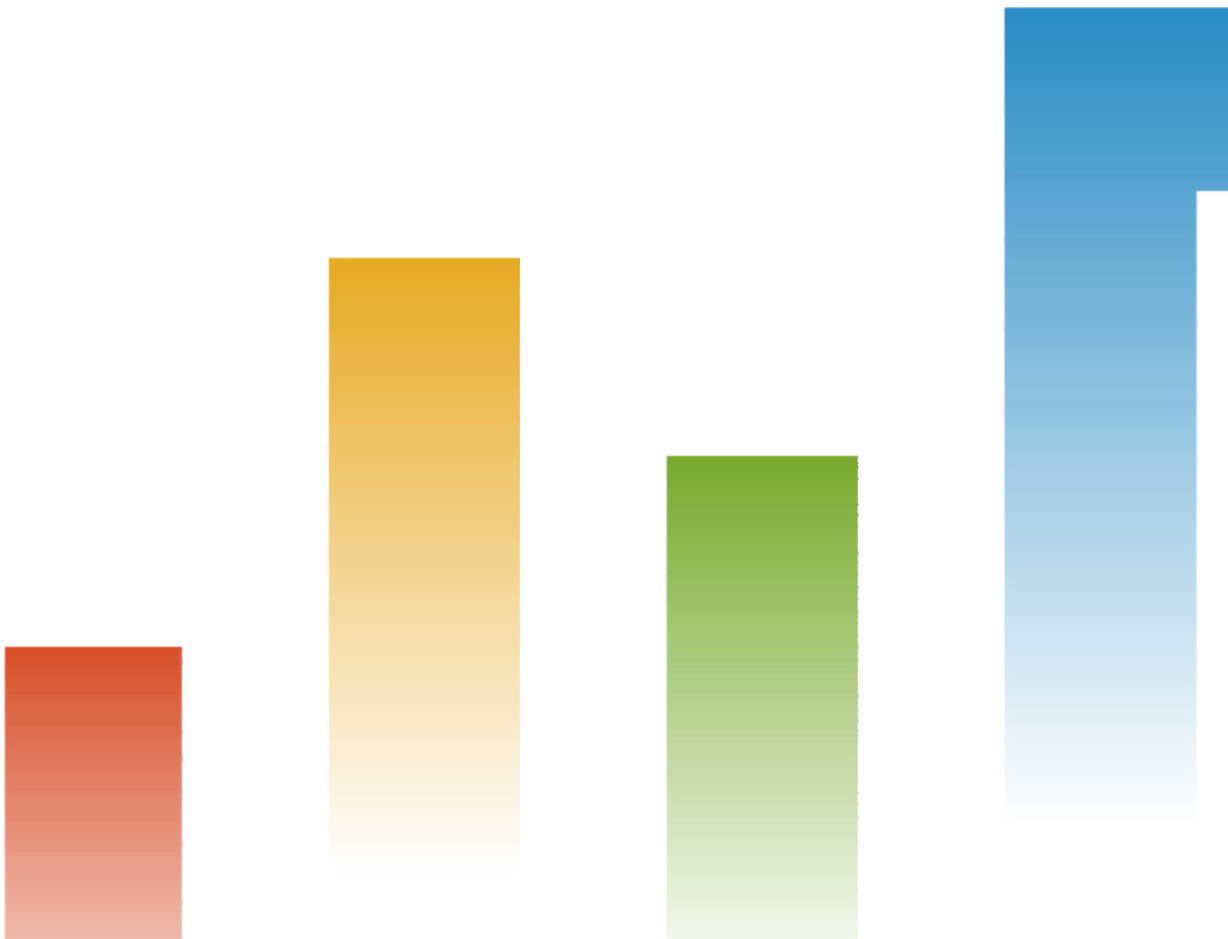
WhaTap

Java Monitoring

release date. 2026. 09. 11.



Javaドキュメント目次

- JAVAモニタリング 7
  - はじめに 9
  - サポート環境 11
  - エージェントのインストール 26
    - Tomcat 32
    - JBoss 36
    - JEUS 39
    - WebLogic 43
    - WebSphere 47
    - Resin 51
    - Liberty 55
    - Jetty 58
    - Spring Boot 61
    - Play2 66
    - IBM Bluemix 69
    - Elastic Beanstalk 73
    - Docker 78
    - バッチプログラム 83
    - 旧バージョンのJava 102
  - オープンソース追跡
    - Java Helper 105
    - Weaving 109
  - インストール点検 118
  - トラブルシューティング 120
  - 設定 127

- エージェント名 139
- 機能制御 143
- エージェント使用量 147
- 通信設定 148
- 性能設定 150
- ログ設定 151
- トランザクション設定 157
- DB/SQL 175
- HTTPC/API Call 180
- スクリプトプラグイン 182
- ユーザー数 195
- 負荷制御 197
- 通知設定 200
- Apdex 203
- 統計 204
- エラースタック 207
- 追加オプション 210
- 非同期トラッキング 212

◦ 管理

- エージェントアップデート 215
- エージェント削除 216
- MSAグループ 217
- AES 256暗号化 218
- スケールイン自動削除 219

◦ ダッシュボード 220

- アプリケーションダッシュボード 221
- TXマップ 249

- アクティブTX 259
- ヒットマップ/トレース 264
- エージェントリソースダッシュボード 294
- ビジネスダッシュボード 300
- Flexボード 306
  - Flexボードガイド 309
  - ウィジェット管理 320

◦ 分析 325

- 日別APP現況 327
- 性能推移 329
- スタック 333
- キューブ 344
- ヒットマップ 348
- マルチTX 376
- TX検索 382
- MSA分析 392

◦ メトリクス 400

- アプリケーションメトリクス 403
- 性能カウンター 423
- メトリクスチャート v2 431
- メトリクスチャート 454
- メトリクス照会 457
- 異常検知 459

◦ トランザクション 462

- トランザクション照会 464
- トランザクションプロファイル 474

- アクティブTX 502
- マルチトレース 506
- エンドポイント設定 510
- スタック収集 514

◦ アプリケーショントポロジー 516
◦ 統計 520
◦ レポート 533

- APMレポート 536
- 統合レポート 544

◦ アラート通知 553

- イベント設定 0
  - メトリクスイベント 560
  - 複合メトリクスイベント 581
  - ヒットマップパターン 588
  - 異常検知イベント 593
  - リアルタイムログ 597
  - 複合ログ 609
  - トランザクションイベント 623
- イベント設定(クラシック) 634
- 通知設定 638
  - 通知フォーマット 660
- イベント履歴 669
  - リアルタイムアラート 677
  - システムイベント 680

◦ インスタンス性能 682
◦ 連携プロジェクト 700

◦ Logモニタリング 716
◦ 実験室 0

Javaアプリケーションモニタリング

Web Application Server（WAS）、バッチアプリケーションなど、JVM環境で動作するすべてのアプリケーションに適用できます。アプリケーションの障害分析のためにThread Dump（スレッドダンプ）を取得する手間をかけることなく、既に取得されたスタックを利用して、障害の再現なしに正確な原因を分析できます。

WhaTapアプリケーションのモニタリングは、障害を再現することなく、リアルタイムでモニタリングできます。

主な機能

- **リアルタイムなトランザクションモニタリング**

  WhaTapは、現在進行中のトランザクションの実行状況をリアルタイムで表示します。すべてのアプリケーションの問題は、進行中のトランザクションの増加で現れます。増加した瞬間に実行中のSQL、外部呼び出し、Methodをすぐに確認できることで問題を早く認識して分析することができます。

- **トランザクション性能の詳細分析**

  WhaTapは、すべてのトランザクションの実行履歴を収集して分析できます。トランザクションの性能は、SQLやHTTPコールだけでなく、メソッドレベルまで追跡できます。

  特に、**アクティブスタック**機能により、煩わしいMethodトレース設定なしでMethodレベルの分析が可能です。**アクティブスタック**はWhaTapの特許技術であり、既存のサービスでは発見できなかった隠れた問題の原因を特定し、改善ポイントを提示します。

  > ⓘ アクティブスタック機能は、Java、Python、.NET の環境でのみ対応しています。

- **MSA呼び出しパターン分析**

  MSA構造の複雑なアプリケーション間の呼び出し関係を、トランザクションURLを基準に統計的に可視化して提供します。時点ごとに変化する呼び出し関係や呼び出し比率の変化を統計的に把握できます。

- **AIを活用した応答時間分布パターン分析**

  WhaTapは、特定のパターンをマシンラーニングで自動的に認識して通知アラートを生成し、システム運用者が気づいていない問題もお知らせします。オンプレミス環境でも提供しており、適用範囲は別途契約により定めます。
  出願番号：10-2020-0037381

- **マルチプロジェクトのトランザクション連携分析**

MSA環境においてトランザクションの連携追跡機能を提供します。全体のトランザクションを追跡し、遅延が発生しているアプリケーションを特定できます。

- **スタック統計の分析**

  統計分析では、リアルタイムのモニタリングやトレース情報で見つけることが困難なチューニングポイントを見つけることができます。 マクロ的な問題の推移もグラフと表を通じて改善対象を選択したり、特定の時間帯の応答遅延現象を確認することができます。

- **事後分析(Cube)**

  時間を軸にしてさまざまな指標を一度に確認できます。応答時間、エラー件数、スループットなどが高い時間帯を特定することで、同時間帯における問題要素を特定したり、サービスの状況を正確に把握することができます。

- **サービス輻輳時の負荷制御**

  システムが処理できない過剰なリクエストが発生した場合、一部のリクエストを拒否したり、不正なサービスリクエストを遮断することで、安定したサービス運用を支援します。

- **非同期トランザクショントレース**

  WhaTap独自のBCI(Byte Code Instrumentation) 技術で、非同期トランザクションの性能を効果的に追跡します。WebFlux、Reactor、Hystrixなどを使用するアプリケーションのトランザクショントレース情報を収集します。

Javaモニタリングを始める

WhaTapモニタリングは次の手順で開始します。

ステップ1. 会員登録
ステップ2. 製品選択
ステップ3. プロジェクト作成
ステップ4. アクセスキー確認

その後、サポート環境を確認してからエージェントのインストール手順に進みます。

会員登録

WhaTapモニタリングサービスを利用するには、まず会員登録が必要です。 会員登録に関する詳細は、アカウント管理 > 会員登録ドキュメントを参照してください。

製品選択

1. WhaTapモニタリングサービスにログインします。
2. **プロジェクト全体** > ＋ **Project**ボタンをクリックします。
3. **商品選択**画面でモニタリングするGo製品を選択してから作成ボタンをクリックしてください。

プロジェクト作成

モニタリングする製品を選択したら、エージェントをインストールする前にまずプロジェクトを作成します。

1. **プロジェクト作成**画面でプロジェクトを作成するには、次の項目を入力または選択してください。
    - **プロジェクト名**(必須): プロジェクト名を入力します。
    - **データサーバーリージョン**(必須): データサーバーが位置するリージョン(地域)を選択します。リージョンはクラウドサービスを提供するデータセンターのグループです。選択したリージョンにユーザーのデータが保存されます。

- **プロジェクトタイムゾーン**(必須): アラートおよびレポート生成時の基準となる時間を設定します。
- **通知言語の設定**(必須): プロジェクトで発生する警告通知メッセージの言語を設定します(韓国語、英語対応)。
- **プロジェクトグループ**: 複数のプロジェクトをグループにまとめて管理できます。所属するグループがあれば選択してください。
- **プロジェクト説明**: プロジェクトに関する追加説明を入力します。

2. すべての設定を完了したら**プロジェクトを作成する**ボタンをクリックしてください。

> ⓘ 組織を選択した状態でプロジェクトを追加する場合、**組織下位のグループ**を必須で設定する必要があります。 グループに関する詳細は、次のドキュメントを参照してください。

アクセスキー確認

アクセスキーは、WhaTapサービスを有効化するための固有IDです。

インストールガイドセクションで**プロジェクトアクセスキーの発行**ボタンを選択してください。**アクセスキー**を自動的に発行した後、次のステップに進みます。

> ⓘ プロジェクトを作成すると、自動的に**エージェントインストール**ページに移動します。**エージェントインストール**ページに移動しない場合は、画面左側のメニューから**管理** > **エージェントインストール**を選択してください。

Javaサポート環境

Javaエージェント動作環境

エージェントをインストールする前に、JVMアプリケーションサーバーが、次の仕様を満たしていることを確認します。

Web Application Server（WAS）だけでなく、デーモンやバッチアプリケーションなど、JVMベースのすべてのアプリケーションに適用できます。Java 1.6未満の場合は、support@whatap.io までお問い合わせいただくか、次の文書を参照してください。

サポートバージョン

区分 / JDKバージョン	Java 6 ~ 21	Java 22	Java 23
サポート	✅	✅	✅
サポート範囲	限定サポート	完全サポート	

オペレーティングシステム

OS系列	サポートバージョン	アーキテクチャ
Unix	HP-UX 11.x	32bit, 64bit
	Oracle Solaris 2.8, 2.9, 10, 11	32bit, 64bit
Linux	Red Hat	Itanium 64bit
	x86 Intel Linux	32bit
Windows	Windows Server	2000以上

OS系列	サポートバージョン	アーキテクチャ
	Windows	XP以上
その他	Itanium	64bit

ⓘ JDK 6をインストールできるすべてのオペレーティングシステムをサポートします。

アプリケーションサーバー

ベンダー	製品	サポートバージョン
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	-
Fujitsu	Interstage	5.x ~ 7.x
Hitachi	Cosminexus	7.x ~ 9.x

ベンダー	製品	サポートバージョン
AWS	Elastic Beanstalk	-
Docker	Container Environment	-

ⓘ Java EE Specに準拠するすべてのアプリケーションサーバーをサポートします。詳細はこちらのリンクを参照してください。

フレームワーク / ライブラリ

カテゴリ	フレームワーク	サポートバージョン
Webフレームワーク	Spring Boot	1.5.x ~ 3.x
非同期フレームワーク	Netty	-
	Akka HTTP	-
	Play Framework	-
マイクロサービス	Quarkus	-
	Quarkus Reactive	1.x ~ 2.x
統合フレームワーク	Mule Framework	3.9.5, 4.5

ⓘ Java EE Specに準拠するアプリケーションサーバーで動作するすべてのライブラリをサポートします。

データベースドライバー

カテゴリ	データベース	ドライバー	サポートバージョン
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

アプリケーションサーバー別の互換性とデータソース

Tomcat

互換性

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ファイル名	JDBC Ver.	JDK Ver.	互換性
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

互換性

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ファイル名	JDBC Ver.	JDK Ver.	互換性
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

互換性

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ファイル名	JDBC Ver.	JDK Ver.	互換性
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

互換性

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

DataSource

DB	JDBC Driverファイル名	JDBC Ver.	JDK Ver.	互換性
Mysql	mysql-connector-java-5.1.39-bin.jar	5.1.39	JDK 1.7.0_80	O

DB	JDBC Driverファイル名	JDBC Ver.	JDK Ver.	互換性
Oracle	ojdbc6-11.2.0.2.0.jar	11.2.0.2.0	JDK 1.7.0_80	O

JETTY

互換性

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ファイル名	JDBC Ver.	JDK Ver.	互換性
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

共通サポート環境

サービス提供形態

WhaTapモニタリングは、SaaSまたはオンプレミス（インストール型）方式でご利用いただけます。どちらを選択しても、同じエージェントで同じモニタリング機能をご利用いただけます。

Deployment	Support	Description
SaaS	✅	WhaTapが運用する収集サーバーに接続。リージョン別アドレスはファイアウォール項目を参照
オンプレミス（インストール型）	✅	収集サーバーを自社で構築して運用（閉域網環境に対応）。導入範囲は営業担当者と協議

> ⓘ インターネットに接続しない閉域網環境でも、オンプレミスで構築できます。

提供形態による機能の違い

どの形態を選択しても、モニタリング機能は同じです。モバイルアプリのみが例外で、service.whatap.io SaaSでのみご利用いただけます。

> ⓘ オンプレミス環境で一部の機能をご利用いただくには、別途契約が必要です。

対応ブラウザ

WhaTapモニタリングサービスは、ウェブブラウザとモバイルアプリで利用できます。

Webブラウザ	推奨	対応バージョン
Google Chrome	O	111以降

Webブラウザ	推奨	対応バージョン
Mozilla FireFox	X	最新バージョン
Edge	X	最新バージョン
Safari	X	最新バージョン

サポート案内

- ブラウザの互換性および性能の観点から、Chrome の最新バージョンの使用を推奨します。
- ユーザーインターフェース（User Interface, UI）は HTML5 標準技術で実装されているため、Internet Explorer はサポートしていません。

制約事項

WhaTap の Web インターフェースはモバイルブラウザをサポートしていません。モバイル端末から WhaTap にアクセスするには、Android または iOS 用の WhaTap アプリをインストールしてください。WhaTap モバイルアプリはモバイル環境に最適化されています。WhaTap モバイルアプリの詳細については、こちらのドキュメント を参照してください。

ファイアウォール

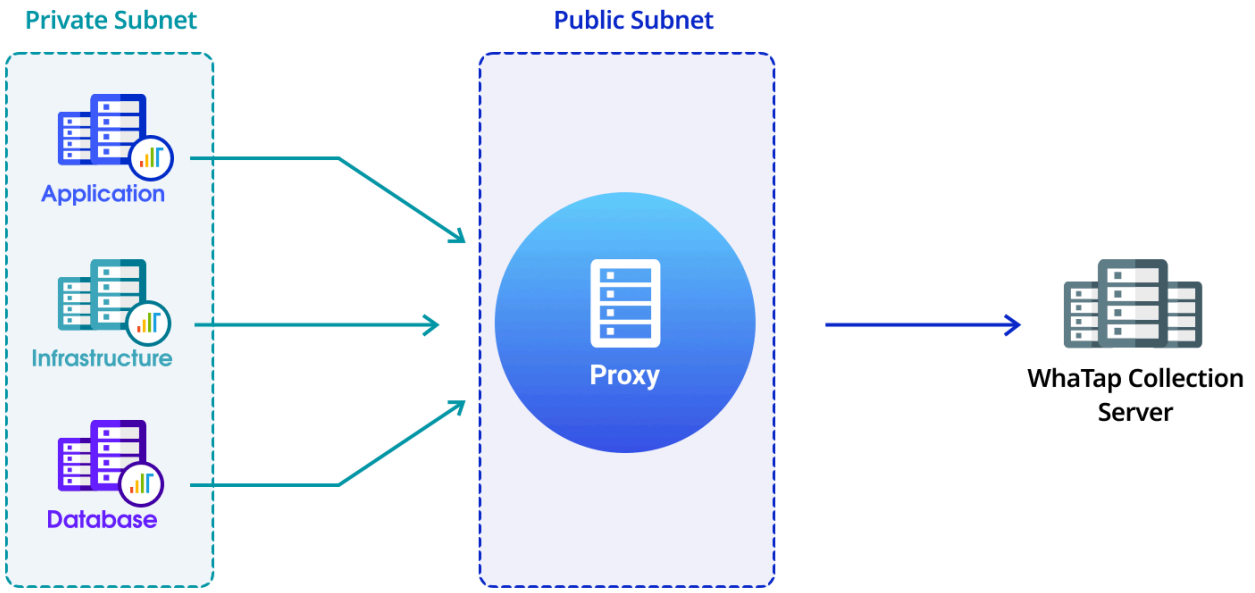
WhaTap エージェントは、収集サーバーの **TCP 6600** ポートにアクセスできる必要があります。監視対象に最も近い収集サーバーのアドレスを許可してください。

出発地：WhaTapエージェント

クラウド	目的地	目的地IP	ポート番号
AWS	WhaTapソウル収集サーバー	13.124.11.223 / 13.209.172.35	TCP 6600
	WhaTap東京収集サーバー	52.68.36.166 / 52.193.60.176	TCP 6600
	WhaTapシンガポール収集サーバー	18.138.0.93 / 18.139.67.236	TCP 6600

クラウド	目的地	目的地IP	ポート番号
	WhaTapジャカルタ収集サーバー	108.136.91.69 / 108.137.158.44	TCP 6600
	WhaTapタイ収集サーバー	43.209.109.219 / 43.209.48.86	TCP 6600
	WhaTapムンバイ収集サーバー	13.127.125.69 / 13.235.15.118	TCP 6600
	WhaTapカリフォルニア収集サーバー	52.8.223.130 / 52.8.239.99	TCP 6600
	WhaTapバージニア収集サーバー	107.23.220.101 / 54.236.221.105	TCP 6600
	WhaTapフランクフルト収集サーバー	3.125.142.162 / 3.127.76.140	TCP 6600
Azure	WhaTapソウル収集サーバー	52.231.66.38 / 20.194.5.115	TCP 6600
	WhaTap東京収集サーバー	52.246.169.54 / 20.210.27.232	TCP 6600
Kakao	WhaTapソウル収集サーバー	61.109.237.237 / 61.109.238.166	TCP 6600

エージェントから収集サーバーに直接接続できない場合は、提供するProxyモジュールを使用して経由します。



モバイルアプリ

WhaTap モバイルアプリは、Android および iOS 環境をサポートしています。リンクへ移動するか、QR コードをスキャンして WhaTap アプリをインストールできます。モバイルアプリの詳しい説明は、次の文書を参照してください。

iOS	Android

iOS	Android
iOS 12バージョン以降	Android 5.0バージョン以降

Javaエージェントのインストール

エージェントダウンロード

アクセスキーが発行されると、**エージェントダウンロド**セクションで**ダウンロード**ボタンがアクティブになります。

1. **ダウンロード**ボタンを選択してから、WhaTapエージェントファイルをダウンロードしてください。

   > 次のコマンドを使用して、サーバーから直接にダウンロードすることもできます。
   >
   > ```
   > wget https://api.whatap.io/agent/whatap.agent.java.tar.gz
   > ```

2. モニタリング対象サーバーに接続してください。
3. 目的のモニタリング対象サーバーのパスにエージェントファイルを圧縮して解凍してください。圧縮を解凍したパスは `$WHATAP_HOME` になります。

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

エージェントのインストールファイルを構成

エージェントファイルは、アプリケーションモニタリングに必要な情報を抽出し、WhaTap収集サーバーに送信するトレーサーとトレーサーがデータを送信するのに役立つ要素で構成されています。

- whatap

whatap.confファイルを設定する

基本情報を入力する

圧縮を解いた`$WHATAP_HOME`パスに移動してください。_$WHATAP_HOME/whatap.conf_ファイルにアクセスキーとデータ収集サーバーのIPアドレスを入力します。WhaTapモニタリングサービス画面に移動し**インストールの案内**セクションの**エージェントダウンロード**項目から**アクセスキー**、**収集サーバIPアドレス**値をコピーして貼り付けます。

```
license={アクセスキー}
whatap.server.host={収集サーバーIP}
```

> ⓘ whantap.confファイルの詳細については、次の文書を参照してください。

セキュリティキーの設定

SQL変数、HTTPクエリを検索、Thread停止機能を使用する場合は、セキュリティキーを設定してください。

- Javaエージェント2.2.2バージョン以降

  `$WHATAP_HOME`パスにsecurity.confファイルを作成した後、次のように 6桁の英数字のパスワードを入力します。

  security.conf

  ```
  paramkey=ABCDEF # SQL変数およびHTTPクエリの検索
  threadkill=ABCDEF # Thread停止機能
  ```

- Javaエージェント2.2.2バージョン未満

  `$WHATAP_HOME`パスにparamkey.txtファイルを作成した後、次のように 6桁の英数字のパスワードを入力します。

  paramkey.txt

  ```
  ABCDEF # SQL変数およびHTTPクエリの検索、スレッド停止機能
  ```

Javaエージェント 2.2.2 のバージョン未満から2.2.2バージョンにアップデートする場合、既存のparamkey.txtファイルの設定値は、security.confファイルに自動的に適用されます。例えば、paramkey.txtの設定値が `FEDCBA` で、バージョン2.2.2にアップデートすると、security.confファイルが次のように適用されます。

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

paramkey.txtファイルが存在しない場合、security.confファイルのキー値は、**WHATAP**というキー値で自動的に生成されます。

- SQLパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
- HTTPパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。

JVMオプション追加

使用するアプリケーションサーバー環境に応じて、以下の設定を行います。エージェントをインストールして適用する前に 動作環境を確認してください。

Tomcat

Tomcat環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

JBoss

JBoss環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

JEUS

JEUS環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

WebLogic

WebLogic環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

WebSphere

WebSphere環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

Resin

Resin環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

Liberty

Liberty環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

Jetty

Jetty環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

Spring Boot

Spring Boot環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

Play2

Play2フレームワークでJavaエージェントを適用するためのJVMオプション設定方法を案内します。

IBM Bluemix

IBM Bluemix環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

Elastic Beanstalk

AWS Elastic Beanstalk環境でJavaエージェントを適用するための設定方法を案内します。

Docker

Docker環境でJavaエージェントを適用するためのJVMオプション設定方法を案内します。

バッチプログラム

バッチアプリケーションにWhaTapエージェントを適用するためのインストールおよび設定手順を説明します。

旧バージョンのJava

Java 5〜8以下の環境でエージェントを適用する際に必要なJVMオプションを案内します。

TomcatにJVMオプションを追加

環境変数の追加

Tomcatアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。使用するOSに合わせて設定してください。

Linux

catalina.shファイルの先頭に `JAVA_OPTS` 環境変数を追加します。

{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` 項目は、エージェントのアップデートで最新のWhaTapエージェントを適用します。

Windows

- setup.batファイルを実行した場合

  catalina.batファイルの先頭に `JAVA_OPT` 環境変数を追加します。

  {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 ############
```

- TomcatをWindowsインストーラーでインストールした場合
    1. **開始** > **Apache Tomcat** `X.Y.Z.` > **Configure Tomcat**を選択します。
    2. **Java**タブを選択します。
    3. **Java Options**テキストボックスに `-javaagent` オプションを追加します。

```
-javaagent:{WhaTapインストールパス}/whatap.agent-X.Y.Z.jar
```

Apache Tomcat

Java 17バージョン以降

Java 17 以降では、reflectionに関する次のオプションを追加します。

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイ

ズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

JBossにJVMオプションを追加

環境変数の追加

JBossアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。

JVMオプションに `-javaagent` および `-Djboss.modules.system.pkgs` オプションを追加します。

JBoss7.0以降、EAP6.0以降

設定場所：(JBossインストールパス)/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 ############
```

> JBoss5.0バージョン以下の場合は_(JBossインストールパス)/bin/run.conf_パスで設定してください。エラーが発生した場合は、 `-Djboss.platform.mbeanserver=true` オプションを追加します。

Java 17バージョン以降

Java 17 以降では、reflectionに関する次のオプションを追加します。

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

JEUSにJVMオプションを追加

JVMオプションの追加

JEUSアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。

1. JEUSバージョンによる開始オプションの設定ファイルのパスを参照して、`-javaagent`オプションを追加します。

- JEUS 7 JEUS 6

```xml title="$JEUS_HOME/domains/jeus_domain/config.xml" {6-9} showLineNumbers
<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>
```

```xml title="$JEUS_HOME/config/$hostname/JEUSMain.xml" {6-9} showLineNumbers
<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. アプリケーションサーバーを再起動します。

   ```
   jdown && jboot
   ```

2. アプリケーションサーバーのログファイルとエージェントのログファイルを確認して、エージェントが正常に動作しているか、エラーが発生していないかを確認してください。ログファイルの場所については、以下を参照してください。
   - エージェント：`$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. エージェントがアプリケーションサーバーの種類とサービスコンテナ名を認識していることを確認してください。

ⓘ WhaTapモニタリングサービスにログインし、プロジェクトを選択します。**アプリケーション** > **インスタンスパフォーマンス分析**メニューを選択し、**実行環境変数**タブを選択します。**whatap.name**と**whatap.type**項目を確認してください。**whatap.name**項目の最後の要素は、コンテナ名にしてください。**whatap.type**項目には、アプリケーションサーバーの種類を指定してください。

Boot Environment

ⓘ Java 17 以降では、reflectionに関する次のオプションを追加します。

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

WebLogicにJVMオプションを追加

JVMオプションの追加

WebLogicアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。

使用するオペレーティング システムを確認してから設定を完了してください。

- 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 以降では、reflectionに関する次のオプションを追加します。

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーショ**

ンメニューの詳細については、次の文書を参照してください。

WebSphereにJVMオプションを追加

JVMオプションの追加

WebSphereアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。JVMオプションを追加するには、管理コンソール(Admin console)に接続してください。

1. ウェブブラウザで管理コンソールにログインしてください。
2. **Environment** > **Virtual Hosts** > **default host** > **Host Aliases**項目を選択し、エージェントをインストールするサーバーのポート番号を確認してください。

   WebSphere port
3. **Servers** > **Server Types** > **Websphere application servers**項目に移動し、エージェントをインストールするサーバーを選択します。

   WebSpherer application servers
4. 選択したサーバー画面から**Configuration**タブを選択し、**Server Infrastructure**セクションの**Java and Process Management** > **Process definition**メニューを選択します。

   WebSphere Configuration
5. 右側のメニューから**Additional Properties** > **Java Virtual Machine**メニューを選択します。
6. **Configuration**タブから**Generic JVM arguments**テキストボックスに `-javaagent` と `-Dwhatap.port` を追加します。

   ```
   -javaagent:/whatap/whatap.agent-X.Y.Z.jar
   -Dwhatap.port={ポート番号} ## 2番 項目 ポート番号 ##
   ```

   Add JVM arguments

> Java 17 以降では、reflectionに関する次のオプションを追加します。

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

クライアントのIPアドレスが収集されない場合

ダッシュボードやその他のメニューでクライアントIPアドレスが正常に収集されない場合は、以下を参照してください。

IBM HTTPサーバーでバックエンドのWebSphereアプリケーションサーバーに実際の(original)クライアントIPアドレスに転送すると、Javaエージェントから有効なクライアントIPアドレスを収集できます。実際のクライアントIPアドレスを転送するにはIBM HTTPサーバーでHTTPヘッダー(header)に `X-Forwarded-For` を追加する必要があります。

これについての詳細は、IBM対応の文書で扱われています。次のリンクを参照してください。Passing the real client IP from IBM HTTP Server 9.0 to WebSphere Application Server

> ⓘ IBM対応文書を確認するには、IBMアカウントが必要になる場合があります。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

ResinにJVMオプションを追加

JVMオプションの追加

Resinを使用するアプリケーションサーバー環境でJVMオプションを追加する方法を確認してください。Javaバージョンによって適用するJVMオプションが異なる場合があります。ユーザーの環境に合わせたJVMオプションを適用してください。オプションを適用して、アプリケーションサーバーを再起動します。

- オプション1: resin.propertiesファイルに `jvm_args` 属性を使って `javaagent` 引数を追加します。
- オプション2: `conf/resin.conf` または `conf/resin.xml` ファイルに `<jvm-args>` セクションを追加して `-javaagent` 引数を設定してください。

> ⓘ
> - アプリケーションサーバーのログファイルとエージェントのログファイルを使用して、エージェントが正常に動作していることとエラーが発生していないことを確認してください。ログファイルの場所については、以下を確認してください。
>   - エージェント： `$WHATAP_HOME` /logs/whatap- `{SERVER_NAME}` - `{DATE}` .log
>   - RESIN 4.x: $RESIN_HOME/log/jvm-app-#.log
> - エージェントが正常に動作しないか、エラーが発生した場合は次の文書を確認してください。

Java 6 ~ Java 8バージョン

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 ~ Java 15バージョン

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バージョン以降

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>
```

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

LibertyにJVMオプションを追加

JVMオプションの追加

Libertyアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。

jvm.optionsファイルにエージェントオプションを追加します。ファイルがない場合は、新規に作成してください。

```
-javaagent:/app/whatap/whatap.agent-2.1.1.jar
-Dorg.osgi.framework.bootdelegation=whatap.* ## OSGIフレームワーク設定のための追加オプション##
```

> Java 17 以降では、reflectionに関する次のオプションを追加します。
>
> ```
> --add-opens=java.base/java.lang=ALL-UNNAMED
> ```

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

  example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

JettyにJVMオプションを追加

JVMオプションの追加

Jettyアプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。

- jetty.shファイルにJVMオプションを追加

  $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
...
```

- シェルスクリプトがいない状態でJettyの開始

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

> ⓘ Java 17 以降では、reflectionに関する次のオプションを追加します。
>
> ```
> --add-opens=java.base/java.lang=ALL-UNNAMED
> ```

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

Spring BootにJVMオプションを追加

Spring Bootを使用するアプリケーションサーバー環境でJVMオプションを追加する方法を確認してください。

JVMオプションの追加

- Java 17バージョン以降の場合、reflection関連のJVMオプションを追加でアプリケーションを起動できます。

```
# Java 17以降のバージョンに対応
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}
```

- 起動スクリプトにJVMオプションとして、 `-javaagent` を追加します。

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

JVMオプションを追加した後、使用するSpring Bootのバージョンとエージェントのバージョンに合わせて適切なエージェント設定を進めてください。

Spring Boot 3

Spring Boot 3.0.0 バージョン以降

- Javaエージェントv2.2.9バージョン以降(whatap.agent-2.2.9.jar)

  whatap.conf

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

- Javaエージェントv2.2.5バージョン以降(whatap.agent-2.2.5.jar)

  whatap.conf

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

- Javaエージェントv2.2.4バージョン以降(whatap.agent-2.2.4.jar)
  - `jakarta.servlet.http.HttpServlet` 追跡(同期方式)

    whatap.conf

    ```
    weaving=tomcat10
    ```

  - WebClient追跡(非同期方式)

    whatap.conf

    ```
    weaving=webflux-6.0
    ```

Spring Boot 2

Spring Boot 2.7.0バージョン以降

- Javaエージェントv2.2.9バージョン以降(whatap.agent-2.2.9.jar)

  whatap.conf

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

- Javaエージェントv2.2.5バージョン以降(whatap.agent-2.2.5.jar)

  whatap.conf

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

- Javaエージェントv2.2.4バージョン以下(whatap.agent-2.2.4.jar)

  WebClient追跡(非同期方式)

whatap.conf

