPU > 70%
- Critical: CPU > 90%

메트릭 값: 60%  →  75%  →  92%  →  85%  →  60%
상태:      정상     Warning   Critical   Warning   정상
                    (발생)    (발생)     (유지)    (해소)
                              Warning
                              (유지)
```

진행 중 상태를 갖는 이벤트는 조건이 충족되는 동안 계속 활성화 상태를 유지하며, 조건이 해제될 때까지 지속됩니다.

조건 설정 가이드

지표명이 숫자로 시작하거나 특수문자가 포함된 경우 `${지표명}` 형태로 작성해야 합니다.

- 괄호 `()`, `[]`
- 연산자 `+`, `-`, `*`, `/`, `%`
- 구분자 `:`, `@`, `#`, `,`, (공백)
- 기타 특수문자 `!`, `^`, `&`, `|`, `~`, ```, `=`

`${}` 사용 예시:

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` 없이 사용 가능: 영문자, 언더스코어(`_`), 점(`.`)만 포함된 경우

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. 비교 연산자

숫자 비교

```
크다:          cpu > 80
크거나 같다:    cpu >= 80
작다:          memory < 1000
작거나 같다:    memory <= 1000
같다:          status == 200
```

```
같지 않다:      error != 0
```

문자열 비교

```
status == 'OK'
region == "us-east-1"
```

2. 산술 연산자

- 기본 산술 연산

```
덧셈:   cpu + 10 >= 90
뺄셈:   memory - 100 >= 500
곱셈:   cpu * 2 >= 100
나눗셈:  disk / 1024 >= 100
나머지:  value % 10 == 0
```

- 괄호를 이용한 우선순위 제어

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 음수 표현

```
cpu > -100
```

3. 논리 연산자

- AND 연산 (&&)

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 연산 (||)

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 복합 논리 연산

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. 패턴 매칭 연산자

- LIKE 연산자

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 연산자

```
oname not like 'test-*'
url not like '*debug*'
```

5. 내장 함수

- Null 체크 함수

```
isNull(value)          # null인지 확인
isNotNull(value)       # null이 아닌지 확인
nvl(value, 0)          # null이면 기본값 반환
isEmpty(str)           # 빈 문자열인지 확인
isNotEmpty(str)        # 빈 문자열이 아닌지 확인
```

- 집계 함수

```
sum(cpu, memory, disk)          # 합계
avg(cpu, memory)                # 평균
max(cpu, memory, disk)          # 최대값
```

```
min(cpu, memory, disk)             # 최소값
count(value1, value2, value3)      # 개수
```

- 수학 함수

```
round(cpu, 2)          # 반올림 (소수점 2자리)
```

- 문자열 함수

```
length(str)              # 문자열 길이
startsWith(str, 'prefix')  # 접두사 확인
endsWith(str, 'suffix')    # 접미사 확인
indexOf(str, 'search')     # 문자열 위치
substring(str, 0, 10)      # 부분 문자열
trim(str)                # 공백 제거
replace(str, 'old', 'new') # 문자열 치환
hasStr(str, 'search')      # 문자열 포함 여부
```

문자열 함수

```
equalsIgnoreCase(str, 'match_text')     # 대소문자를 구분하지 않는 일치 비교
notEqualsIgnoreCase(str, 'match_text')  # 대소문자를 구분하지 않는 불일치 비교
```

`==` 와 `!=` 는 대소문자를 구분하므로 `WebServer` 와 `webserver` 를 다른 값으로 판단합니다. 표기가 일정하지 않은 태그나 필드를 비교할 때 위 두 함수를 사용하면 대소문자 차이를 무시하고 값을 맞출 수 있습니다.

예시

```
equalsIgnoreCase(okindName, 'WebServer')
notEqualsIgnoreCase(container, 'PROD.billing')
```

이벤트 발생 조건과 이벤트 대상 선택은 같은 문법을 사용하므로 두 함수를 양쪽 조건식에 모두 작성할 수 있습니다.

- 조건 함수
  - **if:** 조건식이 참(`true`)이면 `true값` 을 반환하고, 거짓(`false`)이면 `false값` 을 반환하는 조건 함수

형식

```
if(조건, true값, false값)
```

예시

```
if(value >= 90, 'High', 'Medium') == 'High'
```

- **decode**: 값과 조건을 비교하여 일치하는 조건의 결과를 반환, 없으면 기본값 반환

형식

```
decode(값, 조건1, 결과1, 조건2, 결과2, ..., 조건N, 결과N, 기본값)
```

예시

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

- **in**: 값이 지정한 후보값 중 하나와 일치하면 `true`, 아니면 `false`

형식