```
weaving=webflux-5.3
```

Spring Boot 2.5.0バージョン以降

- Javaエージェントv2.2.9バージョン以降(whatap.agent-2.2.9.jar)

  whatap.conf

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

- Javaエージェントv2.2.5バージョン以降(whatap.agent-2.2.5.jar)

  whatap.conf

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

- Javaエージェントv2.2.4バージョン以下(whatap.agent-2.2.4.jar)

  WebClient追跡(非同期方式)

  whatap.conf

  ```
  weaving=webflux-5.3
  ```

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

Play2にJVMオプションを追加

JVMオプションの追加

Play2アプリケーションサーバー環境でJavaモニタリングエージェントを効果的に使用するために必要なJVMオプションを追加する方法を案内します。

_$PLAY_HOME/framework/build_ファイルにエージェントオプションを設定してください。別途利用する開始または停止スクリプトがある場合は、そのファイルに適用してください。

```
#! /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"
...
```

ⓘ Playは基本設定に、 `-Dwhatap.play2=2.2.6` オプションを追加する必要があります。

ⓘ Java 17 以降では、reflectionに関する次のオプションを追加します。

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

IBM BluemixにJVMオプションを追加

エージェントの適用

IBM BlueMixを使用してデプロイする場合、WhaTapエージェントを有効にする方法です。

IBM BlueMixは、コンテナで**WebSphere Liberty**環境を提供します。**Liberty**は、**WebSphere Application Server**と異なる環境で**Spring Boot**が動作する方式と類似しています。

> ⓘ **Platform as a Service**(**PaaS**) の環境では、対象アプリケーションにWhaTapエージェントの設定を含めて配布します。

以下は設定環境の例です。 ガイドによる環境を設定する場合、ローカル開発環境で生成するファイルです。

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

1. `${APP_HOME}`で_src/main/resources/whatap-agent/_ディレクトリを作成し、jarファイルと confファイルをコピーします。

```
$ 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_ファイルにオプションを追加します。ymlファイルですので、スペースとインデントに従って作成します。

```
1  ---
2  applications:
3  - name: sample-java-helloworld
4  random-route: true
5  memory: 256M
6  path: target/JavaHelloWorldApp.war
7  # ここから追加します。
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 以降では、reflectionに関する次のオプションを追加します。

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

ⓘ whatap.conf設定は、PaaS以外の環境と同様に適用します。適用後、エージェント名の識別が特定するのが難しい場合があるため、状況に応じたエージェント名を適用してください。 エージェントネーミングの詳細については、次の文書を参照してください。

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒ**

ープヒストグラムタブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

Elastic BeanstalkにJVMオプションを追加

エージェントの適用

Elastic Beanstalkを使用してデプロイする場合、WhaTapエージェントを適用します。

> ⓘ **Platform as a Service**(**PaaS**) の環境では、対象アプリケーションにWhaTapエージェントの設定を含めて配布します。

1. Spring Boot の結果をjarでデプロイする場合、.ebextensionsの内容は適用できません。次のファイルを圧縮して、zip形式でデプロイしてください。
   - .elasticbeanstalk/config.yml：ebコマンドを実行するディレクトリ下位に自動生成
   - .ebextensions/`{config_name}`.config：ebの設定ファイル
   - Procfile：JVM command lineオプションを設定するためのファイル
   - `{application}`.jar：実行するapplicaion.jarのファイル
2. Serviceに適用するWhaTapエージェントのファイルを圧縮し、S3(またはダウンロードできるpublicパス)にアップロードしてください。
   - paramkey.txt
   - whatap.agent-2.0_25.jar
   - whatap.conf

   ```
   # ディレクトリ圧縮
   zip -r whatap-agent.zip agent
   ```
3. _.ebextensions/`{config_name}`.config_ファイルにダウンロードするWhaTapエージェントパスを入力します。

   ダウンロード用の圧縮ファイルがある場合、ebを実行して自動的にダウンロードして解凍します。

```
sources:
 target directory:
  S3のパス
```

Example

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

4. WhaTapエージェントオプションを追加したJVM commandを入力してProcfileを作成してください。

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ファイルと*.ebextensionsファイル、`{application}`.jar*ファイルを圧縮してください。

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

6. _.elasticbeanstalk/config.yml_ファイルにデプロイする zipファイルのパスを追加します。

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

7. eb配布コマンドを実行してください。

```
eb deploy
```

Java 17 以降では、reflectionに関する次のオプションを追加します。

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

詳細は、AWSガイド文書を参考にしてください。

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

DockerにJVMオプションを追加

Dockerはコンテナベースの仮想化プラットフォームで、Javaアプリケーションを Dockerコンテナ内で実行する時、JVMオプションを追加して、アプリケーションの性能とリソースの使用を調整できます。この文書ではDockerfileを修正してJVMオプションを追加する方法を案内します。

> ⓘ Dockerがインストールされており、JavaアプリケーションのDockerイメージがビルドできる必要があります。

エージェントの適用

既存のDockerイメージに基づいて、WhaTapエージェント構成を追加したイメージをビルドします。

> ⓘ この文書では、Javaエージェントをインストールする場合のみ説明します。WhaTap Kubernetesモニタリングと一緒に適用する場合、次の文書を参照してください。

Dockerのビルドディレクトリを作成する

Dockerのビルドディレクトリを作成します。

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

エージェント設定ファイルの作成する

whatap.confファイルを作成します。

```
1 cat >{Docker build Dir}/whatap.conf <<EOL
2 # アクセスキーを入力します。
3 license=XXXXXXXXXXXXX-XXXXXXXXXXXXX-XXXXXXXXXXXXX
4 # 収集サーバーのIP情報を入力します。
5 whatap.server.host=xx.xx.xx.xx/yy.yy.yy.yy
```

```
6  EOL
```

Dockerfile生成する

イメージをビルドするには、WhaTapのイメージから `-javaagent` のオプションに適用するjarファイルをコピーします。

```
cat >/home/silver/whatap/docker/Dockerfile <<EOL
FROM whatap/kube_mon as build
## 実際イメージの作成(既存のイメージにWhatapを追加)
## $Image_Name(イメージ名)
FROM $Image_Name
RUN mkdir -p /whatap
COPY --from=build /data/agent/micro/whatap.agent-*.jar /whatap
COPY ./whatap.conf /whatap/
#...
EOL
```

`JAVA_OPTS` 適用する

`JAVA_OPT` に次の内容を追加します。

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

Dockerビルドする

Dockerをビルドします。

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

> ⓘ Java 17 以降では、reflectionに関する次のオプションを追加します。

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

Javaエージェントファイル名は、`Rename`機能を活用して変更できます。Javaエージェント名を変更した場合は、`Java_OPTS`に新しいJavaエージェント名を登録してください。

Javaエージェント名の変更方法の例

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

ヒープヒストグラム照会

WhaTapのJavaモニタリングでは、JVMメモリのヒープ占有オブジェクト現況(ヒープメモリ上のオブジェクト別サイズ)を照会できる機能を標準として提供します。 **アプリケーション** > **インスタンスパフォーマンス分析**メニューから**ヒープヒストグラム**タブを選択します。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

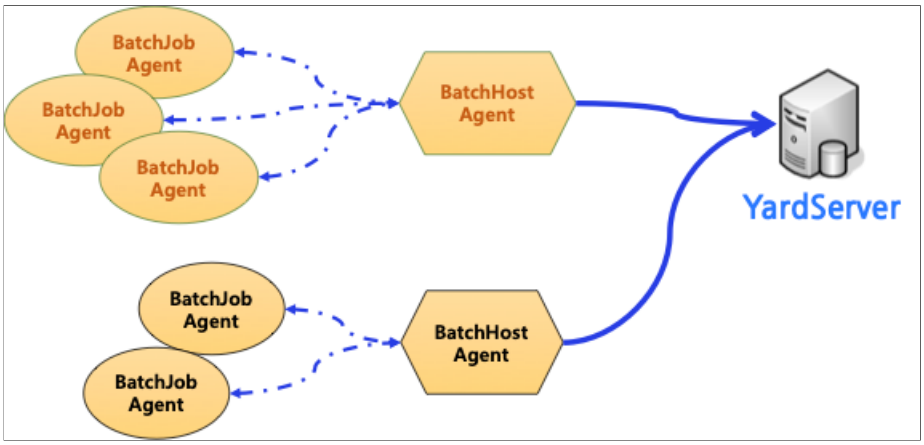
エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

バッチプログラムモニタリング

バッチアプリケーションにWhaTapエージェントを適用するためのインストールおよび設定手順を説明します。



バッチモニタリングの構成

バッチアプリケーションはWASとは異なり、実行と停止を繰り返します。そのため、バッチモニタリングは次の2つのプロセスに分かれています。

プロセス	役割	実行単位
バッチホスト(Batch Host)	バッチエージェントから受け取ったデータを収集サーバーに送信	ホストごとに1つ
バッチエージェント(Batch Agent)	バッチジョブのデータを収集してバッチホストに送信	バッチジョブごとに1つ

2つのプロセスはUDPで通信します。この構成により、複数のバッチジョブを並列に実行できます。バッチホストとバッチエージェントは、バッチジョブを実行するサーバーに一緒にインストールしてください。

通信要件

経路	プロトコル/ポート
バッチホスト → 収集サーバー	TCP 6600(アウトバウンド)
バッチエージェント → バッチホスト	UDP 6611



バッチプログラムモニタリングはJavaモニタリングの付加機能ですが、Javaモニタリングエージェントとは別に動作する独立したシステムです。Javaエージェントの設定オプションはバッチモニタリングには適用されません。この文書に記載したオプションのみ使用してください。Javaエージェントで `profile_` 接頭辞を使うオプションは、バッチプログラムでは `trace_` 接頭辞を使います。

エージェントのダウンロード

次のコマンドを実行して、バッチアプリケーションモニタリング用のエージェントをダウンロードしてください。

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

圧縮を解凍すると、次のように構成されます。

```
{BATCH_HOME}/
 ├── whatap.agent.batch.host-X.Y.Z.jar  バッチホスト
 ├── whatap.agent-X.Y.Z.jar          Javaエージェント
 ├── whatap.conf                 バッチホスト設定ファイル
 └── agent/
   ├── whatap.agent.batch-X.Y.Z.jar   バッチエージェント
```

```
└── whatap.conf          バッチエージェント設定ファイル
```

`whatap.conf` ファイルがない場合は新しく作成した後、**設定** > **エージェントインストール**メニューで確認できる `license` と `whatap.server.host` 項目を追加してください。

設定ファイルの場所

> 導入時に最も失敗しやすい部分です。必ず確認してください。
>
> WhaTapは設定ファイルとログの場所を `whatap.home` を基準に決定します。 `whatap.home` が決まる方式は起動方法によって異なります。設定ファイルが見つからない場合、起動時のバナーに `PCODE(0)` が表示され、データを収集しません。

whatap.homeの決定方式

優先順位は次のとおりです。

1. `-Dwhatap.home` を指定した場合は、常にその値が優先されます。
2. 指定がない場合、 `-javaagent` で起動したプロセス(バッチエージェント)はエージェントjarがあるディレクトリを `whatap.home` として使用します。
3. `-cp` でメインクラスを指定して起動したプロセス(バッチホスト)は上記の自動設定が適用されないため、起動時点の作業ディレクトリが `whatap.home` になります。

プロセス	起動方法	whatap.home	読み込む設定ファイル
バッチホスト	`-cp` + メインクラス	作業ディレクトリ	作業ディレクトリの `whatap.conf`
バッチエージェント	`-javaagent`	エージェントjarがあるディレクトリ	jarと同じディレクトリの `whatap.conf`

> バッチホストは必ずインストールディレクトリに移動してから起動してください。ジョブスケジューラやジ

ョブ管理製品から起動する場合、作業ディレクトリが想定と異なることがあります。起動スクリプト内で作業ディレクトリを明示的に移動してください。

バッチホストの設定

`license` と `whatap.server.host` のオプション値は、プロジェクトの**設定** > **エージェントインストール**メニューで確認できます。

{BATCH_HOME}/whatap.conf

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

Option	Default	Description
`license`	-	プロジェクトのアクセスキー
`whatap.server.host`	-	収集サーバーアドレス。二重化した場合はスラッシュ(/)で区切る。例) `10.0.0.1/10.0.0.2`
`whatap.server.port`	`6600`	収集サーバーポート

上記3項目だけで動作します。以降で説明するオプションは、必要な場合にのみ追加してください。

UDP受信の設定

バッチホストがバッチエージェントからデータを受け取るアドレスとポートです。

{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データを受け取るポート。既定ポートを他の用途で使う場合に変更
net_udp_listen_ip	0.0.0.0	UDPデータを受け取るためにバインドするIPアドレス。変更可能だが既定値を推奨

UDPポートを変更する場合は、バッチホストとバッチジョブの両方を変更する必要があります。

変更対象	設定ファイル	オプション
受信側(バッチホスト)	{BATCH_HOME}/whatap.conf	net_udp_listen_port
送信側(バッチジョブ)	バッチジョブの whatap.conf	net_udp_port

片方だけ変更すると、エラーが表示されないままトランザクションデータを失います。バッチジョブは従来のポートに送信し続け、バッチホストは変更したポートでのみ待機するためです。変更後はバッチホストとバッチジョブの両方を再起動してください。

トレースステップ収集の設定

個別のトランザクショントレースは、ステップを無制限に収集できません。トランザクションごとに長さが制限されたバッファにトレースステップを保存し、既定の設定では**1つのバッチジョブあたり1,000個**まで収集します。

{BATCH_HOME}/whatap.conf

```
split_trace_enabled=false
```

Option	Default	Description
split_trace_enabled	false	トレースを分割して送信

上限を超えたステップは**トレースに表示されないまま欠落します。** トレース分析画面のステップ番号が1,000付近で途切れている場合がこれに該当します。

ステップ数が多いバッチジョブの処理内容をすべて取得する必要がある場合は、テクニカルサポートチーム

にお問い合わせください。

プロファイリング終了待機時間

バッチホストは実行中のジョブを定期的に点検し、一定時間ステップが届かないジョブのプロファイリングを強制終了します。

{BATCH_HOME}/whatap.conf

```
profile_expired_time=30000
```

Option	Default	Description
profile_expired_time	30000	最後のステップを受け取った時点からこの時間(ミリ秒)が経過するとプロファイリングを強制終了

バッチジョブが異常終了したときに、トランザクションが実行中の状態で残らないよう自動的に整理する機能です。強制終了したトランザクションには lost-profile ステップが記録されます。

外部システムの応答待ちなど、計測対象の処理が発生しない区間がこの時間を超えると、ジョブが正常に実行中であっても強制終了します。このような区間があるバッチジョブでは、実際の無入力区間より余裕のある値を設定してください。

バッチエージェントの設定

{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	データを送信するバッチホストのIPアドレス。同一サーバーで実行する場合は変更不要
net_udp_port	6611	データを送信するバッチホストのUDPポート。バッチホストの net_udp_listen_port と同じ値
job_name	-	バッチジョブを識別する名前。トランザクション名として表示。ジョブごとに固有の値を設定
time_limit	300000	遅延警告を表示するしきい値(ミリ秒)。ジョブを中断させる値ではない
trace_httpc_limit	1000000	バッチジョブ1件で収集するHTTP呼び出しの最大件数
trace_sql_limit	1000000	バッチジョブ1件で収集するSQLの最大件数
active_stack_enabled	true	アクティブスタックを収集するかどうか
active_stack_start_wait_time	1000	バッチジョブ開始後、最初のアクティブスタックを収集するまでの待機時間(ミリ秒)
active_stack_time1	5000	最初の5回までのアクティブスタック収集間隔(ミリ秒)
active_stack_time2	10000	6回目以降のアクティブスタック収集間隔(ミリ秒)

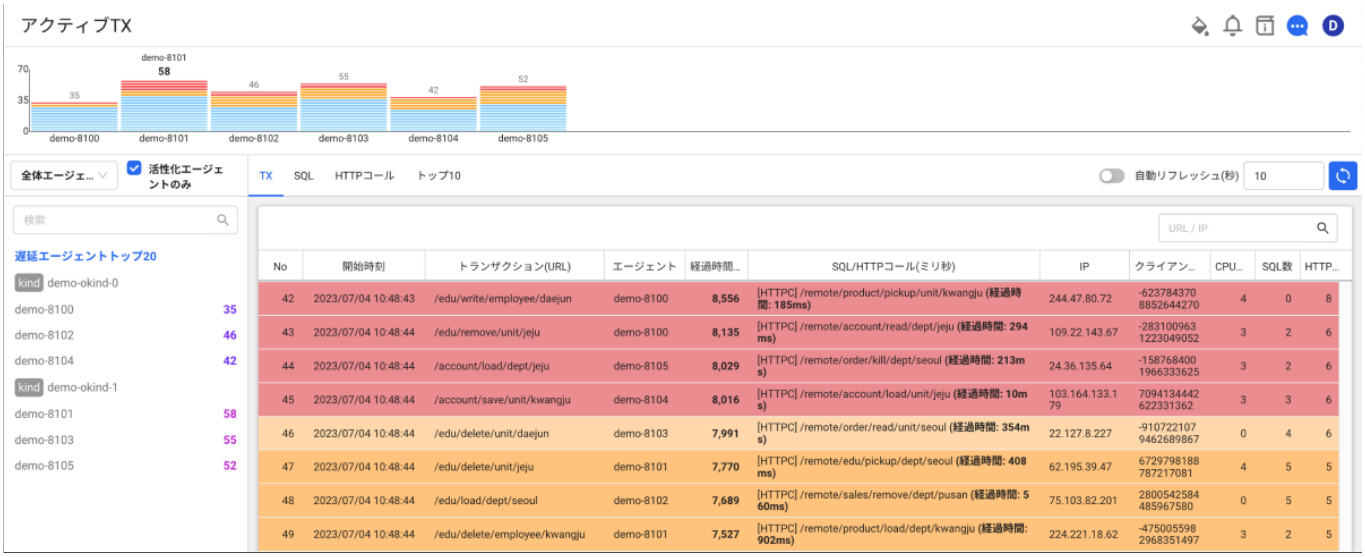
net_udp_ip と net_udp_port は、バッチホストと同じサーバーで実行する一般的な構成であれば既定値で動作します。その他のオプションは必要な場合にのみ追加してください。

time_limitの設定

バッチジョブは通常は長時間実行され、ジョブごとに想定処理時間が異なります。ジョブごとに `time_limit` を設定してください。

`time_limit` を設定すると、アクティブトランザクション画面の警告表示しきい値が自動的に計算されます。個別の設定は必要ありません。

経過時間	表示
`time_limit` の70%まで	既定の色
`time_limit` の70%超過	黄色
`time_limit` 超過	赤色



色の判定はジョブ全体の経過時間を基準にします。トレースを分割した場合も、区間ごとではなくジョブ全体の経過時間で判定します。

アクティブスタックの収集タイミング

バッチエージェントは、実行中のバッチジョブのスタックトレースを一定間隔で収集します。収集したアクティブスタ

ックは、トレース分析画面で**ActiveStack**ステップとして確認できます。

`active_stack_time1` と `active_stack_time2` はしきい値ではなく収集間隔です。既定値を基準にした収集タイミングは次のとおりです。

回数	収集時点	適用オプション
1回目	バッチジョブ開始後1秒	`active_stack_start_wait_time`
2回目 ~ 5回目	直前の収集から5秒間隔	`active_stack_time1`
6回目以降	直前の収集から10秒間隔	`active_stack_time2`

既定値で90秒間実行するバッチジョブでは、約12回のアクティブスタックを収集します。

収集間隔が切り替わる基準である5回は固定値であり、変更できません。変更できる項目は `active_stack_start_wait_time` 、 `active_stack_time1` 、 `active_stack_time2` の3つの時間設定です。

`active_stack_time1` と `active_stack_time2` の最小値は100ミリ秒です。100未満を設定しても100として適用されます。

アクティブスタックを収集しない場合は、 `active_stack_enabled=false` に設定してください。

収集件数の上限

バッチジョブ1件で収集するHTTP呼び出しとSQLの最大件数です。既定値はそれぞれ100万件です。

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

設定した件数を超えると、その種類のデータ収集はその時点で停止し、超過分は統計に反映されません。バッチジョブ自体は実行を継続し、トランザクションも正常に完了します。

> 収集が停止してもトレースに表示は残りません。実際の処理件数が上限を超える可能性がある場合は、あら

> かじめ十分に大きな値を設定してください。
>
> トランザクション一覧に表示されるSQL件数またはHTTP呼び出し件数が設定した上限値と正確に一致する場合、上限に達した可能性があります。
>
> 2つのオプションに製品上の上限値はありません。ただし値を大きく設定するとバッチプロセスのメモリ使用量が増えるため、検証環境で確認してから運用環境に適用してください。

ジョブごとに設定を分ける

`job_name` と `time_limit` はジョブごとに異なる値を設定します。設定ファイルは `whatap.home` を基準に読み込むため、 `-Dwhatap.home` でジョブ別ディレクトリを指定する方式を推奨します。

エージェントjarは1か所に置き、設定ファイルのみジョブごとに用意してください。

```
{BATCH_HOME}/agent/whatap.agent.batch-X.Y.Z.jar   すべてのジョブで共有

{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
```

設定ファイルを読み込んだディレクトリには `logs` ディレクトリが作成されます。ジョブディレクトリに `logs` がない場合は、そのディレクトリの設定が使われていません。問題を確認するときの指標として活用してください。

実行

実行順序

バッチホストは、すべてのバッチジョブに対して1つだけ起動します。

1. すべてのバッチジョブを実行する前に、バッチホストを1回起動してください。

2. 各バッチジョブを `-javaagent` オプションとともに実行してください。複数のジョブを並列に実行できます。
3. すべてのバッチジョブが終了した後にバッチホストを終了してください。

> 設定ファイルは起動時に1回だけ読み込みます。設定を変更した場合、バッチエージェントは次に実行するジョブから反映され、バッチホストは再起動が必要です。
>
> バッチホストのプロセスを強制終了すると、実行中のジョブから送信したデータを失うことがあります。すべてのジョブが終了した後に停止してください。

バッチホストの起動

- 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以上の場合は、リフレクション関連のオプションを追加してください。

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以上の場合は、リフレクション関連のオプションを追加してください。

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
```

Windowsではクラスパスの区切り文字がセミコロン(;)です。コマンドプロンプトはセミコロンを引数の区切りとして解釈するため、クラスパス全体をダブルクォーテーションで囲む必要があります。

バッチジョブの実行

- 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以上の場合は `--add-opens=java.base/java.lang=ALL-UNNAMED` を追加してください。

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以上の場合は `--add-opens=java.base/java.lang=ALL-UNNAMED` を追加してください。

起動確認

起動時に表示されるバナーで設定の適用状況を確認します。バッチホストとバッチエージェントのバナーは異なります。

バッチホスト

```
WhaTap vX.Y.Z YYYYMMDD PCODE(プロジェクトコード)
Java <JDKバージョン> asmlib <バージョン> asm <バージョン>
```

```
whatap.agent-X.Y.Z.jar
./whatap.conf
```

Display	Description
PCODE(数値)	正常。数値はプロジェクトコード
PCODE(0)	license または設定ファイルが適用されていない。トラブルシューティングを参照

3行目にはバッチホストのjarではなく、Javaエージェントのjarファイル名が表示されます。正常な動作です。

バッチエージェント

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

バッチエージェントのバナーには PCODE が表示されません。**4行目に出力された設定ファイルのパス**が、ジョブ別の設定が適用されたかを確認する基準です。意図したジョブディレクトリのパスでない場合は、 whatap.home が想定と異なる形で決定されています。

バナーのJava・asmの値は、プロセスを実行したJDKによって決まります。2つのプロセスを異なるJDKで起動すると、それぞれ異なる値が表示されます。

画面での確認

バッチジョブは、ヒットマップとトランザクション一覧でトランザクションとして確認できます。トランザクション名は job_name に設定した値です。

仮想トランザクション

split_trace_enabled=true の場合、ステップ数が多いジョブはトレースが一定の単位で分割されて送信されます。分割した各区間は仮想トランザクションとして表示され、名前に連番が付きます。**既定値は false のため、既定の設定では仮想トランザクションは表示されません。**

`job_name=sample-job` でステップ数が約5,000の場合の例です。

トランザクション名	内容	表示される所要時間
`sample-job-1` ~ `sample-job-4`	分割した区間(仮想トランザクション)	各区間の処理時間
`sample-job`	実際のトランザクション	ジョブ全体の所要時間

ジョブ全体の所要時間は、連番が付かないトランザクションで確認してください。ステップ番号はジョブ全体で連続します。

- 仮想トランザクションはヒットマップとTPSに反映されますが、統計にはジョブ1件として集計されます。トランザクション件数が多く計上されることはありません。
- 仮想トランザクションはCPU使用時間が0、クライアントIPアドレスが `0.0.0.0` と表示されます。
- トレース分析画面でステップの実行内容をファイルとしてダウンロードする際、トレースが分割されているとファイルも分割単位に分かれます。ジョブ全体のステップ内容が必要な場合は、仮想トランザクションをそれぞれダウンロードしてください。

バッチエージェントのメトリクス指標の収集

ダッシュボードにはバッチホスト(Batch Host)のHeap Memoryを収集し、バッチアプリケーションのパフォーマンス指標はメトリクス(TagCount: `batch_job_counter`)として収集します。

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`

トラブルシューティング

バッチホストの起動バナーにPCODE(0)が表示されます

1. バナーの4行目に出力されたパスに `whatap.conf` があるか確認してください。
2. ない場合は `whatap.home` が想定と異なる形で決定されています。設定ファイルの場所を確認し、必要であれば `-Dwhatap.home` で明示的に指定してください。
3. 設定ファイルがある場合は、 `license` の値が正しいか確認してください。値に問題があると `WhaTap AccessKey Error` が記録されます。

java.net.BindExceptionエラー

UDP 6611がすでに使用中です。バッチホストを強制終了した場合、Javaプロセスが残ってポートを占有し続けることがあります。

- Linux Windows

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

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

残っているプロセスを終了してから、バッチホストを再起動してください。

ジョブごとに設定した内容が反映されません

そのディレクトリが `whatap.home` として認識されていません。起動バナーの4行目に出力されたパスを確認してください。

最も多い原因は、エージェントjarを共有の場所に置き、設定ファイルだけをジョブディレクトリに置いた場合です。 `-javaagent` で指定したjarのディレクトリが `whatap.home` になるため、ジョブディレクトリの `whatap.conf` は読み込まれません。 `-Dwhatap.home` でジョブディレクトリを指定してください。

モニタリングデータが表示されません

次の順序で確認してください。

1. バッチホストが起動しているか確認してください。
2. 起動バナーの `PCODE` が0でないか確認してください。
3. バッチホストから収集サーバーへのTCP 6600が許可されているか確認してください。
4. バッチエージェントからバッチホストへのUDP 6611が通信できるか確認してください。
5. UDPポートを変更した場合、バッチホストの `net_udp_listen_port` とバッチジョブの `net_udp_port` が同じ値か確認してください。
6. `{whatap.home}/logs/whatap.log` にエラーが記録されているか確認してください。

ステップが途中で途切れています

収集の上限に達した場合です。上限に達してもトレースに表示は残らないため、次を確認してください。

確認項目	関連設定
ステップ番号が1,000付近で途切れているか	トレースステップ収集の設定
SQL件数が `trace_sql_limit` と一致するか	収集件数の上限
HTTP呼び出し件数が `trace_httpc_limit` と一致するか	収集件数の上限
トレースの最後に `lost-profile` ステップがあるか	プロファイリング終了待機時間

トランザクションにlost-profileが記録されます

`profile_expired_time` (既定30秒)の間ステップが届かず、プロファイリングを強制終了した状態です。

- ジョブが `kill -9` などで異常終了した場合は、正常な自動整理の結果です。
- ジョブが正常に実行中だった場合は、外部システムの応答待ちなど計測対象の処理が発生しない区間が30秒を超えた可能性があります。プロファイリング終了待機時間を参照して値を調整してください。

自動的に計算される設定

次の値は `time_limit` から自動的に計算されます。設定ファイルに直接記載する必要はありません。

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

適用確認チェックリスト

#	項目	確認方法
1	エージェントの解凍	`{BATCH_HOME}` 配下と `agent` ディレクトリにjarがあります。
2	ホストの設定	`license` と `whatap.server.host` を記載しました。
3	通信要件	TCP 6600アウトバウンドを許可しました。
4	バッチホストの起動	バッチホストのバナーの `PCODE` が0ではありません。
5	エージェントの設定	ジョブごとに `job_name` と `time_limit` を設定しました。
6	ジョブの実行	`-Dwhatap.home` でジョブディレクトリを指定しました。
7	設定の適用	ジョブディレクトリに `logs` が作成されました。
8	データの確認	ヒットマップに `job_name` のトランザクションが表示されます。
9	所要時間の確認	連番が付かないトランザクションでジョブ全体の時間を確認しました。

#	項目	確認方法
10	並列実行	複数のジョブを同時実行し、`job_name`別に区分されて表示されます。
11	終了処理	すべてのジョブ終了後にバッチホストを停止しました。

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイル

の管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

旧バージョンのJava用JVMオプション

Java1.6バージョン未満でもWhaTapエージェントを適用できますが、一部の機能に制約があります。

Java1.5

Java1.5バージョンは、 `-javaagent` オプションを提供します。 ただし、外部スレッドからの情報を参照するには制限があり、一部の機能は制限されます。

次の機能は動作不可

- アクティブTXにおけるリアルタイムトレース
- スタック分析（トップスタック、ユニークスタック、アクティブスタック）
- スレッド一覧の詳細スタック

Java1.4

Java1.4バージョンは、 `-javaagent` オプションがありません。別の方法でWhaTapエージェントをインストールする必要があります。whatap.agent.jarファイルは、Java1.4バージョンを使用する必要があります。

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

boot.jar作成

setup.shファイルを利用して、boot.jarファイルを作成してください。[JAVA_HOME]を入力しない場合、現在のパスは、JDKの場所を自動的に認識します。

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

setup.sh実行結果：whatap.java14.boot- `X.Y.Z` .jar

bootclasspathにインストール

`bootclasspath` オプションを使用し、ビルドされたjarとtracerjarをprependに追加します。

Java実行オプション

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

制約事項

jdk1.4バージョンでは、次の機能は動作しません。

- アクティブTXにおけるリアルタイム特性
- スタック分析（トップスタック、ユニークスタック、アクティブスタック）
- コンポーネントのバージョン
- スレッド一覧、ヒープヒストグラム
- ロードされたクラス
- オープンソケット
- ダンプエージェント

次のステップ

オープンソース追跡

アプリケーションで使用するフレームワークやオープンソースライブラリによって、エージェントオプションの追加が必要になる場合があります。 たとえば、フレームワークやオープンソースでspring-boot-3.x、feign-client-11、okhttp3-4.4使用の際は、次のようにオプションを設定します。

whatap.conf

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

ユーザーのアプリケーション環境に合わせたフレームワークやオープンソースライブラリを設定する方法は、次の

文書を参照してください。

インストールの点検

プロジェクトの作成とエージェントのインストール、JVMオプションまですべて適用したら、次の文書でチェック事項を確認します。

インストール問題の解決

エージェントをインストールする際に発生する可能性がある問題と、これを解決するための具体的なガイドを提供します。 ファイアウォールの設定、SpringBoot及びTomcat JMXの設定、OSGIフレームワーク使用サーバーの設定など、様々な状況に対する解決策を案内します。 詳細については、次の文書を参照してください。

エージェント設定

エージェント設定(whatap.conf)ファイルにオプションを適用して、モニタリングのための様々な機能を提供します。 デフォルト設定から、サーバーの接続とデータ転送の設定、複数のアプリケーションサーバーの設定ファイルの管理方法、トランザクションの追跡などが含まれます。 詳細については、次の文書を参照してください。

モニタリングの開始

すべての設定を完了してから、アプリケーションサーバーを再起動します。 エージェントが情報の収集を開始します。 まず、**アプリケーション**メニューでモニタリングデータが収集されていることを確認します。 **アプリケーション**メニューの詳細については、次の文書を参照してください。

Java Helper

Java Helperはアプリケーションのライブラリを自動で分析し、最適な `weaving` 設定を生成するCLIツールです。Javaアプリケーションで使用するフレームワークやオープンソースライブラリをWhaTapエージェントが追跡するには、`weaving` 設定が必要です。Java Helperはエージェントパッケージに `whatap.javahelper.jar` として含まれており、**JDK 8以上**で動作します。

`weaving` オプションを直接指定するには、手動設定文書を参照してください。

サポートフォーマット

JAR、Spring Boot Fat JAR、Maven Shade JAR、WARフォーマットをサポートします。EARはルートレベルのJARモジュールまで分析します。

Java Helperコマンド

```
java -jar whatap.javahelper.jar conf generate <入力> [オプション]
```

入力

Input	Description	Example
<JARパス>	JARファイルを直接指定して分析	/app/myapp.jar
--pid <PID>	実行中のJavaプロセスのPIDで分析	--pid 12345

JARパスと `--pid` のいずれかを必ず指定してください。

Options

Option	Description	Default
`-o <ファイル>`	マッチング結果をwhatap.confファイルに保存	未指定時は画面出力のみ
`-v`	分析過程の詳細ログを出力 (verbose)	off

使用シナリオ

新規エージェントインストール

JARファイルパスを指定してwhatap.confを直接作成します。

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

生成されたwhatap.confの例:

whatap.conf

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

# 推奨モジュール（必要に応じてコメント解除）
#weaving+=,reactor-kafka-1.3
#weaving+=,resilience4j
```

作成したファイルにライセンスキーとサーバー情報を追加すればインストールが完了します。

運用中の設定点検

実行中のプロセスをPIDで分析してマッチング結果を確認します。

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

出力例:

```
========================================
 WhaTap Weaving Module マッチング結果
========================================