```
in(값, 후보값1, 후보값2, ..., 후보값N)
```

예시

```
in(status, 200, 201, 204) == false
```

작성 시 주의사항

- 권장사항
  - 간단한 변수명은 `${}` 없이 사용 (예: cpu, memory)
  - 특수문자 포함 시 반드시 `${}` 사용
  - 복잡한 연산은 괄호로 명확하게 구분
  - 함수명은 정확히 입력 (대소문자 구분)

- 피해야 할 사항
  - 불완전한 표현식 작성 (예: `cpu >` , `memory &&`)
  - 괄호 미스매칭 (예: `(cpu > 80 && cpu > 70)`)
  - 연속된 연산자 사용 (예: `cpu >> 80` , `cpu >< 80`)

복합 로그 이벤트 설정

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 설정** > **복합 로그** 탭

수집한 로그 데이터를 조건에 맞춰 필터링하려 경고 알림을 설정할 수 있습니다. 로그 이벤트를 사용하려면 로그 모니터링이 활성화되어야 합니다. **로그 모니터링 활성화**에 대한 자세한 내용은 (각 제품 로그 문서 링크)를 참고하세요.

- **복합 로그 이벤트**: 설정한 **데이터 조회 범위** 내 로그를 기준으로, 로그 검색 조건을 충족하는 로그 수가 임계 개수 이상일 경우 이벤트가 발생합니다.

> **권한**
>
> 복합 메트릭스 이벤트를 설정하려면 **알림 설정** 권한이 필요합니다. 알림 설정 권한은 **홈 화면 > 프로젝트 선택 > 관리 > 통합 멤버 관리**에서 권한을 부여할 멤버의 아이콘을 클릭해 설정할 수 있습니다.

기본 옵션

- **검색**: 이벤트 이름 또는 이벤트 수신 태그를 선택해 원하는 이벤트 검색
- **JSON 다운로드/업로드**: 이벤트 규칙은 JSON 편집기를 통해 바로 수정하거나, 파일로 다운로드하여 수정 후 다시 업로드 가능
- **알림 메시지 설정**: 수신할 메시지의 상세 항목 선택 및 항목 별 내용 편집
- **+ 이벤트 추가**: 새로운 로그 이벤트 추가

표 | 복합 로그 이벤트 목록 구성

항목	설명
이벤트 이름	알림 제목으로 사용될 이벤트 이름
템플릿	복합 로그 템플릿
카테고리	로그 구분 명칭(로그 폴더명)
규칙	이벤트 발생하는 조건

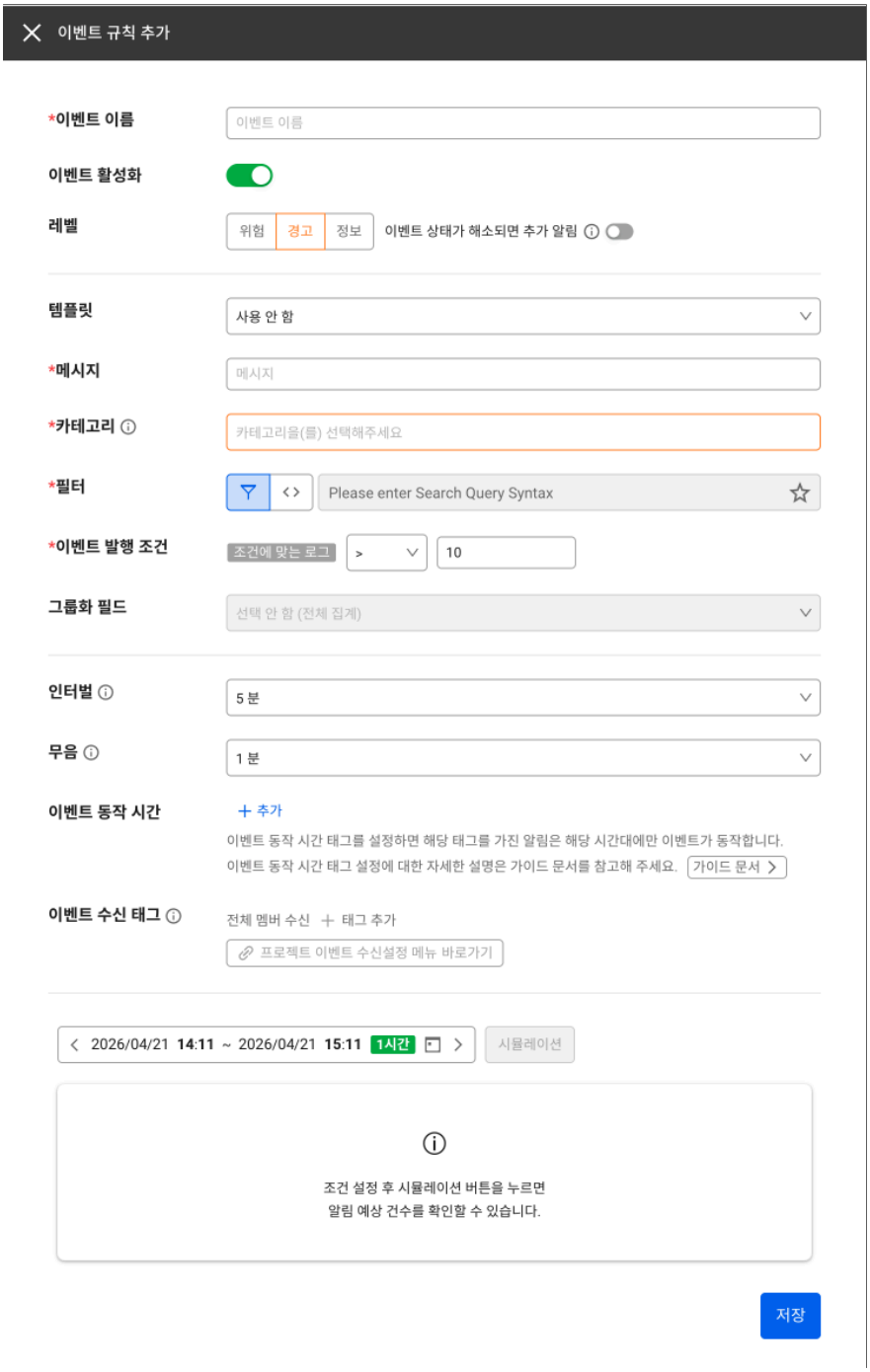
항목	설명
이벤트 상태가 해소되면 추가 알림	Critical과 Warning 레벨의 이벤트가 해소되면 정상(RECOVERED) 상태의 알림 수신 여부 - 활성화: 진행 중 → 정상 상태로 전환될 때 RECOVERED 알림을 보내며, 이벤트 기록에서 진행 중/해소 상태를 구분해 볼 수 있음 - 비활성화: 임계값을 초과할 때 마다 단발성 이벤트 발생
이벤트 발생 일시 중지	이벤트 알림 발생 후, 같은 이벤트 발생을 일정 시간 동안 멈춤 - 선택 가능 시간: 사용 안 함, 5분, 10분, 15분, 20분, 30분, 1시간, 2시간, 3시간, 6시간, 12시간, 1일 - 해소된 알림이 활성화된 경우, 정상(RECOVERED) 상태 알림을 받은 후 선택한 시간 동안 같은 이벤트가 발생해도 알림이 발생하지 않음
인터벌	로그 데이터를 조회할 시간 범위
이벤트 수신 태그	해당 이벤트 알림을 받을 사용자 또는 그룹(수신 태그)
	이벤트 수정 및 삭제
	이벤트의 활성화 여부 - 활성화: 조건을 충족하면 이벤트 발생 - 비활성화: 이벤트가 동작하지 않으며, 조건을 검사하지 않음

이벤트 설정

복합 로그 이벤트는 설정한 **데이터 조회 범위** 내 로그를 기준으로, 로그 검색 조건을 충족하는 로그 수가 임계 개수 이상일 경우 경고 알림을 설정할 수 있는 기능입니다.

복합 로그 이벤트 추가

1. **경고 알림** > **이벤트 설정**의 **복합 로그** 탭을 클릭하세요.
2. 화면 오른쪽의 [+ 이벤트 추가] 버튼을 클릭하세요.
3. **이벤트 규칙 추가** 창에서 로그 이벤트 조건을 설정하세요. 자세한 내용은 기본 설정, 상세 설정, 추가 설정를 참고하세요.



4. **이벤트 규칙 추가** 창에서 설정이 완료되면 [저장] 버튼을 클릭하세요.

기본 설정

- **이벤트 이름**: 이벤트 제목으로 사용할 이름을 입력하세요.
- **이벤트 활성화**: 토글 버튼 활성화 시, 조건을 충족하면 이벤트 발생합니다.
- **레벨**: 이벤트 레벨(Critical, Warning, Info)을 선택하세요.

상세 설정

템플릿(프리셋), 메시지, 카테고리, **필터**, 이벤트 발생 조건 항목을 설정합니다.

템플릿(프리셋)

미리 정의된 프리셋을 선택하면 **필터, 쿼리 타입, 이벤트 발행 조건**이 자동으로 설정됩니다. 프리셋을 선택한 후 필요에 맞게 수정할 수 있습니다.

복합 로그 템플릿 유형

프리셋	쿼리 타입	기본 규칙
사용 안 함	none	`rows > 10`
2xx 상태코드 건수	count	`count > 10`
3xx 상태코드 건수	count	`count > 10`
4xx 상태코드 건수	count	`count > 10`
5xx 상태코드 건수	count	`count > 10`
정상 상태코드(2xx, 3xx) 건수	count_v2	`include_minus_exclude_count > 10`
에러 상태코드(4xx, 5xx) 건수	count_v2	`include_minus_exclude_count > 10`
에러 수신 건수	count	`count > 10`
예외 수신 건수	count	`count > 10`

프리셋	쿼리 타입	기본 규칙
초당 로그 발생건수	rps	`rps > 100`
필드 집계 (Aggregation)	aggr	`avg > 1000`
비율 (Rate)	rate	`rate > 10`
고유값 수 (Cardinality)	cardinality	`cardinality > 100`
백분위 (Percentile)	percentile	`p99 > 3000`

메시지

이벤트 발생 시 전달받을 알림 메시지를 입력하세요. 메시지에 변수를 사용하여 동적 메시지를 구성할 수 있습니다.

- 예시: 메시지 `${host}`에서 에러 `${count}`건 발생 → 알림 메시지 web-01에서 에러 15건 발생

표 | 메시지 변수

변수	설명	사용 조건
`${count}`	조건에 맞는 로그 건수	항상 사용 가능
`${groupField}`	그룹화 필드 이름	그룹화 알림만
`${groupValue}`	그룹화 필드 값	그룹화 알림만
`${필드명}`	그룹화 필드 값 (필드명으로 직접 참조)	그룹화 알림만

`${count}` 변수는 그룹화 여부와 관계없이 모든 알림에서 사용할 수 있습니다. `${groupField}`, `${groupValue}`, `${필드명}` 변수는 그룹화 필드를 설정한 경우에만 치환됩니다.

카테고리

로그 구분 명칭(로그 폴더명)을 목록에서 선택하거나 직접 입력하세요. 카테고리를 선택해야 검색 키와 검색 값을 입력할 수 있습니다.

필터

Lucene 쿼리 문법으로 로그 검색 조건을 입력합니다. 두 가지 입력 모드를 지원합니다.

- **필터 모드** : 필드와 값을 선택하여 쿼리를 구성합니다.
- **코드 모드** : Lucene 쿼리를 직접 입력합니다.

표 | Lucene 쿼리 예시

쿼리	설명
`level:ERROR`	level 필드가 ERROR인 로그
`status_nxx:status_4xx OR status_nxx:status_5xx`	4xx 또는 5xx 상태코드
`appender:accessLog`	accessLog appender의 로그
`response_time.n:*`	response_time.n 필드가 존재하는 로그
`host:web-01 AND level:ERROR`	web-01 호스트의 에러 로그

이벤트 발생 조건

쿼리 타입별로 설정합니다. 선택한 템플릿(프리셋) 또는 쿼리 타입에 따라 추가 설정 항목이 표시됩니다.

표 | 이벤트 발행 조건 키 의미

키	의미
`rows` / `count`	필터 조건에 맞는 로그 건수
`rps`	초당 로그 발생건수 (건수 / 인터벌 초)

키	의미
avg	대상 필드 값의 평균
sum	대상 필드 값의 합계
min	대상 필드 값의 최솟값
max	대상 필드 값의 최댓값
rate	분자 건수 / 분모 건수 × 100 (%)
cardinality	대상 필드의 고유한 값 개수
p99 / p95 / p90 / p75 / p50	대상 필드의 백분위 값
include_minus_exclude_count	포함 필터 건수 - 제외 필터 건수

- **건수 (count / none)**: 설정한 필터 조건에 맞는 로그의 건수를 카운트합니다.

표 | 건수 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
이벤트 발행 조건	건수 또는 조건에 맞는 로그 {연산자} {임계값} (예: count > 10)

- **초당 로그 발생건수 (rps)**: 설정한 필터 조건에 맞는 로그의 초당 발생건수(Requests Per Second)를 계산합니다. 인터벌 시간 동안의 총 건수를 인터벌 초로 나누어 산출합니다.

표 | 초당 로그 발생건수 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
이벤트 발행 조건	RPS {연산자} {임계값} (예: rps > 0.5)

- **필드 집계 (aggr)**: 지정한 숫자 필드의 집계 값(평균, 합계, 최솟값, 최댓값)을 계산합니다. 이벤트 발행 조건에서 집계 방식을 드롭다운으로 선택할 수 있습니다.

표 | 필드 집계 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
대상 필드	집계 대상 숫자 필드 (예: `response_time.n`)
이벤트 발행 조건	`평균` / `합계` / `최솟값` / `최댓값` `{연산자}` `{임계값}` (예: `avg > 1000`)

- **비율 (rate)**: 분자 필터와 분모 필터의 건수 비율(%)을 계산합니다. 전체 로그 대비 특정 조건의 비율을 모니터링할 때 유용합니다.

표 | 비율 설정 항목

설정 항목	설명
필터 (분자)	비율 계산의 분자가 되는 로그 조건 (예: `level:ERROR`)
분모 필터	비율 계산의 분모가 되는 로그 조건 (예: `level:*`)
이벤트 발행 조건	`비율 (%)` `{연산자}` `{임계값}` (예: `rate > 10`)

  - **예시**: 필터 `level:ERROR`, 분모 필터 `level:*`, rule `rate > 10` → 전체 로그 중 에러 비율이 10%를 초과하면 알림 발생

- **고유값 수 (cardinality)**: 지정한 필드의 고유한 값 개수를 계산합니다. 비정상적으로 많은 종류의 값이 발생하는 경우를 감지할 때 유용합니다.

표 | 고유값 수 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
대상 필드	고유값을 계산할 필드 (예: `host`)

설정 항목	설명
이벤트 발행 조건	`고유값 수` `{연산자}` `{임계값}` (예: `cardinality > 100`)

- **백분위 (percentile)**: 지정한 숫자 필드의 백분위(Percentile) 값을 계산합니다. 응답 시간의 tail latency를 모니터링할 때 유용합니다. Percentile 선택을 변경하면 이벤트 발행 조건의 키가 자동으로 업데이트됩니다.

표 | 백분위 설정 항목

설정 항목	설명
필터	로그 검색 조건 (Lucene 쿼리)
대상 필드	백분위를 계산할 숫자 필드 (예: `response_time.n`)
Percentile	P99, P95, P90, P75, P50 중 선택
이벤트 발행 조건	`P99` / `P95` /... `{연산자}` `{임계값}` (예: `p99 > 3000`)

표 | 백분위 의미

백분위	의미
P50	중앙값. 전체 데이터의 50%가 이 값 이하
P75	전체 데이터의 75%가 이 값 이하
P90	전체 데이터의 90%가 이 값 이하
P95	전체 데이터의 95%가 이 값 이하
P99	전체 데이터의 99%가 이 값 이하

- **포함/제외 건수 (count_v2)**: 포함 필터의 건수에서 제외 필터의 건수를 뺀 순수 건수를 계산합니다.

표 | 포함/제외 건수 설정 항목

설정 항목	설명
필터 (포함)	포함할 로그 조건 (예: `status_nxx:status_4xx OR status_nxx:status_5xx`)
필터 제외	제외할 로그 조건 (예: `status_nxx:status_4xx`)
분모 필터	비율 계산용 분모 (선택사항)
이벤트 발행 조건	`포함-제외 건수` `{연산자}` `{임계값}` (예: `include_minus_exclude_count > 10`)

- **예시**: 필터 `status_nxx:status_4xx OR status_nxx:status_5xx`, 필터 제외 `status_nxx:status_4xx` → 4xx+5xx에서 4xx를 제외한 순수 5xx 건수가 10을 초과하면 알림 발생

그룹화 필드(Group By)

그룹화 필드를 설정하면 해당 필드의 값별로 독립적으로 알림 조건을 판정합니다. 드롭다운에서 그룹화할 필드를 선택합니다. 드롭다운을 클릭하면 현재 카테고리에서 사용 가능한 필드 목록이 자동완성됩니다.

- **예시:** 필터 `level:ERROR`, 그룹화 필드 `host`, rule `rows > 3`

표 | 그룹화 설정 비교

항목	그룹화 없음	그룹화 사용
판정 단위	전체 로그를 하나로 집계	그룹 값별 독립 집계
알림 발생	1건	조건 충족 그룹 수만큼
알림 제목	`이벤트 제목`	`[그룹 값] 이벤트 제목`
Stateful	전체 기준 상태 전이	그룹별 독립 상태 전이

- `[web-01] 에러 알림` — web-01에서 에러 8건 → 알림 발생
- `[web-02] 에러 알림` — web-02에서 에러 5건 → 알림 발생
- `[api-01] 에러 알림` — api-01에서 에러 2건 → 조건 미충족, 알림 미발생

> **이벤트 상태가 해소되면 추가 알림**을 활성화한 경우, 그룹별로 독립적으로 ABNORMAL → RECOVERED 상태가 전이됩니다. 예를 들어 `web-01` 에서 에러가 해소되면 `[web-01] RECOVERED` 알림이 발생하며, 다른 호스트의 상태에는 영향을 미치지 않습니다.

추가 설정

- **인터벌**: 선택한 시간(`unset` , `1분` , `5분` , `10분` , `30분` , `1시간` , `12시간`) 간격으로 알림 조건을 확인합니다.
- **무음**: 이벤트 발생 후, 선택한 시간(`unset` , `1분` , `5분` , `10분` , `30분` , `1시간` , `12시간`) 동안 같은 이벤트가 발생되지 않습니다.
- **이벤트 동작 시간**: 이벤트가 특정 시간대(근무/비근무/점검 시간 등)에 동작하도록 태그를 설정하세요. 태그를 설정하지 않으면 이벤트가 활성화된 상태에서 24시간 상시 동작합니다.
    1. **+ 추가**를 클릭하세요.
    2. **이벤트 동작 시간**에서 **+새로운 태그 생성**을 클릭하세요.
    3. **이벤트 동작 시간 태그 생성**에서 **태그 이름, 요일, 시간, 색상**을 선택하고 [적용] 버튼을 클릭하세요.
    4. 생성한 태그는 태그 목록에서 확인할 수 있으며, 체크 박스를 선택하면 적용됩니다.
    5. 수정 또는 삭제할 태그의 ✎ 아이콘을 클릭해 **이벤트 동작 시간 태그 수정**에서 태그를 수정하거나 태그 삭제할 수 있습니다.

> 태그 삭제 시 해당 태그가 적용된 모든 사용자의 태그가 제거됩니다. 단, 이벤트 규칙에서 사용 중인 태그는 삭제할 수 없습니다.

- **이벤트 수신 태그**: 이벤트에 수신 태그를 설정하면, 해당 태그를 가진 프로젝트 멤버에게 알림을 전송합니다. 수신자 태그를 설정하지 않으면 프로젝트 내 모든 사용자에게 알림이 전송됩니다.
    1. **+ 태그 추가** 또는 **+**를 클릭하세요.
    2. **이벤트 수신 태그** 창 아래의 **+ 새 태그 생성**을 클릭하세요.
    3. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
    4. 생성된 태그는 태그 목록의 ✎ 아이콘을 클릭해 수정 또는 삭제하세요.
    5. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

> 이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. **경고 알림 > 이벤트 수신 설정**에서 프로젝트 멤버와 3rd-party 플러그인에 각각 태그를 지정할 수 있습니다.

시뮬레이션

알림을 저장하기 전에 과거 데이터로 시뮬레이션하여 알림 발생 예상 건수를 확인할 수 있습니다.

1. 알림 설정을 완료한 후 하단의 시간 범위를 선택하세요.
2. [시뮬레이션] 버튼을 클릭하세요.
3. 차트에 시간대별 데이터와 알림 발생 횟수가 표시됩니다.

   > 시뮬레이션 시간 범위를 조절하여 다양한 시간대의 알림 발생 패턴을 확인할 수 있습니다. 최근 1~2분의 데이터는 인덱스 빌드 지연으로 조회되지 않을 수 있습니다.

4. **이벤트 규칙 추가** 창에서 설정이 완료되면 [저장] 버튼을 클릭하세요.

로그 탐색기에서 알림 생성

로그 탐색기에서 직접 알림을 생성할 수 있습니다.

1. **로그** > **로그 탐색** 메뉴로 이동하세요.
2. 카테고리와 검색 쿼리를 설정하세요.
3. 화면 오른쪽 위의 [이벤트 추가] 버튼을 클릭하세요.
4. Drawer에서 알림 설정 폼이 열리며, 현재 카테고리와 검색 쿼리가 자동으로 채워집니다.
5. 추가 설정을 완료하고 [저장] 버튼을 클릭하세요.
6. 저장 완료 후 [이동] 버튼을 클릭하면 알림 목록으로 이동합니다.

이벤트 수정/삭제

1. **경고 알림** > **이벤트 설정**의 **복합 로그** 탭으로 이동하세요.
2. 로그 이벤트 목록에서 수정 또는 삭제하려는 이벤트의 아이콘을 클릭하세요.
3. **이벤트 설정** 화면에서 옵션을 수정하고, [저장] 버튼을 클릭하세요.

   a. 선택한 이벤트를 삭제하려면 **이벤트 설정** 화면의 오른쪽 위의 [삭제] 버튼을 클릭하세요.

이벤트 공유

메트릭스 이벤트 설정을 JSON 파일로 저장해 다른 사용자와 설정을 공유하거나 다른 사용자의 설정을 가져올 수 있습니다.

- JSON 파일명: event-rules-`YYYYMMDD`.json

내보내기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. JSON 편집 창이 나타나면 [내보내기] 버튼을 클릭하세요.
   - 이벤트를 검색 후 내보내기 기능을 이용하면 검색한 목록만 JSON 파일로 다운받을 수 있습니다.
3. JSON 파일이 다운로드되면 공유할 다른 사용자에게 전달하세요.

가져오기

1. 화면 오른쪽 위에 [ ] 버튼을 선택하세요.
2. **내보내기** 기능을 통해 다운로드한 JSON 파일을 선택하세요.
3. JSON 편집 창이 나타나면 [목록에 추가하기] 또는 [덮어쓰기] 버튼을 선택하세요.

> 같은 종류의 제품 간에 이용할 것을 권장합니다. 다른 제품의 프로젝트로부터 이벤트 설정을 가져올 수는 있지만 정상 작동하지 않습니다.

Json 형식으로 수정하기

1. 화면 오른쪽 위에 [**JSON**] 버튼을 클릭하세요.
2. 편집 창이 나타나면 JSON 형식에 맞춰 내용을 수정하세요.
3. 수정을 완료하면 화면 아래 [저장] 버튼을 선택하세요.

> 수정한 내용이 JSON 형식에 맞지 않으면 화면 아래에 에러 메시지가 표시되며, 저장할 수 없습니다. 표시되는 에러 메시지는 형식에 따라 다를 수 있습니다.
>
> ```
> Expected ',' or '}' after property value in JSON at position 1964 (line 80 column 42)
> ```

이벤트 수신 설정

사용자별 이벤트 수신 설정

> **사용자별 이벤트 수신 설정** 의 다른 멤버의 정보는 **멤버 관리** 권한을 가진 멤버만 확인할 수 있습니다. 그 외 멤버에게는 필수 정보가 마스킹 처리됩니다.
>
> - 이벤트 알림의 일괄 수신설정 및 접근 설정을 위한 모바일 기기 관리는 **계정 정보**에서 할 수 있습니다.
> - 멤버 권한에 대한 자세한 설명은 다음 문서를 참고하세요.

수신 수단 설정

이메일, **SMS**, **WhatsApp**, **모바일** 알림을 제공합니다.

사용자별 이벤트 수신 설정 섹션에서 원하는 알림 수신 수단의 체크 박스를 선택하면 경고 알림을 받을 수 있습니다. 알림 수신 수단의 체크 박스를 해제하면 경고 알림을 보내지 않습니다.

> 프로젝트 최고 관리자를 제외한 모든 사용자는 자신의 수신 설정만 변경할 수 있습니다.

이메일 알림

이메일 알림은 회원 가입 시 입력한 이메일 주소로 알림을 보냅니다.

SMS 알림

SMS로 알림을 수신할 수 있습니다.

> SMS 알림 기능은 유료 프로젝트에 한정됩니다.

1. 화면 오른쪽 위의 **자신의 프로필 아이콘** > **계정 관리**로 이동하세요.
2. **계정 정보**의 **사용자 전화번호** 섹션에서 [일반 휴대전화] 버튼을 클릭하세요.
3. **전화번호**에 인증번호를 수신을 전화번호를 입력하세요.
    - SMS를 알림으로 수신할 수 있는 전화번호는 **한국의 휴대전화 번호만 등록**할 수 있습니다.
4. [저장] 버튼을 클릭하세요.

> **등록한 전화번호 변경 및 삭제**
>
> - 전화번호를 변경할 경우, [변경] 버튼을 클릭하세요. SMS 인증이 필요합니다.
> - 전화번호를 삭제할 경우, [삭제] 버튼을 클릭하세요. 전화번호 삭제 시 SMS로 수신받는 알림은 수신할 수 없습니다.

WhatsApp 알림

WhatsApp을 통해 알림을 수신할 수 있습니다.

> WhatsApp 알림 기능은 유료 프로젝트에 한정됩니다.

1. 화면 오른쪽 위의 **자신의 프로필 아이콘** > **계정 관리**로 이동하세요.
2. **계정 정보**의 **사용자 전화번호** 섹션에서 [WhatsApp] 버튼을 클릭하세요.
3. 인증번호를 받을 **전화번호**를 입력하세요.
4. [인증번호 전송] 버튼을 클릭하세요.
5. WhatsApp 애플리케이션으로 전송된 인증번호 6자리를 입력하세요.
    - 10분 내로 인증번호를 받지 못했다면, [인증번호 재전송] 버튼을 클릭하세요.
6. [인증하기] 버튼을 클릭하세요.

모바일 알림

모바일 알림 수신이 필요한 사용자는 모바일 기기에 와탭 애플리케이션을 설치한 후 로그인해야 합니다.

- 와탭 애플리케이션 다운로드 - **안드로이드**
- 와탭 애플리케이션 다운로드 - **iOS**

반복 알림(에스컬레이션) 설정

경고 알림 발생 시간으로부터 알림 발생 상황이 해소되지 않을 경우 최초 알림 발생 시각으로부터의 알림 반복 간격을 설정할 수 있습니다. 위험(Critical) 레벨의 알림만 적용됩니다.

예를 들어, 경고 알림 발생 시간으로부터 0분(즉시), 1시간 후, 1일 후에 경고 알림을 반복하려면 `0,1H,1D` 를 **반복 알림(에스컬레이션)** 컬럼 항목에 입력하세요.

1. **경고 알림** > **이벤트 수신 설정**의 **사용자별 이벤트 수신 설정** 섹션으로 이동하세요.
2. **반복 알림(에스컬레이션)** 항목에서 설정하세요.
   - **M**: 분, **H**: 시간, **D**: 일, 단위를 생략하면 분 단위로 시간을 설정합니다.
   - 숫자 또는 숫자+단위(**M**, **H**, **D**)로 입력하세요. 입력이 올바르지 않으면 메시지가 표시됩니다.
3. [저장] 버튼을 클릭하세요. 저장하지 않으면 설정되지 않습니다.

이벤트 수신 태그

이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다. 이벤트 수신 태그를 설정하지 않으면 전체 멤버에게 경고 알림이 전송됩니다.

이벤트 수신 태그 추가/수정/삭제하기

> 이벤트에 적용 중인 **이벤트 수신 태그** 항목은 삭제할 수 없습니다.

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **사용자별 이벤트 수신 설정** 섹션의 사용자 목록에서 **+ 태그 추가** 또는 **+** 를 클릭하세요.
3. **이벤트 수신 태그** 창에서 **+ 새 태그 생성**을 클릭하세요.
4. **태그 생성** 창에서 태그 이름을 입력하고, 색상을 선택한 후, [태그 생성] 버튼을 클릭해 태그를 생성하세요.
5. 생성된 태그는 태그 목록의 아이콘을 클릭해 수정 또는 삭제하세요.
6. **이벤트 수신 태그** 창의 **태그 목록**에서 원하는 태그를 선택하면 적용됩니다.

이벤트 수신 태그 해제하기

1. **경고 알림** > **이벤트 수신 설정**의 **사용자별 이벤트 수신 설정** 섹션으로 이동하세요.
2. 이벤트 수신 태그를 해제할 사용자의 **이벤트 수신 태그** 항목의 **+** 를 클릭하세요.
3. **이벤트 수신 태그** 창에서 해제할 태그의 X 를 클릭하면 해당 태그가 해제됩니다.

수신 태그 미설정 알림 받기

이벤트 설정 시 이벤트 수신 태그를 선택하여 해당 태그를 가진 프로젝트 멤버와 3rd-party 플러그인에 알림을 전송할 수 있습니다.

- 이벤트 수신 태그가 설정되지 않은 경고 알림을 받으려면 **수신 태그 미설정 알림 받기** 옵션을 선택하세요.
- 이벤트 수신 태그가 설정된 경고 알림만 받고 싶다면 선택을 해제하세요.

> ⓘ 모든 경고 알림을 받지 않으려면 해당 옵션을 해제하고 선택한 **이벤트 수신 태그**가 없어야 합니다.

이벤트 수신 모드 선택하기

사용자별 이벤트 수신 설정 섹션에서 멤버 또는 3rd 파티 채널별로 이벤트 알림 수신 범위를 다음 네 가지 모드 중에서 선택할 수 있습니다. 수신 모드는 각 멤버 또는 채널 행의 **이벤트 수신 태그** 열에서 라디오 버튼으로 선택하세요.

수신 모드	설명
모든 알림 받기	이벤트 수신 태그와 무관하게 모든 경고 알림 수신
선택한 태그만 받기	선택한 이벤트 수신 태그를 가진 경고 알림만 수신. 태그를 최소 한 개 이상 선택
선택한 태그만 제외하기	선택한 이벤트 수신 태그를 가진 경고 알림만 제외하고 나머지 전부 수신. 제외 태그를 최소 한 개 이상 선택
모든 알림 받지 않기	어떤 이벤트 알림도 받지 않음. 선택해 둔 태그는 삭제되지 않고 그대로 보존

- 이벤트 수신 태그가 지정되지 않은 경고 알림까지 받으려면 태그 선택 영역에서 **미분류** 가상 태그를 선택하세요. **미분류** 태그는 일반 태그와 동일하게 선택하거나 해제할 수 있으며, 태그 자체를 수정하거나 삭제할 수는 없습니다.

> ⓘ **모든 알림 받지 않기** 모드에서는 해당 멤버 또는 채널이 이벤트 알림을 전혀 받지 않습니다. 이때 선택해 둔 태그는 삭제되지 않고 보존되므로, 다시 **선택한 태그만 받기** 모드로 전환하면 이전에 선택한 태그 설정을 그대로 사용할 수 있습니다.

선택한 태그만 받기

1. **사용자별 이벤트 수신 설정** 섹션에서 설정할 멤버 또는 3rd 파티 채널 행의 **이벤트 수신 태그** 열을 클릭하세요.
2. **선택한 태그만 받기** 라디오 버튼을 선택하세요.
3. 태그 선택 영역에서 알림을 받을 태그를 하나 이상 선택하세요.

> ⓘ **선택한 태그만 받기** 모드에서 태그를 하나도 선택하지 않았을 때 나타나는 완전 차단 경고는 **선택한 태그만 제외하기** 모드에는 표시되지 않습니다. 제외 태그가 없는 상태는 알림을 전부 차단하는 상태가 아니라 전부 수신하는 상태이기 때문입니다.

선택한 태그만 제외하기

경고 알림을 대부분 받되 특정 분류만 걸러 내고 싶을 때 사용합니다. 받을 태그를 하나씩 고르는 **선택한 태그만 받기**와 방향이 반대입니다.

1. **사용자별 이벤트 수신 설정** 섹션에서 설정할 멤버 또는 3rd 파티 채널 행의 **이벤트 수신 태그** 열을 클릭하세요.
2. **선택한 태그만 제외하기** 라디오 버튼을 선택하세요.
3. 아래 태그 선택 영역에서 알림을 받지 않을 태그를 하나 이상 선택하세요.
    - 이벤트에 붙은 태그와 제외 태그가 하나라도 겹치면 해당 경고 알림은 전송되지 않습니다. 겹치는 태그가 없는 알림은 모두 수신합니다.
    - 제외 태그는 최소 한 개 이상 선택해야 합니다. 제외 태그가 하나도 없는 상태는 **모든 알림 받기**와 결과가 같아지므로, 마지막 태그는 제거할 수 없고 제외 태그가 비어 있는 동안 **최소한 하나의 태그를 선택해 주세요.** 안내가 표시됩니다.
    - 제외 태그 목록은 **선택한 태그만 받기** 모드의 수신 태그 목록과 별도로 저장됩니다. 두 모드를 번갈아 선택해도 각 모드에서 고른 태그가 그대로 남습니다.
    - **미분류** 가상 태그도 제외 대상으로 선택할 수 있으며, 태그 한 개로 계산됩니다.

알림 수신 언어 선택

사용자의 알림 수신 언어를 선택할 수 있습니다.

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **사용자별 이벤트 수신 설정** 섹션의 사용자 목록에서 언어를 설정할 사용자의 가장 왼쪽에 ▶ 버튼을 클릭하세요.
3. **알림 수신 언어**를 선택하세요.

요일 및 시간별 알림 설정

사용자의 요일별, 시간별 알림 수신 여부를 선택할 수 있습니다.

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **사용자별 이벤트 수신 설정** 섹션의 사용자 목록에서 요일 및 시간별 알림을 설정할 사용자의 가장 왼쪽에 ▶ 버튼을 클릭하세요.
3. 경고 알림 수신을 원하는 요일을 선택하거나 시간을 입력하세요.
    - 알림 수신 수단별로 설정할 수 있습니다.

3rd 파티 플러그인

외부 애플리케이션을 통해 경고 알림을 받을 수 있습니다.

- 지원 애플리케이션: Slack, Telegram, Teams, Teams Workflows, Jandi, AlertNow, Webhook, Webhook JSON, Opsgenie, PagerDuty, LINE, ilert

> ⓘ 와탭랩스의 지원 범위에 포함하지 않는 사내 메신저는 표준 Webhook, Webhook json을 통해 연동할 수 있습니다.

표 | 3rd 파티 플러그인 목록 구성

항목	설명
플러그인 이름	플러그인의 이름

항목	설명
인증 키	인증 키
인증 값	인증값
수신 레벨	경고, 전체, 위험으로 구분
반복 알림(에스컬레이션)	알림 발생 시간으로부터 알림 발생 상황이 해소되지 않았을 경우, 최초 알림 발생 시각으로부터의 알림 재발행 주기 - 위험(Critical) 단계의 알림에만 적용되는 기능
Smart 메시지(AI)	Telegram 3rd 파티 플러그인 추가 시 AI 알림 기능 제공 - 지원 언어: 한국어, 영어, 일본어
이벤트 수신 태그	프로젝트의 멤버 중 특정 멤버 또는 팀을 대상으로 알림 수신 여부
삭제	3rd 파티 플러그인 삭제

3rd 파티 플러그인 추가

1. **경고 알림** > **이벤트 수신 설정**으로 이동하세요.
2. **3rd 파티 플러그인** 섹션의 [추가하기] 버튼을 클릭하세요.
3. **서드 파티 플러그인 추가** 창에서 원하는 서비스 탭으로 선택하세요.
4. 선택한 서비스의 안내(3rd 파티 플러그인 알림 추가 방법)에 따라 설정을 진행하세요.
5. [수신 테스트] 버튼을 클릭해 설정이 잘 되었는지 확인하세요.
6. 모든 과정을 완료하면 [추가하기] 또는 [등록] 버튼을 클릭하세요.

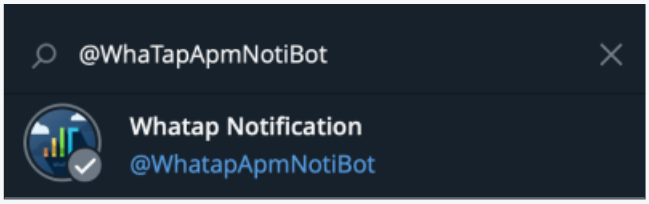
3rd 파티 플러그인 알림 추가 방법

Slack

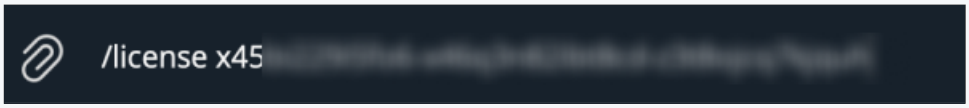
> **서드 파티 플러그인 추가**의 **Slack** 탭 화면의 [**슬랙에 추가하기**] 버튼을 클릭하면 Slack의 팀 및 채널에 **WhaTapNotiBot**을 등록할 수 있습니다.

Telegram

1. **Telegram**에서 **@WhaTapApmNotiBot**을 검색하세요.



2. Bot을 선택하고 [시작] 버튼을 클릭하세요.
3. 채팅창에 `/license` 명령어와 알림을 받을 프로젝트의 액세스 키를 채팅창에 입력하세요.

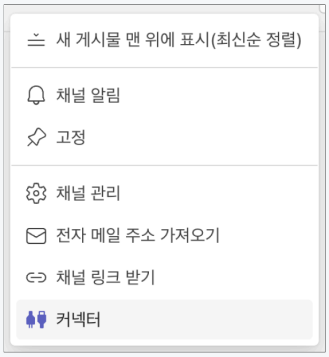


사용할 수 있는 명령어 목록

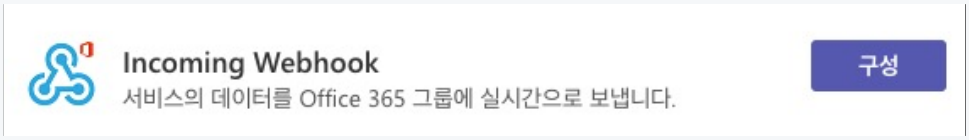
- `/help` : 명령어 목록 보기
- `/license` : 알림 받을 프로젝트의 액세스 키 입력
- `/remove` : 알림을 받지 않을 프로젝트 코드 입력
- `/list` : 현재 알림을 받고 있는 프로젝트 코드 보기

Teams

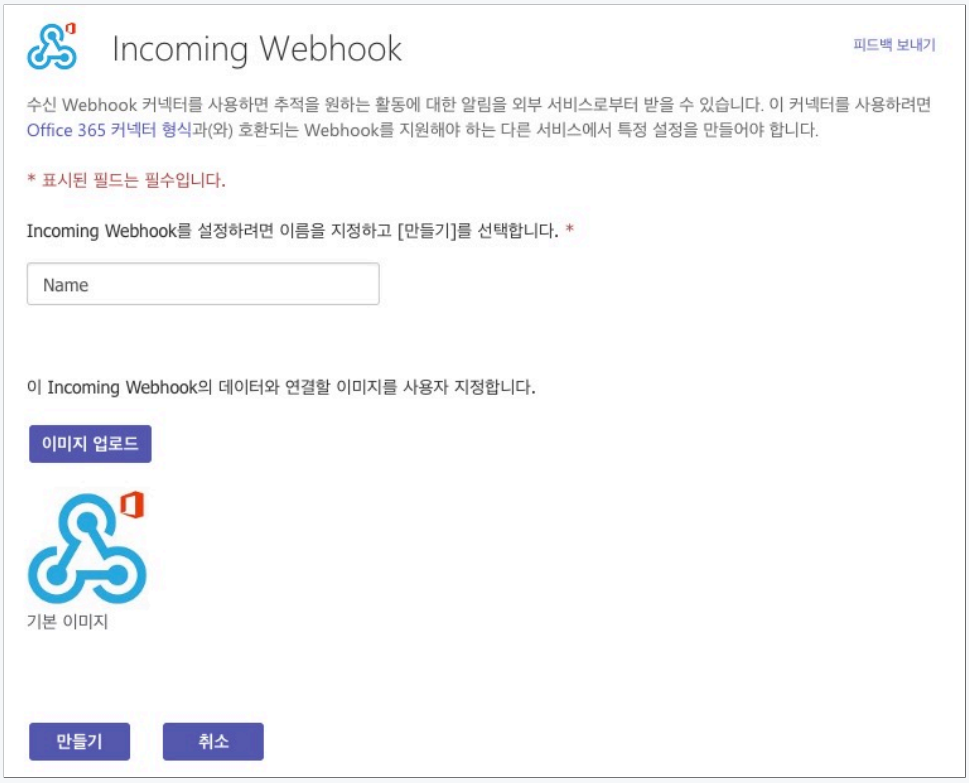
1. 알림을 받을 채널의 **커넥터**로 이동하세요.



2. **Webhook**을 추가하고 [구성] 버튼을 클릭하세요.



3. Webhook 이름을 입력하고 [만들기] 버튼을 클릭하세요.



4. 생성된 Webhhok URL을 복사하세요.

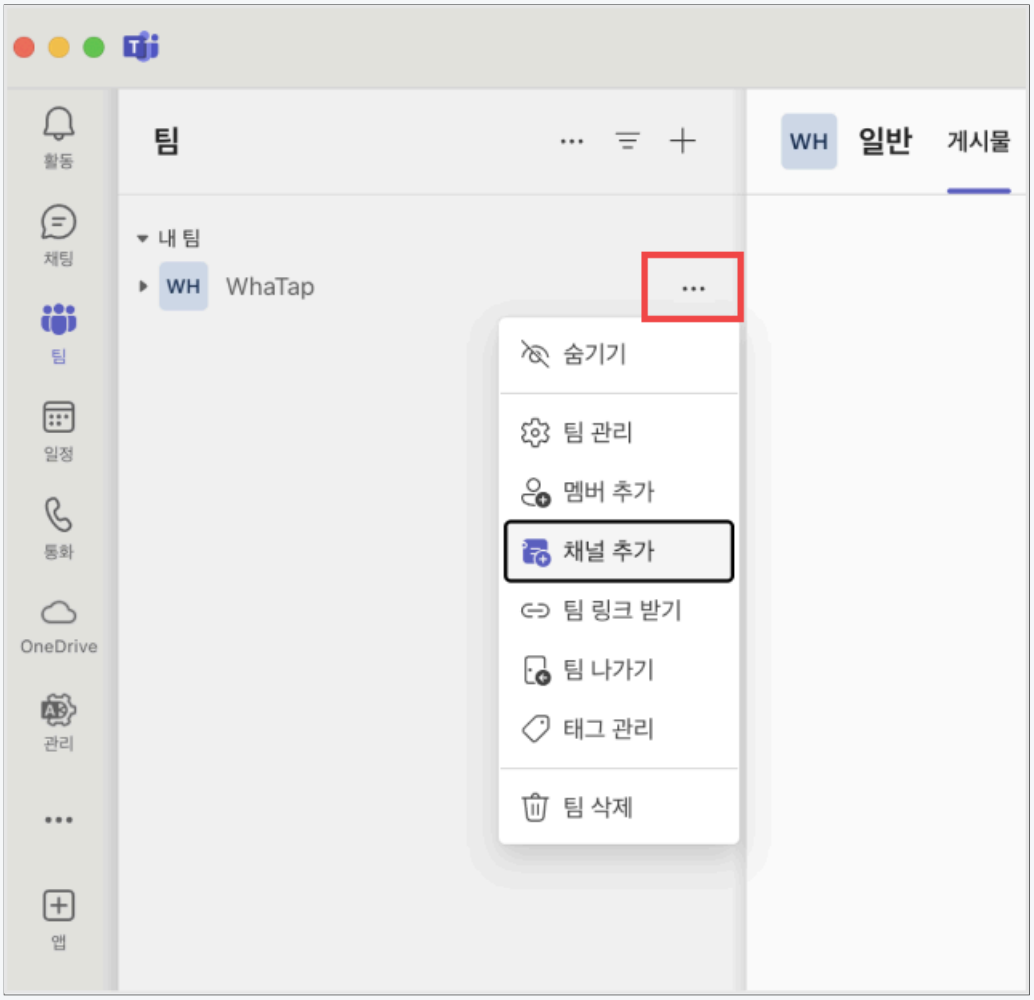


5. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **Teams** 탭 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
6. [**등록**]버튼을 클릭하세요.

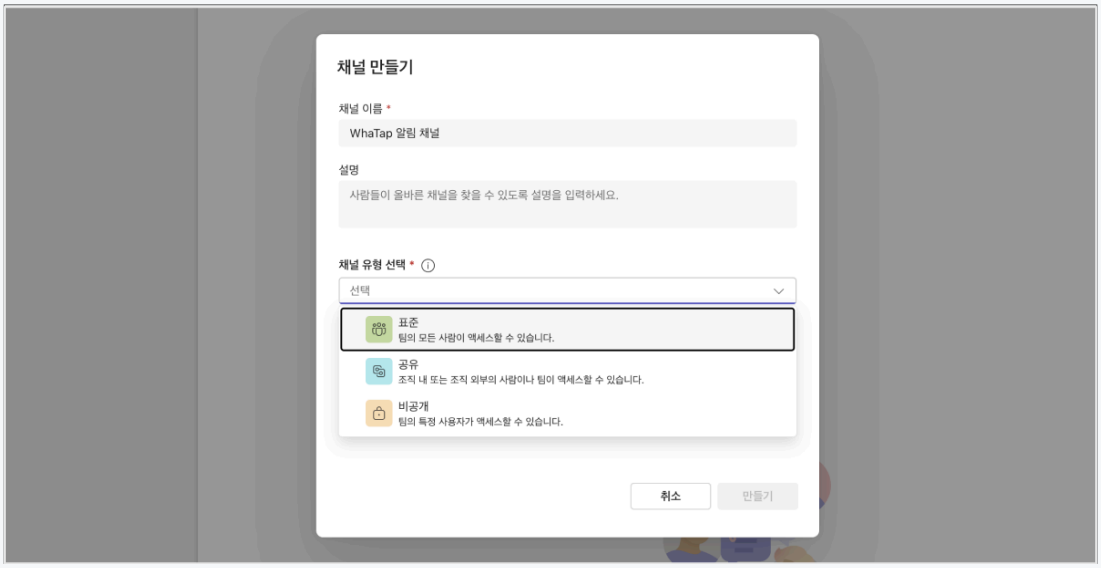
Teams Workflows

워크플로를 사용하려면 **Microsoft Teams**에 액세스할 수 있는 계정과 비즈니스 플랜 구독이 필요합니다.

1. **팀** 메뉴의 더보기(...) > **채널 추가**를 클릭하세요.

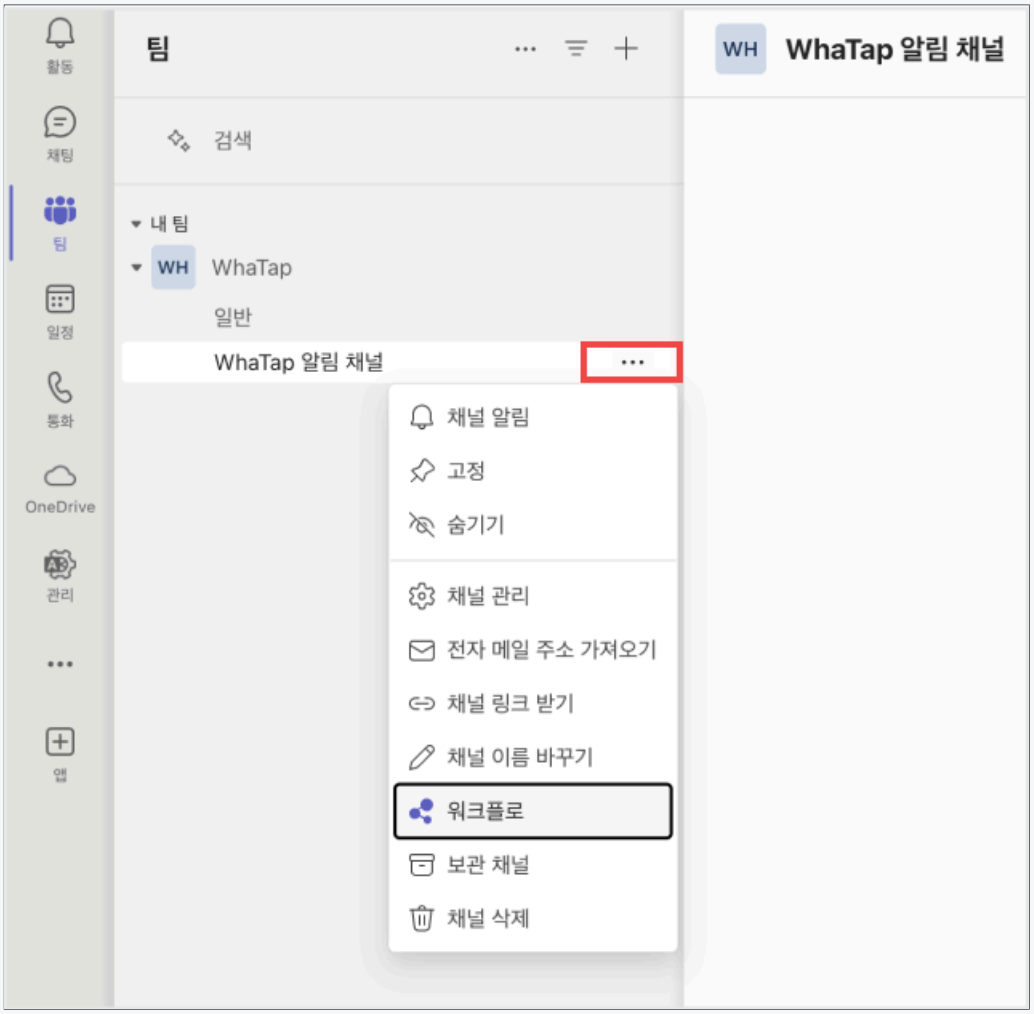


2. **채널 만들기** 창에서 **채널 이름**과 **설명**을 입력하고, **채널 유형 선택** 항목은 **표준**으로 선택한 후, [만들기] 버튼을 클릭하세요.



> 채널 유형을 **공유**로 선택하면 워크플로를 사용할 수 없으며, **비공개**를 선택하면 워크플로를 추가할 수 있지만 알림을 수신할 수 없습니다.

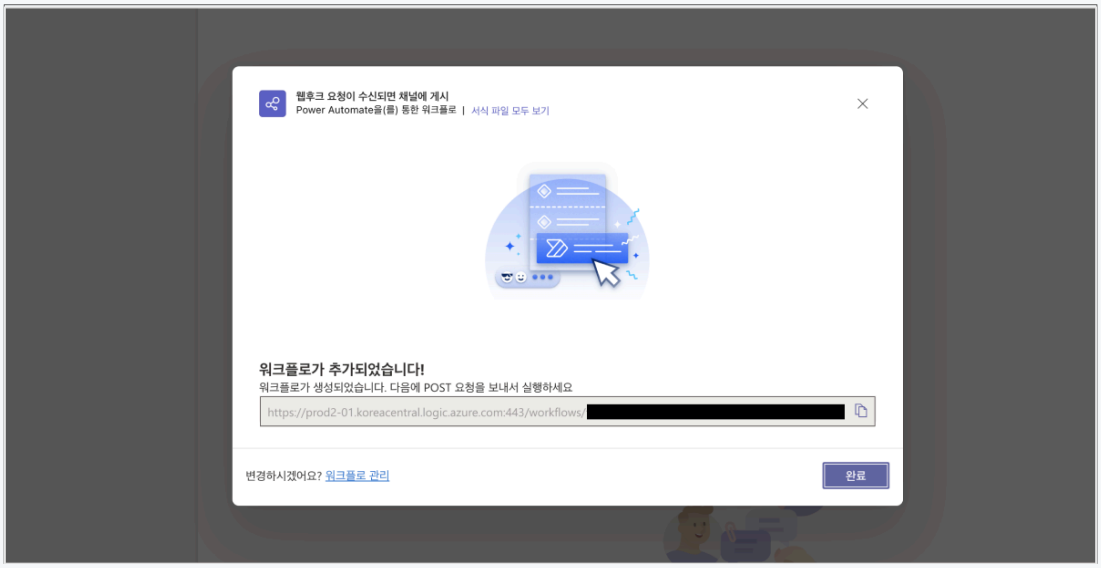
3. 생성한 채널의 더보기(...) > **워크플로**를 클릭하세요.



4. **워크플로** 대화상자의 **워크플로**의 목록에서 **웹후크 요청이 수신되면 채널에 게시**를 선택 후, [추가 워크플로] 버튼을 클릭하세요.



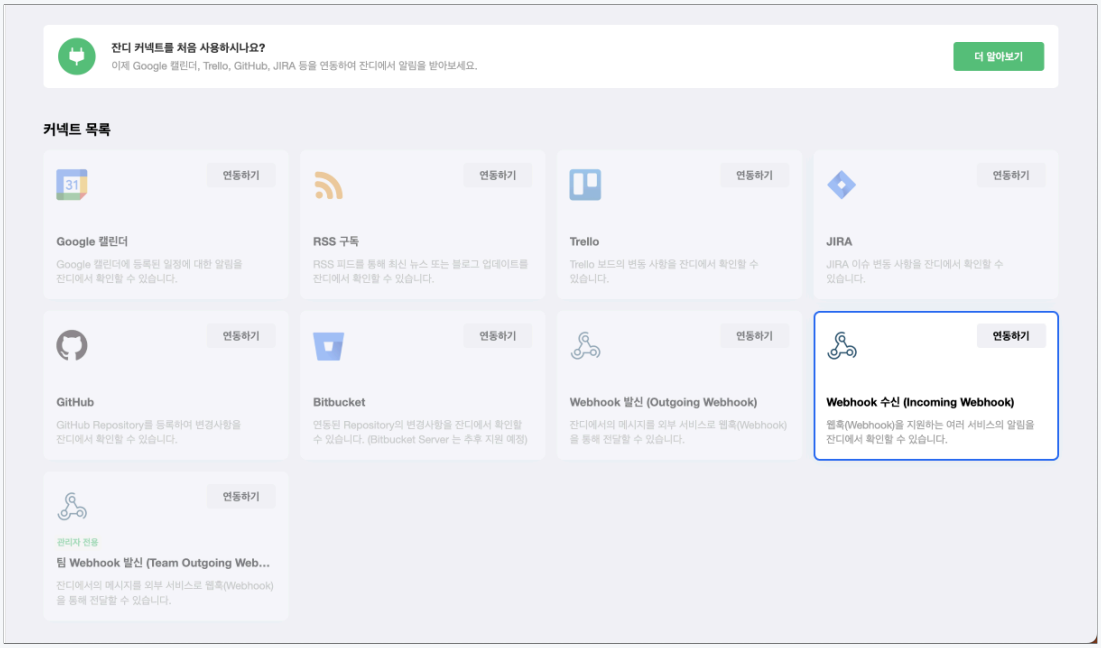
5. **이름**과 **연결** 상태를 확인한 후, [다음] 버튼을 클릭하세요.
6. **세부 정보**를 확인한 후, [워크플로 추가] 버튼을 클릭하세요.
7. 워크플로가 추가되었습니다. 생성된 주소를 복사하세요.



8. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **Teams Workflows** 탭 안내 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
9. [**등록**] 버튼을 선택하세요.

잔디(Jandi)

1. 알림을 받을 토픽 또는 채팅에서 잔디 커넥트 설정으로 이동하세요.
2. **커넥트 목록**에서 **Webhook 수신 (Incoming Webhook)** 항목에서 [연동하기] 버튼을 클릭하세요.



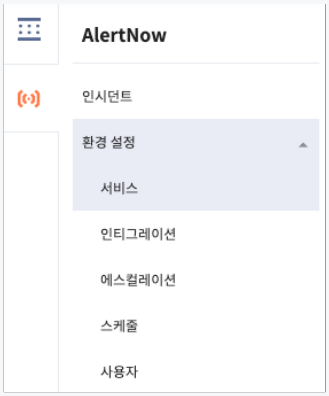
3. [Webhook 수신 추가] 버튼을 클릭하세요.
4. **Webhook URL 등록** 항목에 Webhook URL을 복사하세요.



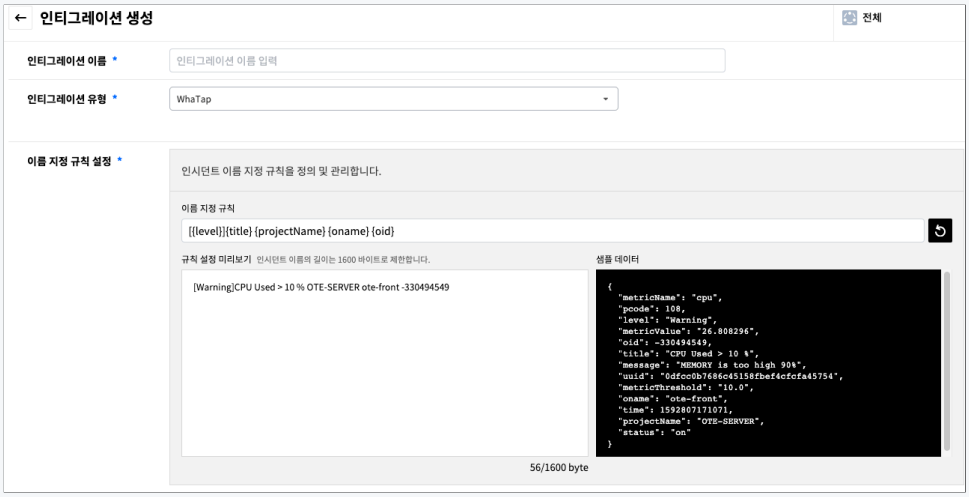
5. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **Jandi** 탭 안내 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
6. [**등록**] 버튼을 클릭하세요.

AlertNow

1. **AlerNow** 서비스를 접속한 후, **환경 설정** > **Integration** 메뉴로 이동하세요.



2. **인티그레이션** 화면에서 [인티그레이션 생성] 버튼을 클릭하세요.
3. Integration 유형 중 **WhaTap**을 선택하세요.
4. **인티그레이션 생성** 화면에서 이름 및 이름 지정 규칙을 설정하세요.



5. [확인] 버튼을 클릭하면 Integration이 생성됩니다.
6. 생성된 인티그레이션 목록을 확인하고 생성된 Integration 항목을 선택하세요.



7. 상세 화면에서 URL 오른쪽의 복사 아이콘을 클릭해 URL을 복사하세요.



8. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와 **AlertNow** 탭 안내 하단의 **Channel Name**과 **Webhook URL** 항목을 입력하세요.
9. [**등록**] 버튼을 클릭하세요.

Webhook

1. 알림을 받을 Webhook을 등록하세요. 등록한 URL에 POST Method로 알림의 `title` , `text` 가 전송됩니다.