 検出されたライブラリ: 25個
 マッチしたモジュール: 8個
 バージョン不一致:    1個
 自動含有:          3個

 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
 ...
```

現在のwhatap.confと比較して不足しているモジュールがないか確認できます。設定ファイルに保存するには `-o` オプションを追加してください。

ライブラリアップグレード後

Spring Bootのバージョンアップグレードなどライブラリ変更時、アップグレードしたJARで再分析すると新しい `weaving` 設定を取得できます。

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

マッチング状態

Status	Description	Action
`MATCHED`	artifactId、groupId、versionがすべて一致し、weavingモジュール適用可能	自動適用
`AUTO`	自動含有対象モジュール（weaving設定なしでデフォルト適用）	自動適用

Status	Description	Action
VERSION_MISMATCH	ライブラリは検出されたがバージョンがweavingモジュールのサポート範囲外	エージェントアップデート確認
NO_MATCH	weavingレジストリに該当するモジュールなし	該当なし

参考

依存性分析方式

JARファイルからライブラリバージョンを確認する過程は3段階の優先順位で動作します。上位段階で抽出した情報は下位段階で上書きされません。

1. **Maven pom.properties**: JAR内部の `META-INF/maven/{groupId}/{artifactId}/pom.properties` から `groupId`、`artifactId`、`version` を抽出します。最も正確な方法です。
2. **MANIFEST.MF**: pom.propertiesがない場合、`META-INF/MANIFEST.MF` から `Implementation-Title`、`Implementation-Version` などの標準属性を読み取ります。
3. **ファイル名パーシング**: 上記2つの方法が失敗した場合、JARファイル名から推論します。（例: `jackson-databind-2.14.0.jar` → `version="2.14.0"`）

Weavingプラグイン設定

whatap.confファイルに `weaving` オプションを直接指定してフレームワークとオープンソースライブラリを追跡します。ライブラリを自動で分析して設定を作成するには、自動設定 (Java Helper)文書を参照してください。

例えば、`spring-boot-3.x` 、`feign-client-11` 、`okhttp3-4.4` を使用する場合、次のようにオプションを設定してください。

whatap.conf

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

追跡可能なオープンソースリスト

Javaエージェントを通じてトレースしているフレームワークまたはオープンソースに対する設定方法は、次を参照してください。

Apache Camel

CXF

camel-cxf Version	設定値	エージェント最小バージョン
3.15 以上	weaving=camel-cxf-3.15	v2.2.27

Netty

camel-netty4 Version	設定値	エージェント最小バージョン
2.25 以上	weaving=camel-netty4-2.25	v2.2.42

SEDA

camel-seda Version	設定値	エージェント最小バージョン
2.22 以上	weaving=camel-seda-2.22	v2.2.20
3.2 以上	weaving=camel-seda-3.2	v2.2.20

DynamoDB

DynamoDB Version	設定値	エージェント最小バージョン
1.11	weaving=dynamodb-1.11	v2.2.39
2.25	weaving=dynamodb-2.25	v2.2.39

Feign Client

Feign Version	設定値	エージェント最小バージョン
11 以上	weaving=feign-11	v2.2.6

Hystrix

Hystrix Version	設定値	エージェント最小バージョン
1.5 以上	weaving=hystrix-1.5	v2.0_21

Kafka

Kafka Version	設定値	エージェント最小バージョン
kafka-clients 2.4.0 以上	weaving=kafka-clients-2.4.0	v2.2.15
reactor-kafka 1.3 以上	weaving=reactor-kafka-1.3	v2.2.5

Logging

Log4j2

Log4j2 Version	設定値	エージェント最小バージョン
2.17 以上	weaving=log4j-2.17	v2.2.28

> ⓘ 特定appenderを除外するエージェントオプション提供

Logback

Logback Version	設定値	エージェント最小バージョン
1.2.8 以上	weaving=logback-1.2.8	v2.2.28

> ⓘ 特定appenderを除外するエージェントオプション提供

MongoDB

MongoDB Version	設定値	エージェント最小バージョン
3.8.2 以上	weaving=mongodb-3.8	v2.2.11
4.0.3 以上	weaving=mongodb-4.0	v2.2.11
4.4 以上	weaving=mongodb-4.4	v2.2.11
4.8 以上	weaving=mongodb-4.8	v2.2.11
4.11 以上	weaving=mongodb-4.11	v2.2.45
5.0 以上	weaving=mongodb-5.0	v2.2.45

Mule Framework

Mule Version	設定値	エージェント最小バージョン
3.9.5 以上	weaving=mule-3.9.5	v2.2.23
4.5 以上	weaving=mule-4.5	v2.2.23

OkHttp

OkHttp Version	設定値	エージェント最小バージョン
2.7 以上	weaving=okhttp-2.7	v2.0_15
3.x	weaving=okhttp3	v2.0_15
3.x (4.4 以上)	weaving=okhttp3-4.4	v2.2.9

Quarkus Reactive

Quarkus Reactive Version	設定値	エージェント最小バージョン
1.13 以上	weaving=quarkus-reactive-1.13	v2.2.19
2.10 以上	weaving=quarkus-reactive-2.10	v2.2.19

RabbitMQ

reactor-rabbitmq Version	設定値	エージェント最小バージョン
1.2 以上	weaving=reactor-rabbitmq-1.2	v2.0_06

Redis

Jedis

Jedis Version	設定値	エージェント最小バージョン
2.9.0 以上	weaving=jedis-2.9	v2.2.43
2.9.3 以上	weaving=jedis-2.9	v2.0_33
3.2 以上	weaving=jedis-3.2	v2.0_09

Lettuce

Lettuce Version	設定値	エージェント最小バージョン
5.1 以上	weaving=lettuce-5.1	v2.2.7

Lettuce Version	設定値	エージェント最小バージョン
6.2 以上	weaving=lettuce-6.2	v2.2.16

Retrofit

Retrofit Version	設定値	エージェント最小バージョン
2.5 以上	weaving=retrofit-2.5	v2.2.39

> okhttp設定と同時使用不可

Ribbon

Ribbon Version	設定値	エージェント最小バージョン
全バージョン	weaving=ribbon	v2.2.10

Spring Boot

Spring Boot Version	設定値	エージェント最小バージョン
2.1 以上	weaving=spring-boot-2.1	v2.2.23
2.5 以上	weaving=spring-boot-2.5	v2.2.9
2.7 以上	weaving=spring-boot-2.7	v2.2.9
3.0 以上	weaving=spring-boot-3.0	v2.2.9
3.2 以上	weaving=spring-boot-3.2	v2.2.38

Spring Boot Version	設定値	エージェント最小バージョン
4.0 以上	weaving=spring-boot-4.0	v2.2.69

含まれる技術スタックを表示

- **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	設定値	エージェント最小バージョン
9.x	weaving=tomcat9	v2.2.5
10.x	weaving=tomcat10	v2.2.5

Undertow

Undertow Version	設定値	エージェント最小バージョン
2.2 以上	weaving=undertow-2.2	v2.2.39

Undertow Version	設定値	エージェント最小バージョン
2.3 以上	weaving=undertow-2.3	v2.2.14

Vert.x

Vert.x Version	設定値	エージェント最小バージョン
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エージェントのCVE誤検知防止

Javaエージェントに含まれるライブラリクラスは、アプリケーションで実際に使用する時のみロードされます。CVE脆弱性検査時にライブラリが検出されても、使用していなければ実際の脆弱性はありません。

不要なトレースクラスを削除するには、次のコマンドを実行してください。

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

Javaエージェントアップデート時には、削除作業を再度実行する必要があります。

特定のライブラリクラスを削除するコマンドは、**Javaエージェント2.2.37バージョン**以上でサポートされます。

使用法:

```
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` または `-remove` : Javaエージェントからweaving jarファイルを削除し、新しいJavaエージェントを生成します。
- `-l` または `-list` : weaving jarファイルリストを出力します。
- `-d` または `-debug` : デバッグロギング(debug logging)を有効化します。
- `-e` または `-error` : エラーに対するフルスタックトレース情報を表示します。
- `-h` または `-help` : ヘルプを出力して終了します。

Javaインストールの点検

エージェントをインストールした後の確認事項です。もし、インストールに問題がある場合は、次の文書を参照してください。

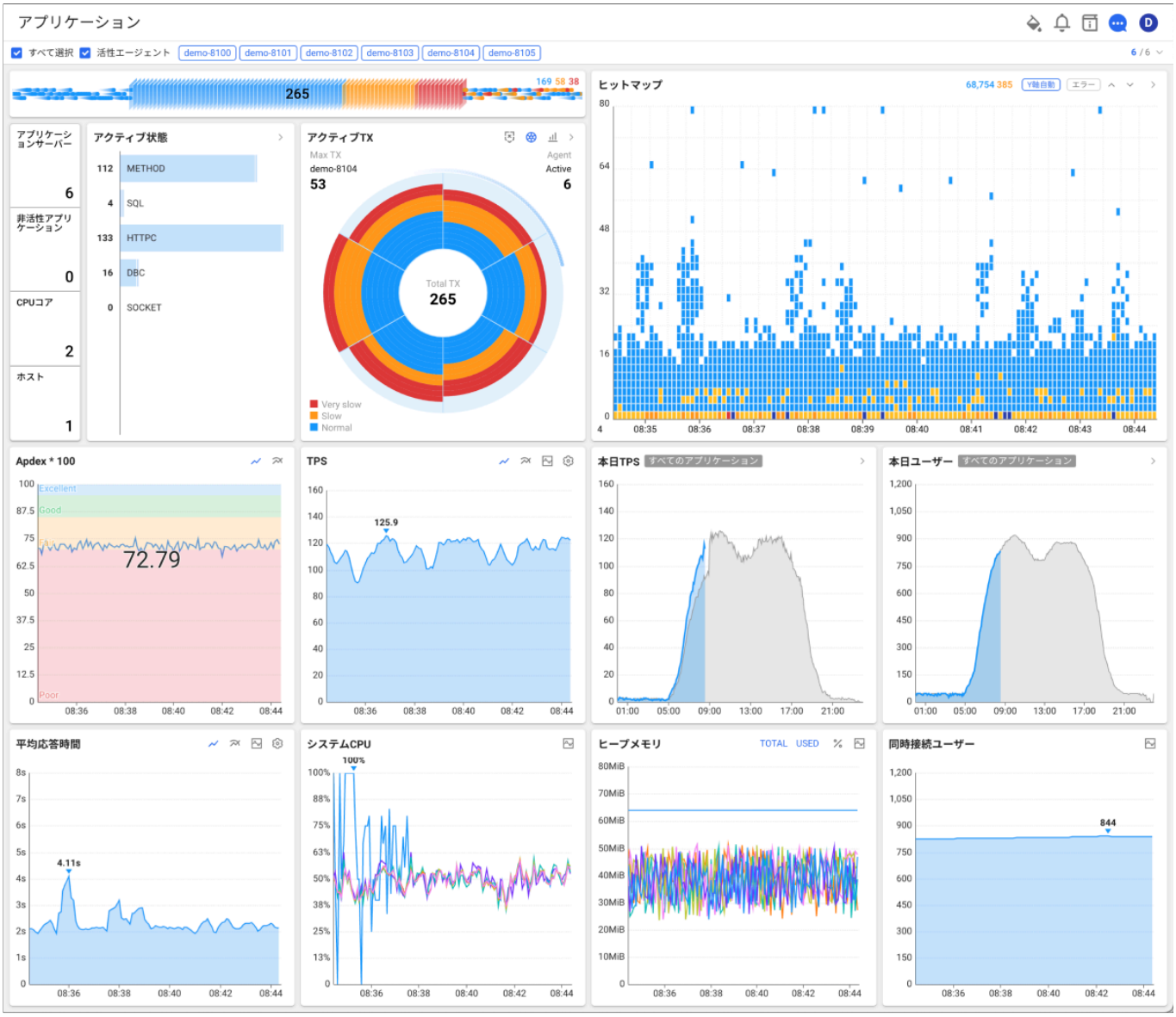
WAS再起動

アプリケーションサーバーを再起動してください。動作ログに次のようなWhaTapキャラクターログが表示された場合は、`JAVA_OPTS` 設定は完了です。

```
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
```

エージェント接続の確認

WhaTapモニタリングサービスに移動します。ログイン後、プロジェクトの一覧で作成したプロジェクトを選択します。**アプリケーション**メニューで接続したサーバーのデータがウィジェットを通じて表示されたら、すべての設定が完了したことになります。



ⓘ エージェント名の既定値は `{type}-{ip2}-{ip3}-{port}` です。 エージェントの名前を変更またはプロパティを追加するには、次の文書を参照してください。

Javaエージェントのトラブルシューティング

ファイアウォール設定の確認

WhaTapのデータ収集サーバーにTCPアウトバウンドポリシーを設定しない場合、モニタリング情報を送信できません。 **ファイアウォール遮断を解除**してください。

`telnet` コマンドを使用して、"Connected to `IP` "を確認します。

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

> ⓘ 収集サーバーの情報は、WhaTapモニタリングサービスからプロジェクトを選択した後、管理 > エージェントインストール メニューから確認できます。

SpringBoot2.2以降、Tomcat JMXのOFF状態

SpringBoot 2.2バージョン以降でEmbedded Tomcatを使用する場合、既定値としてJMX関連の機能が動作しません。この場合、JMX機能を有効してください。関連リンク を参照してください。

application.properties

```
spring.jmx.enabled=true
server.tomcat.mbeanregistry.enabled=true # tomcat embeddedの場合 #
```

application.yml

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

```
tomcat:
  mbeanregistry:
    enabled: true #tomcat embeddedの場合
```

OSGI フレームワーク対応アプリケーション・サーバーの場合

OSGIフレームワーク構造を持つアプリケーション・サーバーの場合は、JVMオプションにエージェント・パッケージプリフィックス(whatap)を登録します。

JBoss EAP6.0以降、 Jboss AS7.0以降、 Wildfly 8.0以降、IBM WebSphere AS 7.0以降が含まれます。

JBoss AS, Wildfly, JBoss EAP6.0以降

次のファイルにプレフィックスを登録します。

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

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

JBOSS EAP 7.0

WebSphere

JVM オプションとして、以下を追加します。

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

- 既定値として '*' で指定されている場合、設定は必要ありません。
- 設定位置は、WebSphereを参照してください。

security.policy権限を次のように追加します。

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

```
grant codeBase "file:$WHATAP_HOME/-"
{
```

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

Log Manager関連エラー

JBoss ASバージョン7.0以降、JBoss EAPバージョン6.0以降でLog Manager関連エラーが発生した場合、JVM オプションを追加します。

- `-Djava.util.logging.manager` 項目にLog Manager package名を設定してください。
- `-Xbootclassloader` 項目にJoboss Log Manager JAR fileを設定してください。

(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エラーが発生した場合

JBoss5.0以下のバージョンからMBeanServerBuilder関連エラーが出力された場合、JVMオプションを追加します。

(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エラーが発生した場合

Java Security Policyに関連するエラーが発生すると、_$JAVA_HOME/jre/lib/security/java.policy_ファイルに権限設定を追加します。

すべての権限を一括で適用するには、次のようにjava.policyファイルに設定を追加します。

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

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

java.io.FilePermissionエラーが発生した場合

```
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)
```

次のようにjava.policyファイルに設定を追加します。

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

```
grant {
  ...
  permission java.io.FilePermission {エラーメッセージから確認されたパッケージパス}, "read"
};
```

java.util.PropertyPermissionエラーが発生した場合

次のようにjava.policyファイルに設定を追加します。

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

```
grant {
  ...
  permission java.util.PropertyPermission {エラーメッセージから確認されたパッケージパス}, "read"
};
```

Sigar libraryをロードできない場合

$WHATAP_HOME/lib1/*.soファイルに実行権限があることを確認してください。アクセス権限がない場合は、実行するアクセス権限を付与してください。

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

AIX7で_$WHATAP_HOME/lib1_以下の階層にlibsigar-ppc64-aix-7.soファイルが存在しないエラーが発生した場合は、そのファイルをコピーしてください。

```
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指標が-1で表示される場合

最新バージョンのJavaエージェントを使用する環境でIBM Javaを実行する時、CPU指標が**-1**と表示される場合があります。この問題は、特定のプラットフォームでSystem Load Average**値に対応していないために発生します。

Javaエージェントは基本的にJava Management Extensions(JMX)を活用して指標を収集します。`OperatingSystemMXBean`インターフェースの`getSystemLoadAverage()`メソッドを使用します。このメソッドは、オペレーティング システムに応じて、システムの最近1分間の平均負荷を返します。しかし、一部のプラットフォームではその値に対応していないため**-1**を返還できます。

> ⓘ `getSystemLoadAverage()`メソッドはOpenJDKソースコードで確認できます。例えば、OpenJDK 7uバージョンのソースコードはGitHubリポジトリで確認できます。

推奨事項: 組み込みライブラリによるメトリック収集の設定

CPU指標が**-1**と表示される現象を解決するには、Javaエージェント設定により組み込みライブラリ(OSHIまたはSIGAR)を活性化します。

- **Java 8 以降環境**: **OSHI**ライブラリを活用

  オペレーティングシステムレベルの指標を安定的に収集するために**OSHI**ライブラリを使用できます。エージェントの設定方法については、以下を参照してください。

  whatap.conf

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

- **Java 8 未満環境**: **SIGAR**ライブラリを活用

  Java 8 未満の環境では**SIGAR**ライブラリを活用して指標を収集できます。エージェントの設定方法については、以下を参照してください。

  whatap.conf

  ```
  # default false
  ```

```
sigar_enabled=true
```

OSHI(oshi_enabled)と**SIGAR**(sigar_enabled)オプションは既定値として非活性化されています。必要に応じて明示的に設定する必要があります。

- 両方の設定を同時に活性化しないように注意します。環境に合ったライブラリを選択して設定します。

Javaエージェント設定

WhaTapエージェントは、エージェントごとに必要な設定をwhatap.confファイルに作成します。 エージェントは、環境変数を使用して設定ファイルの位置を把握して読み込みます。

他の設定がない場合、エージェントは(whatap.agent-X.Y.Z.jar)にあるディレクトリからwhatap.conf ファイルを実装します。5秒ごとにwhatap.confファイルの確認してリロードします。

ユーザーの利便性を高めるため、WhaTapモニタリングサービスでもエージェント設定機能を提供しています。

基本設定

whatap.conf

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

- whatap.home ：既定値はwhatap.agent.jarファイルへのパスを入力します。
- whatap.config : ファイル名のみを入力することをお勧めします。相手パスを一緒に入力することができます。

エージェントがデータを収集し、サーバーにデータを送信するには、アクセスキーとサーバーアドレスが設定されている必要があります。

whatap.conf

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

WhaTapエージェントを初めて実行すると、次のようにどのjarファイルがエージェントとして使用され、どのパスでwhatap.confファイルをロードされたのかを確認できます。



サーバー接続およびデータ送信

WhaTapエージェントは、whatap.confファイルに設定したサーバーアドレスに接続します。

whatap.conf