```
{
    "title":"Alert Title",
    "text": "Sample warning message"
}
```

1. **Webhook Name**과 **Webhook URL** 항목을 입력하세요.
2. [**등록**] 버튼을 클릭하세요.

Webhook JSON

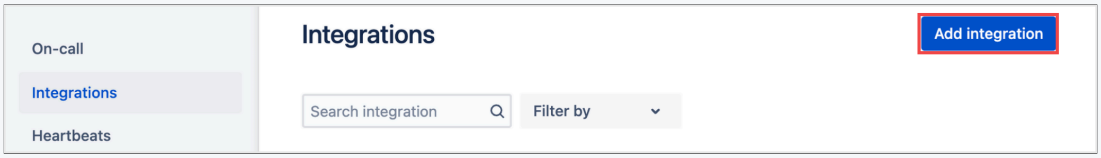
1. 알림을 받을 Webhook을 등록하세요. 등록한 URL에 POST Method로 알림 데이터가 JSON 형태로 전송됩니다. JSON 내에 포함 시킬 수 있는 대체 변수에 대해선 **사용 가능 변수** 부분을 참고하세요.
   - `${something}` 은 와탭에서 제공하는 대체될 값입니다. 지원하는 전체 목록과 의미는 사용 가능 변수를 참고하세요.
   - 이벤트 상태(`status`)는 이벤트 발생(`on`), 이벤트 해소(`off`)로 구분할 수 있습니다.
   - `level` 은 `None` , `Info` , `Warning` , `Critical` 로 구분합니다.