```
whatap.server.host=10.10.1.1
whatap.server.port=6600 # 既定値
```

- TCPセッションを接続すると `license` を利用してサーバーから通信キーを受け取ります。誤った `license` を設定すると、サーバーはセッションを終了します。
- TCP接続が繰り返し終了する場合は、ファイアウォールの確認または `license` の値が正しいことを確認してください。
- TCPセッションを接続すると、サーバーから受信した秘密鍵に基づいてセキュリティで保護された通信データが送信されます。

複数のアプリケーションサーバーの設定ファイル管理

1つのサーバーで複数のアプリケーションを実行する場合、各アプリケーション別の設定ファイルを個別に管理する場合は、whatap.confファイル名を変更し、起動スクリプトにJVMオプションを追加します。

Service A

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

Service B

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

> whatap.conf設定ファイルは、 ${WHATAP_HOME} パスに配置する必要があります。

サービス画面でエージェントを設定する

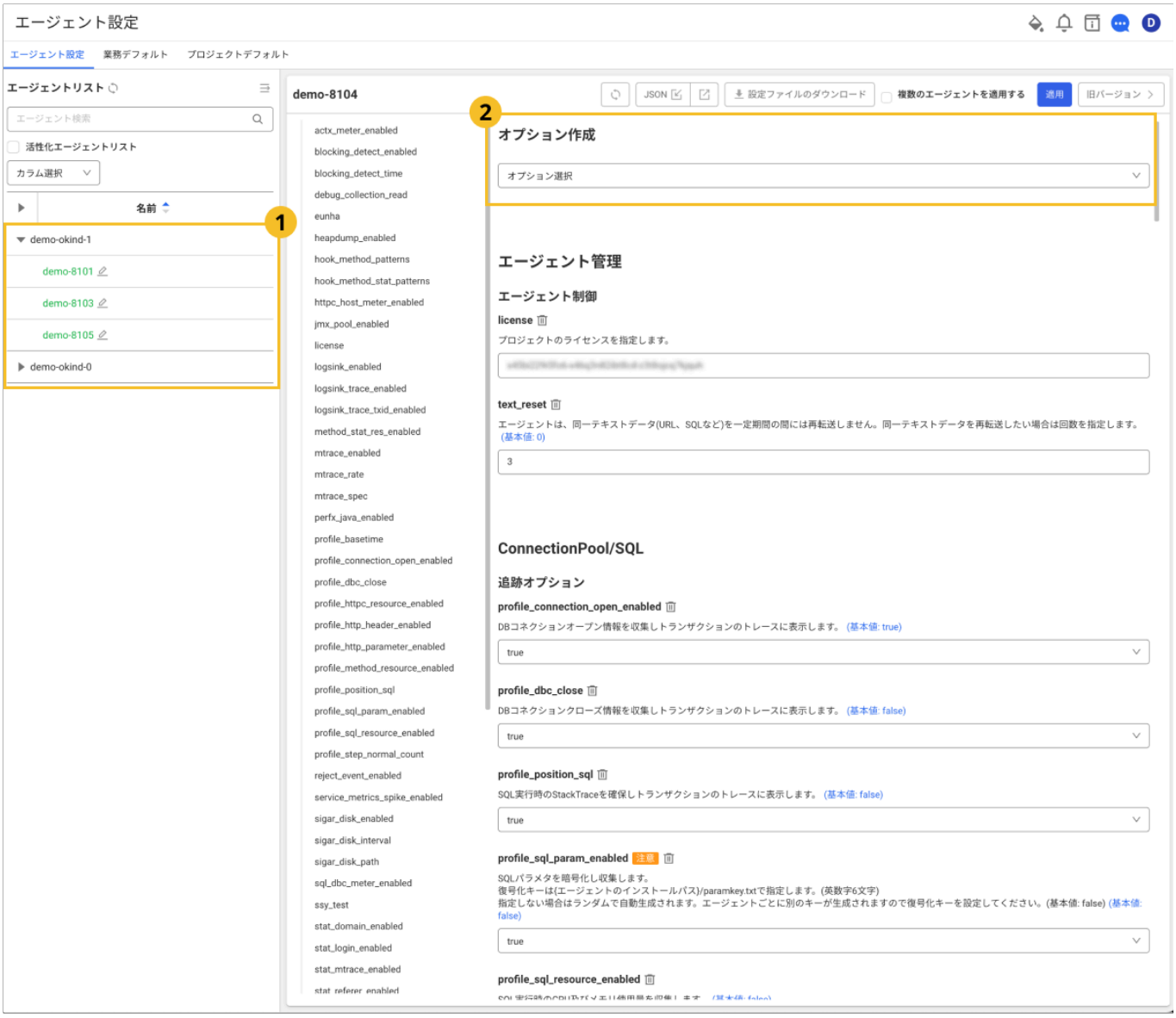
ホーム画面 > プロジェクト選択 > **管理** > **エージェント設定**

モニタリング対象サーバーにあるwhatap.confファイルを修正することなく、WhaTapモニタリングサービスからエージェント設定のオプションを追加または修正、削除することができます。また**設定ファイルのダウンロード**ボタンをクリックしてwhatap.confファイルをダウンロードできます。

- この機能は、**修正**権限を持つメンバーのみ利用できます。**修正**権限のないメンバーは、設定内容の表示のみ可能です。
- オプション値として設定できる形式は次のとおりです。
  - Boolean形式の値は、 true または false を選択します。
  - 数値の値は数字のみ入力できます。
  - テキスト(String) 形式の値を入力または修正する場合は、オプションの説明を詳しく確認してください。
- 修正できないオプションは選択できません。（例、 license ）

- 追加または修正、削除したオプションによっては、エージェントの再起動が必要になる場合があります。
- アプリケーションの種類やエージェントのバージョンによって、適用できるオプションキーが異なる場合があります。

オプションを追加する



1. **エージェントリスト**からオプションを追加する1エージェントを選択します。

2. ②**オプション作成**から追加するオプションを選択します。
   - **検索**で追加するオプションを確認できます。テキストを入力すると、一致するオプションをフィルタリングします。
   - **直接入力**を選択すると、オプションキーと値を入力できます。
3. 選択したオプションキーの説明と既定値を確認し、設定値を入力します。

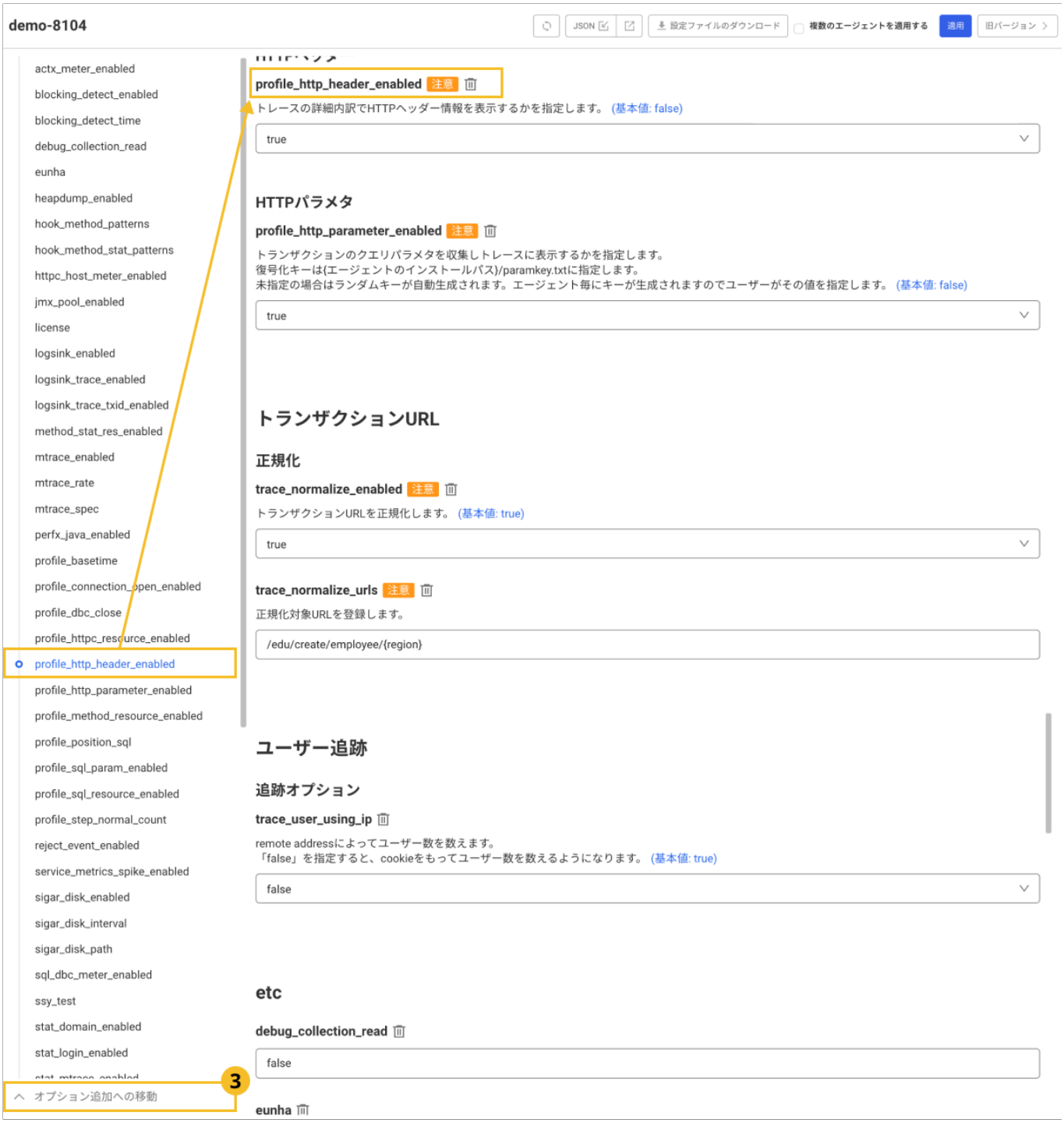


   - 選択したオプションをキャンセルするには🗑ボタンを選択します。
   - オプションを追加設定するには**追加する**ボタンを選択し、上記2の手順を繰り返します。
4. 必要なすべてのオプションを追加した場合は、画面右上の**適用**ボタンを選択します。

選択したオプションと設定値をエージェントに適用します。

- 変更内容を初期化するには、画面右上の↻ボタンを選択します。
- オプション値に何も入力せずに**適用**ボタンを選択すると、該当オプションを削除します。
- すでに追加したオプションは、オプション一覧から選択することはできません。
- アプリケーションの種類やエージェントのバージョンによって、適用できるオプションキーが異なる場合があります。
- 例として提供された画像は、アプリケーションの種類やエージェントによって異なる場合があります。

オプションを修正・削除する



1. 画面を上下にスクロールするか、左側のオプション一覧から修正または削除するオプションを選択します。
2. 変更するオプションから目的の値を選択または修正してください。オプションを削除するには🗑ボタンを選択します。
3. 変更した事項を適用するには**適用**ボタンを選択します。

> ⓘ
> - 画面の上部に移動するには、3 **オプション追加への移動**ボタンを選択します。
> - オプション値に何も入力せずに**適用**ボタンを選択すると、該当オプションを削除します。

複数のエージェントに同時適用する

プロジェクトに所属する複数のエージェントに変更したオプションを同時に適用できます。

1. 画面右上の**複数のエージェントを適用する** チェックボックスを選択すると、各オプション項目にチェックボックスが表示されます。
2. 同時に適用するオプションのチェックボックスを選択します。複数選択できます。
3. 画面右上の**適用**ボタンを選択します。
4. **エージェント 適用**画面が表示されたら、変更したオプションを適用するエージェントを選択します。すべてを選択するには、**すべて選択**チェックボックスを選択します。



5. **適用**ボタンを選択します。

エージェントの既定値を設定する

プロジェクトに新しいエージェントを追加する場合、既存の設定値を繰り返し適用する手間を省くには、プロジェクト別、業務別の基本設定値を作成して適用します。



- **業務デフォルト**：whatap.confファイルから `whatap.okind` 項目で分類されたエージェントにオプションを適用できるし、適用されたオプションを確認できます。
- **プロジェクトデフォルト**：プロジェクトに所属するすべてのエージェントに共通のオプションを適用できるし、適用されたオプションを確認できます。

> ⓘ
> - **業務デフォルト**または**プロジェクトデフォルト**タブを選択し、オプションを適用するのは、エージェントをまとめて管理できる方法です。
> - オプションの優先順位は、**業務デフォルト**、**プロジェクトデフォルト**、**エージェント設定**の順です。

- **プロジェクトデフォルト**で、bオプション値に1000を適用し、**業務デフォルト**で、bオプション値に2000を適用すると、**業務デフォルト**で設定したオプション値の2000を優先します。
- **プロジェクトデフォルト**で、bオプション値に入力せずに、**業務デフォルト**の bオプション値に2000が適用された場合でも、**業務デフォルト**で設定したオプション値が優先されます。

エージェント設定を共有する

エージェントの設定内容を json形式のファイルとして保存し、そのファイルを読み込んで他のエージェントに適用することができます。

1. **エージェントリスト**で設定内容をJSONファイルに書き出しするエージェントを選択します。
2. 画面右上の↗ボタンを選択します。
3. jsonファイルをダウンロードしてください。
4. **エージェントリスト**で他のエージェントを選択します。
5. 画面右上の↙ボタンを選択します。

jsonファイルを通じて読み込んだエージェント設定を確認してください。

エージェント設定オプションの案内

次のリンクは、設定できるオプションについて説明します。

エージェント名

Javaエージェント名を設定してモニタリング対象を識別する方法を案内します。

機能制御

Javaエージェントの機能を有効または無効にする設定オプションを案内します。

エージェント使用量

Javaエージェントが使用するシステム資源量を確認する方法を案内します。

通信設定

Javaエージェントと収集サーバー間のネットワーク通信を設定する方法を案内します。

性能設定

Javaエージェントの性能関連オプションを設定しオーバーヘッドを最適化する方法を案内します。

ログ設定

Javaエージェントのログファイルの場所とログレベル設定オプションを案内します。

トランザクション設定

Javaエージェントのトランザクション追跡とプロファイリング関連の設定オプションを案内します。

DB/SQL

Javaエージェントのデータベース接続とSQLトラッキング関連の設定オプションを案内します。

HTTPC/API Call

Javaエージェントの外部HTTP呼び出しとAPI連携トラッキングの設定オプションを案内します。

スクリプトプラグイン

Javaエージェントにユーザー定義スクリプトを追加しモニタリングを拡張する方法を案内します。

ユーザー数

Javaエージェントの同時ユーザー数集計方式と関連設定オプションを案内します。

負荷制御

Javaエージェントのスロットリングと負荷制御関連の設定オプションを案内します。

通知設定

Javaエージェントで発生するイベントの通知関連の設定オプションを案内します。

Apdex

JavaエージェントのApdexスコア設定オプションを案内します。

統計

Javaエージェントの統計データ収集関連の設定オプションを案内します。

エラースタック

Javaエージェントのトランザクションエラー発生時のスタックトレース収集設定を案内します。

追加オプション

Javaエージェントの追加設定オプションについて案内します。

非同期トラッキング

Java非同期呼び出しのトランザクション連結とトラッキング設定方法を案内します。

エージェント名の識別

WhaTapはモニタリングは、情報収集の対象であるアプリケーションサーバーを識別するための情報で、基本的にアプリケーションサーバーから収集した情報を使用します。基本的な使用状況の情報は、アプリケーションサーバーの種類、IPアドレスなどを組み合わせであり、アプリケーションサーバーを固有識別子として使用します。必要に応じて、ユーザーが設定した名前を使用するか、パターンを変更して固有の値に変更して使用します。エージェントの名前は固有の値である必要があります。

アプリケーションサーバーから抽出した情報を使用する理由は、アプリケーションサーバーの停止またはネットワークの切断、エージェント問題による収集サーバーとエージェントの通信状態が復旧した場合に、再接続されたエージェントから送信される情報が既存エージェントから送信された情報との継続性を維持するためです。

- エージェント名は、プロジェクトごとに一意である必要があります。
- エージェントID／名前を変更しても、古いデータには関連付けられません。

- WhaTapモニタリンサービスでは、モニタリング対象をオブジェクトまたはエージェントとして混在して使用できます。

デフォルト名の決定方法

JVMオプションで `whatap.oname` 設定がない場合、エージェント名に次の優先順位で自動的に割り当てられます。

1. `whatap.type` を特定します。
2. `whatap.type` がサーブレットコンテナの場合、アプリケーションが使用するポートを識別します。
3. ポートを特定できない場合は、process ID(PID)を検索します。
4. 識別された情報にIPv4アドレスの3番目と4 番目を追加して、最終的に名前を決定します。

例えば、IPアドレスが10.11.**12**.**13**のサーバーで**Tomcat**が**8080**番ポートで起動した場合、 `TC-12-13-8080` 名でエージェント名が自動的に割り振られます。

エージェント名の変数

使用可能な名前の変数を次のように指定します。

変数	説明
文字列	入力した文字列を使用します。
{type}	コンテナタイプを自動的に識別して割り当てます。 自動識別の対象：JAVA、 TOMCAT、 JBOSS、 RESIN、 SPRINGBOOT、 JETTY、 JEUS、 WEBLOGIC、 WEBSPHERE、 UNDERTOW、 PLAY
{ip0}	IPv4アドレスのうち、第1オクテットを使用します。(例：**10**.11.12.13のうち**10**)
{ip1}	IPv4アドレスのうち、第2オクテットを使用します。(例：10.**11**.12.13のうち**11**)
{ip2}	IPv4アドレスのうち、第3オクテットを使用します。(例：10.11.**12**.13のうち**12**)
{ip3}	IPv4アドレスのうち、第4オクテットを使用します。(例：10.11.12.**13**のうち**13**)
{pid}	JavaのProcess ID(PID)を使用します。
{port}	アプリケーションがListenするポートを使用します。

エージェント名の設定

起動スクリプトにJVMオプションを追加して、自動的に指定する名前の代わりに独自の名前を設定できます。

オプション	設定位置	説明
-Dwhatap.name	JVM Options	エージェント名をパターンで設定します。 既定値 {type}-{ip2}-{ip3}-{port}
-Dwhatap.oname	JVM	エージェント名を固定値で設定します。

オプション	設定位置	説明
	Options	-Dwhatap.oname=java-sales-01 オプションを設定しないと-Dwhatap.nameオプションに設定されたパターンが適用されます。

IPアドレスが10.11.12.13で、ポート番号が8000の環境で

- **-Dwhatap.onameオプションを適用しない場合：whatap.oname**は、 -Dwhatap.name に設定されたパターンによって JAVA-12-13-8000 に設定されます。
- **-Dwhatap.onameオプションを適用した場合**: -Dwhatap.name オプションを無視して -Dwhatap.oname オプションに設定した値でエージェント名が適用されます。

- クラウドのScale Out環境では、VMをコピーして使用できます。したがって、オブジェクト名が自動的に付与されることが重要です。
- 2つのVPC環境を作成し、同じprivate IPを使用するようにネットワークを構成した場合、2つのVPCが重複しないようにする必要があります。それらを別々のプロジェクトに分割したり、 -Dwhatap.name オプションと名前変数を異なる方法で設定できます。

次は、whatap.confファイルにエージェント名で適用できるオプションです。

- **whatap.name** String

  既定値 {type}-{ip2}-{ip3}-{port}

  エージェント名をパターンとして設定します。

- **auto_oname_enabled** Boolean

  既定値 false

  サーバーからエージェント名(whatap.oname)を自動的に受信してサーバーに登録する機能をアクティブにします。 このオプションを適用する場合、JVMオプションとして設定した -Dwhatap.name 、 -Dwhatap.oname の値は無視します。 収集サーバーとの通信により、 whatap.oname 値が与えられ、エージェントの通常の動作を開始します。

- **auto_oname_prefix** String

  既定値 agent

エージェント名をサーバーから自動的に割り当てる場合は、エージェント名のprefixや業務名を使用します。prefixにシリアル番号「1~)」を付与します。

> ⓘ エージェント名を自動的に割り当てる「auto_oname_enabled」オプションは、使用する前に有効(「true」)にする必要があります。

- **auto_oname_reset** Int

  既定値 `0`

  サーバーから新しいエージェント名を付与するために修正します。エージェント名を自動付与すると、`whatap.oname` というシステム環境変数に設定されます。一度定められたシステムの環境変数は、JAVAインスタンスが再起動するまで維持します。リセットしたいときは、`auto_oname_reset` 値を修正してください。現在の設定値と異なる値に変更すると適用されます。

- **whatap.okind** String

  複数のエージェントを'種類別'グループにまとめて、モニタリング単位を設定できます。 このオプションにより、複数のエージェントをグループ単位でのモニタリングと管理ができます。また、性能指標の分析や通知の設定もできます。

  > ⓘ **ダッシュボード**及び**分析**、**統計**メニューなどで**種類別**に分類されたグループオプションで確認できます。

- **whatap.onode** String

  複数のエージェントを'ノード別'グループにまとめて、モニタリング単位を設定できます。 このオプションにより、複数のエージェントをグループ単位でのモニタリングと管理ができます。また、性能指標の分析や通知の設定もできます。

  > ⓘ **ダッシュボード**及び**分析**、**統計**メニューなどでノード別に分類されたグループオプションで確認できます。

エージェント機能の制御

Javaアプリケーションサーバーのモニタリングのためのエージェント機能を細かく調整する方法を案内します。エージェントの活性化・非活性化、トランザクションと性能カウンターの追跡、OS情報収集などの機能を含み、各機能を制御する様々な設定オプションを確認できます。

- **shutdown** Boolean

  既定値 `false`

  `true` 場合は、エージェントのすべての動作を停止し、サーバーとの接続を終了します。

- **enabled** Boolean

  既定値 `true`

  機能のすべてを有効にします。 `false` の場合でも、サーバーと最小限の通信を維持するために情報が送信されます。

- **transaction_enabled** Boolean

  既定値 `true`

  トランザクション追跡機能を有効にします。ヒットマップに記録されるトランザクション情報に対応します。 `enabled` オプションが `'false` の場合、無効にします。

- **counter_enabled** Boolean

  既定値 `true`

  パフォーマンスカウンターを追跡する機能を有効にします。アクティブTX数およびユーザ数、JVMリソースの使用量、Process CPUの使用量、DB Poolの使用量情報などが含まれます。 `enabled` オプションが `false` の場合、無効にします。

- **stat_enabled** Boolean

  既定値 `true`

  統計情報を追跡する機能を有効にします。5分単位で収集するトランザクション、SQL、HTTPCALL、UserAgent、Client IPなどの統計データなどが含まれます。 `enabled` オプションが `false` の場合、無効にします。

- **sigar_enabled** Boolean

  既定値 `false`

sigarライブラリによるOS情報の収集を有効にします。sigarライブラリを通じて5秒単位で収集するCPU、Memory、DiskなどのOSリソースデータが含まれます。 `enabled` オプションが `false` の場合、無効にします。

> ◦ whatap.agent.2.1.0バージョン以下では既定値は `true` です。whatap.agent.2.1.0バージョン以降では既定値は `false` です。
>
> ◦ Java 1.6以下のバージョン環境では、 `sigar_enabled=true` オプションを追加してsigarライブラリを有効にし、パフォーマンス指標を収集できます。

- **active_stack_enabled** Boolean

  既定値 `true`

  アクティブスタックの追跡を有効にします。スタックメニューのトップスタック、ユニークスタック、アクティブスタックが対象です。 `enabled` または `counter_enabled` オプションが `false` の場合、無効にします。

- **license** String

  エージェントのインストール時にサーバーから付与されたアクセスキーを入力します。 アクセスキーは、エージェントが属するプロジェクトとセキュリティ通信のためのパスワードキーが含まれます。

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

  既定値 `128`

  AESセキュリティアルゴリズムの暗号レベルを設定します。256ビットまで拡張できます。AES256暗号化の設定については、次の文書を参照してください。

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

  既定値 `2`

  WhaTapエージェントは、サーバーにデータを送信する際、データ属性に基づいて選択的に暗号化します。高いセキュリティを維持しながらも、パフォーマンス上の利点を持っています。データ タイプに関係なく、一括した暗号化ポリシーを適用するには、次のオプションを参照してください。

  ◦ `1` ：暗号化送信機能を無効

  ◦ `2` ：SQLパラメータ、Plain Textなどの機密属性の暗号化された転送

  ◦ `3` ：すべての項目の暗号化送信

- **stat_ip_enabled** Boolean

  既定値 `true`

プロジェクトのサイドメニューの中、**統計** > **クライアントIP**のメニューでIP統計の使用有無を有効にします。

> ⓘ 次の文書を参考にしてください。

- **stat_ipurl_enabled** Boolean

  既定値 false

  プロジェクトサイドメニューの中で**統計** > **IP Transaction**統計の収集有無を設定します。

  > ⓘ 次の文書を参照してください。

- **realtime_user_enabled** Boolean

  既定値 true

  プロジェクトのサイドメニューの中、**ダッシュボード** > **アプリケーション**メニューで確認できる指標のうち、**同時接続ユーザー**指標の収集有無を設定します。

- **hook_direct_patch_classes** ClassFile_FullPath

  特定のクラスをロードタイムに置き換えるときに直接使用します。クラスをコンパイルしてから別のファイルを作成し、そのファイルの完全パスを設定します。

- **active_stack_second** Seconds

  既定値 10

  アクティブスタックを追跡する間隔を設定します。

  > ⚠ 指定された既定値を変更しないことをお勧めします。

- **boot_redefine_size** Int

  既定値 100

  エージェントをAttach方式またはWatcher方式でインストールされている場合、すでにロードされているクラスの追跡のためにBCIを新たに実行します。このとき、同時にredefineするクラスの数を設定します。

- **trace_component_enabled** Boolean

  既定値 true

プロジェクトのサイドメニューの中、**インスタンス性能管理** > **ライブラリバージョン**のメニュー機能を有効にします。

- **realtime_user_thinktime_max / wclient_thinktime_max** MiliSeconds

  既定値 `300000`

  リアルタイムにブラウザのユーザー数を測定する場合、同じユーザーとして認識される呼び出し間隔の最大値を設定します。

  > ⓘ Javaエージェント 2.2.0 バージョン以降の場合は、 `wclient_thinktime_max` オプションを使用します。

- **time_sync_interval_ms** MiliSeconds

  既定値 `300000`

  エージェントとサーバー間で同期化する時間の間隔を設定します。同期しない場合は、 `0` に設定します。

- **detect_deadlock_enabled** Boolean

  既定値 `false`

  Javaスレッドがデッドロック(DeadLock)しているかどうかをチェックして検知する場合、イベントを発生させます。発生間隔は5秒単位です。同じDeadLockに対するイベントは1時間に1回だけ発生します。

- **text_reset** Int

  既定値 `0`

  WhaTapエージェントは、送信されるテキストタイプのデータをhash処理するため、翌日まで再送信されません。以前の設定値と異なる値を入力する場合は、再度送信します。

  > ⓘ トランザクションURL、SQL Stringなどがテキストタイプデータに対応します。

エージェント使用量

JavaエージェントがCPU、メモリ、ディスク使用量を収集する方法を案内します。JavaのJMXライブラリを基本的に活用し、必要に応じてLinuxのprocディレクトリやoshi、sigarライブラリによる性能指標の収集方法を確認できます。

- **linux_proc_stat_enabled** Boolean

  既定値 `false`

  CPU、 メモリ、ディスクの使用量をLinuxの_proc_ディレクトリから収集するかどうかを設定します。

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

  既定値 `false`

  CPU、メモリ、ディスクの使用量を測定するとき、oshiライブラリを使用するかどうかを設定します。

  > ⓘ Java 8 以降の環境での使用をお勧めします。

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

  既定値 `false`

  netstat指標の収集有無を設定します。Maximum Transmission Unit(MTU) 指標が追加されます。

- **sigar_enabled** Boolean

  既定値 `false`

  sigarライブラリによるOS情報の収集を有効にします。sigarライブラリを通じて5秒単位で収集するCPU、Memory、DiskなどのOSリソースデータが含まれます。 `enabled` オプションが `false` の場合、無効にします。

  > ⓘ
  > - whatap.agent.2.1.0バージョン以下では既定値は `true` です。whatap.agent.2.1.0バージョン以降では既定値は `false` です。
  > - Java 8 未満またはJava 1.6 以下のバージョン環境では、 `sigar_enabled=true` オプションを追加してsigarライブラリを有効にし、パフォーマンス指標を収集できます。
  > - アプリケーション環境によってCPU指標が**-1**と表示される現象がある可能性があります。WhaTapが提供する勧告事項を確認し、現状を解決できます。次の文書を参照してください。

エージェント通信設定

エージェントがアプリケーションサーバーから収集したデータをサーバーに転送するためのネットワーク設定方法を案内します。主な設定オプションとしては、収集サーバーのホスト、ポート、タイムアウト値、ネットワーク転送サイズ、データ転送キューサイズなどがあり、エージェントの効率的なデータ管理と安定したサーバー通信を保証するために重要です。

- **whatap_server_host** ip_address

  既定値 `127.0.0.1,127.0.0.1`

  エージェントが収集したデータを送信するサーバーを設定します。収集サーバーの二重化により2つ以上のIPアドレスを持つ場合は、コンマ(,)を区切り文字として使用します。設定するIPアドレスには収集サーバーであるproxyサーバーがリスニング(Listening)状態でサービスする必要があります。

- **whatap_server_port** tcp_port

  既定値 `6600`

  収集サーバーのポートを設定します。ポートは1つだけ設定できるので、 `whatap_server_host` オプションに設定した収集サーバーは同一のポートを使用する必要があります。

- **tcp_so_timeout** MiliSecond

  既定値 `60000`

  収集サーバーと通信するTCPセッションのSocket Timeout値を設定します。

- **tcp_connection_timeout** MiliSecond

  既定値 `5000`

  収集サーバーと通信するTCPセッションのConnection Timeout値を設定します。

- **net_send_max_bytes** Byte

  既定値 `5242880`

  エージェントがデータを収集して、ネットワークに一度に送信できる最大byteサイズです。

- **net_send_queue1_size** Int

  既定値 `256`

  トレース情報とアクティブスタックを除く、データ転送に使用するQueueのサイズを設定します。設定するサイズ

が10以下の場合は、10に設定します。

- **net_send_queue2_size** Int

  既定値 `512`

  トレース情報とアクティブスタックデータの転送に使用するQueueのサイズを設定します。設定したサイズが10以下の場合は、10に設定します。

エージェント性能設定

Javaエージェントの性能に関するオプションを案内します。同時に処理できる最大トランザクション数とトランザクションデータを保存するバッファの初期サイズの設定を確認できます。これは、システムの負荷とリソースの使用を最適化する重要な役割を果たします。

whatap.agent.2.1.0バージョン以前

- **whatap.tx_max_count** Int

  既定値 `5000`

  トランザクションを保存する最大数です。同時に処理するトランザクションが多い場合は、そのサイズをさらに大きくしてください。

- **whatap.tx_default_capacity** Int

  既定値 `101`

  トランザクションを保存するバッファのデフォルトサイズです。

エージェントログ設定

Javaアプリケーション環境で発生するログデータをエージェントを通じて管理する方法を案内します。ログファイルのパスと名前の設定、保存期間の設定、ログのモニタリング、ユーザー定義型のログ追跡の活性化方法などが含まれます。ログ管理のための様々な設定オプションを提供し、システムの効率的なログ管理をサポートします。

パスおよび名前設定

エージェントログのパスと名前は、whatap.confファイルで設定できます。 `log_root` の既定値として_ `${WHATAP_HOME}` /logs_パスを使用します。

whatap.conf

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

ログ管理オプション

whatap.confファイルにログ関連オプションを作成し、ログ内容を管理することができます。

whatap.conf

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

- **watchlog_enabled** Boolean

  既定値 `false`

  値が `true` の場合、ログモニタリングを有効にします。

- **watchlog_check_interval** Int

  既定値 `2000`

ログをモニタリングする間隔です。一定の間隔でログファイルに記録を追加したことを確認します。

- **watchlog_read_count** Int

  既定値 8

  ログを読み取る最大回数です。一度に watchlog_buffer_size を読み取ります。

- **watchlog_buffer_size** Int

  既定値 131072(128k)

  ログを一度に読み取るログのサイズです。

- **watchlog.LOGKEY**

  1つのログファイルをモニタリングするための設定です。 file 名を入力し、モニタリングした複数の word をコンマ(,)で絞り込んで設定します。 check_interval 間隔でログをモニタリングし、キーワードを見つけたら警告します。一度警告が出れば silent ほど警告を止めます。

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

- **log_root** String

  既定値 ${WHATAP_HOME} /logs

  エージェントログのパスを設定します。

- **log_datasource_lookup_enabled** Boolean

  既定値 true

  InitialContext Lookup時のDataSourceの場合、ログを記録する機能を有効にします。

- **log_rotation_enabled** Boolean

  既定値 true

  エージェントログファイルを日付ごとに保存する機能を有効にします。ログファイル名は、whatap- YYYYMMDD .log形式で保存します。値が false の場合、ログファイル名はwhatap.log形式で保存されます。

- **log_keep_days** Int

既定値 7

ログファイルの保存期間を設定します。 log_rotation_enabled の値が true の場合にのみ動作します。

- **log_oname_enabled** Boolean

  既定値 false

  値がtrueの場合は、各ログごとにエージェント名を追加します。

```
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.
```

ログシンク(LogSink)

whatap.agent-2.1.0以降のバージョンからは、アプリケーションサーバーを統合モニタリングするログシンク(LogSink)機能を提供します。

> LogSinkオプションはwhatap.agent-2.1.0以降で使用できます。

ログモニタリング機能の有効化

- **logsink_enabled** Boolean

  既定値 false

  ログモニタリング機能を有効にします。

- **logsink_trace_enabled** Boolean

  既定値 false

  Logにトランザクション IDを挿入して、トランザクショントレースのログタブの表示有無を指定します。

フレームワークログ内容の収集

フレームワークで収集されるログをモニタリングするためには、モジュールからログをインターセプトするための設定とプラグインを追加する必要があります。フレームワークで収集されるログの基本カテゴリ名を `AppLog` に設定します。既定値は、`hooklog_enabled` オプションの設定と同じです。

- **hooklog_enabled** Boolean

  既定値 `logsink_enabled`

  ログ追跡を有効にするかどうかを設定します。

  > ◦ Javaエージェント2.2.4バージョン以降では、アプリケーションの再起動なしでオプションの変更が適用されます。
  >
  > ◦ Javaエージェントバージョン2.2.4未満では `hooklog_enabled` の値を変更すると再起動が必要です。`hooklog_enabled` の既定値は `logsink_enabled` です。そのため `logsink_enabled=true` に設定し、Javaアプリケーションを再起動すると、`hooklog_enabled` が `true` に設定されます。

- **hooklog_custom_methods** String

  カスタムログを登録します。特定のログフレームワークの内容を収集します。サイトで個別に作成したログモジュールのログを追跡する際に利用してください。

  Java

```java
package io.home.test;

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

  whatap.conf

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

特定のログAppender除外を設定する

特定のログappenderを除外するエージェントオプションを案内します。次に提供するエージェントオプションにより、モニタリング対象から特定appenderを除外できます。

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

  logback-1.2.8を追跡する際に特定appenderを除外するエージェントオプションです。例えば、`CONSOLE`と`ACCESS_APPENDER`を除外するには、次の例を参照します。

  whatap.conf

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

  ⓘ
  - このエージェントオプションを適用する場合、`weaving=logback-1.2.8`オプションを追加適用する必要があります。
  - `ACCESS_APPENDER`は例示として用いた値です。ユーザー環境に合わせたappender名前を指定する必要があります。

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

  log4j-2.17を追跡する際に、特定appenderを除外するエージェント オプションです。 例えば、`CONSOLE`と`ACCESS_APPENDER`を除外するには、次の例を参照します。

  whatap.conf

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

  ⓘ
  - このエージェントオプションを適用する場合、`weaving=log4j-2.17`オプションを追加適用する必要があります。
  - `ACCESS_APPENDER`は例示として用いた値です。ユーザー環境に合わせたappender名前を指定する

必要があります。

エージェントトランザクション設定

トランザクションとは、ユーザーブラウザからのリクエストを処理するサーバーサイドの Logical Unit of Work（LUW）です。Web アプリケーションにおいては、1 つの HTTP リクエストを受け取り、レスポンスを返すまでの一連の処理全体がトランザクションに該当します。

アプリケーションの性能は、トランザクション処理性能によって評価できます。トランザクション性能には、トランザクション開始から終了までの応答時間、リソース使用量、呼び出し元の属性などの情報が含まれます。

トランザクション性能は、応答時間の分布や統計データを基に分析できます。

トランザクショントレーシング

以下は、アプリケーションの各トランザクションの実行パスを詳細に追跡できるエージェントのオプションです。最小応答時間からトランザクションの実行ステップ数まで、様々な部分を調整可能です。

- **profile_basetime** MiliSecond

  既定値 `500`

  トレースの詳細情報を収集する最少応答時間を設定します。設定した値以下の時間内にトランザクションが終了した場合、トレース情報を収集しません。ただし、5分単位で最初に呼び出されたURLとエラーが発生したトランザクションに関するトレース情報は収集します。

  > ⓘ `profile_basetime` オプションに関係なくエラーを収集するには、`profile_concern_error_enabled` オプションを `true` に設定します。デフォルトを `false` に設定すると、エラーがサンプリングされます。

- **profile_step_normal_count** Int

  既定値 `1000`

  トランザクショントレースの最大ステップ数を設定します。

- **profile_step_heavy_count** Int

  既定値 `1020`

  Heavyなステップの場合、トレースの基本ステップ数を超えても決められた値だけ記録します。

- **profile_step_max_count** Int

  既定値 `1024`

  トレースステップの最大数を設定します。収集されたトレースステップの数が設定値を超えると、その後に収集されるステップはすべて破棄されます。 `profile_step_heavy_count` を最大1000に設定すると、 `profile_step_max_count` までアクティブスタックが収集されます。

- **profile_step_heavy_time** MiliSecond

  既定値 `100`

  Heavyなステップの基準を設定します。 設定した値より実行時間が長い場合、 `profile_step_normal_count` の値を超えても、 `profile_step_heavy_count` の設定値のみ記録します。

- **active_stack_count** Int

  既定値 `100`

  トランザクション内で収集するアクティブスタックの最大数を設定します。

- **profile_method_resource_enabled** Boolean

  既定値 `false`

  トレースからmethodステップを収集する場合は、そのステップで使用したCPUとメモリ使用量を追跡します。

- **profile_position_method** String

  設定したメソッド実行時のStackTraceを記録します。

- **profile_position_depth** Int

  既定値 `50`

  position追跡のStackTraceを記録する場合、最大ライン数を設定します。

- **trace_error_callstack_depth** Int

  既定値 `50`

  Error発生時に収集するStackTraceの最大ライン数を設定します。

- **trace_active_callstack_depth** Int

  既定値 `50`

  アクティブスタックから収集するStackTraceの最大ライン数を設定します。

- **trace_active_transaction_yellow_time** MiliSecond

既定値 3000

アクティブTXのアークイコライザーで黄色で表現する基準を設定します。

- **trace_active_transaction_red_time** MiliSecond

  既定値 8000

  アクティブTXのアークイコライザーで赤色で表現する基準を設定します。

- **hook_method_patterns** String

  レスポンス時間を測定するメソッドを設定します。最後のピリオド(.)を区切り文字としてクラスのFullNameとメソッドに区分され、アスタリスク(*)をワイルドカードとして使用できます。対象が複数ある場合は、コンマ(,)で区切ります。

  whatap.conf

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

  > ◦ このオプションで登録したメソッドは、**トレース分析**または**トランザクション情報**画面からMethodのステップで確認できます。
  >
  > ◦ なるべく小さい範囲に適用してください。広い範囲に適用すると、アプリケーションの性能が低下する恐れがあります。

- **hook_method_supers** String

  特定のクラスを継承するメソッドの応答時間を測定する場合は、Super Classを設定します。 クラスFullNameを設定し、対象が複数ある場合は、コンマ(,)で区切ります。

  whatap.conf

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

- **hook_method_interfaces** String

  特定のインターフェイスを実装するメソッドの応答時間を測定する場合は、Interfaceを設定します。インターフェイスのFullNameを設定し、対象が複数ある場合は、コンマ(,)で区切ります。

- **hook_method_ignore_classes** String

  メソッドトレースを設定する場合は、トレースから除外するクラスをprefixに設定します。

- **hook_method_access_public_enabled** Boolean

  既定値 `true`

  メソッドトレースを設定する場合、publicメソッドを個別のターゲットとしてのみ処理するかどうかを設定します。

- **hook_method_access_private_enabled** Boolean

  既定値 `false`

  メソッドトレースを設定する場合、privateメソッドに限り、個別の対象として処理するかどうかを設定します。

- **hook_method_access_protected_enabled** Boolean

  既定値 `true`

  メソッドトレースを設定する場合、protectedメソッドに限り、個別の対象として処理するかどうかを設定します。

- **hook_method_access_none_enabled** Boolean

  既定値 `true`

  メソッドトレースを設定する際、no access indicatedメソッドに限り、別途対象として処理するかを設定します。

- **stacklog_socket_port** TCP_PortNumber

  既定値 `0`

  宛先のTCPポートを設定すると、Socket.connect()は、特定の時点でStackTraceをエージェントログとして記録します。基本設定で確認されていないDB接続、HTTPC接続などを追跡する場合に使用できます。

  > 設定した宛先への接続は、毎回StackTraceを記録します。これらを行うと、パフォーマンスが低下する可能性があります。デバッグ用に選択したエージェントに一時的に使用する必要があります。
  >
  > DB接続状態追跡の例） `stacklog_socket_port=1521`

- **trace_concern_error_enabled** Boolean

  既定値 `true`

  `trace_basetime` または `profile_basetime` オプションに関係なく、エラーのある詳細なトレースの収集有無を設定します。

例外処理

以下は、アプリケーションで発生する例外を管理およびモニタリングするためのオプションです。ビジネス例外を登録し、関連する統計を収集することで、アプリケーションの安定性を確認できます。また、特定の例外を無視したりHTTPステータスコードがエラーの場合、エラーとして処理するかどうかを設定できます。例外として処理された項目は、**ヒットマップ**または**トレース分析**画面で、正常なトランザクションとして表示されます。

> Javaエージェントの、例外処理基準についての詳細は、次の文書を参照してください。

- **biz_exceptions** String

  Business Exceptionを登録します。

  whatap.conf

  ```
  # example
  biz_exceptions=com.home.test.exception.BizException
  ```

  > エラークラス収集 / ヒットマップ表示レベル：INFO(青色) / エラー統計を含む

  > Javaエージェントv2.2.20 以降では、登録したビジネスExceptionをエラー統計で収集します。 v2.2.20未満では、登録したビジネスExceptionをエラー統計から除外します。 ただし、トレースの詳細では表示されます。

- **biz_exceptions_status** String Java Agent v2.2.20 or later

  Business Exception発生時、イベント通知を抑制するBusiness ExceptionとStatusを設定します。 このオプションで設定した**Business Exception:Status**については、イベントレベルがINFOに変更されます。 Business ExceptionとStatusはコロン(:)で区分し、一つのBusiness Exceptionに複数のStatusを使用する場合、アンパサンド(&)で区分します。 コンマを区切り文字として使用して複数設定できます。

  whatap.conf

  ```
  # example
  ```

```
biz_exceptions_status=java.util.concurrent.TimeoutException:400&404&408,java.lang.Exception:200
```

> エラークラス収集 / ヒットマップ表示レベル：INFO(青色) / エラー統計を含む

- **ignore_exceptions** String

  登録したExceptionエラを無視します。

  whatap.conf

```
# example
ignore_exceptions=com.home.test.exception.BizException
```

> エラークラス無視 / ヒットマップ表示レベル：INFO(青色) / エラー統計を含まない

- **transaction_status_error_enable** Boolean

  既定値 `true`

  正常応答ではなくHTTP401、403のようなステータスコードがリターンされた場合、エラーとして処理するかどうかを設定します。

- **status_ignore** String

  無視しようとするHTTPステータスコードを設定できます。 複数の値を対象とする場合は、コンマ（,）を区切り文字として使用してください。

  whatap.conf

```
# example
status_ignore=408,500
```

> Statusエラー無視 / ヒットマップ表示レベル：INFO(青色) / エラー統計を含まない

- **status_ignore_set** String

  `whatap.error.STATUS_ERROR` を無視します。 複数の値を対象とする場合は、コンマ（,）を区切り文字として使

用してください。

whatap.conf

```
# example
status_ignore_set=/api/test/timeout/{time}:408,/error:500
```

Status エラーセット無視 / ヒットマップ表示レベル：INFO(青色) / エラー統計を含まない

- **httpc_status_error_enable** Boolean

  既定値 `true`

  HTTPステータスコードがエラーの場合は、収集有無を設定します。 クライアントエラーレスポンス(`400` 以上)、サーバーエラーレスポンス(`500` 以上)が該当します。

  HTTPステータスコードの詳細については、次のリンクを参考にしてください。

- **httpc_status_ignore** String

  HTTPのステータスコードがエラー(`HTTPC_ERROR`)の場合、無視します。 複数の値を対象とする場合は、コンマ（,）を区切り文字として使用してください。

  whatap.conf

  ```
  # example
  httpc_status_ignore=404,500
  ```

  HTTPステータスエラーコード無視 / エラー統計を含まない

- **httpc_status_url_ignore_set** String

  `whatap.error.HTTPC_ERROR` を無視します。 複数の値を対象とする場合は、コンマ（,）を区切り文字として使用してください。

  `httpc_status_url_ignore_set=/a/b/c:400,/a/ab/c:404` のように設定してください。 この場合、URLは統計/エラー分析に `HTPC_URL` の値を入力します。

whatap.conf

```
# example
httpc_status_url_ignore_set=/a/b/c:404,/d/e/f:500
```

> HTTPステータスエラーコード無視 / エラー統計を含まない

- **trace_sql_exception_enabled** Boolean

  既定値 `true`

  JDBCドライバーでSqlException発生時に追跡可否を設定します。 オプション値を `false` に設定すると、ユーザー定義の例外処理により `biz_exceptions` 処理できます。

- **trace_exception_handler_overwrite_enabled** Boolean

  既定値 `true`

  spring-boot ExceptionHandlerを通過する例外(exception)が発生すると、以前に発生した例外に置き換えできます。 **トレース分析**画面の**レコード要約**タブで例外(exception)を、spring-bootのExceptionHandlerに登録した例外で表示できます。

HTTPトランザクションの追跡

以下は、HTTPリクエストと応答に関連する様々な情報を追跡および記録するエージェントのオプションです。これにより、ユーザーはアプリケーションのHTTPトランザクションを細かくモニタリングし分析できます。トランザクション名にパラメータ値を追加または特定のURLやHTTPメソッドを除外できます。またHTTPヘッダー情報やパラメータ情報を含め、トランザクションをより詳細に記録できます。

- **trace_normalize_enabled** Boolean

  既定値 `true`

  トランザクションURLをパーシングし、正規化する機能を有効にします。

> `false` に値を変更すると、パスパラメータのパーシングが無効になります。この場合、統計データの意味が弱くなるため、デバッグ目的のみで使用することをお勧めします。

- **trace_auto_normalize_enabled** Boolean

  既定値 `true`

  トランザクションURLを正規化すると、パターン値をアノテーションから抽出して自動的にパーシングする機能がアクティブになります。

- **trace_normalize_urls** String

  正規化するトランザクションURLパターンを設定します。呼び出しURLパターンをパーシングして、パスパラメータを削除します。

  > ⓘ 例） `/a/{v}/b` を宣言する場合は、 `a/123/b` → `a/{v}/b` に置き換えます。複数登録する場合は、コンマ(,)を区切り文字として使用します。置換パターンをクリーンアップした後、補足する必要があります。

- **web_static_content_extensions** String

  既定値 `js, htm, html, gif, png, jpg, css, swf, ico`

  静的コンテンツであると判断する拡張子を設定します。このオプションで設定した拡張子を持つトランザクションは、トレース追跡とカウントから除外されます。

- **trace_transaction_name_header_key** String

  設定したHTTP header keyから抽出した値をトランザクションの名前の末尾に追加します。

- **profile_http_header_enabled** Boolean

  既定値 `false`

  トレース履歴にHTTPヘッダー情報を記録するには、 `true` に設定します。

- **profile_http_parameter_enabled** Boolean

  既定値 `false`

  トレース履歴にHTTPパラメータ情報を記録するには、 `true` に設定します。パラメータは、他のセキュリティ キーを入力して参照できます。

  > ⓘ ◦ **Javaエージェント2.2.2バージョン未満**：セキュリティキーは、WASサーバー_ `${WHATAP_AGENT_HOME}` /paramkey.txt_ファイル内に6桁で作成します。paramkey.txtファイルが存在しない場合は、ランダム値で自動生成します。

> ◦ **Javaエージェント2.2.2バージョン以降**：セキュリティキーは、WASサーバー_${WHATAP_AGENT_HOME}/security.conf_ファイル内に `paramkey` キー値を確認してください。security.confファイルが存在しない場合は、`paramkey` キー値を**WHATAP**で自動生成します。
>
> ◦ セキュリティキー設定ファイルの詳細については、次の文書を参照してください。

- **profile_http_header_url_prefix** String

  トレース履歴にHTTPヘッダー情報を書き込むための宛先URLのprefixを定義するために使用します。

- **profile_http_parameter_url_prefix** String

  トレース履歴にhttpパラメータ情報を記録する対象URLのprefixを定義するときに使用します。

- **trace_transaction_name_key** String

  HTTPリクエストパラメータの変数名を設定すると、その変数の値を取得してトランザクション名の最後に追加します。

  例えば、HTTPパラメータで変数名 `paramKey` を持った `/api/test` というURLを呼び出す場合、次の例のように設定すると、トレース履歴に `/api/test$paramKey={value}` の形式で表示されます。ただし、複数のパラメータを登録できません。

  whatap.conf

  ```
  # example
  trace_transaction_name_key=paramKey
  ```

- **recursive_max** Int

  既定値 `1000000`

  トランザクションが再帰呼び出されているかどうかを検知するためのオプションです。１つのトランザクションから派生する再帰呼び出し回数をカウントすると、イベント通知を発行するための条件が設定されます。

  > HTTP URLの再帰呼び出しを対象とします。jsp:forwardによって再呼び出しするケースもカウントに含まれます。

- **hook_httpservlet_classes** String

  HTTPトランザクションのEND POINTを設定します。メソッドの2つのパラメータは、`HttpServletRequest` と

`HttpServletResponse`を設定できます。

- **hook_jsp_patterns** String

  既定値 `org.apache.jasper.servlet.JspServlet.serviceJspFile`

  JSPファイルをロードするためのメソッドを設定します。トランザクションの呼び出しの結果として戻されたJSP情報をトレースに表示します。このオプションで追加された設定に既定値が自動的に追加されます。

- **trace_ignore_url_set** String

  トランザクション追跡から除外するURLを設定します。2つ以上の値を設定する場合は、コンマ(,)を区切り文字として使用します。

  > このオプションを通じて登録したURLは、**ヒットマップ**, **トレース分析**または**トランザクション情報**画面で収集が除外されます。

- **trace_ignore_active_url_set** String

  アクティブTXリストから除外するURLを設定します。2つ以上の値を設定する場合は、コンマ(,)を区切り文字として使用します。

  > このオプションを通じて登録したURLは、**アクティブTX**画面でのみ収集が除外され、ヒットマップ、トレース分析、トランザクション情報など他のトランザクションデータには影響しません。

- **trace_ignore_url_prefix** String

  トランザクション追跡から除外するURL prefixを設定します。2つ以上の値を設定する場合は、コンマ(,)を区切り文字として使用します。

  > このオプションを通じて登録したURLは、**ヒットマップ**, **トレース分析**または**トランザクション情報**画面で収集が除外されます。

- **ignore_http_method** String

  既定値 `PATCH,OPTIONS,HEAD,TRACE`

  設定したHTTPメソッド(Method)で要求されたトランザクション情報は収集しません。複数設定するには、コンマ(,)を区切り文字として利用してください。

whatap.conf

```
# e.g. http_methodがOPTIONS、HEADであるトランザクションを「追跡」
ignore_http_method=PATCH,TRACE

# e.g. http_methodがOPTIONS、HEADであるトランザクションを「無視」
ignore_http_method=OPTIONS,HEAD
```

> ⓘ このオプションを通じて登録したHTTPメソッドは、**ヒットマップ**, **トレース分析**または**トランザクション情報**画面で収集が除外されます。

- **trace_tx_name_with_method_enabled** Boolean

  既定値 `false`

  統計データでURLを変更したURL+methodで収集できます。**統計**または**TX検索**メニューでURL+methodで確認できます。

  例、/api/user+GET、/api/user+POST

非HTTPトランザクションのトレース

以下は、HTTP以外のトランザクションを追跡してモニタリングするためのオプションです。このオプションにより、ユーザーはバックグラウンド作業や他のプロトコルに対するトランザクションを細かく追跡できます。

- **trace_auto_transaction_enabled** Boolean

  既定値 `false`

  トレース対象メソッドがトランザクション開始点(`Javax.http.httpservlet` , `hook_service_*`)内で実行されない限り、収集されません。 この場合、トレース対象メソッドをトランザクションの開始点になるように設定します。

  > ⓘ 主に開発環境でバックグラウンドトランザクションのEND POINTを見つめるために使用します。

- **trace_auto_transaction_backstack_enabled** Boolean

  既定値 `true`

`trace_auto_transaction_enabled`オプションの値が`true`の場合、トランザクション開始時にStackTraceを記録します。これを通じて、トランザクションの開始点を見つけることができます。

- **trace_background_socket_enabled** Boolean

  既定値 `true`

  トランザクションではなく、バックグラウンドスレッドによるソケットがオープンしたときも記録します。

- **async_stack_enabled** Boolean

  既定値 `false`

  バックグラウンドスレッドに対するActive Stack機能を使用するかどうかを設定します。

- **async_thread_match** String

  アクティブスタックをダンプするバックグラウンドスレッド名を設定します。複数を設定するには、コンマ(,)を区切り文字として使用します。 名前を設定時、「Thread-*」のようにアスタリスク(*)を使用して比較パターンを使用できます。

- **async_thread_parking_class** String

  既定値 `sun.misc.Unsafe`

  スタックの上位メソッドが`async_thread_parking`に登録されたクラス/メソッドの場合、スレッドがパーキング状態にあると判断してダンプを生成しません。

- **hook_service_patterns** String

  NON-Httpトランザクション追跡のための開始点パターンを設定します。ワイルドカード文字は使用できません。2つ以上の値を設定するにはコンマ(,)を区切り文字として利用します。

- **hook_serivce_ignore_methods** String

  `hook_service_patterns`で設定した項目のうち、出発点として不要なメソッドを追加できます。

- **hook_service_supers** String

  NON-HTTPトランザクション追跡のための始点の共通分母が特定のクラスのメソッドを継承した場合、共通分母のメソッドを設定します。メソッド全体のパスを入力します。ワイルドカード(`*`)が使用でき、コンマ(,)を区切り文字として利用して複数のメソッドを登録できます。

  whatap.conf

```
hook_service_supers=a.b.C.method,a.b.C.*
```

- **hook_service_interfaces** String

  NON-HTTPトランザクション追跡のための開始点の共通分母が特定のインターフェイスを実装している場合は、これを設定します。

- **hook_service_access_public_enabled** Boolean

  既定値 `true`

  NON-HTTP Demonプロセスのプロセスのトランザクションを設定する場合は、publicメソッドに対してのみAccess権限条件をon/offで設定します。

- **hook_service_access_private_enabled** Boolean

  既定値 `true`

  NON-HTTP Demonプロセスのトランザクションを設定する場合は、privateメソッドに対してのみAccess権限条件をon/offで設定します。

- **hook_service_access_protected_enabled** Boolean

  既定値 `true`

  NON-HTTP Demonプロセスのトランザクションを設定する場合は、protectedメソッドに対してのみAccess権限条件をon/offで設定します。

- **service_name_mode** String

  既定値 `full`

  トランザクション名には、次のオプションを使用できます。

  - `full` : Full Class名を使用
  - `class` : サービス名をClass名として使用
  - `method` : サービス名をMethod名として使用
  - `string` : サービス名を文字列の最初のパラメータとして使用します。
  - `arg` : パラメータで、 `service_name_index` オプションで設定されたインデックスにパラメータをサービス名として使用します。

マルチトランザクショントレース

以下は、アプリケーション間の呼び出しを追跡しモニタリングするためのオプションです。このオプションにより、ユ

ーザーは複数のアプリケーション間の相互作用を理解できます。提供する様々なオプションにより、マルチトランザクションの追跡に対する詳細な動作を調整できます。

- **mtrace_enabled** Boolean

  既定値 `true`

  マルチトランザクション追跡機能(Multi Transaction ID、以下MTID) の使用有無を設定します。MTIDを追跡することで、登録したすべてのアプリケーション間の呼び出しを確認できます。

- **mtrace_rate** Percentage

  既定値 `10`

  最初のトランザクションが発生したときに発行されるMTIDの発行比率を設定するオプションです。

- **mtrace_caller_key** String

  既定値 `x-wtap-mst`

  MTID追跡に使用するCaller Key Nameを設定します。

- **mtrace_callee_key** String

  既定値 `x-wtap-tx`

  MTID追跡に使用するCallee Key Nameを定義します。

- **mtrace_send_url_length** Int

  既定値 `80`

  Http Callerは、CalleeにそのURLを渡します。このとき、URLの長さが制限され、この長さの値が設定されます。

- **mtrace_callee_id_send_enabled** Boolean

  既定値 `false`

  MTIDを追跡する場合、HTTPC呼び出しで発行されたCallee IDをトレースに表示します。

- **mtrace_callee_id_recv_enabled** Boolean

  既定値 `false`

  MTID追跡時に受信したCallee IDのトレースを表します。

- **mtrace_alltx_enabled** Boolean

  既定値 `false`

log4jなどのロギングシステムに接続するために、すべてのトランザクションのMTIDを追跡できます。

- **mtrace_basetime** Miliseconds

  既定値 100

  mtrace_alltx_enabled オプションの値が true の場合、多くのログが残る可能性があります。この場合、トレースログの量を減らす必要があります。

- **stat_mtrace_enabled** Boolean

  既定値 false

  CallerとCalleeの相関統計を収集します。Callerに適用するとCallerの詳細情報が送信され、Calleeを適用するとurl単位のCaller-Calleeの呼び出し統計情報が収集サーバーに送信されます。

トレースデータサンプリング

以下は、トレースデータのサンプリングを調整し、データの送信有無を設定するオプションです。このオプションにより、ユーザーはアプリケーションの性能とデータ転送量を管理できます。

エージェント対応の範囲：Java Agent v2.2.4以降

- **trace_sampling_enabled** Boolean

  既定値 false

  トレースデータサンプリングの収集有無を設定できます。

- **trace_sampling_tps** Int

  既定値 10000

  トレースデータサンプリングの数を設定できます。設定したサンプリング数を超えるデータは送信せず、5秒ごとに初期化します。

- **trace_send_enabled** Boolean

  既定値 true

  トレースデータの送信有無を設定できます。

- **debug_trace_samling** Boolean

  既定値 false

トレースデータサンプリングのデバッグオプションをオンまたはオフにできます。

トレースURL名を修正する

- **hook_tx_name_patterns / hook_tx_name_mode** String Java Agent v2.2.4 or later

  特定URLおよび特定メソッド(method)を呼び出す時にトレースURLの名前を修正できます。例えば `io.home.test.TestController.test1` に登録したURLを呼び出すと `test100000()` メソッドが呼び出される場合、サービス画面から `/api/100000+test1+test100000` で確認できます。

  `hook_tx_name_mode` オプションを通じて**class**、**method**、**string**、**return**の中で1つを選択できます。

  whatap.conf

```
# 呼び出しメソッドのフルパス、ワイルドカード(*)使用可能
hook_tx_name_patterns=ab.cd.ef.GH.ij

# class, method, string, returnから1つを選択
hook_tx_name_mode=method
```

特定のexceptionを無視する

- **ignore_exception_tx_pattern** String Java Agent v2.2.4 or later

  特定のサービスで特定のexceptionを無視できるように設定できます。複数登録するときは、コンマ(,)を区切り文字として利用してください。

  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を追跡する

次は、`HttpURLConnection`クラスを使ってHTTP呼び出しを追跡するためのオプションです。

- **HttpURLConnection** Boolean

  既定値 `true`

  `HttpURLConnection`クラスのメソッドを追跡するためのオプションです。

- **hook_HttpURLConnection_startup_enabled** Boolean Java Agent v2.2.4 or later

  既定値 `false`

  `HttpURLConnection`クラスのメソッドの最初の呼び出し時にも追跡できるように設定できます。

- **HttpURLConnection_weblogic** Boolean Java Agent v2.2.4 or later

  既定値 `true`

  Weblogicで`HttpURLConnection`でhttp呼び出しをする場合、`weblogic.net.http.HttpURLConnection`クラスのメソッドを追跡するためのオプションです。

- **hook_HttpURLConnection_weblogic_startup_enabled** Boolean Java Agent v2.2.4 or later

  既定値 `false`

  `weblogic.net.http.HttpURLConnection`クラスのメソッドの最初の呼び出し時にも追跡できるように設定できます。

メモリ割当量の収集

- **trace_malloc_enabled** Boolean

  既定値 `false`

  トランザクションごとの平均メモリ割当量を収集します。

  > ⓘ 当該オプションの値を `true` に設定し、活性化時に**トレース分析**画面の**レコード要約**タップで**メモリ割当量**が確認できます。詳細は次の文書を参照してください。

DB/SQL設定

データベース及びSQL性能データ収集のためのJavaエージェントの様々なオプションを案内します。DBCP、Hikari、Tomcatなど様々なDB接続プール(Connection Pool)情報の追跡及びSQL実行の詳細を記録できます。また、SQLパラメータ情報の記録、DB接続漏れの追跡など、性能最適化に欠かせない情報収集オプションを提供します。

- **dbcp_pool_enabled** Boolean

  既定値 `true`

  JMXを使わず、DBCPのDB Connection情報を追跡するために使用します。

- **hikari_pool_enabled** Boolean

  既定値 `false`

  JMXを使わず、hikari poolのDB Connection情報を追跡するために使用します。

- **tomcat_ds_enabled** Boolean

  既定値 `false`

  JMXを使ってTomcat DB Connection Pool情報を追跡する機能を活性化します。

- **tomcat_pool_enabled** Boolean

  既定値 `true`

  JMXを使わずにTomcat DB Connection Pool情報を追跡する機能を活性化します。

- **weblogic_ds_enabled** Boolean

  既定値 `false`

  JMXを使ってWeblogic DB Connection Pool情報を追跡する機能を活性化します。

- **weblogic_pool_enabled** Boolean

  既定値 `true`

  JMXを使わずにWeblogic DB Connection Pool情報を追跡する機能を活性化します。

- **jeus_pool_enabled** Boolean

  既定値 `true`

  JMXを使わずにJEUS DB Connection Pool情報を追跡する機能を活性化します。

- **profile_connection_open_enabled** Boolean

  既定値 `true`

  トレース履歴にDBConnectionオープン情報を記録します。

- **trace_dbc_close_enabled** Boolean Java Agent v2.2.3 or later

  既定値 `false`

  `profile_connection_open_enabled` の値が `true` の場合のみ動作します。トレース履歴にDBConnectiondのクローズ情報を記録します。

  > ⓘ **Javaエージェント2.2.3バージョン前**では `profile_dbc_close` オプション名を使用します。

- **profile_sql_param_enabled** Boolean

  既定値 `false`

  トレース履歴にSQLパラメータ情報を記録するときに使用します。パラメータは、別途のセキュリティ キーを入力しないと照会できません。

  > ⓘ
  > - **Javaエージェント2.2.2バージョン以前**：セキュリティキーは、WASサーバー_ `${WHATAP_AGENT_HOME}` /paramkey.txt_ファイル内に6桁で作成します。paramkey.txtファイルが存在しない場合は、ランダム値で自動生成します。
  > - **Javaエージェント2.2.2バージョン以降**：セキュリティキーは、WASサーバー_ `${WHATAP_AGENT_HOME}` /security.conf_ファイル内に `paramkey` キー値を確認してください。security.confファイルが存在しない場合は、 `paramkey` キー値を**WHATAP**で自動生成します。
  > - セキュリティキー設定ファイルの詳細については、次の文書を参照してください。

- **profile_sql_resource_enabled** Boolean

  既定値 `false`

  トレースからSQLを収集する場合、そのステップで使用したCPUとメモリ使用量を追跡します。

- **profile_update_count** Boolean

  既定値 `false`

  `executeUpdate()` メソッドを通じてSQL UPDATE文を実行した場合、UPDATE件数を収集します。

- **custom_pool_classes** String

  pre-defineされない別途のConnection Poolを使用する場合は、そのクラス名を設定します。

- **ds_update_interval** MiliSeconds

  既定値 `5000`

  DB Connection情報のCount間隔を設定します。

- **profile_position_sql** Boolean

  既定値 `false`

  SQLを実行する時点のStackTraceを記録します。

- **profile_sql_param_length** Int

  既定値 `40`

  SQLパラメータの長さを設定します。

  > ⓘ 最大値は128です。その以上の値を入力してもパラメータは128まで保存します。

- **trace_dbc_leak_enabled** Boolean

  既定値 `false`

  DBConnection Leakを追跡する機能を有効にします。

  > ⚠ Connection Wrapperを使用してLeakを追跡するため、運営サービスに影響を与える可能性があります。必ずテストしてから適用してください。

- **trace_dbc_leak_fullstack_enabled** Boolean

  既定値 `false`

  DBConnection Leakが検出された場合、その時点のStackTraceを収集します。

  > ⚠ ピークタイム(Peak Time)にはなるべく適用しないでください。オプションを適用すると、CPU使用量が多少増加する可能性があります。問題解決用途にのみ一時的に適用することをお勧めします。

- **trace_sql_normalize_enabled** Boolean

既定値 `true`

SQL文からリテラル部分を抽出し、SQL文を正規化する機能を活性化します。

- **profile_error_jdbc_fetch_max** Int

  既定値 `10000`

  SQL Fetch Count(`ResultSet.next()` 呼び出し件数)が設定した値を超えると、**TOO_MANY_RECORDS**エラーで処理します。 `0` に設定するとエラー処理されません。

- **profile_error_sql_time_max** Int

  既定値 `30000`

  SQLの実行時間が設定した値を超えると、**SLOW_SQL**エラーで処理されます。 `0` に設定する場合、エラー処理は実行されません。

- **hook_connection_open_patterns** String

  DB Connection Open時に呼び出すメソッドを登録します。事前設定していないConnection PoolのgetConnectionを登録するのが一般的です。

  Example

  ```
  hook_connection_open_patterns=mypool.ConPool.getConnection
  ```

- **hook_jdbc_con_classes** String

  未登録のJDBC Connectionクラスを指定します。

  Example

  ```
  hook_jdbc_con_classes=mypool.ConPool
  ```

- **hook_jdbc_pstmt_classes** String

  未登録のjdbc Prepared Statementクラスを設定します。生成者パラメータにSQL文字列が伝達される構造である必要があることに注意してください。

  Example

  ```
  hook_jdbc_pstmt_classes=org.apache.derby.impl.jdbc.EmbedPreparedStatement
  ```

- **hook_jdbc_cstmt_classes** String

  未登録のjdbc Callable Statementクラスを設定します。

  Example

  ```
  hook_jdbc_cstmt_classes=org.apache.derby.impl.jdbc.EmbedCallableStatement
  ```

- **hook_jdbc_stmt_classes** String

  未登録のJDBC Statementクラスを設定します。

  Example

  ```
  hook_jdbc_stmt_classes=org.apache.derby.impl.jdbc.EmbedStatement
  ```

- **hook_jdbc_rs_classes** String

  未登録のJDBC ResultSetクラスを設定します。

  Example

  ```
  hook_jdbc_rs_classes=org.apache.derby.impl.jdbc.EmbedResultSet
  ```

- **hook_jdbc_wrapping_driver_patterns** String

  DB2ドライバーのように難読化したJDBCドライバーは、`hook_jdbc_xx`オプションで直接BCIが難しいです。このような場合、Wrapper方式でSQLを追跡できます。このとき、`Driver.connect`を設定して追跡します。

- **debug_dbc_stack_enabled** Boolean

  既定値`false`

  DB Connection時点のStackTraceをトレースに保存します。アプリケーションで使用するConnection Pool情報を取得するために使用します。

- **ignore_sql_hash_set** String Java Agent v2.2.4 or later

  該当オプションに入力したSQLハッシュ値と一致するSQL文は無視するように設定できます。

HTTPC/API Call設定

HTTP外部呼び出しとAPI呼び出しに関するデータを収集、分析するためのJavaエージェントのオプション設定方法を案内します。**SLOW_HTTPC**エラー処理及びCPUとメモリ使用量の追跡、呼び出し時点のスタックトレース記録、URL正規化など、細かいモニタリングのための様々なオプションを提供します。これにより開発者は、アプリケーションの外部依存性を効率的に管理し、性能問題を識別できます。

- **profile_error_httpc_time_max** Int

  既定値 `10000`

  HTTPCの実行時間が指定した値を超えると、**SLOW_HTTPC**エラーで処理します。`0` に設定するとエラー処理されません。

- **profile_httpc_resource_enabled** Boolean

  既定値 `false`

  トレースからHTTP Call ステップを収集する場合は、それらのステップで使用されるCPUとメモリ使用量を追跡します。

- **profile_position_httpc** Boolean

  既定値 `false`

  HTTPCが実行されている時点のStackTraceを記録します。

- **trace_httpc_normalize_enabled** Boolean

  既定値 `true`

  トランザクション内のHTTPC URLをパーシングおよび正規化する機能を有効にします。

- **trace_httpc_normalize_urls** String

  正規化するHTTPC URLパターンを設定します。呼び出しURLパターンをパーシングして、pathパラメータを削除します。

  > ⓘ 例） `/a/{v}/b` と宣言すると、 `a/123/b` → `a/{v}/b` に置き換えます。複数を登録する場合は、コンマ(,)を区切り文字として使用します。 置換パターンをクリーンアップした後、補足する必要があります。

- **hook_httpc_patterns** String

HTTP outbound呼び出しを行い、 `full packageクラス名` . `メソッド` を設定します。HTTP Callを実行するメソッド(method)のフルパス(full path)を登録して使用します。登録後のクラス再定義(class redefine)をするか、エージェントを再起動する必要があります。

whatap.conf

```
hook_httpc_patterns=io.home.test.baseapp.app.post.service.HookHttpcService.*
```

- クラスを複数登録するに場合は、コンマ(,)を区切り文字として使用してください。
- ワイルドカードは、アスタリスク(*)のみ使用でき、正規表現は使用できません。

スクリプトプラグイン

Javaエージェントが提供するプラグインオプションを通じて、ユーザーが希望するコードをトレースデータに注入するか、メソッド実行前後に付加情報を追加する方法を提供します。メソッドの開始・終了時点で特定のコードを実行する場所を設定するオプションと、ユーザー定義型poolをモニタリングするためのクラス設定方法が含まれます。実際のプラグインの適用事例を通じて具体的な活用方法を提示し、モニタリングの柔軟性と細さを高める様々な設定例とAPIの使い方を案内します。

プラグインオプション

Javaエージェントプラグインは、メソッドの開始/終了部分に挿入して実行します。プラグインを適用する場所(クラス、メソッド)は、エージェントオプションで設定できます。

- `hook_trace_helper_start_patterns`

  メソッドの開始部分にトレースプラグインを挿入するポイント(クラス、メソッド)を設定します。プラグインコードは、_ `$WHATAP_HOME` /plugin/TraceHelperStart.x_ファイルに記述してください。

- `hook_trace_helper_end_patterns`

  メソッドの最後にトレースプラグインを挿入するポイント(クラス、メソッド)を設定します。プラグインコードは、_ `$WHATAP_HOME` /plugin/TraceHelperEnd.x_ファイルに記述してください。

- `hook_trace_helper_patterns`

  メソッドの開始/終了の両方にトレースプラグインを挿入するポイント(クラス、メソッド)を設定します。プラグインコードは、 `$WHATAP_HOME` /plugin/TraceHelperStart.x, _ `$WHATAP_HOME` /plugin/TraceHelperEnd.x_ファイルにそれぞれ記述してください。

- `custom_pool_classes`

  ユーザー定義プールをモニタリングするには、プール使用量の情報を含むクラスを設定します。プラグインコードは、_ `$/WHATAP_HOME` /plugin/CustomPool.x_に作成します。

プラグインの設定例

複数のクラスを設定する場合は、コンマ(,)を区切り文字として利用してください。パッケージ名の文字列の一部または

全部を「*」に置き換えることができます。

- **パッケージとメソッド名を設定する場合**

```
whatap.bytecode.instrument.PluginTestA.testA,whatap.bytecode.instrument.PluginTestB.testB
```

- **パッケージとメソッド名を `*` に置き換える場合**

```
*PluginTestA.testA,whatap.bytecode.instrument.PluginTestB.*
```

```
*.testA, *PluginTestB.testB
```

- **全体を対象とする場合**

```
*.*
```

- **Custom Poolを設定する場合**

  Custom Poolは、識別子@パッケージ名の形に設定します。

  例）Class名の前に `whatap_plugin_guide` という識別子を設定します。識別子とクラスは@で区切ります。

```
whatap_plugin_guide@com.ibm.ws.connectionpool.monitor.ConnectionPoolStats
```

プラグインコードの作成例

ほとんどの場合はオブジェクトを返すので、明示的な型変換処理が必要です。以下はプラグインコードの作成例です。

設定したメソッド開始時間をトレースに記録する

1. Javaエージェント `hook_trace_helper_patterns` オプションに適用するメソッドをwhatap.confファイルに記述してください。

   $WHATAP_HOME/whatap.conf

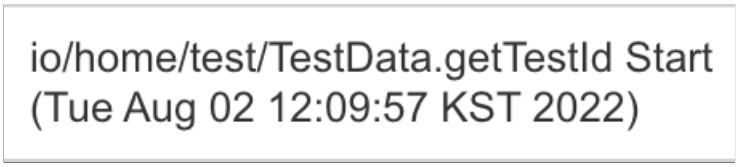
```
hook_trace_helper_patterns=org.apache.catalina.connector.RequestFacade.*
```

2. TraceHelperStart.xファイルにプラグインコードを作成します。

   $WHATAP_HOME/plugin/TraceHelperStart.x

```
// String prefixにクラス名、メソッド名を代入します。
String prefix = $point.class1 + "." + $point.method;
// WhaTapトレース属性名称を `prefix` + "st"にして現在の時間を設定します。
$ctx.setAttribute(prefix + "st", new Long(System.currentTimeMillis()));
// トレース情報に現在の時間を追加します。
$ctx.profile(prefix + " Start", new java.util.Date().toString());
```

トレース履歴にメソッドの開始時間を次のように表示します。



設定したメソッド終了時間と実行時間をトレースに記録する

1. Javaエージェント `hook_trace_helper_patterns` オプションに適用するメソッドをwhatap.confファイルに記述してください。

   $WHATAP_HOME/whatap.conf

```
hook_trace_helper_patterns=org.apache.catalina.connector.RequestFacade.*
```

2. _TraceHelperEnd.x_ファイルにプラグインコードを作成します。

   $WHATAP_HOME/plugin/TraceHelperEnd.x

```
1  String prefix = $point.class1 + "." + $point.method;
2  // TraceHelperStart.xファイルで追加したトレース属性を取得します。
3  long st = ((Long) $ctx.getAttribute(prefix + "st")).longValue();
4  long gap = System.currentTimeMillis() - st;
```

```
StringBuilder sb = new StringBuilder();
sb.append(new java.util.Date().toString() + " (Gap:" + gap + " milliseconds)");
// トレース情報に開始時間と現在時間とのGap(実行時間)を追加します。
$ctx.profile(prefix + " End", sb.toString());
```

トレース履歴にメソッド終了時間と実行時間を次のように表示します。

```
io/home/test/TestData.getTestId Start
(Tue Aug 02 12:09:57 KST 2022)
```

プラグインAPI

プラグインAPIは、共通項目である `$ctx` 、 `$point` 、 `$req` 、 `$res` で区分します。トランザクション開始/終了時、HTTPC間隔、特定メソッドの実行間隔などに適用できます。

> プラグインAPIを使用して個人識別情報の保存を禁止し、個人情報収集時にその内容を個人情報処理方針に明示する必要があります。

パラメータ提供 - 共通プラグインファイル

パラメータ	メソッド／変数	説明
$ctx	属性管理	
	void setAttribute(String key, Object value)	Attributeを設定します。
	Object getAttribute(String key)	Attributeの値を取得します。
	トレースデータ	

パラメータ	メソッド／変数	説明
	void profile(String desc)	トレースメッセージ記録
	void profile(String name, String msg)	トレースStep名、メッセージの記録
	void profile(String name, String msg, int elapsed)	トレースStep名、メッセージ、実行時間の記録
	void profile(String name, String msg, int elapsed, int value)	トレースStep名、メッセージ、実行時間、順序の記録
	HTTP Request	
	String service()	サービス名をリターン
	void service(String name)	サービス名の設定
	int serviceHash()	サービス名のHash値リターン
	String remoteIp()	サービスリクエストIPリターン
	void remoteIp(String ip)	サービスリクエストIPを設定
	boolean isError()	エラーの有無のリターン
	void login(String id)	ログインID設定
	String login()	ログインIDのリターン
	String httpMethod()	HTTP Methodリターン
	String httpQuery()	HTTP Queryリターン
	String httpContentType()	HTTP ContentTypeリターン
	String userAgent()	User-Agentリターン

パラメータ	メソッド／変数	説明
	int status()	HTTP statusリターン
$point	String class1	クラス名
	String method	メソッド名
	Object this1	Hooking対象クラス・メソッド
	Object[] args	ファクター
	Object return1	リターン

パラメータ未指定 - 共通プラグインファイル

パラメータ	メソッド／変数	説明
無し	void log(Object c)	LoggerによるLogging
	void println(Object c)	System.out.println()で出力
	Object field(Object o, String field)	Field値をリターン
	Object method(Object o, String method)	Invokeメソッド
	Object method(Object o, String method, String param)	Invokeメソッド
	String toString(Object o)	ObjectをtoSting() にリターン
	String toString(Object o, String def)	ObjectをtoSting() にリターン、nullの場合def リターン

パラメータ	メソッド／変数	説明
	int syshash(Object o)	hash値リターン
	int syshash(HookArgs hook, int x)	x番目のargumentのhash 値リターン
	int syshash(HookArgs hook)	argumentのhash値リターン
	int cint(Object o)	intでリターン
	float cfloat(Object o)	floatでリターン
	String cString(Object o)	Stringでリターン
	long clong(Object o)	longでリターン
	double cdouble(Object o)	doubleとしてリターン
	String desc(Object o)	Class signatureをリターン
	String toJson(Object o)	jsonでリターン
	void shell(final String cmd, final String env)	shell実行

デーモンおよびバッチ

デーモン、バッチなどのアプリケーション サーバーをモニタリングする場合、開始点として設定したメソッドに適用するAPIです。次のファイルにコードを作成します。

- サービス開始部分適用：`$WHATAP_HOME` /plugin/AppServiceStart.x
- サービス終了部分適用：`$WHATAP_HOME` /plugin/AppServiceEnd.x

プラグインファイル	パラメータ	メソッド／変数	説明
AppServiceStart.x	$ctx	共通属性を参照	

プラグインファイル	パラメータ	メソッド／変数	説明
	$point	共通属性を参照	
AppServiceEnd.x	$ctx	共通属性を参照	

HTTPサービス

`javax.servlet.http.HTTPServlet` クラスを使用する一般的なウェブアプリケーションに適用できるAPIです。次のファイルにコードを作成します。

- サービス開始部分適用：`$WHATAP_HOME` /plugin/HttpServiceStart.x
- サービス終了部分適用：`$WHATAP_HOME` /plugin/HttpServiceEnd.x

プラグインファイル	パラメータ	メソッド／変数	説明
HttpServiceStart.x/ HttpServiceEnd.x	$ctx	共通属性を参照	
	$req	String getCookie(String key)	Cookie値リターン
		String getRequestURI()	RequestURIリターン
		String getRemoteAddr()	RemoteAddrリターン
		String getMethod()	HTTP Methodリターン
		String getQueryString()	HTTP QueryStringリターン
		String getParameter(String key)	Parameter値リターン
		Object getAttribute(String key)	Attribute値リターン
		String getHeader(String key)	Header値リターン
		Enumeration getParameterNames()	getParameterNamesリターン
		Enumeration getHeaderNames()	getHeaderNamesリターン

プラグインファイル	パラメータ	メソッド／変数	説明
		WrSession getSession()	Session Wrapperリターン
		Set getSessionNames()	getSessionNamesリターン
		Object getSessionAttribute(String key)	getSessionAttributeリターン
		boolean isOk()	Plugin状態リターン
		Throwable error()	Errorリターン
	$res	String getContentType()	ContentTypeリターン
		String getCharacterEncoding()	CharacterEncodingリターン
		boolean isOk()	Plugin状態リターン
		Throwable error()	Errorリターン

HTTP Outbound

`HttpClient`などのライブラリを使用してHTTP Outbound Callを実行するときに適用できるAPIです。次のファイルにコードを作成します。

- サービス開始部分適用：`$WHATAP_HOME`/plugin/HttpCallStart.x
- サービス終了部分適用：`$WHATAP_HOME`/plugin/HttpCallEnd.x

プラグインファイル	パラメータ	メソッド／変数	説明
HttpCallStart.x	$ctx	共通属性を参照	
	$req	String url()	URLリターン
		String host()	Hostnameリターン
		int port()	Portリターン

プラグインファイル	パラメータ	メソッド／変数	説明
		boolean isOk()	Plugin状態リターン
		Throwable error()	Errorリターン
HttpCallEnd.x	$ctx	共通属性を参照	
	$res	String getContentType()	ContentTypeリターン
		String getCharacterEncoding()	CharacterEncodingリターン
		boolean isOk()	Plugin状態リターン
		Throwable error()	Errorリターン

特定メソッド

Javaエージェントオプション(`hook_trace_helper_*`)で設定したメソッドに適用するAPIです。

プラグインファイル	パラメータ	メソッド／変数	説明
TraceHelperStart.x	$ctx	共通属性を参照	
	$point	共通属性を参照	
TraceHelperEnd.x	$ctx	共通属性を参照	
	$point	共通属性を参照	

ユーザー定義Pool

ユーザー定義プールをモニタリングするためのAPIです。プール使用量の情報を使用してメソッドを設定すると、WhaTapモニタリングサービスで監視し、統計情報を確認できます。

プラグインファイル	パラメータ	メソッド／変数	説明
CustomPool.x	$id	custom_pool_classesに設定したID値	
	$pool	custom_pool_classesに設定したclass	
	$result	active(Object o)	Active Pool Count設定
		int idle(Object o)	Idle Pool Count設定