```
{
  "pcode": "${pcode}",
  "projectName": "${projectName}",
  "time": "${time}",
  "oid": "${oid}",
  "oname": "${oname}",
  "title": "${title}",
  "message": "${message}",
  "level": "${level}",
  "status": "${status}",
  "metricName": "${metricName}",
  "metricValue": "${metricValue}",
  "metricThreshold": "${metricThreshold}",
  "uuid": "${uuid}",
  "okind": "${okind}",
  "onode": "${onode}"
}
```

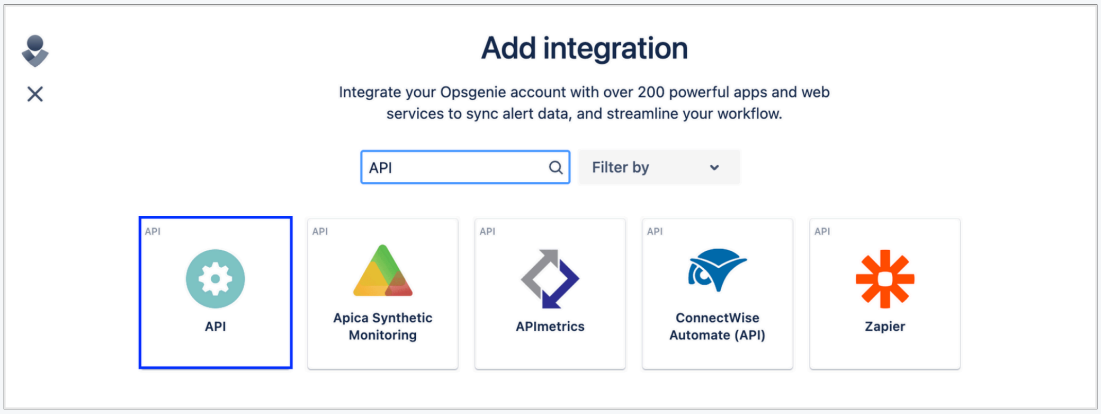
2. **Webhook Name**과 **Webhook URL** 항목을 입력하세요.
3. Header 값을 추가하려면 **사용자 Header 값 추가**의 **+ 추가**를 클릭해 key, value을 추가하세요.
4. [**등록**] 버튼을 클릭하세요.

Opsgenie

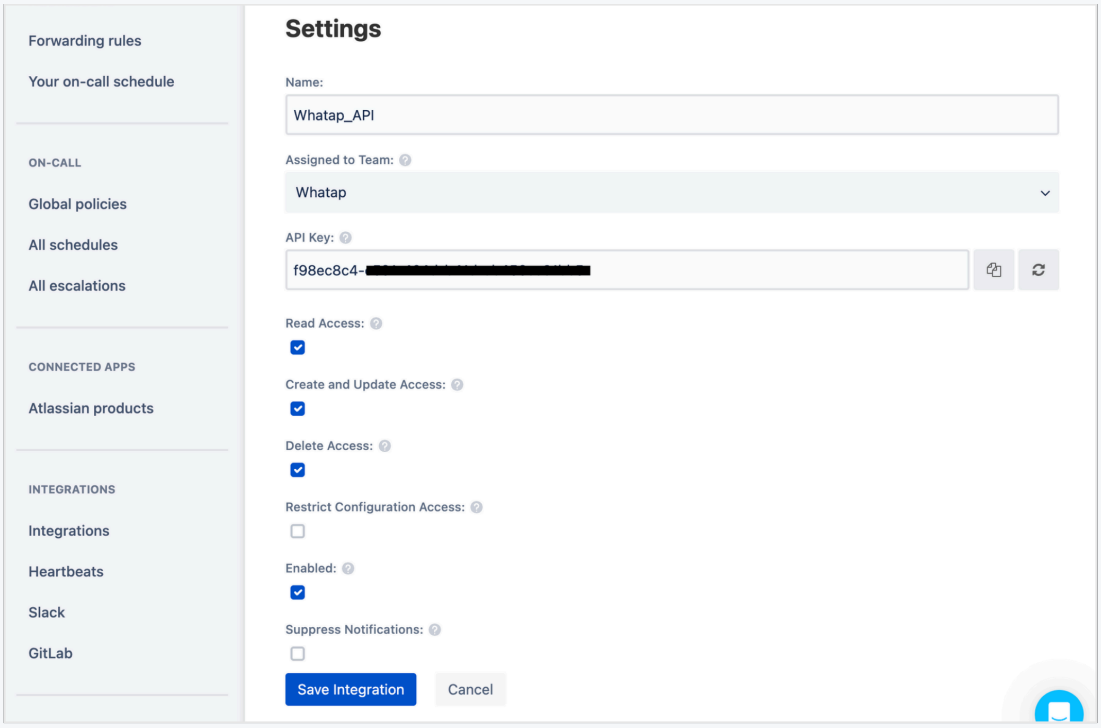
1. **Teams** 메뉴에서 알림을 수신할 팀을 선택하세요.
2. **Integration** 메뉴로 이동해 [Add Integration] 버튼을 클릭하세요.



3. Integration 목록에서 'API' 검색 후 선택하세요.



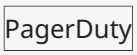
4. API 명과 Team을 할당하고 각 권한을 설정한 후, [Save Integration] 버튼을 클릭하세요.



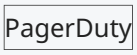
5. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 창으로 돌아와, **Opsgenie** 탭 안내 하단의 **Name**과 **API Key** 항목을 입력하세요.
6. [**등록**] 버튼을 선택하세요.

PagerDuty

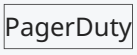
1. PagerDuty 계정에 등록하고 로그인하세요.
2. PagerDuty에서 **Services** > **Service Directory** 메뉴에서 [+ New Service] 버튼을 클릭하세요.

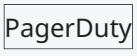
3. Service 정보(**Name**)를 입력한 후 [Next] 버튼을 클릭하세요.

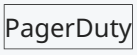
4. Escalation Policy를 설정 후 [Next] 버튼을 클릭하세요.

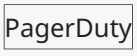
5. **Alert Grouping**과 **Transient Alerts** 항목을 설정한 후 [Next] 버튼을 클릭하세요.

6. **Integrations**에서 **Event API V2**를 선택한 후 [Create Service] 버튼을 클릭하세요.

7. 생성한 Service 화면에서 **Integration Name**과 **Integration Key** 값을 복사하세요.

8. 와탭 모니터링 서비스의 **서드 파티 플러그인 추가** 화면으로 돌아와 **Integration Name**과 **Integration Key** 항목을 입력하세요.
9. [등록] 버튼을 클릭하세요.

> 메시지를 수신하지 못했거나 플러그인이 등록되지 않았다면 **Integration Name** 또는 **Integration Key**값이 올바른지 확인하세요.

LINE

1. LINE 앱을 설치합니다.
2. 아래 QR 코드를 스캔하여 **WhaTapLabs** 공식 채널을 친구로 추가합니다.



3. 다음 명령어를 사용하여 프로젝트를 라인과 연동하고 알림을 설정하세요.

명령어

```
/help # 사용 가능한 명령어 안내
/list # 연동된 프로젝트 목록 조회
/license [액세스키] # 새 프로젝트 연동 추가
/remove [프로젝트코드] # 프로젝트 연동 해제
/language [ko|ja|en] # 알림 언어 설정 (한국어/일본어/영어)
```

- 프로젝트 액세스 키는 **관리 > 프로젝트 관리**에서 확인하실 수 있습니다.
- 라인 내에서 `/language` 명령어를 사용하는 것은 라인 내 채팅방에만 적용됩니다. 다른 3rd 플러그인이나 프로젝트의 알림 수신 언어가 변경되는 것은 아닙니다.

사용 예시

```
/license XXXXXXXXXX # 프로젝트 연동
/language ko # 한국어로 언어 설정
```

```
/list # 연동된 프로젝트 확인
```

- 프로젝트 액세스 키: WhaTap 프로젝트 내 **관리 > 프로젝트 관리**에서 확인할 수 있음
- 언어 설정: `/language` 명령어는 LINE 채팅방 내에서만 적용되며, 다른 알림 채널의 언어는 변경되지 않음

ilert

1. ilert 콘솔에서 **Alert sources** 화면의 오른쪽 위의 [+ Create alert source]를 클릭하세요.
2. **Integration type 단계**의 하단에 있는 **Filter integrations 검색창**에 `WhaTap` 을 입력해 검색합니다.
3. 검색 결과에서 **WhaTap 타일**을 선택한 후 [Next] 버튼을 클릭합니다.
4. 알림 소스 이름을 입력하고 팀을 지정한 후 [Next] 버튼을 클릭합니다.
   - **에스컬레이션 정책은** 새 정책을 생성하거나 기존 정책을 선택할 수 있습니다.
5. 알림 그룹화 방식을 선택한 후 [Continue setup] 버튼을 클릭합니다.
   - 우선 선택한 **Do not group alerts는** 이후 변경할 수 있습니다.
6. 다음 화면에서 **알림 템플릿, 우선순위 등 추가 설정**을 할 수 있습니다. 우선 [Finish setup] 버튼을 클릭해 설정을 완료합니다.
7. 설정이 완료되면 완료 화면에서 **API 키** 또는 **Webhook URL**이 생성됩니다.
8. **Webhook JSON** 탭에서 생성된 **Webhook URL**을 복사해 아래의 **Webhook URL 입력란**에 붙여넣습니다.
9. [등록] 버튼을 클릭하세요.

대량 알림 발생 방지

알림이 대량으로 발생하면 지정한 시간 동안 알림이 일시적으로 중지됩니다. 예를 들어, 5분 동안 20회의 이벤트가 발생하면 5분

동안 경고 알림을 중지합니다. 설정한 **정지 시간** 시간이 지나면 대량 알림 발생 방지 기능은 해제됩니다.

1. **경고 알림** > **이벤트 수신 설정**의 **대량 알림 발생 방지** 섹션으로 이동하세요.
2. 토글 버튼을 클릭해 **활성화** 또는 **비활성화** 할 수 있습니다.
   - **활성화**: 대량 알림 일시적 중지
   - **비활성화**: 대량 알림 재개
3. **탐지 시간, 탐지 횟수, 정지 시간**을 설정하세요.
   - **탐지 시간** 동안 **탐지 횟수** 이상의 이벤트가 발생하면 **정지 시간** 동안 경고 알림을 중지합니다.

> ⓘ 대량 알림으로 차단되는 경우 차단 안내 박스가 표시됩니다. 대량 알림 차단 기능을 해제하려면 박스의 [대량 알림 발생 중지] 버튼을 클릭하세요.

이벤트 수신 포맷

공용 알림 템플릿

공통 템플릿은 대부분의 경우 적용 가능합니다. 다양한 환경에서 동일한 포멧으로 알림을 받을 수 있습니다.

- Product Type: 애플리케이션, 데이터베이스, 쿠버네티스
- Event Type: 애플리케이션 알림, 데이터베이스 알림, 메트릭스 알림
- Event Channel: sms, mobile, 3rd party plugin, plugin

이벤트 제목

Event title format

```
[Level][Platform][ProjectName][ApplicationName][EventTitle]
```

Event title example

```
[Info][JAVA][애플리케이션 프로젝트][TC-0-1-8081][CRITICAL_HIGH_MEMORY]
```

> ⓘ ApplicationName을 설정하지 않은 경우 이벤트 제목에서 생략합니다.
>
> - Platform은 다음 중 한 가지로 표시합니다.
>   - JAVA
>   - NODEJS
>   - PYTHON
>   - PHP
>   - DOTNET
>   - GO
>   - POSTGRESQL
>   - ORACLE

- MYSQL
- MSSQL
- BSM_JAVA
- CLOUDWATCH
- TIBERO
- KUBERNETES
- KUBE_NS
- URLCHECK
- URLCHECK_ADMIN
- CUBRID
- ALTIBASE
- CLUSTER
- REDIS
- MONGODB
- VR
- RUM

이벤트 메시지

이벤트 메시지에 포함할 수 있는 정보입니다. `optional` 이 `false` 인 경우 항상 메시지에 포함합니다. `optional` 이 `true` 인 경우 해당 데이터를 확인할 수 있으면 표시합니다.

Event message example

```
Project Name : 애플리케이션 프로젝트
Project Code : 3
Agent Name : TC-0-1-8081
Message : RECOVERED: Memory is too high. less than 10%
Event Time : 2022-04-12 18:53:24 +0900
Event OFF Time : 2022-04-12 18:53:24 +0900
Alert Type : APPLICATION_MEMORY
```