プラグインの適用例

以下は、実際プラグインの適用例です。

Elasticsearchエンジンの検索要求モニタリング

Elasticsearch(以下ES) エンジンは、サブレットエンジンではないため、非定型モニタリングを行う必要があります。他の製品が指標中心のモニタリングをする場合、WhaTapのプラグインを使用して、ESの要請および処理時間、検索キーワードを追跡できます。

whatap.conf

```
# トランザクションEndPointとしてorg.elasticsearch.search.SearchService.executeQueryPhaseを設定します。
hook_service_patterns=org.elasticsearch.search.SearchService.executeQueryPhase

# hook_trace_helper_start_patternsに登録されたメソッドが実行されるたびにTraceHelperStart.xファイルのコードを実行します。
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 {
```

```
    String tclass = "org.elasticsearch.search.query.QueryPhase";
    String tmethod = "execute";

    if (tclass.equals($point.class1) && tmethod.equals($point.method)) {
      // 一番目のargumentのqueryメソッドの呼び出し結果をString queryに保存、
      // org.elasticsearch.search.query.QueryPhase.execute(SearchContext searchContext)です。
      // それから、SearchContext.query()メソッドをinvokeした結果をString queryに保存します。
      String query = " " + method($point.args[0], "query");
      // String queryをトレース情報として出力
      $ctx.profile(query);
    }
  } catch(Exception e) {
    $ctx.profile(e.toString());
  }
```

特定のIPで流入されるトランザクションの収集を除外

Health Check、内部ユーザーによるトランザクションの収集を除外する場合は、プラグインを使用できます。

${WHATAP_HOME}/plugin/HttpServiceStart.x

```
// ignoreIP値を収集するIPとして設定します。
String ignoreIP = "123.234.123.234";
String remoteIP = $req.getRemoteAddr();

if ($ctx.ok() && remoteIP != null) {
  if (remoteIP.equals(ignoreIP)) {
    $ctx.ignore();
  }
}
```

Atomikos Poolモニタリング

API文書を参照して適用します。

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);
```

トレースにCookie値を表現

クッキーの値をトレースから確認できます。

${WHATAP_HOME}/plugin/HttpServiceStart.x

```
1  // $req.getCookie() パラメータで確認したいCookieを設定してください。
2  String cookie = $req.getCookie("AWSALB");
3
4  if ($ctx.ok()) {
5     if (cookie != null) {
6        $ctx.profile(cookie);
7     }
8  }
```

ユーザー数設定

Javaウェブアプリケーションサーバーに接続されたユーザー数をエージェント設定を通じて集計する方法を案内します。リアルタイムのユーザー集計の活性化有無の設定、ユーザー数を集計するためのクッキー制限設定、IPアドレスまたはHTTPヘッダーの特定値に基づいたユーザーの集計方法、クライアントIP情報追跡のためのHTTPヘッダー設定などを含みます。

- **trace_user_enabled** Boolean

  既定値 `true`

  リアルタイムのユーザー集計の有無を設定します。ユーザー追跡オプションが重複設定されている場合、動作の優先順位は次のとおりです。

  1. **trace_user_using_ip / wclient_using_ip**
  2. **trace_user_using_jsession**
  3. **user_header_ticket / wclient_header_ticket**

  > Javaエージェント 2.2.0バージョン以降の場合、 `wclient_using_ip` 、 `wclient_header_ticket` オプションをご利用ください。

- **trace_user_cookie_limit / wclient_cookie_limit** Int

  既定値 `2048`

  ユーザー集計のためにクッキーを発行する場合、既存のクッキーが多い場合、クッキーオーバーフローが発生する可能性があります。これを防止するため、limitを設定します。

  > Javaエージェント 2.2.0バージョン以降の場合は、 `wclient_cookie_limit` オプションを使用して下さい。

- **trace_user_using_ip / wclient_using_ip** Boolean

  既定値 `true`

  リアルタイムユーザーをIPアドレスで集約します。IPアドレスでなく、クッキーでユーザーを区分したい場合は、値を `false` に変更してください。

> Javaエージェント 2.2.0バージョン以降の場合は、 `wclient_using_ip` オプションを使用して下さい。

- **user_header_ticket / wclient_header_ticket** String

  HTTP Headerの特定の値でユーザー数を集計する場合は、その対応するKey値を設定します。 モバイルクライアントの場合、ユーザ区分にheaderを使用する場合は、次のオプションが利用できます。

  whatap.conf

```
user_header_ticket=login
```

> Javaエージェント 2.2.0バージョン以降の場合は、 `wclient_header_ticket` オプションを使用して下さい。

- **trace_http_client_ip_header_key** String

  クライアントIP(Remote IP)情報を特定HTTPヘッダーの値に変更して設定する機能です。プロキシ(Proxy) 環境では、 `X-Forwarded-For` ヘッダーの値をクライアントIPとして設定できます。

  WEB/WAS の前にL4 などのロードバランサーが設置されている場合、クライアントのIPアドレスではなく、L4 のIPアドレスがRemote Adressである場合があります。この状況でクライアントのIP情報がHTTPヘッダーに特定キーの値として記録される場合は、そのキー値に置き換えることができます。

- **trace_user_using_jsession** Boolean

  既定値 `false`

  リアルタイムユーザーをSESSIONIDベースで集計します。

負荷制御設定

Javaアプリケーションサーバーのトラフィックスロットリングオプションを設定するエージェントオプションを案内します。アプリケーションの最大同時リクエスト数を制限し、特定のユーザーまたはURLに基づいてサービスへのアクセスを制御する機能を含めます。ユーザー定義型メッセージの送信、URLリダイレクト設定、イベント通知の活性化など、様々なオプションを提供し、過負荷状況でのアプリケーションの安定性を保障できます。

> Javaエージェントのスロットリングオプションは、Servletベースのアプリケーションにのみ適用されます。このオプションに対応するサーバーはTomcat、Weblogic、UndertowなどのServletであり、最新バージョンのSpringはそのオプションに対応していません。

- **throttle_enabled** Boolean

  既定値 `false`

  アプリケーションの最大同時処理数を制限するスロットリング機能を活性化します。throttle_で始まるすべてのオプションは、`throttle_enabled=true` 状態でのみ機能します。

  スロットリング制御に関するポリシーには、次の優先順位があります。

  1. Block：URLとユーザーIPアドレスに基づいてサービスをブロックし、最優先します。
  2. Passing：通過として適用されたURLは、Rejectリシーよりも優先されます。
  3. Reject: Passingポリシーの後にRejectポリシーをブロックして適用します。

- **throttle_limit** Int

  既定値 `10000`

  エージェントごとに同時処理できるリクエスト(トランザクション) の数が設定値を超えたらと、追加で流入するリクエストはRejectされます。

- **throttle_rejected_message** String

  既定値 `too many request!!`

  スロットリングを制限する場合は、ユーザーに案内するメッセージを設定します。

- **throttle_rejected_forward** String

  ユーザーのリクエストがlimit値を超えてrejectされた場合、ユーザーに送信される案内ページのURLを設定します。

`throttle_rejected_message`を同時設定した場合、`throttle_rejected_forward`が優先されます。

> 案内ページを同じコンテナでサービスする場合、これらも同時処理数としてカウントされ、再帰呼び出しによる障害につながる場合があります。そのため、案内ページはstatic htmlページにするか、外部におく必要があります。

- **reject_event_enabled** Boolean

  既定値`false`

  ユーザーリクエストがRejectされた場合、イベント通知を発生させるかどうかを設定します。

- **reject_event_interval** MiliSecond

  既定値`30000`

  Rejectのイベント通知が発生した後、設定した時間の間に重複したイベントの通知は発生しません。

- **throttle_blocking_url** String

  `throttle_limit`を超えない場合でも、ブロッキング(処理拒否)するURLを設定します。システム障害の原因になり得るURLを強制的にブロックする場合に有効です。

- **throttle_blocking_ip** ip_address

  ユーザーのIPアドレスを基準にブロックする場合に設定します。DDoS攻撃や悪意のあるユーザーをIPアドレスベースでブロックするときに使用します。複数の場合はコンマ(,)で区切ります。

- **throttle_target_urls** String

  登録したURLを対象にスロットリング機能を適用します。複数の場合はコンマ(,)で区切ります。

- **throttle_passing_url** String

  `throttle_limit`を超えた場合でも処理するURLを設定します。複数の場合はコンマ(,)で区切ります。

- **throttle_passing_url_prefix** String

  `throttle_limit`を超えた場合でも処理するURLを設定します。複数の場合はコンマ(,)で区切ります。

- **throttle_blocked_message** String

  既定値`request blocked!!`

  リクエストをブロックしたユーザーに案内するメッセージを設定します。

- **throttle_blocked_forward** String

  リクエストをブロックしたユーザーに案内するURLを設定します。`throttle_blocked_message`を同時設定した場合、`throttle_blocked_forward`を優先します。

エージェント通知設定

Javaエージェントを通じてアプリケーションサーバーで発生する様々なイベントに対する通知の設定方法を提供します。トランザクションの再帰呼び出し、サービス拒否、HTTPC接続エラー、ヒープおよびディスク使用量超過、CPU使用量しきい値到達、DBコネクション重複割り当て、例外発生時にイベント通知を設定するオプションが含まれます。各イベントごとに発行間隔、発行可否、しきい値の設定など、細かい調整ができます。

- **recursive_event_interval** MiliSeconds

  既定値 `300000`

  トランザクションの再帰呼び出しのイベント通知発行間隔を設定します。

- **reject_event_enabled** Boolean

  既定値 `false`

  サービス拒否(呼び出し負荷制限/拒否) 時、イベント通知発行の可否を設定します。

- **reject_event_interval** MiliSeconds

  既定値 `300000`

  サービス拒否(呼び出し負荷制限/拒絶) 時、イベント通知発行の間隔を設定します。

- **httpc_event_enabled** Boolean

  既定値 `false`

  HTTPC接続エラーが発生した場合、イベント通知を発行するかどうかを設定します。

- **httpc_event_interval** MiliSeconds

  既定値 `300000`

  HTTPC接続エラーが発生した場合、イベント通知発行の間隔を設定します。

- **heap_event_enabled** Boolean

  既定値 `false`

  ヒープ使用量のしきい値に達した場合、イベント通知を発行するかどうかを設定します。

- **heap_event_percent** Percentage

  既定値 `90`

ヒープ使用量イベント通知の発行基準のしきい値を設定します。

- **heap_event_duration** MiliSeconds

  既定値 30000

  ヒープ使用量イベント通知の発行基準の持続時間を設定します。

- **heap_event_interval** MiliSeconds

  既定値 300000

  ヒープ使用量イベント通知の発行間隔を設定します。

- **heap_event_action** String

  既定値 NONE

  ヒープ使用量イベントが発生した場合、実行する動的ローディングコードを設定します。

  > _ $WHATAP_HOME/plugin/ActionScript.x_ファイルに記述されたJavaコードに渡すID($id で渡す)

- **disk_event_enabled** Boolean

  既定値 false

  ディスク使用量がしきい値に達した場合、イベント通知を発行するかどうかを設定します。

- **disk_event_percent** Percentage

  既定値 90

  ディスク使用量イベント通知発行基準のしきい値を設定します。

- **disk_event_interval** MiliSeconds

  既定値 300000

  ディスク使用量イベント通知の発行間隔を設定します。

- **disk_event_action** String

  ディスク使用量イベントが発生した場合、実行する動的ローディングコードを設定します。

  > _ $WHATAP_HOME/plugin/ActionScript.x_ファイルに記述されたJavaコードに渡すID($id で渡す)

- **cpu_event_enabled** Boolean

  既定値 `false`

  CPU使用量がしきい値に達したとき、イベント通知を発行するかどうかを設定します。

- **cpu_event_percent** Percentage

  既定値 `90`

  CPU使用量のイベント通知を発行するしきい値を設定します。

- **cpu_event_duration** MiliSeconds

  既定値 `30000`

  CPU使用量のイベント通知を発行する持続時間を設定します。

- **cpu_event_interval** MiliSeconds

  既定値 `300000`

  CPU使用量のイベント通知を発行する間隔を設定します。

- **cpu_event_action** String

  CPU使用量のイベントが発生した場合、実行する動的ローディングコードを設定します。

  > _ `$WHATAP_HOME` /plugin/ActionScript.x_ファイルに記述されたJavaコードに渡すID(`$id` で渡す)

- **dbc_dup_event_enabled** Boolean

  既定値 `false`

  DB Connectionが冗長に割り当てられる場合、イベント通知の発行可否を設定します。

- **dbc_dup_event_fullstack_enabled** Boolean

  既定値 `false`

  DB Connectionが重複して割り当てられる場合、Stackを確保するかどうかを設定します。

  > _ `$WHATAP_HOME` /plugin/ActionScript.x_ファイルに記述されたJavaコードに渡すID(`$id` で渡す)

Apdex設定

ユーザー満足度を測定するApdexスコアを設定するJavaエージェントオプションを案内します。Apdex指標を通じてアプリケーションの性能を客観的に評価し、ユーザー体験を改善できます。

whatap.apdex_time millisecond

既定値 `1200`

満足するS既定値は、エージェント設定メニューから変更できます。

Application Performance Index(Apdex)は、アプリケーションのパフォーマンス指標です。ウェブアプリケーションの顧客満足度を測定します。ユーザー満足度の指標として活用でき、0~1の間の値で表示します。

(満足回数+ (許容回数*0.5))／全体リクエスト数

- 満足(Satisfied, S)：タスク処理に問題なし≤ 1.2秒(満足S既定値)
- 許容 (Tolerating, T)：ユーザーが遅延を感じるが、タスク処理は可能≤4.8秒(満足S*4)
- 不満 (Frustrated, F)：タスク処理の不可能 > 4.8秒 (許容T超過及びエラー)

> エージェントバージョン2.0以降でサポーされています。

統計設定

Javaアプリケーションモニタリングのための様々な統計収集機能関連エージェントオプションをご案内します。性能カウンタの拡張、ドメイン別トランザクションの収集、マルチサーバートランザクションの依存性分析、ログインタイプ別及びRefererer別の統計収集などが含まれます。また、SQL、HTTP Call、エラー統計とユーザエージェント情報収集の最大レコード数制限設定も確認できます。

- **addin_monitor_num** Int

  既定値 `0`

  パフォーマンスカウンターの拡張のために追加して宣言するクラス数を設定します。`addin_monitor_num` に設定した値と等しく `addin_monitor_x` を宣言します。

- **addin_monitor_0** Int

  既定値 `0`

  whatap.conf

  ```
  addin_monitor_num=2
  addin_monitor_0=a.a.Class1
  addin_monitor_1=a.a.Class2
  ```

- **stat_domain_enabled** Boolean

  既定値 `false`

  クライアントに接続中のドメインによるトランザクション統計の収集機能を有効にします。

- **stat_domain_max_count** Int

  既定値 `7000`

  1つのJVMから5分間収集されるドメインごとのトランザクション統計の最大レコード数です。

- **stat_mtrace_enabled** Boolean

  既定値 `false`

マルチサーバトランザクションでCaller&Calleeの間のバージョンごとの依存関係の統計を収集します。

- **mtrace_spec** String

  既定値 `v1`

  現在のインスタンスのアプリケーションバージョンを設定します。任意の文字列を設定できます。このデータは、呼び出し統計に使用されます。

- **stat_mtrace_max_count** Int

  既定値 `7000`

  マルチサーバトランザクションでCaller & Calleeの間のバージョンごとの依存関係の統計における最大レコード数です。

- **stat_login_enabled** Boolean

  既定値 `false`

  ログインごとのトランザクション統計を収集します。

- **stat_login_max_count** Int

  既定値 `7000`

  1つのJVMから5分間に収集するログインごとのトランザクション統計の最大レコード数です。

- **stat_referer_enabled** Boolean

  既定値 `false`

  Refererごとのトランザクション統計を収集します。

- **stat_referer_max_count** Int

  既定値 `7000`

  1つのJVMから5分間に収集するRefererごとのトランザクション統計の最大レコード数です。

- **stat_tx_max_count** Int

  既定値 `5000`

  トランザクション統計情報の数を制限します。5分間に収集してサーバーに送信される統計情報の最大レコード数を制限します。

- **stat_sql_max_count** Int

既定値 `5000`

SQL統計情報の数を制限します。5分間に収集してサーバーに送信される統計情報の最大レコード数を制限します。

- **stat_httpc_max_count** Int

  既定値 `5000`

  HTTP Call統計情報の数を制限します。5分間収集してサーバーに送信する統計情報の最大レコード数を制限します。

- **stat_error_max_count** Int

  既定値 `1000`

  Error統計情報の数を制限します。5分間に収集してサーバーに送信される統計情報の最大レコード数を制限します。

- **stat_useragent_max_count** Int

  既定値 `500`

  User Agent統計の情報の数を制限します。5分間に収集してサーバーに送信される統計情報の最大レコード数を制限します。

トランザクションエラースタック設定

Javaアプリケーションでトランザクション中に発生するエラーを感知し、これに対するスタックトレースを表示するエージェント設定オプションを案内します。 トランザクション、メソッド、SQL、HTTP呼び出し、SOCKET呼び出しなど様々な領域からエラー情報を収集し、エラースタックの長さ、タイトルの長さなどを設定する方法が含まれます。エージェント設定によりエラー管理を改善し、アプリケーションのエラー診断をより効率的に行えるようにサポートします。

呼び出しサービスの、 `status_code` とは無関係であり、複数のエラーが発生した場合、エラースタックは最初のエラースタックのみを表示します。

> この文書で案内するエージェント設定は、**Javaエージェント2.2.15**バージョン以降で対応しています。

エージェント設定

全体設定

whatap.conf

```
#エラースタック表示の有無
trace_exception_stack=false

#エラースタック長さの既定値
trace_exception_depth=50

#エラースタックメッセージタイトルの長さ
trace_exception_title_len=128
```

メソッド

whatap.conf

```
#メソッドエラースタック表示の有無
```

```
# trace_exception_stack値に従って、別途にtrueまたはfalseで機能をオン、オフできます。
trace_method_error_stack=trace_exception_stack

#メソッドエラースタック長さ
# trace_exception_depth値に従って、別途に設定できます。
trace_method_error_depth=trace_exception_depth
```

SQL

whatap.conf

```
# SQLエラースタック表示の有無
# trace_exception_stack値に従って、別途にtrueまたはfalseで機能をオン、オフできます。
trace_sql_error_stack=trace_exception_stack

# SQLエラースタック長さ
# trace_exception_depth値に従って、別途に設定できます。trace_sql_error_depth=trace_exception_depth
```

HTTP Call

whatap.conf

```
# http callエラースタック表示の有無
# trace_exception_stack値に従って、別途にtrueまたはfalseで機能をオン、オフできます。
trace_httpc_error_stack=trace_exception_stack

# http callエラースタック長さ
# trace_exception_depth値に従って、別途に設定できます。trace_httpc_error_depth=trace_exception_depth
```

SOCKET

whatap.conf

```
# socketエラースタック表示の有無
# trace_exception_stack値に従って、別途にtrueまたはfalseで機能をオン、オフできます。
trace_socket_error_stack=trace_exception_stack
```

```
# socketエラースタック長さ
# trace_exception_depth値に従って、別途に設定できます。trace_socket_error_depth=trace_exception_depth
```

トランザクション

whatap.conf

```
# トランザクションエラースタック表示の有無
# trace_exception_stack値に従って、別途にtrueまたはfalseで機能をオン、オフできます。
trace_endtx_error_stack=trace_exception_stack

# トランザクションエラースタック長さ
# trace_exception_depth値に従って、別途に設定できます。trace_endtx_error_depth=trace_exception_depth
```

エージェント追加オプション

Javaアプリケーションのモニタリングレベルをユーザーのニーズに合わせて調整できる追加のエージェント設定オプションを案内します。SQLパラメータの収集、HTTPパラメータとヘッダ情報の収集、ユーザーIP情報の抽出、ユーザー数の追跡方法の選択、SAP Functionの追跡などを含みます。これらのオプションにより、より詳細なデータ収集と分析が可能になり、特にセキュリティが重要な環境で必要に応じてオプションを調整できます。

以下は、頻繁に使用されるがセキュリティなどの理由でエージェントの基本設定から除外したオプションです。必要に応じて、以下のオプションを追加して適用できます。

- **profile_sql_param_enabled** Boolean

  既定値 `true`

  SQLパラメータを収集します。

- **profile_http_parameter_enabled** Boolean

  既定値 `true`

  HTTPパラメータを収集します。

- **profile_http_header_enabled** Boolean

  既定値 `false`

  HTTP Header情報を収集します。

- **trace_http_client_ip_header_key** String

  既定値 `X-Forwarded-For`

  ユーザーIP情報をHTTPヘッダーのキーから抽出します。

- **trace_user_method / wclient_cookie_limit** Boolean

  既定値 `cookie`

  ユーザー数の追跡をクッキーベースに行います。

  > ⓘ Javaエージェント 2.2.0バージョン以降の場合、 `wclient_trace_type` オプションをご利用ください。

- **hook_httpc_sap3_enabled** Boolean

既定値 `true`

SAP JCOを使用する場合は、SAP Functionを追跡します。

非同期トラッキング

Javaエージェント設定を介して非同期アプリケーション活動を追跡する方法を案内します。 `CompletableFuture` メソッドの収集方法と、ユーザーアプリケーションにそのメソッドをhookingする方法が含まれます。 設定例とともに whatap.confファイルに追加すべきオプションを提供し、非同期呼び出しの性能モニタリングを強化できるようにサポートします。

CompletableFuture

Javaエージェント設定を通じてCompletableFutureメソッドを収集する方法をご案内します。

CompletableFutureメソッドの追加

ユーザーのJavaアプリケーションプロジェクトにエージェントとしてhookingするCompletableFutureメソッドを追加します。

- メソッド名：'trace'で統一
- 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;
}
}
```

エージェント設定の追加

エージェント設定のため、whatap.confファイルに次のオプションを追加します。

whatap.conf

```
hook_completablefuture_patterns=io.home.test.util.W.*
```

> ⓘ
> - Contextがある場合は、 `trace` メソッドを追加してトランザクションを接続します。
> - Contextがない場合は、 `hook_service_patterns` オプションでサービスを開始する必要があります。

使用例

- **イベントソース**

```
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()` 適用

```
1  // io.home.test.util.W.trace()適用
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エージェントのアップデート

最新バージョンのエージェントをダウンロード

1. WhaTapモニタリングサービスに移動してログインしてください。
2. エージェントを更新するプロジェクトを選択します。
3. 左側のメニューから**管理** > **エージェントアップデート**を選択します。
4. ⬇ **最新バージョン** または、 ⬇ ボタンをクリックします。
5. エージェントをダウンロードして、アプリケーションサーバーを再起動してください。アップデートを自動的に適用します。

エージェントアップデートの自動設定

Javaエージェントをアップデートするには、サーバーを再起動する必要があります。再起動前にwhatap.agent.jarファイルが変更されると、アプリケーションサーバが再起動する過程で安全なシステム終了(shutdown)が失敗する可能性があります。

Javaエージェントを更新するには、次の手順で進める必要があります。

1. app shutdown
2. agent jar変更
3. app start

このプロセスを簡単に処理するために、WhaTapのエージェントjarファイル名にはバージョン番号が含まれています。アップデート中に最新バージョンを自動適用するように設定することが重要です。

```
########## 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エージェントの削除

エージェント削除

アプリケーションサーバーでJVMオプションの `-javaagent` 設定を削除してからサーバーを再起動してください。次に、`$WHATAP_HOME` 環境変数を削除します。

エージェントのロールバック

1. WhaTapモニタリングサービスでロールバックするプロジェクトを選択します。
2. 左側のメニューから**管理** > **エージェントアップデート**を選択します。
3. **他のバージョン**ボタンを選択します。
4. ロールバックするバージョンを選択し**選択バージョン**またはボタンをクリックします。



5. エージェントをインストールする時に修正したスクリプトから `WHATAP_JAR` 環境変数をロールバックするバージョンに変更します。

```
########## 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グループ

Auto Scale Out環境で自動登録されたエージェントのIP／Port／PID情報だけでは業務単位の情報を認識することが難しい場合があります。この場合、 `okind` 属性をJVMオプションを指定することで簡単に識別できます。

次は典型的なDocker環境の説明です。

- 各エージェントは、細かく分けたMSA(Micro Service Architecture) コンポーネントです。
- エージェントは、自動的にScale In/Outされます。
- どのIPを持つサーバーからどのPIDでアプリケーションが起動するか予測できません。

上記のようなDocker環境でエージェント名とともに業務を識別できるJVMオプションとして `whatap.okind` を追加して `okind` 属性を追加することができます。

JAVA_OPTS

```
-Dwhatap.okind=mobile_ui
```

whatap.confファイルに設定を追加することもできます。

whatap.conf

```
whatap.okind=common_api
```

> MSA環境でWhaTapが提供する様々な機能を利用する場合は、 `okind` 属性を追加します。

AES 256暗号化

WhaTapアプリケーション エージェントは、収集したデータを暗号化してサーバーに送信します。このために、XOR(Exclusive-OR) 演算と高度暗号化標準 (Advanced Encryption Standard,以下AES) 暗号化を使用し、平文を128bit単位に分割して暗号化と複合化を行います。256ビットに拡張してセキュリティを強化します。

基本的にJava Cryptography Extension(以下JCE)は、128ビットに対応してします。AES256ビットを適用するには、JCEをアップデートする必要があります。

基本環境でAES256を適用すると、次のようなエラーが発生します。

```
Unsupported keysize or algorithm parameters.
##あるいは、
Illegal key size or default parameters.
```

1. 次のリンクから、バージョンに適合するファイルをダウンロードしてください。

   > ⓘ **JavaバージョンのJCEダウンロードリンク**
   >
   > - JAVA 8
   > - JAVA 7
   > - JAVA 6
   > - JAVA 5
   > - JAVA 1.42

2. _ `$JAVA_HOME` /jre/lib/security_パスにファイルを上書きしてください。
3. JCEを適用した場合は、_ `${WHATAP_HOME}` /whatap.conf_ファイルに次の設定を追加します。

   whatap.conf

   ```
   cypher_level=256
   ```

4. アプリケーションサーバーを再起動すると、AES256の暗号化が適用されます。

スケールイン自動削除

Auto Scale環境でエージェント名を自動付与すると、Scale Outが発生した場合、オペレータの介入は必要ありません。システムを自動的に拡張します。一方、Scale Inが発生すると、システムが正常なShut downとして認識する必要があります。

エージェントの自動削除の場合、WhaTapサーバーは、Javaエージェントからメッセージを受信する必要があります。エージェントは、`${WHATAP_HOME}`にJavaプロセスの_whatap*`{java-process-pid}`.shutdown*ファイルが生成されたら**SILENT_SHUTDOWN**イベントをWhaTapサーバーに送信します。送信は5秒以内に完了します。

SILENT_SHUTDOWNイベントをWhaTapサーバに送信してから1分以内にエージェントがシャットダウンされた場合、WhaTapサーバはリストからJavaエージェントを削除します。inactiveイベントは発生しません。

> Tomcatの場合、shutdown.shファイルで設定できます。
>
> ```
> touch $WHATAP_HOME/whatap_{java-process-pid}.shutdown
> ```

Javaダッシュボード

WhaTapエージェントのインストールが完了したら、WhaTapモニタリングサービスにログインしてください。初期画面は**プロジェクトリスト**です。**プロジェクトリスト**からエージェントをインストールしたアプリケーションサーバーが確認できます。エージェントをインストールした1つのアプリケーションサーバーはプロジェクトであり、モニタリング単位です。プロジェクトリストからプロジェクトを選択すると、**アプリケーションダッシュボード**に移動できます。

ダッシュボードとは？

WhaTapモニタリングサービスでダッシュボードはプロジェクトの全体的な状態を一目で確認できる機能です。モニタリングされているリソース全体の規模を確認でき、リアルタイムにダッシュボードでモニタリング対象リソースをフィルタリングできます。進行中のトランザクションと終了したトランザクション情報をリアルタイムで更新し、可視化されたチャートを提供します。システム運用者は、サービスとシステムの正確な現在の状態を直感的に把握できます。また、障害を素早く認識し、潜在的な問題要素を迅速に特定することができます。

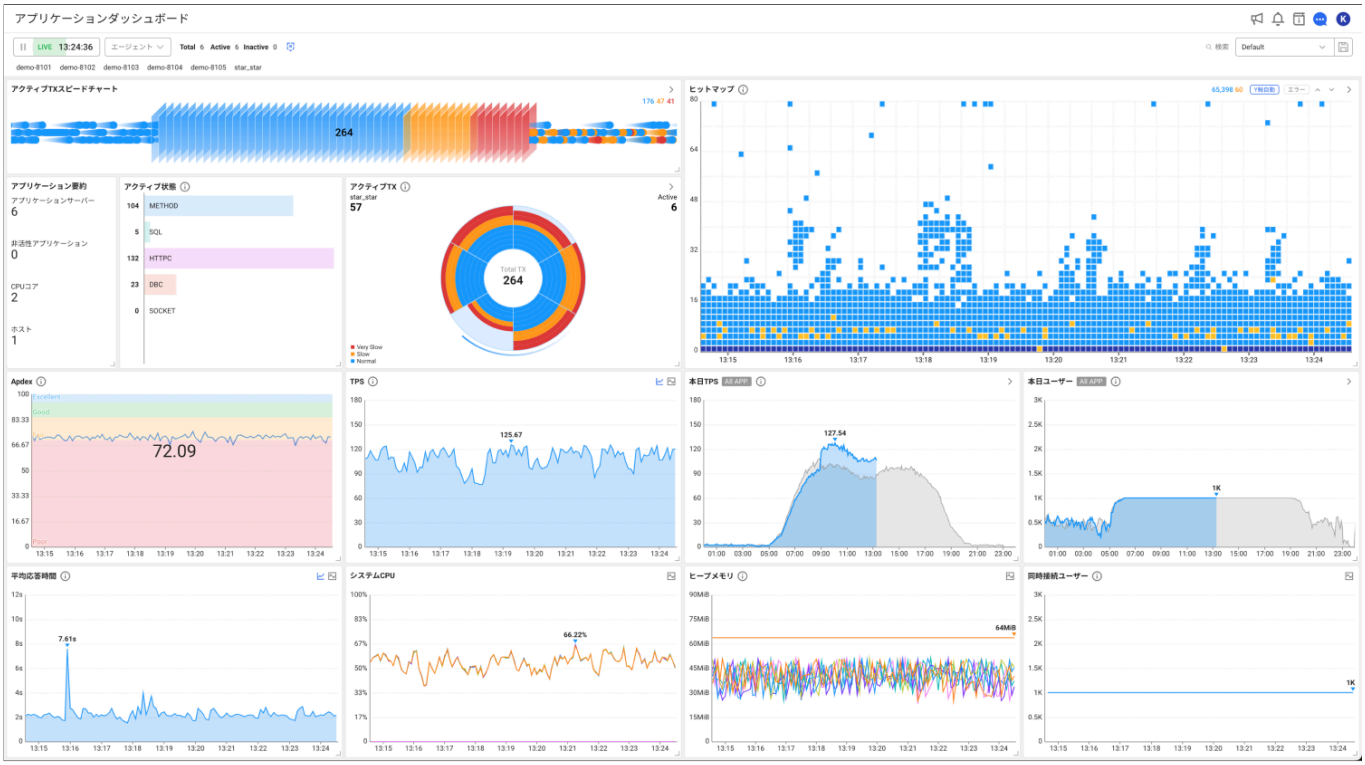
ダッシュボードで把握できる情報は、次の通りです。

- 応答時間分布チャートを使用すると、応答時間範囲を調整したり、エラーデータをフィルタリングしたりできます。
- 進行中のトランザクションを分析し、終了したトランザクションをリアルタイムで更新します。
- TPS、応答時間、CPU、メモリ、リアルタイム使用などのリソース状況を把握できます。
- 最新のデータを自動更新し、リアルタイムのモニタリングサービスを提供します。

次の動画を見てアプリケーションダッシュボードの詳細を確認して下さい。

アプリケーションダッシュボード

ホーム画面 > プロジェクト選択 > **ダッシュボード** > **アプリケーションダッシュボード**



アプリケーションダッシュボードでは、アプリケーションの主な指標をエージェントを通じて収集します。また、チャートを構成してリアルタイムでモニタリングし、過去のデータを表示することができます。 ダッシュボードに配置されたウィジェットを通じて、アプリケーションの問題を素早く特定して調査することができます。

CPU、Memoryを除く一般的なチャートの場合、安定したデータは青色の系列で表現され、問題として識別される要素は赤色の系列で表示されるため、利用状況を簡単に認識できます。

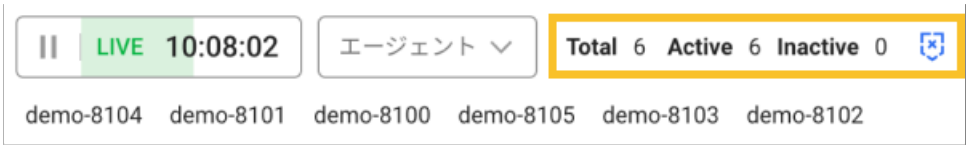
照会時間設定

ダッシュボードではリアルタイムモニタリング機能を基本的に提供しますが、画面上部のタイムセレクターを利用しても過去の時間のデータを照会できます。 タイムセレクターで ⏸ ボタンを選択して時間を設定します。 ユーザーが設定

した時間に基づいて、ダッシュボードに配置したウィジェットのデータを更新します。

エージェント確認

エージェント接続状態を確認する



画面の左上のタイムピッカーの右側には、プロジェクトに関連したエージェントの状態を確認できる情報が表示されます。 この機能で、モニタリング対象サーバーの動作有無がすぐに確認できます。

- Total: プロジェクトに関連付けられているすべてのエージェントの数
- Active: 活性化されたエージェントの数
- Inactive: 非活性化状態のエージェント数
- : 非活性化されたエージェントを表示または非表示します。

エージェント別モニタリング



基本的に、ダッシュボードにはすべてのエージェントから収集した指標をチャートとして表示しますが、エージェントを選択してデータを表示することもできます。 タイムピッカーの下にあるエージェントを1つ以上選択します。 選択したエージェントの指標でダッシュボードに配置されたウィジェットのデータを更新します。

エージェントを1つ以上選択した状態で再度すべてのエージェントを選択するには、選択を解除するか、Totalをクリックします。

プロジェクトに接続されたエージェントの数が多い場合は、エージェント名を短く設定することをお勧めし

ます。 エージェント名の設定に関する詳細は、次の文書を参照してください。

分類別エージェントモニタリング

エージェント設定で分類した単位で複数のエージェントを簡単にモニタリングできます。

- **エージェント**: 個別エージェント
- **種類別**: エージェントの種類(例: ウェブサーバー、データベースなど)に分類されたグループ、 `whatap.okind` に設定
- **サーバー別**: サーバ(ノード)別に分類されたグループ、 `whatap.onode` に設定
- **種類別エージェント**: `whatap.okind` に分類されたエージェントの一覧
- **サーバー別エージェント**: `whatap.onode` に分類されたエージェントの一覧

エージェント設定の詳細については次の文書を参照してください。

エージェントを検索する

プロジェクトに接続されたエージェントの一覧が多く、見つけるのが難しい場合は、エージェント検索機能を利用します。 画面の左上にある**検索**ボタンをクリックすると、文字列を入力するボックスが表示されます。



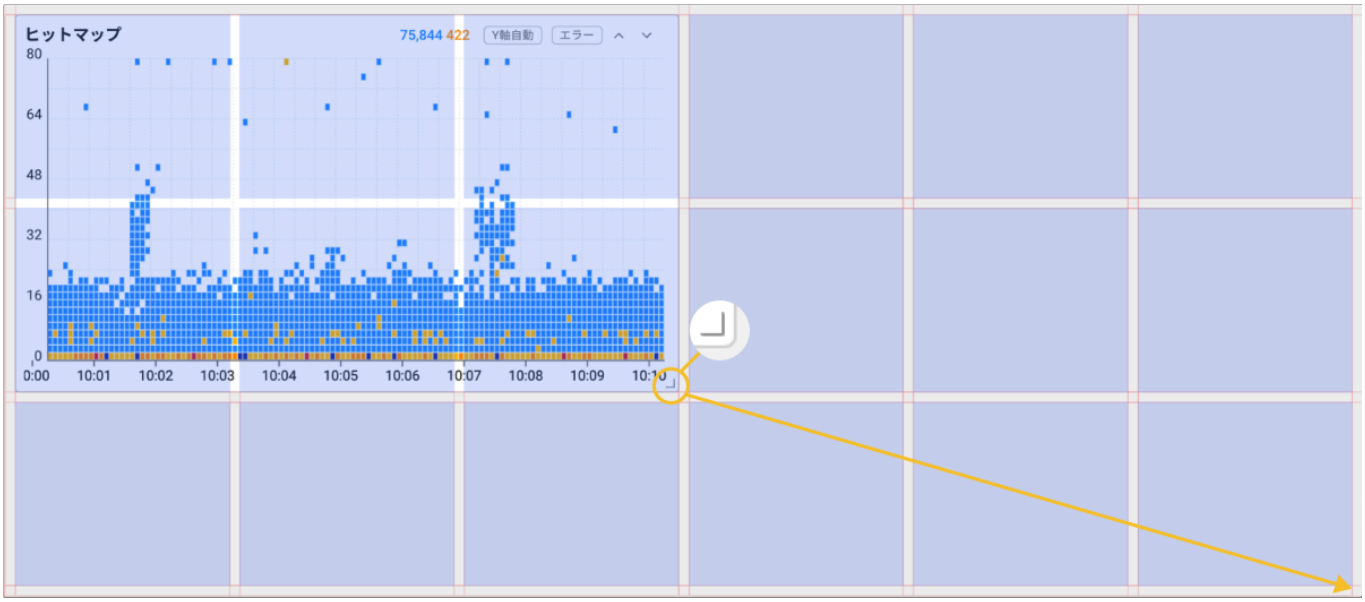
入力した文字列に一致するエージェントのみが、エージェント一覧に表示されます。 検索したエージェントに基づいてダッシュボードのデータをフィルタリングするには、文字入力ボックスの右側にある**選択する**ボタンをクリックします。

選択したエージェント項目を初期化するには、文字入力ボックスの右側の**閉じる**ボタンをクリックします。

ウィジェット編集

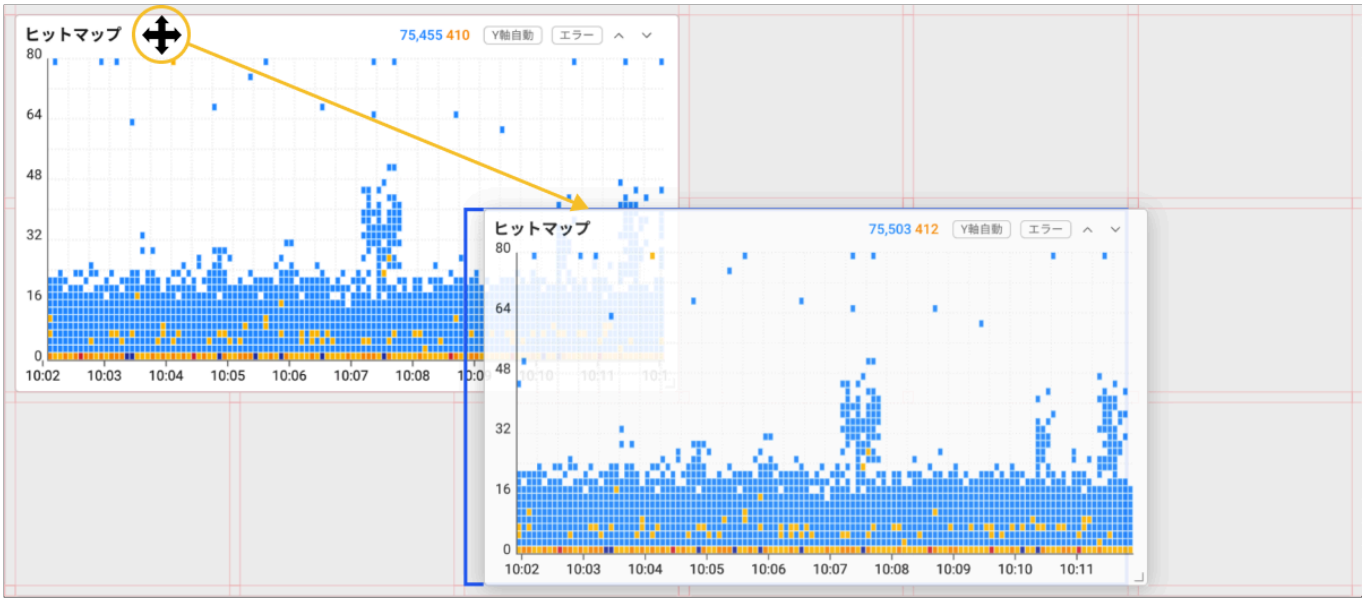
ダッシュボードに配置したウィジェットは、ユーザーが希望するサイズに調整でき、任意の位置に配置できます。 るウィジェットは削除したり、再度追加したりすることもできます。

ウィジェットのサイズ変更



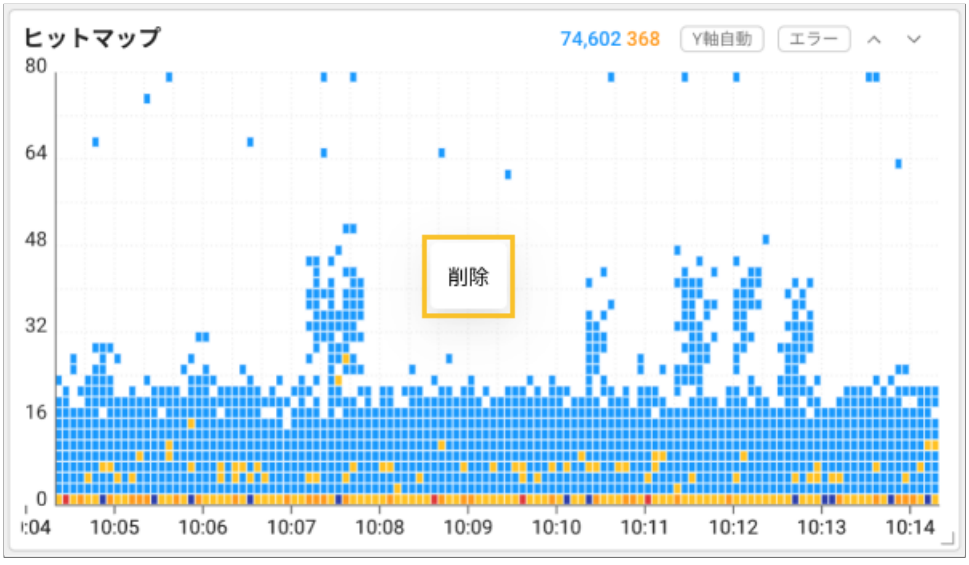
ウィジェットの右下にある⌟をマウスでクリックした状態で希望するサイズにドラッグします。 幅と高さの比率が均一なグリッドが表示され、グリッド単位でウィジェットのサイズを調整できます。

ウィジェットの移動



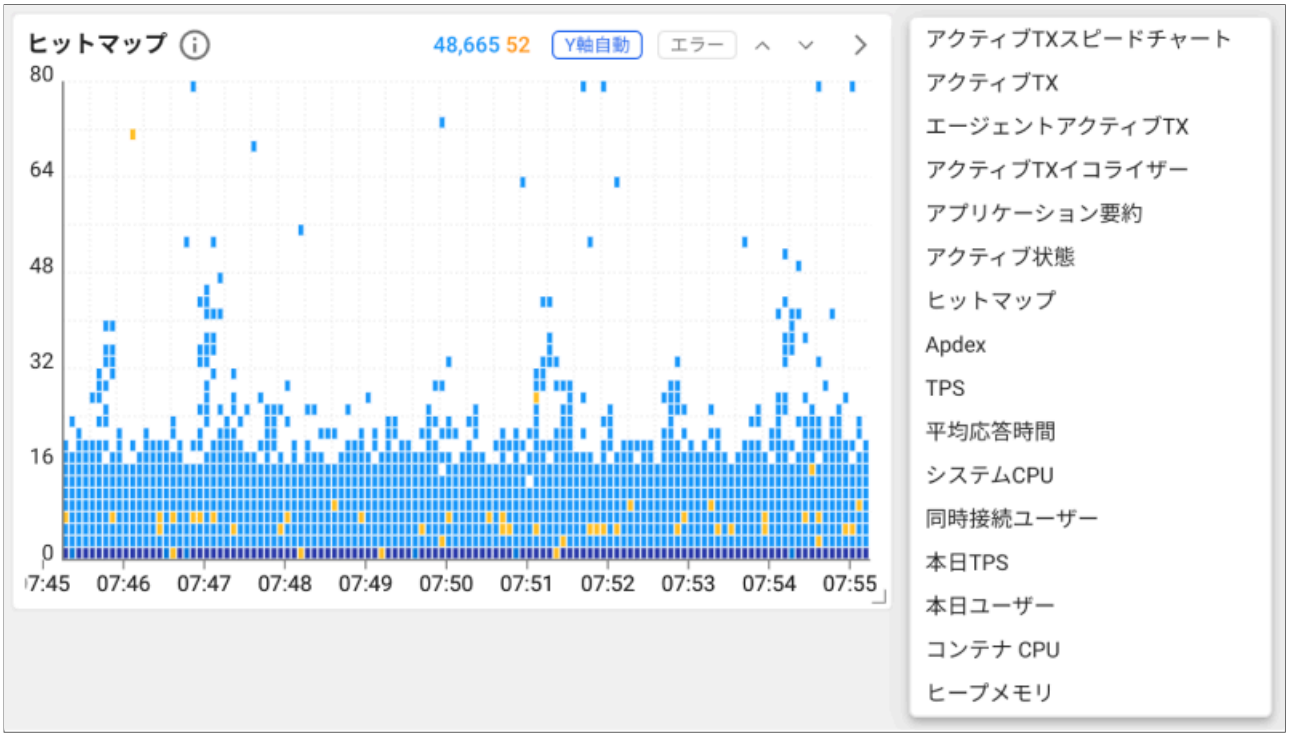
ウィジェットの上部にマウスカーソルを位置すると、カーソルの形が✢の形に変更されます。 この時、マウスの左ボタンをクリックした状態で希望する位置にドラッグすると、ウィジェットを移動することができます。

ウィジェットを削除する



削除するウィジェットで右クリックします。 **削除**ボタンをクリックすると、ウィジェットがダッシュボードから削除されます。

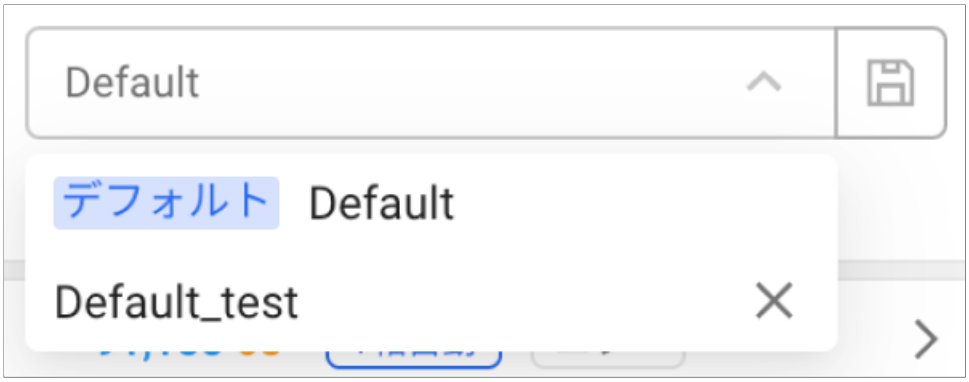
ウィジェットを追加する



ダッシュボードから空白エリアにマウスカーソルを移動し、マウスの右ボタンをクリックします。 ポップアップメニューから追加するウィジェットを選択します。 希望する場所にウィジェットを配置してサイズを変更します。

- ダッシュボードに配置できるウィジェットの詳細については、次の文書を参考にしてください。
- 現在追加できるウィジェットは固定されていますが、今後のアップデートでウィジェットサポートを追加していく計画です。
- ダッシュボードで基本的に提供されるメトリクスの他に、ユーザーが希望するメトリクスを追加できます。 ユーザー定義ウィジェットの詳細については、次の文書を参照してください。

プリセット設定



ダッシュボードからユーザーが設定したウィジェットの設定とレイアウト状態を保存して読み込むことができます。ウィジェットのサイズを調整し、希望する場所に配置して新しいプリセットを作成できます。

- **デフォルト**に設定されたプリセットは変更できません。
- **プロジェクト修正**権限のあるメンバーのみがプリセットを保存または変更できます。 メンバー権限の詳細については、次の文書を参照してください。

新しいプリセットを作成する

1. ダッシュボードから、希望する形にウィジェットを配置してみてください。 サイズを変更して、頻繁に確認するウィジェットのみを配置することもできます。
2. 画面の右上に保存ボタンを選択します。
3. 新しいプリセット名を入力します。



エージェント選択履歴を一緒に保存するには、**エージェント選択履歴を含む**を選択します。

4. **保存**ボタンを選択します。

プリセット一覧から新しく保存したプリセットを確認できます。

> ⓘ
> - 新しく作成したプリセットに変更が加えられた場合は、再度プリセットを保存してください。 ボタンを選択し、同じ名前でプリセットを保存します。 既存のプリセットに変更を上書きします。
> - ダッシュボードの変更を保存せずに別のメニューに移動すると、変更は保存されません。
> - プリセットはプロジェクト単位で保存され、他のユーザーと共有できます。

プリセットを削除する

使用していないプリセットは、プリセット一覧から削除できます。 プリセット一覧から削除したい項目の右側に✕ボタンをクリックします。

> ⓘ **Default**プリセットは削除できません。

基本プリセットを変更する

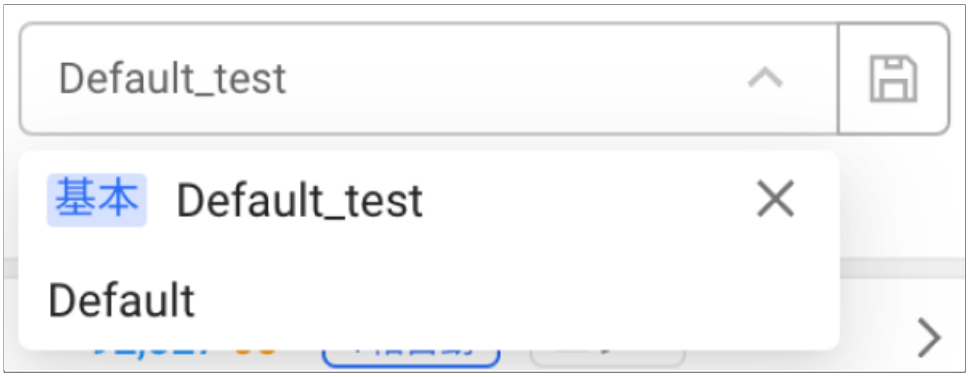
アプリケーションメニューに進入した時に表示されるプリセットを変更できます。

1. ダッシュボードの右上にあるプリセット選択ボックスを選択します。
2. 基本に設定するプリセットに、マウスをオーバーします。



3. **デフォルトで表示**ボタンが表示されたら選択します。

選択したプリセットが基本プリセットに変更され、プリセット一覧に**基本**タグが表示されます。



ウィジェットオプション

ウィジェットに表示されているアイコンボタンの機能は次のとおりです。

- ⓘ：主要ウィジェットに関する機能と情報が確認できます。
- **マージ/個別に表示** : ウィジェット項目のエージェント データを個別またはマージしてグラフで表示します。
- **詳細** : ウィジェット項目のデータをエージェント別に区分して照会できるポップアップが表示されます。

> ⓘ ウィジェットによって使用可能なオプションは異なる場合があります。

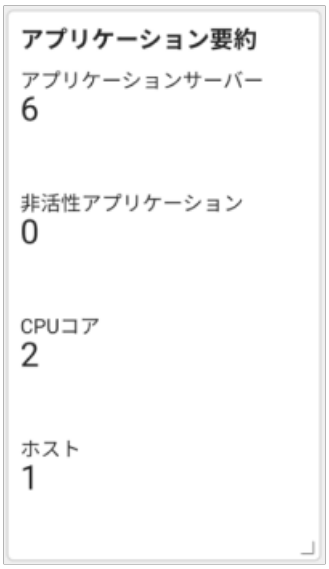
ウィジェットを調べる

> この文書では、**アプリケーションダッシュボード**に配置されたウィジェットについて説明します。 **アプリケーションダッシュボード**を使用したトランザクション、ユーザー、サービス及びリソース分析の方法についての詳細は、次の文書を参照してください。
>
> - アプリケーションダッシュボードを確認する
> - ヒットマップTX
> - DB接続遅延とコネクションプール

エージェント関連ウィジェット

アプリケーションにインストールしたエージェントを使用して、アプリケーションの情報を確認できます。

アプリケーション要約



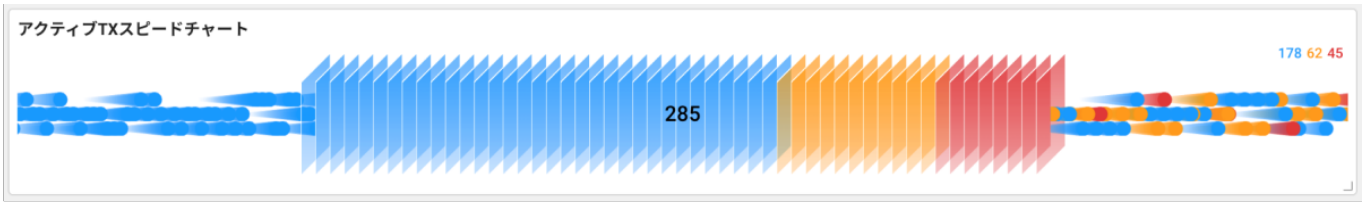
プロジェクトに登録したアプリケーションの実行状況および個数、CPUコア数、Hosts数を表示します。

トランザクション関連ウィジェット

トランザクションを通じて確認できるトラブルの事象は、まず応答時間で把握できます。 また、進行中のトランザクションが終了しない場合は、これもトラブルとして認識すべきです。 WhaTap Monitoringでは、進行中のトランザクションの遂行時間に基づいて状態を表示します。 青色(Normal)は応答時間が正常なトランザクション、オレンジ色(Slow)は応答時間が8秒までの遅いトランザクション、赤色(Very slow)は、応答時間が8秒を超えた非常に遅いトランザクションを意味します。 これにより、ユーザーはトラブル状況が直感的にすぐ認識できるようになります。

> **連携プロジェクト管理**メニューでデータベースプロジェクトを追加すると、次のウィジェットで連携したプロジェクトの性能メトリクスを照会できます。
>
> - **アクティブTXスピードチャート**, **アクティブTX**: 現在進行中のトランザクションと連携されたデータベースのSQL性能を照会できます。
> - **ヒットマップ**: 過去の時点でのトランザクションに連携されたデータベースの性能メトリクスを照会できます。
>
> 連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

アクティブTXスピードチャート



現在進行中のトランザクション(中央領域)と、終了したトランザクション(右領域)の状態を確認できます。 左から右に流れる水玉模様のアイコンは、トランザクションを意味します。 チャート領域をクリックすると、進行中のトランザクション一覧を確認できる画面が表示されます。 トランザクション一覧のいずれかを選択すると、詳細なトランザクション情報とSQLクエリ、コールスタック情報を確認できます。

- 左領域

  リクエストTXの数を意味し、TXの数に比例してより多く、より速く表示します。

- 中央領域

  アクティブTXの状態を速度で色分けし、右上に個数を表示します。 チャートの数字は、アクティブTX全体の数で

す。 チャートの色の意味は次のとおりです。

- 青色(Normal)： 0秒~3秒
- オレンジ色(Slow)：3秒~8秒
- 赤色(Very Slow)：8秒以上

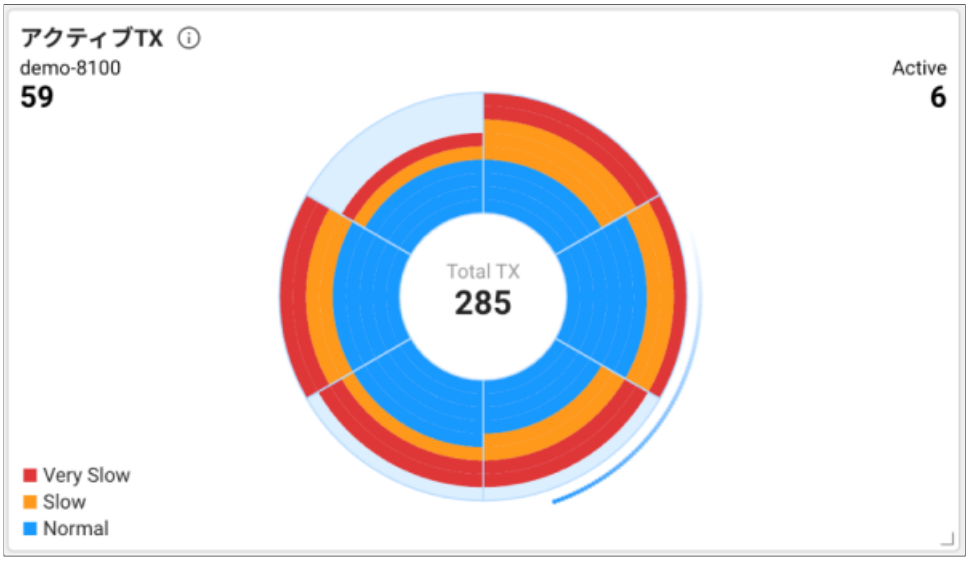
- 右領域

  アプリケーションの性能指標(Application Performance Index, Apdex)を意味します。 ウェブアプリケーションの顧客満足度を測定した指標であり、0と1の間の数です。 水滴が飛んでいく速度はApdexのトランザクションの合計数に比例し、水滴の色の意味は次のとおりです。

  - 青色:Apdex満足(Satisfied)数
  - オレンジ色:Apdex許容(Tolerating)数
  - 赤色:Apdex不満(Frustrated)数= `APDEXトランザクション総数` - (`Apdex満足数` + `Apdex許容数`)

  > ⓘ アプリケーションの性能指標(Application Performance Index、Apdex)についての詳細は次のリンクを参照してください。

アクティブTX

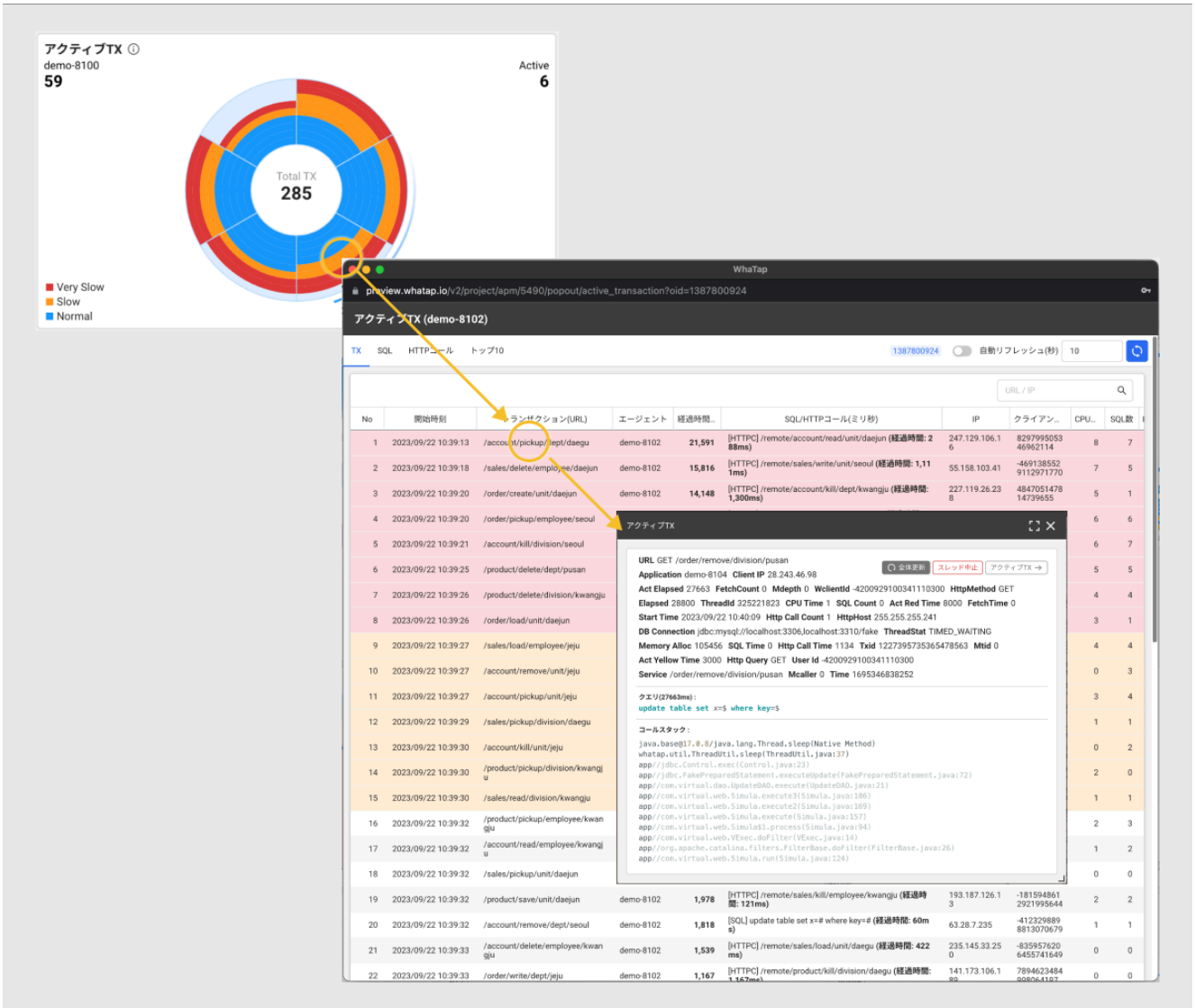


アクティブTXウィジェットでは、進行中のトランザクションを**アークイコライザー**チャートとして表示します。 遅延の発生状況をエージェントごとに確認できます。 エージェントの数程にアークが分割されます。 5秒ごとに現在サーバーで処理中のリクエストの数を表示するため、そのリクエストがそれぞれどのくらいの時間で処理されているのかが把握できます。 5秒間隔で検知されたリクエストは、遅延状況が直感的に把握できるよう、次のように色で分けられます。

- 赤色(Very Slow)：8秒以上
- オレンジ色(Slow)：3秒~8秒
- 青色(Normal)： 0秒~3秒

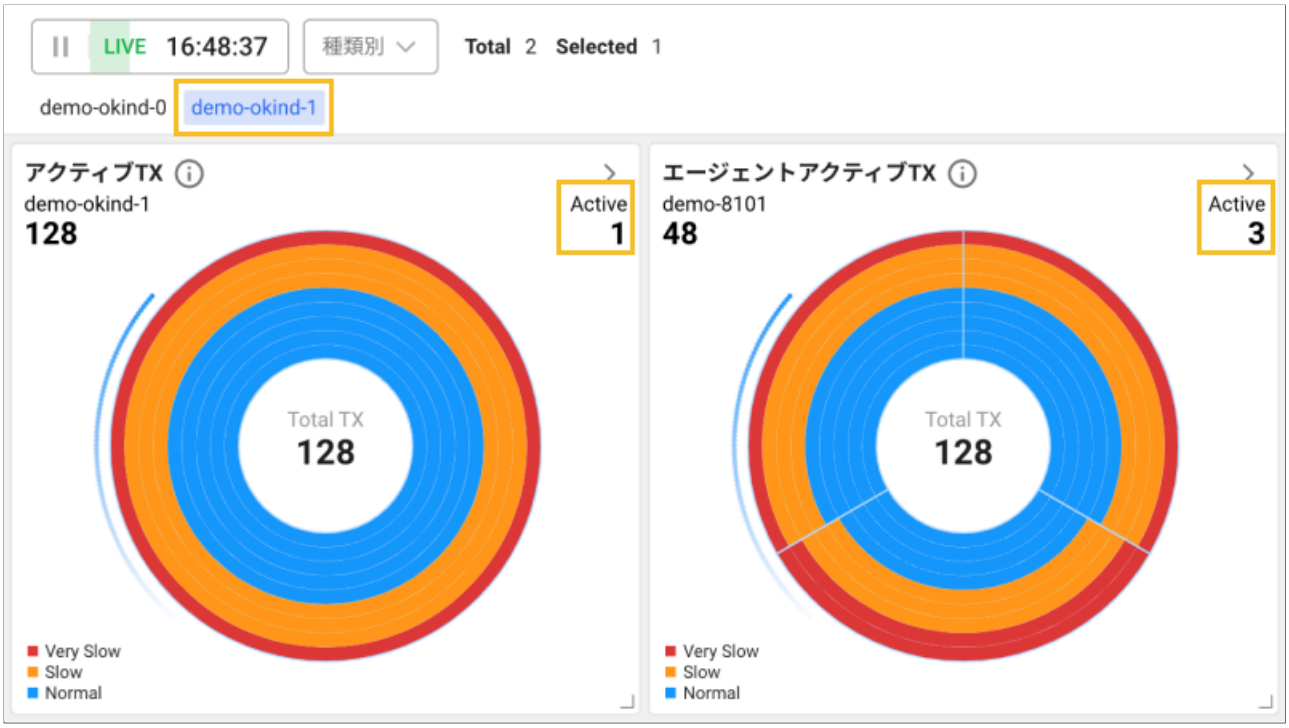
青色が多い状況は問題ありません。 これらの一部がオレンジ色や赤色に変わっていくかを監視する必要があります。 トラブルが発生すると、進行中のトランザクション数が増加し、赤色の割合が高くなります。 一方で、レスポンスが非常に速いシステムの場合、処理されるトランザクション数(TPS)が多くても、進行中のトランザクション数は少ないことがあります。

- 左上には、進行中のトランザクションが最も多いエージェント名を表示します。
- 真ん中の数字は、エージェント全体のアクティブTXの合計数です。
- アクティブTXを囲む2本のバーは、処理量(TPS)に基づいて3段階の速度でグラフの周りを回転します。
- 分割されたアーク領域を選択すると、該当エージェントが監視するアプリケーションから進行中のトランザクション一覧がポップアップ画面に表示されます。 詳細なトランザクション情報を確認するには、トランザクション一覧から任意の項目を1つ選択します。



- トランザクション一覧で選択した項目が既に終了しているトランザクションの場合は、**トレース表示**ボタンを選択します。**トランザクション情報**画面が表示されます。**トランザクション情報**画面についての詳細は、次の文書を参照してください。
- トランザクションの詳細については、次の文書を参照してください。

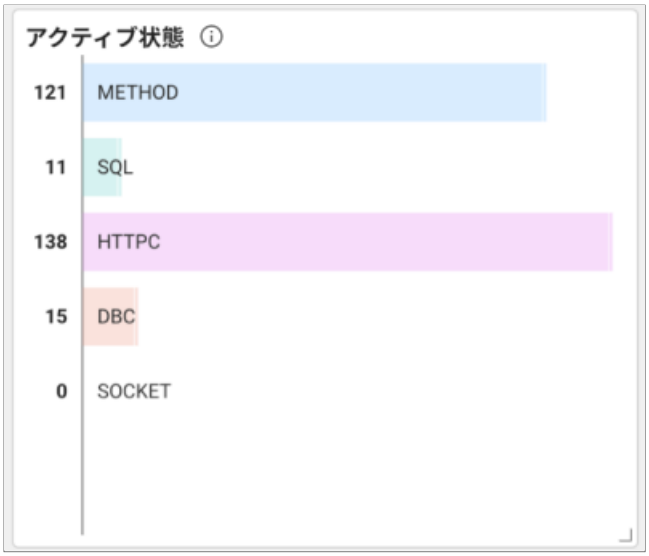
エージェントアクティブTX



アクティブTXウィジェットは、エージェント選択オプションで選択した基準に従って、マージされたデータをチャートに表示します。 一方、**エージェントアクティブTX**ウィジェットでは選択したエージェント分類の基準に含まれたエージェントの状況を確認できます。