```
Metric Name : memory
Metric Value : 20
Metric Threshold : 10
Stateful : true
```

표 | 이벤트 메시지 구성 요소

En	Ko	지원되는 알림 타입	설명
Project Name	프로젝트 이름	전체	-
Project Code	프로젝트 Code	전체	-
Agent Name	에이전트 이름	전체(optional)	oname
Message	이벤트 메시지	전체	-
Alert Type	이벤트 종류	전체	아래의 AlertType 종류 표 참고
Event Time	이벤트 발생 시간	전체	2022-04-13 10:40:49 +0900에서 +0900는 GMT를 의미합니다.
Event Off Time	이벤트 해제 시간	전체(optional)	2022-04-13 10:40:49 +0900에서 +0900는 GMT를 의미합니다.
Metric Name	메트릭스 이름	전체(optional)	이벤트 조건 판단에서 사용하는 메트릭스의 이름
Metric Value	메트릭스 값	전체(optional)	메트릭스 값이 메트릭스 임계치를 넘으면 이벤트 발생 조건이 만족한 경우입니다.
Metric Threshold	메트릭스 임계치	전체(optional)	메트릭스 값이 메트릭스 임계치를 넘으면 이벤트 발생 조건이 만족한 경우입니다.
Stateful	해소된 이벤트 알림	전체(optional)	해소된 이벤트 알림 기능 사용 중이면 true, 아니면 false
Event Rule	이벤트 발생 조건	메트릭스 알림	-

En	Ko	지원되는 알림 타입	설명
Target Filter	이벤트 대상 선택	메트릭스 알림	특정 대상에서 수집된 메트릭스에 대해서만 이벤트 조건을 확인합니다.
Repeat Count	이벤트 반복 횟수	메트릭스 알림	이벤트 조건이 이벤트 반복 시간동안 이벤트 반복 횟수만큼 만족해야 이벤트가 발생됩니다.
Repeat Duration	이벤트 반복 시간	메트릭스 알림	이벤트 조건이 이벤트 반복 시간동안 이벤트 반복 횟수만큼 만족해야 이벤트가 발생됩니다.
Receiver	수신자	메트릭스 알림	-
Query	MXQL 쿼리	복합 메트릭스 알림	-
Rule	이벤트 발생 조건	복합 메트릭스 알림	-
Query Period	쿼리 기간	복합 메트릭스 알림	-
Query Interval	쿼리 간격	복합 메트릭스 알림	-
Silent Time	무음 시간	복합 메트릭스 알림	-
Query	URL	Exception 알림	Exception을 발생시킨 요청의 URL
TXID	트랜잭션 ID	Exception 알림	-
Class	에러 클래스 이름	Exception 알림	-

En	Ko	지원되는 알림 타입	설명
Log Message	로그 메시지	서버 - 파일 로그 알림	-
Log File	로그 파일 경로	서버 - 파일 로그 알림	-
IP	IP	서버 알림 전체	-
CPU	CPU	서버 알림 전체	이벤트 발생 당시의 Snapshot
CPU_load1	CPU_load1	서버 알림 전체	이벤트 발생 당시의 Snapshot
CPU_loadPerCore	CPU_loadPerCore	서버 알림 전체	이벤트 발생 당시의 Snapshot
Memory	Memory	서버 알림 전체	이벤트 발생 당시의 Snapshot
Swap	Swap	서버 알림 전체	이벤트 발생 당시의 Snapshot
Disk Name | Used Percent | Free Size | IO Percent	디스크 퍼포먼스	서버 알림 전체	이벤트 발생 당시의 Snapshot
Name | Bps | Pps	트래픽 퍼포먼스	서버 알림 전체	이벤트 발생 당시의 Snapshot
Message | Time | Name	처리내역 메시지	서버 알림 전체	-

ⓘ 해당 이벤트에서 제공할 수 있는 최대한 많은 정보를 보여줍니다.

- AlertType은 다음 중 한 가지로 표시합니다.

AlertType	설명
APPLICATION_CPU	애플리케이션 CPU 알림
APPLICATION_MEMORY	애플리케이션 MEMORY 알림

AlertType	설명
APPLICATION_DISK	애플리케이션 DISK 알림
APPLICATION_ACTIVE_TRANSACTION	애플리케이션 액티브 트랜잭션 알림
APPLICATION_ERROR_TRANSACTION	애플리케이션 에러 트랜잭션 알림
APPLICATION_SLOW_TRANSACTION	애플리케이션 트랜잭션 응답시간 알림
METRICS	메트릭스 알림
COMPOSITE_METRICS	복합 메트릭스 알림
ANOMALY	이상치 탐지 알림
LOG_REALTIME	로그 실시간 알림
COMPOSITE_LOG	복합 로그 알림
SERVER_REBOOT	서버 - 재시작 알림
SERVER_NO_DATA	서버 - 미수신 알림
SERVER_PORT	서버 - 포트 알림
SERVER_NETWORK_IOPS	서버 - 네트워크 IOPS 알림
SERVER_NETWORK_BPS	서버 - 네트워크 BPS 알림
SERVER_DISK_IO	서버 - 디스크 I/O 알림
SERVER_DISK_QUOTA	서버 - 디스크 사용량 알림
SERVER_DISK_INODE	서버 - inode 알림

AlertType	설명
SERVER_CPU	서버 - CPU 알림
SERVER_MEMORY	서버 - 메모리 알림
SERVER_CPU_STEAL	서버 - steal 알림
SERVER_MEMORY_SWAP	서버 - 스왑 알림
SERVER_LOG_FILE	서버 - 로그 파일 알림
SERVER_WINDOW_EVENT	서버 - 윈도우 이벤트 알림
SERVER_OFF	서버 - 알림 OFF 알림
SERVER_ACKNOWLEDGE	서버 - 처리내역 알림
SERVER_PROCESS_COUNT	서버 - 프로세스 수 알림
SERVER_PROCESS_CPU	서버 - 프로세스 CPU 알림
SERVER_PROCESS_MEMORY	서버 - 프로세스 메모리 알림
SERVER_PROCESS_OFF	서버 - 프로세스 알림 OFF 알림
AGENT_ACTIVE	에이전트 활성화 알림
AGENT_INACTIVE	에이전트 비활성화 알림
AGENT_REACTIVATED	에이전트 재활성화 알림
URL	URL 알림
TOO_MANY_EVENT	너무 많은 이벤트 발생 알림

AlertType	설명
CLOUD_WATCH	Cloud Watch 알림
EXCEPTION	Exception 알림

이벤트 기록 및 모니터링

홈 > 프로젝트 선택 > **경고 알림** > **이벤트 기록**

이벤트 기록은 프로젝트에서 발생한 여러 이벤트를 조회하고 상세 정보를 분석할 수 있습니다. Elasticsearch DSL 쿼리를 지원하며, 원하는 조건에 맞는 이벤트를 정확하고 빠르게 찾을 수 있습니다.

이벤트 기록 조회하기

검색 결과에서 **이벤트 제목을 클릭**하면 해당 이벤트가 발생한 시간대의 상세 데이터를 분석할 수 있는 화면으로 이동합니다. 이를 통해 이벤트 발생 전후 상황을 종합적으로 파악할 수 있습니다.

1. **경고 알림** > **이벤트 기록**으로 이동하세요.
2. **이벤트 기록** 화면의 **시간 선택**에서 이벤트 기록을 조회할 기간를 설정하세요.
3. 필터 입력 창에 쿼리을 입력하고 [돋보기] 버튼을 클릭하세요.
    - 쿼리를 입력하지 않으면 전체 이벤트가 조회됩니다.

컬럼 설정

원하는 컬럼을 표시하거나 숨길 수 있습니다.

1. **이벤트 기록** 화면의 왼쪽 위 [컬럼 설정] 버튼을 클릭하세요.
2. **컬럼 설정** 창에서 표시하거나 숨길 컬럼을 선택하세요.
    - **표시 항목**에서 컬럼의 순서를 변경할 수 있습니다.
    - **컬럼명 검색**창에 원하는 컬럼 항목을 검색할 수 있습니다.

컬럼 항목	설명
이벤트 이름	이벤트의 제목
메시지	이벤트의 메시지 또는 스냅샷 정보

컬럼 항목	설명
이벤트 상태	이벤트의 현재 상태
에이전트 명	이벤트가 발생한 대상 에이전트의 이름(Oname) - 특정 에이전트에 대해 발생한 이벤트가 아닌 경우 빈칸으로 표시됨
에이전트 그룹	이벤트가 발생한 대상 에이전트의 종류(OkindName) - 특정 에이전트에 대해 발생한 이벤트가 아닌 경우 빈칸으로 표시됨
에이전트 노드	이벤트가 발생한 대상 에이전트의 서버(OnodeName) - 특정 에이전트에 대해 발생한 이벤트가 아닌 경우 빈칸으로 표시됨
이벤트 발생 시각	이벤트가 발생한 시간
이벤트 해소 시각	이벤트가 해소된 시간 상태가 없는 이벤트의 경우 `-`로 표시됨

3. 설정을 완료한 후, [확인] 버튼을 클릭해 컬럼 설정을 반영합니다.

CSV 다운로드

조회 결과를 CSV 파일 형식으로 다운로드할 수 있습니다.

1. **경고 알림** > **이벤트 기록**에서 [CSV] 버튼을 클릭하세요.
2. 현재 필터 조건에 맞는 이벤트 목록이 CSV 파일로 저장됩니다.

진행 중인 이벤트만 보기

현재 진행 중인 상태의 이벤트만 조회할 수 있습니다.

1. **경고 알림** > **이벤트 기록**에서 [진행 중인 이벤트만 보기] 버튼을 클릭하세요.
2. 필터 쿼리에 다음 조건이 자동으로 추가되어 검색됩니다.

```
Stateful: true and Status : "ON"
```

이벤트 검색

- **정확한 검색**: 원하는 조건을 정확히 지정하여 필요한 이벤트만 조회
- **복합 조건**: 여러 필드를 동시에 검색하여 복잡한 조건도 한 번에 처리
- **유연한 패턴**: 와일드카드, 부분 검색 등 다양한 검색 방식 지원
- **빠른 성능**: 인덱싱된 데이터를 활용한 고속 검색

검색 가능한 필드

필드와 값에 대해 대소문자를 구분하지 않습니다.

- **필드명 태그**: 필드가 지원되는 이벤트 타입
- **필드 타입**: 필드의 데이터의 타입
  - `#`: Number
  - `T`: String
  - : Boolean
  - : Date

이벤트 정보 필드

필드명	타입	설명	예시 값
Title	String	이벤트 제목	"Database Connection Error"
OffTitle	String	이벤트 해소 제목	"RECOVERED: Database Connection Error"
Message	String	이벤트 메시지	"Connection timeout occurred"
OffMessage	String	이벤트 해소 메시지	"RECOVERED: Connection timeout occurred"
Level	String	현재 이벤트 레벨	`Critical`, `Warning`, `Info`

필드명	타입	설명	예시 값
OriginLevel	String	원본 이벤트 레벨	`Critical`, `Warning`, `Info`
Status	String	이벤트 상태	`ON`, `OFF`, `CANCEL`, `ACKNOWLEDGE`, `MAINTENANCE`, `DISABLED`
ActCount	Number	처리내역 개수	2
MetricName	String	지표 이름	"memory"
MetricValue	String	지표 값	"85.5"
OffValue	String	해소 값	"72.8"
MetricThreshold	String	임계치	"80"
alertType	String	이벤트 종류	"BASIC", "METRICS", "TRANSACTION" 등
alertId	String	이벤트 규칙의 고유식별자	"zf3uojer0fv4v7"

이벤트 타입별 지원 필드

> ⓘ 특정 이벤트 타입에서만 지원되는 필드입니다. 해당 이벤트 타입이 아닌 경우 값이 없을 수 있습니다.

필드명	타입	이벤트 종류	설명	예시 값
eventRule	String	기본, 메트릭스	이벤트 발생 규칙	"memory ≥ 80"
field	String	실시간 로그	로그 검색 키	"content"

필드명	타입	이벤트 종류	설명	예시 값
keyword	String	실시간 로그	로그 검색 값	"Error"
logCategory	String	실시간 로그	로그 카테고리	"AppLog"
logContent	String	실시간 로그	로그 내용	"00:00000:00009:2025/04/06 23:03:52.55 server Error: 1601, Severity: 17, State: 3\n00:00000:00009:2025/04/06 23:03:52.55 server There are not enough 'user connections' available to start a new process."

대상 필드

필드명	타입	설명	예시 값
Oid	Number	에이전트 고유식별자	-98765432
Oname	String	에이전트 이름	"web-server-01"
Okind	Number	에이전트 종류 고유식별자	867318026
OkindName	String	에이전트 종류명	"web-server"
Onode	Number	에이전트 서버 고유식별자	334634079
OnodeName	String	에이전트 서버명	"production-node-1"

이벤트 고유값 필드

필드명	타입	설명	예시 값
Id	Number	이벤트의 고유식별자	5768121

필드명	타입	설명	예시 값
UUID	String	이벤트 고유식별자	"550e8400-e29b-41d4-a716"

상태/플래그 필드

필드명	타입	설명	예시 값
Stateful	Boolean	상태 기반 이벤트 여부	true, false
Disabled	Boolean	종료된 이벤트 여부	true, false
Escalation	Boolean	에스컬레이션 여부	true, false
SystemEvent	Boolean	시스템 이벤트 여부	true, false

검색 유형별 문법

> ⓘ 검색 기본 문법은 WhaTap log search query 문법 문서를 참고하시기 바랍니다.

1. 키워드 검색

Database Connection 문자열이 포함된 이벤트

```
title: "Database Connection"
```

2. 여러 값 검색

여러 옵션 중 하나라도 포함되는 이벤트를 조회합니다.

```
title: Database Connection
```

현재 이벤트 레벨이 Info이거나 Warning인 이벤트

```
level: info warning
```

3. 패턴 검색

와일드카드를 사용하여 패턴을 검색할 수 있습니다.

이름 패턴으로 특정 에이전트에서 발생된 이벤트 찾기

```
oname: web-*-prod
```

제목이 Connection으로 끝나는 이벤트 찾기

```
title: *Connection
```

4. 복합 조건 검색

여러 조건을 조합하여 정확히 검색할 수 있습니다.

Critical 레벨이면서 제목에 Database가 포함된 이벤트

```
level: critical and title: Database
```

특정 에이전트의 Warning 또는 Critical 이벤트

```
oname: web-server-01 and level: Warning Critical
```

5. OR 조건 검색

여러 조건 중 하나라도 만족하는 이벤트를 조회합니다.

제목 또는 메시지에 특정 키워드가 포함된 이벤트

```
title: Connection or message: Connection
```

6. 제외 조건

특정 조건을 제외해 검색할 수 있습니다.

Info 레벨을 제외한 모든 이벤트

```
not level: info
```

실시간 알림

실시간 알림 기능은 **이벤트 설정**을 통해 설정한 이벤트가 발생할 때 팝업 형태 표시됩니다. 이벤트 발생 즉시 알림을 받아 문제를 빠르게 인지하고 대응할 수 있습니다. 이미 발생했거나 놓친 알림 내역을 확인하여 실시간 대응이 어려웠던 상황을 파악하고 해결할 수 있습니다.

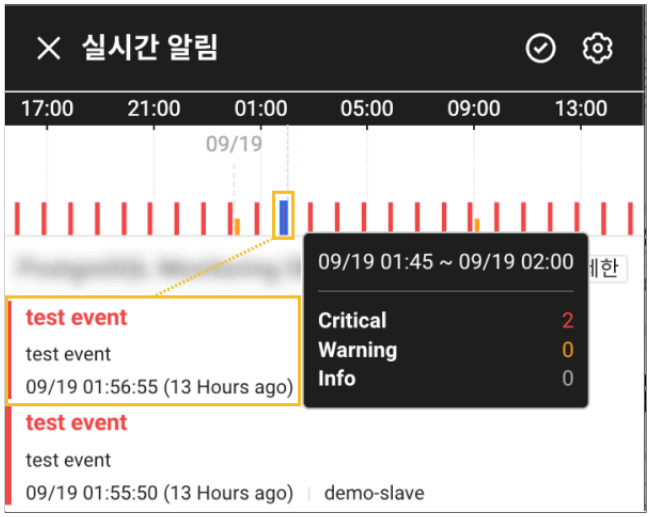
실시간 알림 확인하기

실시간 알림은 화면 오른쪽 위 팝업 형태로 표시됩니다. 알림이 발생한 시간, 제목, 메시지를 확인할 수 있습니다.

- 개별 알림을 닫으려면 메시지 오른쪽 상단의 ✕ 버튼을 클릭하세요.
- 화면에 표시된 모든 팝업 알림을 닫으려면 **전체 알림 닫기**을 클릭하세요.

알림 내역 확인하기

1. 프로젝트를 선택해 진입한 화면의 가장 위에 🔔 아이콘을 클릭하세요.
2. **실시간 알림** 창이 나타납니다.
    - 현재 프로젝트의 최근 24시간 동안의 알림을 최대 100개까지 확인할 수 있습니다.
    - 개별 이벤트 항목을 선택하면 관련 메뉴로 이동합니다.
    - **실시간 알림** 창의 시간별 차트에서 원하는 시간대를 선택하면 해당 이벤트 항목으로 이동할 수 있습니다.



- 전체 알림 내역을 확인하려면 **전체 알림 기록** 버튼을 선택하세요. **경고 알림** > **이벤트 기록** 메뉴로 이동합니다.

> **실시간 알림** 목록이 비어 있을 경우, [전체 알림 기록] 버튼이 표시됩니다. **수신 태그 미설정 알림 받기** 옵션을 해제하세요.

이벤트 일괄 해소하기

실시간 알림 목록에서 진행 중인 모든 이벤트를 일괄 해소합니다. 이벤트 해소 작업 완료까지 약 30초에서 1분 정도 소요됩니다.

- **이벤트 일괄 처리** 기능을 사용하면 현재 진행 중인 이벤트에 대한 해소 알림이 다량 발생할 수 있습니다.
- **이벤트 일괄 처리** 기능을 사용한 후에도 이벤트 규칙이 활성화되어 있고, 데이터가 임계치를 초과했다면 이벤트는 다시 발생할 수 있습니다.
- 이벤트 해소 알림은 **경고 알림** > **이벤트 설정** 메뉴에서 이벤트 수정을 통해 비활성화할 수 있습니다. **이벤트 상태가 해소되면 추가 알림** 또는 **해소된 알림** 옵션을 비활성화하세요.

사용 방법:

1. 프로젝트를 선택해 진입한 화면 상단의 아이콘을 클릭하세요.
2. **실시간 알림** 창이 나타나면 오른쪽 상단의 아이콘을 클릭하세요.

3. `실시간 알림 목록에서 진행중인 이벤트를 일괄 해소합니다.` 메시지 창이 나타나면 [해소] 버튼을 클릭하세요.

실시간 알림 설정하기

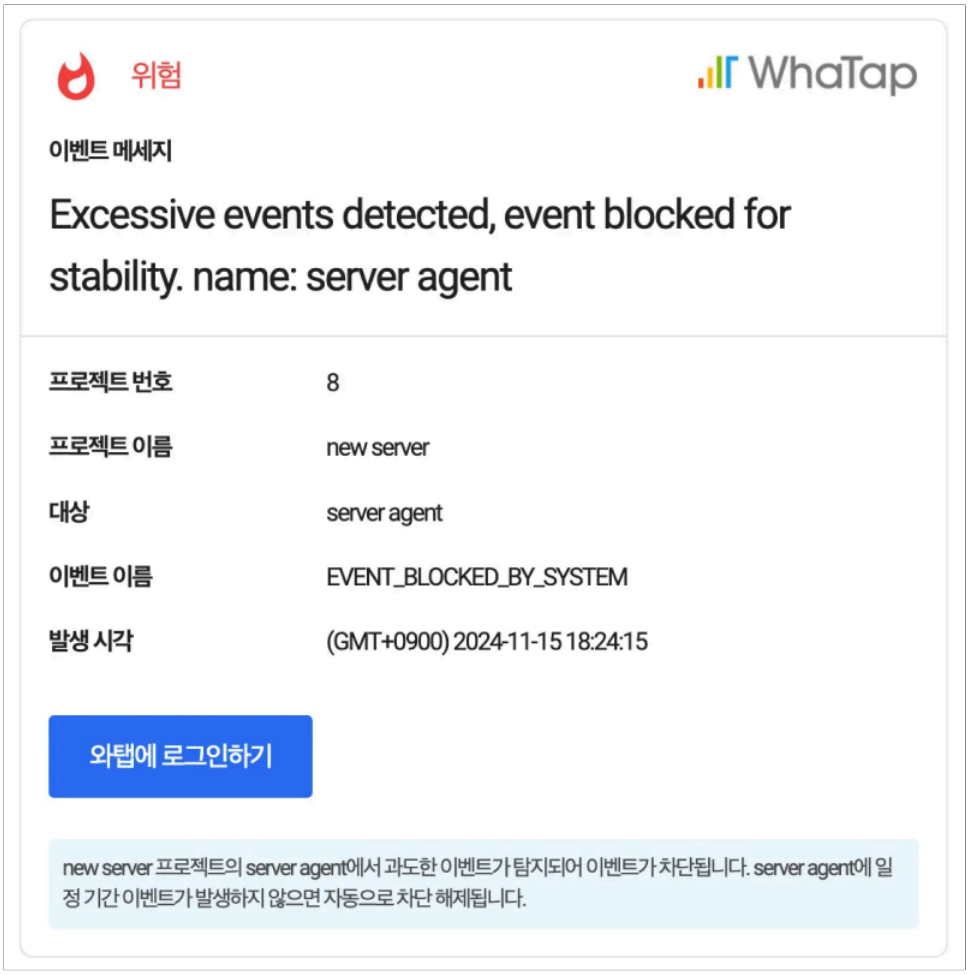
1. 프로젝트를 선택해 진입한 화면 상단의 🔔 아이콘을 클릭하세요.
2. **실시간 알림** 창이 나타나면 오른쪽 상단의 ⚙ 아이콘을 클릭하세요.
3. 실시간 알림 설정 창에서 각 항목을 설정하세요.
   - **팝업 알림**: **실시간 알림** 창이 닫힌 상태에서도 알림을 팝업 형태로 제공합니다.
   - **해소된 이벤트 보기**: **실시간 알림** 목록에서 해소된 이벤트를 포함해 확인할 수 있습니다.
   - **알림음 미리 듣기**: 브라우저에서 발생하는 알림음을 미리 들을 수 있습니다. Critical 또는 Warning 옆의 📢 버튼을 클릭하세요. 알림음이 들리지 않으면 브라우저 설정을 확인하시기 바랍니다.

시스템 이벤트

와탭 모니터링은 플랫폼 전반에서 자동으로 탐지 및 생성되는 공용 이벤트를 제공합니다. 이 문서는 시스템에 의해 발생하는 이벤트의 유형 및 발생 조건, 대응 방법에 대해 안내합니다.

EVENT_BLOCKED_BY_SYSTEM

프로젝트에서 짧은 시간에 과도한 이벤트가 발생하면, 와탭 시스템이 이를 탐지하여 이벤트를 차단하고 다음 메시지를 사용자에게 전달합니다. 이벤트가 차단되었다면 사용자의 이벤트 설정을 점검하세요.



차단 기준

이벤트를 차단하는 기준은 다음과 같습니다.

1. **에이전트 단위 차단**

   한 개의 에이전트에서 **1분** 내에 **10개**의 이벤트가 발생하면 해당 에이전트에서 발생하는 이벤트를 차단합니다.

2. **다중 에이전트 차단 시 전체 차단**

   1의 경우로 이벤트가 차단된 에이전트가 **1분** 내에 **3개** 이상이 되는 경우 프로젝트 전체에서 발생하는 이벤트를 차단합니다.

3. **프로젝트 단위 차단**

   한 개의 프로젝트에서 **1분** 내에 **100개**의 이벤트가 발생하면 프로젝트 전체에서 발생하는 이벤트를 차단합니다.

해제 방법

시스템에 의해 이벤트가 차단된 경우, 1분 동안 대상 에이전트 또는 프로젝트에서 이벤트가 발생하지 않아야 합니다.

- **경고 알림** > **이벤트 설정** 메뉴에서 설정한 이벤트를 모두 비활성화하세요. **이벤트 설정을 점검하여 과도한 이벤트가 발생하지 않도록 임계치를 조정해야 합니다**.
- 이벤트를 설정하기 전 **시뮬레이션** 기능을 활용해 이벤트 발생 횟수를 예측해볼 수 있습니다.

Java 실험실

사용자에게 새로운 기능 또는 실험적인 기능을 제공합니다. 사이드 메뉴의 **실험실** 또는 **사이트맵**을 통해 접근할 수 있습니다. 제품에 따라 제공하는 기능이 다를 수 있습니다.

사이트맵으로 실험실 메뉴 접근하기

사이드 메뉴에서 **실험실**가 보이지 않는다면 **사이트맵**으로 접근할 수 있습니다.

1. 전체 화면 왼쪽 아래에 **사이트맵** 아이콘을 클릭하세요.
2. **사이트맵** 창의 오른쪽 아래에 **실험실** 섹션에서 원하는 메뉴를 선택하세요.

주의

실험실의 기능은 현재 개발 중인 베타 버전으로 예기치 않은 오류가 발생할 수 있습니다.
해당 기능 사용 시 주의가 필요합니다. 중요한 데이터나 운영 환경에서 사용을 권장하지 않습니다. 피드백이나 문제점이 발생하면 지원팀(support@whatap.io)으로 문의하시기 바랍니다.

MXQL 데이터 조회

MXQL은 와탭의 성능 데이터(메트릭스)를 유연하게 조회하기 위한 쿼리 언어입니다. 하나의 프로젝트에 포함된 여러 에이전트에서 수집한 메트릭스들을 종합적으로 조회하고 활용하기 위해서 사용합니다. **MXQL**에 대한 자세한 내용은 다음 문서를 참고하세요.

비즈니스 대시보드

비즈니스 대시보드는 애플리케이션의 비즈니스 단위(App Context)별 성능을 모니터링할 수 있는 대시보드입니다. URL 패턴이나 경로를 기반으로 정의된 각 비즈니스 항목에 대해 실시간 및 과거의 성능 데이터를 조회하고 분석할 수 있습니다. TPS, 응답시간, 에러율 등의 주요 성능 지표를 비즈니스 단위로 확인함으로써, 특정 업무나 서비스의 성능 상태를 빠르게 파악할 수 있습니다.
자세한 내용은 다음 문서를 참고하세요.

에이전트 지원 버전

- **Java**: 2.2.40 이상

- **PHP**: 2.11.0 이상
- **Python**: 1.8.5 이상
- **.NET**: 2.4.3 이상

Collection Map

JVM에 지정된 클래스(Class)가 사용하는 Collection(예. `java.util.Map`) 정보를 제공합니다. 이 기능을 사용하려면 아래 에이전트 옵션을 추가하세요.

whatap.conf

```
collection_nonstatic_holders=io.home.test.baseapp.app.post.web.PostResource
```

에이전트 지원 버전

Java 에이전트 2.1.2 버전 이상에서 지원합니다.

외부 호출 상태

애플리케이션 에이전트의 설정을 통해 각 모니터링 대상별 외부(remote) 호출에 대한 성공/실패 여부를 기록한 로그를 시각화하여 제공합니다.

이 기능을 이용하려면 아래 옵션을 에이전트 설정(whatap.conf) 파일에 추가하거나 **관리** > **에이전트 설정** 메뉴에서 추가하세요.

whatap.conf

```
logsink_rt_enabled=true
logsink_enabled=true
```

알림 메시지

프로젝트 멤버에게 메일 또는 서드 파티 플러그인을 통해 전달하는 경고 알림 메시지를 수정할 수 있는 사용자 정의 기능을 제공합니다. 자세한 내용은 다음 문서를 참고하세요.