ダッシュボードでマウスの右ボタンをクリックし、ウィジェット一覧で**エージェントアクティブTX**ウィジェットを選択します。

アクティブ状態



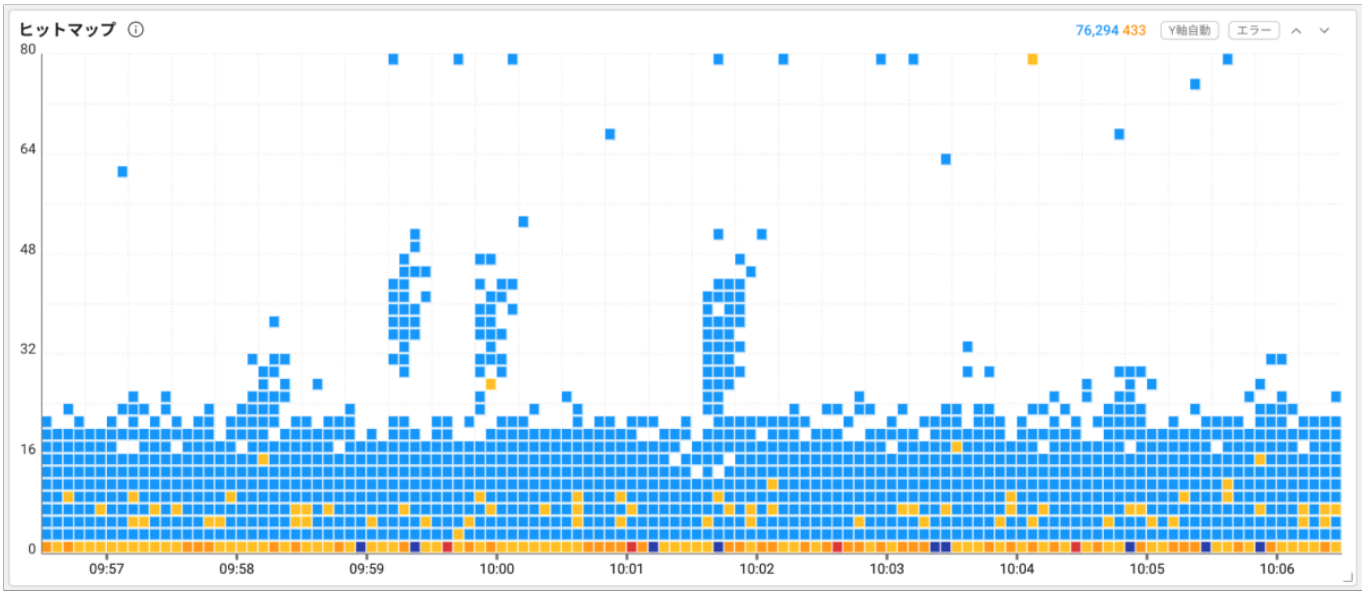
プロジェクト内のすべてのエージェントは、5 秒ごとにアクティブTX数を収集すると同時に、トランザクションのステータスを収集します。 トランザクションのステータスは**METHOD**、**SQL**、**HTTPC**、**DBC**、**SOCKET**に分けられます。

- **METHOD**：メソッドを実行している状態です。 一般的な状況として青色で表示されます。
- **SQL**：SQLを実行している状態です。 **SQL**数の増加が見られた場合は、データベースサーバーとの接続を確認してください。
- **HTTPC**：外部APIを呼び出し中の状態です。 **HTTPC** 数が増加した場合は、外部と接続されたサーバーの応答が正常ではない状況として解釈できます。
- **DBC**：トランザクションがConnection Poolから新しいConnectionを獲得(get)しようと待機している状態です。 DB Connection Poolの数が不足すると、新しい接続リクエストが発生するたびに遅延が発生し、性能トラブルの原因となります。 この場合、**DBC**の数値が増加します。

  > ⓘ DB接続遅延に関する詳細については次の文書を参照してください。

- **SOCKET**：外部システムとのTCP接続試行を意味します。 **SOCKET** 数値が持続的に増加する場合は、外部システムとの接続ができないため障害が発生している可能性があります。

ヒットマップ



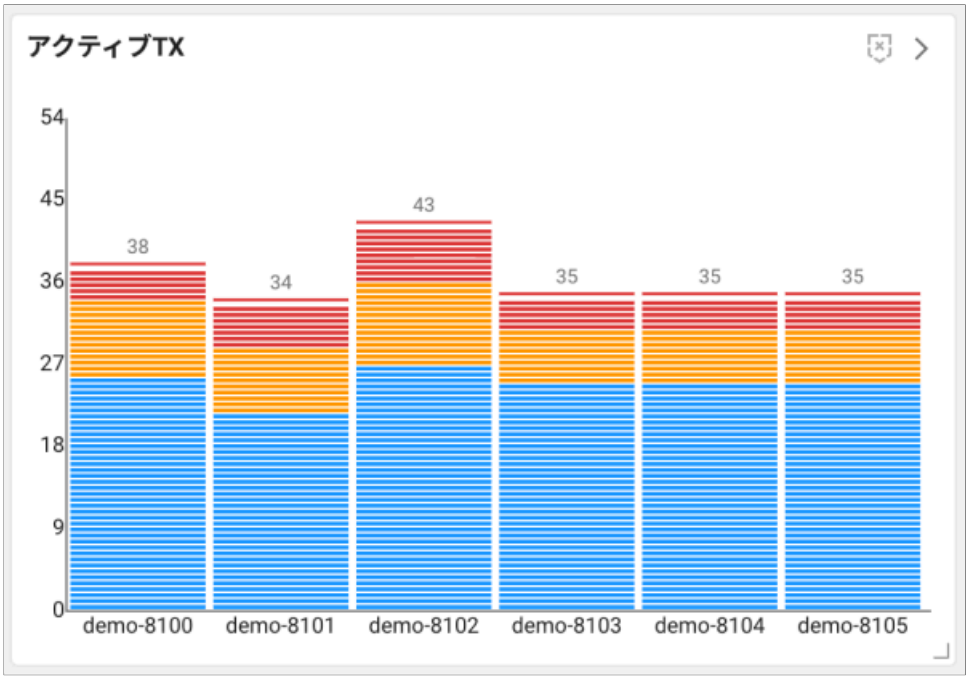
ヒットマップチャートは、終了したトランザクションの応答時間を分布図で表現します。 横軸はトランザクションの終了時間です。 縦軸は実行時間です。 応答時間が伸びたトランザクションは、分布図の上段に位置します。 **ヒットマップ**チャートの中の領域をドラッグすると、詳細なトランザクション情報が確認できる**トレース分析**画面が表示されます。 **トレース分析**の詳細は、次の文書を参照してください。

- 通常のトランザクションは水色→青色→紺色の順に表現されます。
- エラートランザクションは黄色→オレンジ色→赤色の順に表現されます。
- 画面の右上の数字はチャート内のトランザクション数/エラー数を意味します。
- **Y軸自動**ボタンを選択すると、実行時間を示すY軸を自動調整します。
- **エラー**ボタンを選択すると、エラートランザクションのみ表示します。
- ∧または∨ボタンを選択すると、5秒~80秒でチャートを拡大／縮小することができます。

- エラーが発生せず、実行時間が500ms以下のトランザクションの詳細は、URLあたり5分に1回のみ収集されます。
- TPSや平均応答時間などの統計情報には影響しません。
- エージェント設定により、ポリシーを設定できます。 詳細については、次の文書を参照してください。
- ヒットマップウィジェットを使用してトランザクションを分析するプロセスについては、次の文書を参照

してください。

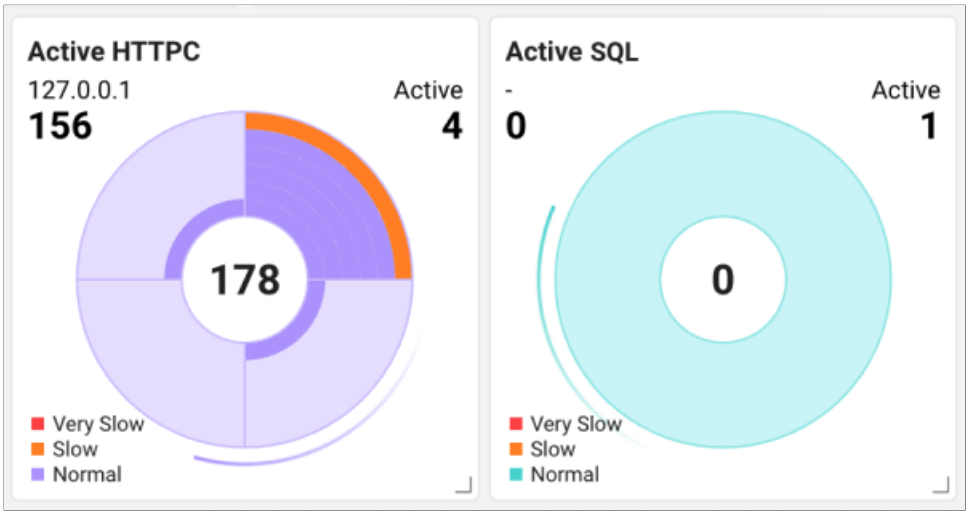
アクティブTXイコライザー



エージェント全体の一覧と進行中のトランザクションの合計数を表示します。 チャートの各グラフにマウスをオーバーすると、トランザクションのステータス速度を確認できます。 トランザクション状態の速度は、Normal、Slow、Very Slowに分類します。

- 非活性化されたエージェントを表示または非表示にするには、ボタンを選択します。
- ＞ボタンを選択すると**アクティブTX**メニューに移動します。

アクティブ HTTPC/SQL



アクティブ HTTPC/SQL ウィジェットは、アプリケーションで発生する外部呼び出し（HTTP 呼び出しおよびデータベースクエリ）のパフォーマンスをリアルタイムでモニタリングします。
すべてのトランザクションを表示する **アクティブTX** ウィジェットとは異なり、外部呼び出しに特化しているため、外部システムのボトルネックを即座に把握できます。2つのウィジェットを一緒に配置することで、トランザクションの増加が外部システムの遅延によるものか、内部ロジックの問題によるものかを区別でき、より効果的な多層分析が可能になります。

進行中の HTTPC/SQL を **アークイコライザー** チャートで可視化し、遅延発生状況をホスト/データベース URL ごとに確認できます。ホスト/データベース URL の数に応じてアークが分割され、5 秒ごとに現在サーバーで処理中のリクエスト数と処理時間が表示されます。5 秒間隔で検知されたリクエストは危険度に応じて直感的な色で分類されます。

- 赤 (Very Slow): 8 秒以上
- オレンジ (Slow): 3 秒〜8 秒
- Normal: 0 秒〜3 秒
  - 紫 (HTTPC Normal)
  - ミント (SQL Normal)

Normal が多く表示される状況は問題ではありません。一部がオレンジや赤に変化するかどうかを観察する必要があります。障害時には進行中トランザクション数が増加し、赤の割合が高まります。一方、応答が非常に速いシステムでは TPS が高くても進行中のトランザクション数は少ない場合があります。

このウィジェットは特にマイクロサービス環境でのサービス間 HTTP 通信遅延や、複数のデータベースを使用する環境で DB パフォーマンスを個別に追跡するのに有用です。外部 API 呼び出しが多いサービスでは、API 提供者ごとの応答時間をモニタリングでき、依存関係管理に効果的です。

アークイコライザーチャート

ウィジェットの中核的な可視化要素で、進行中の HTTPC/SQL をホスト/データベース URL ごとに分割して表示します。全体トランザクションで問題を発見した場合、このチャートを通じて具体的な外部呼び出しのボトルネックを確認できます。

- **左上表示**: 進行中トランザクションが最も多いホスト/データベース URL を表示し、主要対象を容易に特定
- **中央数値**: アクティブ HTTPC/SQL の合計数を表示
- **TPS ベースアニメーション**: イコライザーの外周バーがスループット (TPS) に応じて 3 段階の速度で回転し、システム負荷を視覚的に表現
- **5 秒単位更新**: リアルタイムに近いモニタリングで即時対応可能
- **アーク分割**: ホスト/データベース URL ごとにアークが分割され、どの外部システムで遅延が発生しているかを一目で把握

ホスト/データベース URL 詳細照会

分割されたアーク領域を選択すると、該当ホスト/データベース URL で進行中の **トランザクション Top 100 リスト** ポップアップが表示されます。

- **トランザクションリスト**: 選択したホスト/データベース URL の進行中トランザクション Top 100 を表示
- **トレンドチャート**: 過去 10 分間の TPS/応答時間/エラー件数/エラー率をチャートで表示
- **トランザクション詳細**: リストからトランザクションを選択すると詳細情報を確認可能
  - すでに終了したトランザクションの場合、**トランザクション表示** ボタンを選択して **トランザクション情報** ウィンドウに移動

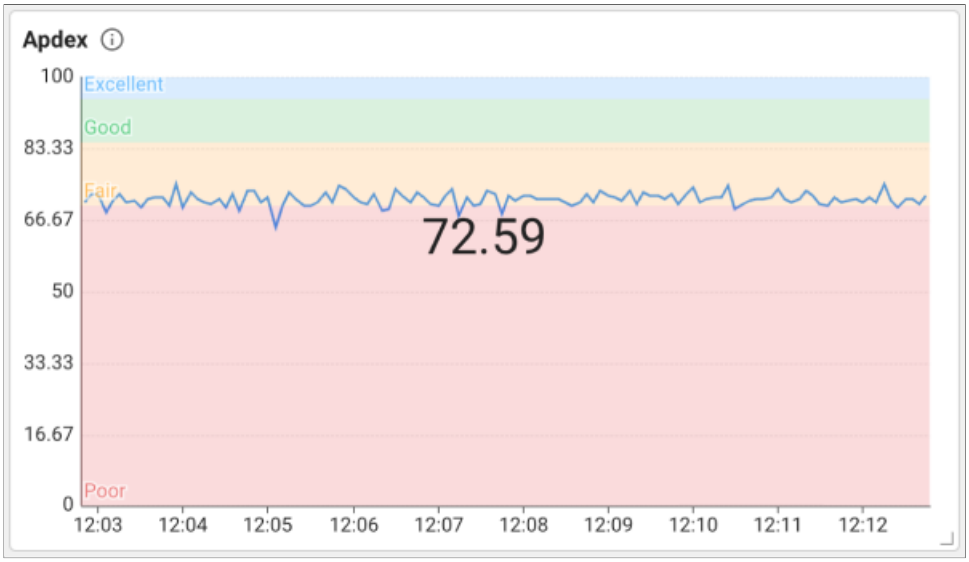
1. **アクティブ HTTPC Top 100** ポップアップでトランザクションリストとトレンドを確認します。
2. 詳細分析が必要なトランザクションを選択します。

   a. アークイコライザーで分析するホストまたはデータベース URL のアークをクリックします。
3. トランザクション情報ウィンドウで詳細内容を確認します。

サービス関連ウィジェット

サービス関連ウィジェットは、アプリケーションのパフォーマンスチューニングの指標として使用できます。

Apdex



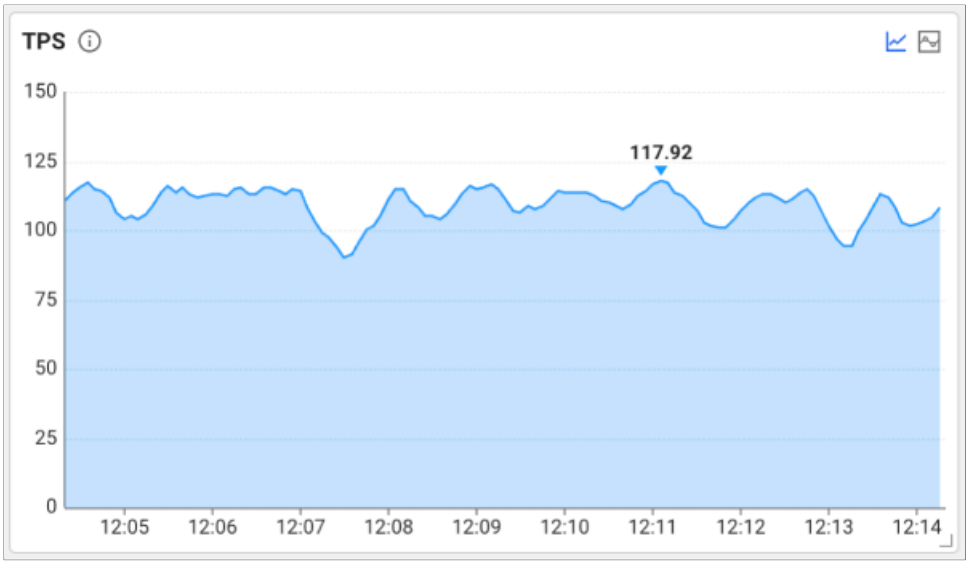
Application Performance Index(Apdex)は、アプリケーションのパフォーマンス指標です。ウェブアプリケーションの顧客満足度を測定します。ユーザー満足度の指標として活用でき、0~1の間の値で表示します。

(満足回数+ (許容回数*0.5))／全体リクエスト数

- 満足(Satisfied, S)：タスク処理に問題なし≤ 1.2秒(満足S既定値)
- 許容 (Tolerating, T)：ユーザーが遅延を感じるが、タスク処理は可能≤4.8秒(満足S*4)
- 不満 (Frustrated, F)：タスク処理の不可能 > 4.8秒 (許容T超過及びエラー)

> ⓘ 満足Sの既定値は**管理** > **エージェント設定**メニューで変更できます。 設定キーの値は `apdex_time` です。 エージェントバージョン2.0以降に対応しています。

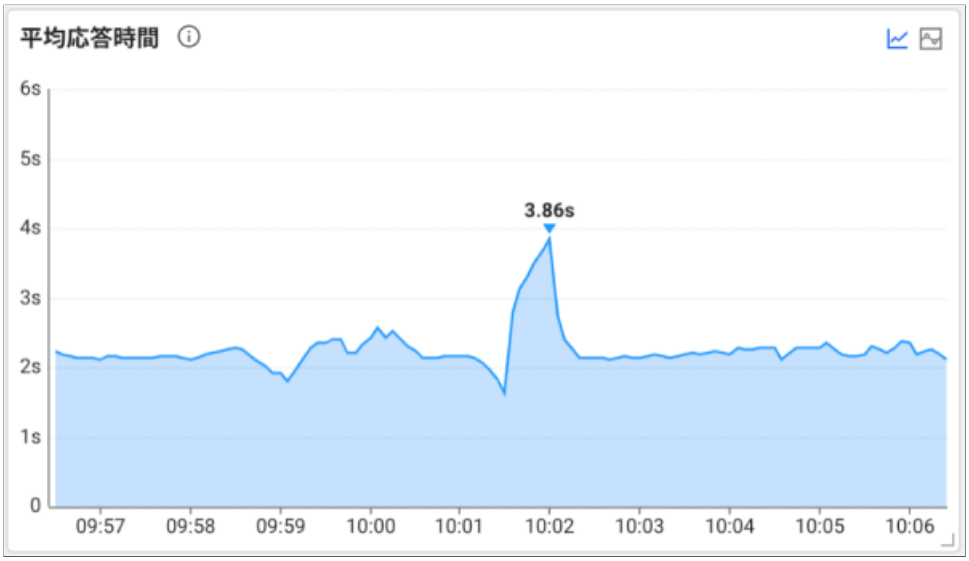
TPS



Tansaction Per Second(**TPS**)は毎秒処理されるトランザクション数を意味し、サービスパフォーマンス指標の基準となります。 5秒間に処理されたトランザクション数を1秒あたりの数として換算してチャートで表現します。 直近10分間のTPSを表示します。

> TPS、応答時間、CPU 使用率分析の詳細については次の文書を参照してください。

平均応答時間



アプリケーションサーバーがユーザーにリクエスト結果を返すのに掛かる時間をリアルタイムでモニタリングします。**平均応答時間**は、チューニング指標として意味があります。 WhaTapのモニタリングサービスは、5秒間隔でトランザクションの平均応答時間を計算します。

> TPS、応答時間、CPU 使用率分析の詳細については次の文書を参照してください。

リソース関連ウィジェット

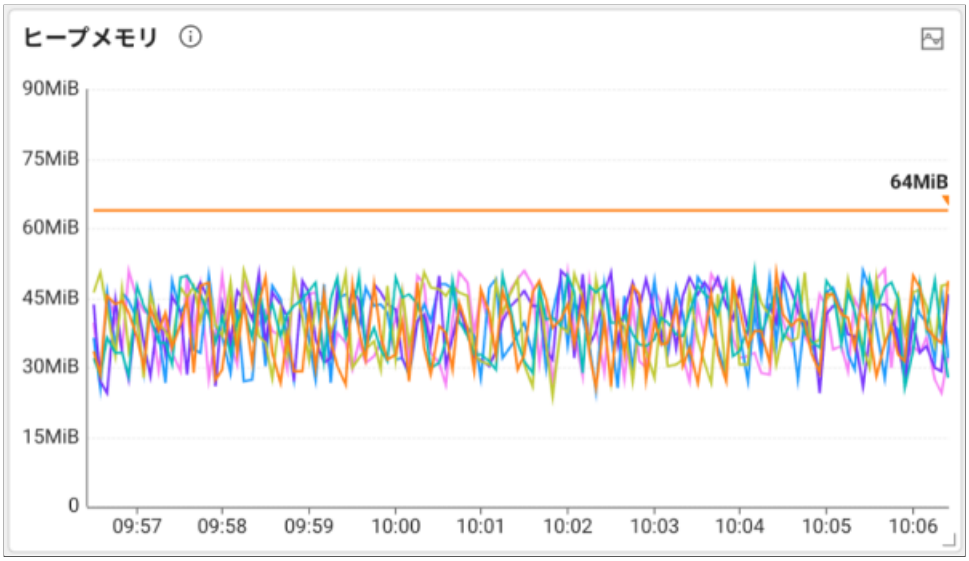
システムCPU



アプリケーションサーバーのCPU使用量です。 リアルタイムでCPU使用量の変化の推移を把握できます。

> TPS、応答時間、CPU 使用率分析の詳細については次の文書を参照してください。

ヒープメモリ



サーバー毎に使用できる最大メモリと現在のメモリを表示して、負荷がかっかているサーバーが容易に把握できます。時間の経過に伴うメモリ使用量の変化をリアルタイムで確認できます。

メモリ折れ線チャートは通常、波線を続けます。 アプリケーションサーバーがリクエストを処理するためにメモリを使用するとメモリ折れ線チャートは増加します。 GC(Garbage Collection)によってメモリをクリーンアップすると減少します。 **ヒープメモリ**指標は平均値を使って表記します。

> ヒープメモリチャート分析に関しての詳細については、次のリンクを参照してください。
>
> - Javaヒープメモリーチャートの分析：Ch.1ヒープチャート観察
> - JAVAヒープメモリーチャートの分析：Ch.2 メモリリークとヒープダンプ分析

ユーザー関連ウィジェット

同時接続ユーザー



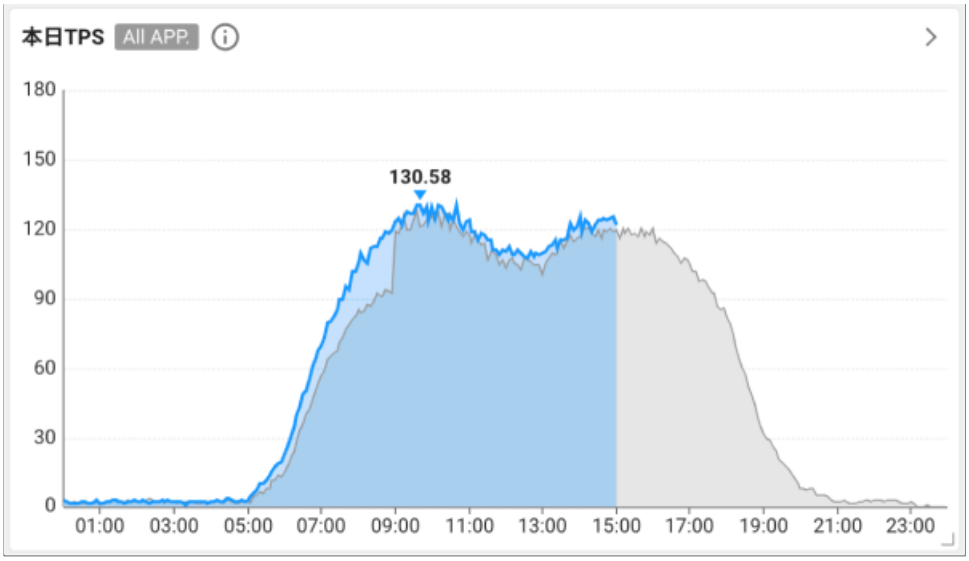
リアルタイムブラウザのユーザー数を表示します。 10秒ごとに、直近5分以内にトランザクションを発生させたユーザーを重複なしでカウントして表示します。 ユーザーブラウザのIP情報でカウントします。

- エージェント設定でユーザーを識別するためにIPまたはクッキーを選択できます。 詳しい内容は、次の文書を参照してください。
- 通常、接続ユーザーはネットワークに接続しているユーザーを意味します。 ただし、ウェブシステムは未接続のネットワークを使用するため異なります。 接続されているという意味ではなく、最近のリクエストを送信したユーザーが測定の対象になる可能性があります。
- リアルタイム(同時接続)ユーザーとは、同じ時間帯にシステムを使用しているユーザーを意味します。 リアルタイム(同時接続)ユーザーを計算または測定する方式は多数あります。
- ユーザー数は、単に合計されるのではなく、** HyperLogLog**によって計算されます。 **HyperLogLog**は、非常に少ないメモリ内の要素の数を推定する確率的なデータ構造を意味します。
- ユーザー数集計のためのエージェントオプションの設定方法は、次の文書を参照してください。

1日基準比較ウィジェット

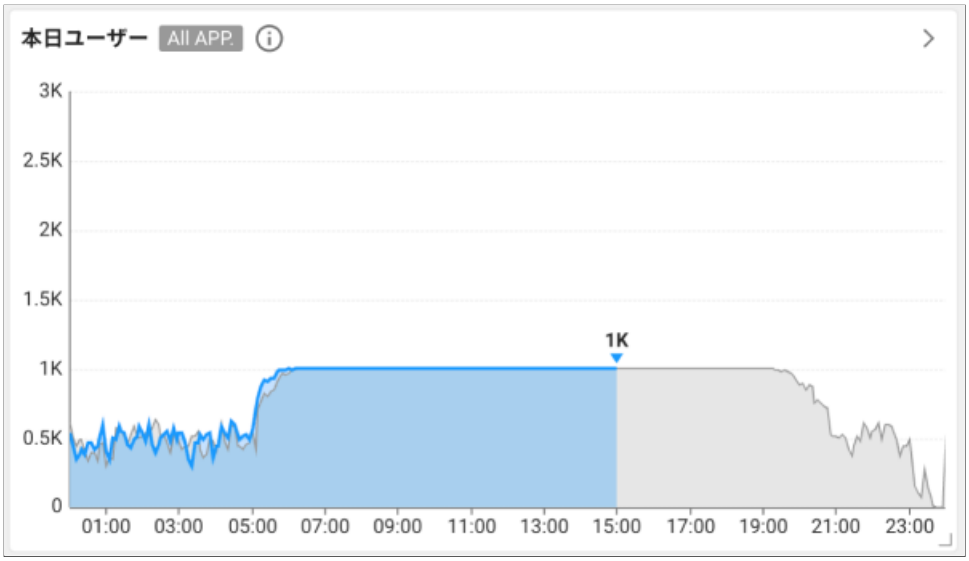
> All APPS. タグ付きウィジェットは、常にすべてのエージェントのデータを表示します。

本日TPS



1日のTPSを時間帯別に表示します。 昨日の同時間帯のTPS(グレー)と比較して確認できます。 > ボタンをクリックすると、**統計/レポート** > **日別APP現況**メニューに移動します。

本日ユーザー



一日あたりの**同時接続ユーザー**の数を時間帯別に表した指標です。 昨日の同時ユーザー数(グレー)と比較して確認できます。 > ボタンをクリックすると、**統計/レポート** > **日別APP現況**メニューに移動します。

リアルタイムアラート

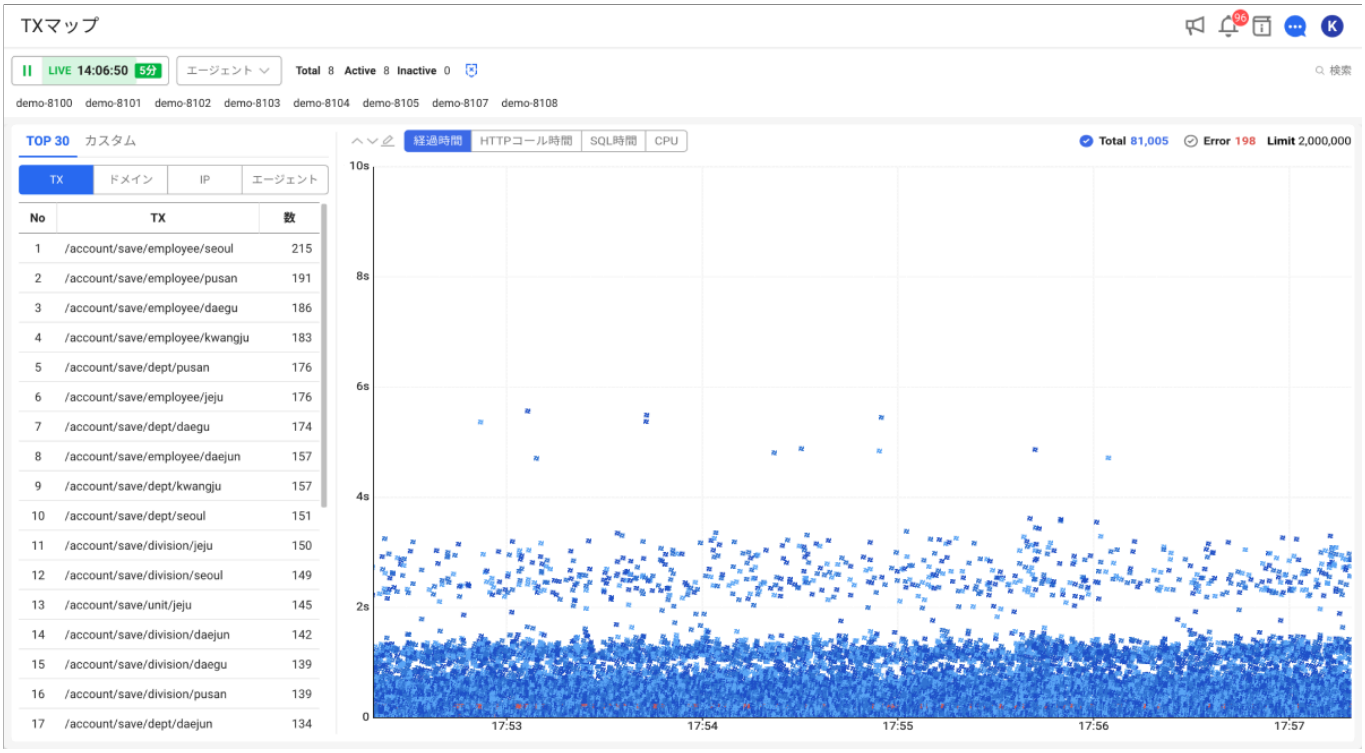
画面の右上に**リアルタイムアラート**ボタンをクリックすると、最近発生したイベントが確認できます。 トグルメニューをクリックして、ブラウザ通知をオン、オフできます。

> 画面上段の固定メニュー領域にある基本項目については、次の文書を参照してください。

TXマップ

ホーム画面 > プロジェクト選択 > **ダッシュボード** > **TXマップ**

TXマップチャートは、終了した個別トランザクションのリスポンス時間を分布図の形で表現したチャートです。 ヒットマップと同様に分布パターンによる問題点を発見し、分析可能です。 **ヒットマップ**は5秒単位でトランザクションをグループ化して表示しますが、**TXマップ**は、トランザクションを個別に表示します。



ヒットマップと違う点は？

ヒットマップは、合算されたデータを使用して、長い間の全体的な推移を確認できます。 一方、TXマップは、個別のトランザクション情報を活用して、比較的短い時間の範囲内でトランザクション分布を実行時間の制限なく確認できます。 また、様々なフィルターを適用し、希望するトランザクションだけを選択して推移を確認できるという点が異なります。

- 長期間・大容量のトランザクション分布図を分析するには、ヒットマップ(**分析** > **ヒットマップ**)をご利用ください。

- パフォーマンステストのトラブル状況で1~5分以内にトランザクションの分布図の分析を行い、実行時間の制限なしでトランザクションの分布図を確認するには、**TXマップ**を確認してください。

- **ヒットマップ**メニューの詳細については、次の文書を参照してください。
- **ヒットマップTX**チャートの分析方法の詳細については、次の文書を参考にしてください。
- ユーザーのPC時間がサーバー時間より過去に設定されていたり、両方の時間差が5分以上の場合、チャートにデータが出力されないことがあります。

トレース分析

TXマップのチャート領域をドラッグすると、詳細なトランザクション情報が確認できる**トレース分析**ウィンドウが表示されます。 トランザクション一覧と各トランザクションの下位ステップの情報を一度に確認できます。

チャート領域をドラッグした後**トレース分析**ウィンドウで照会できるデータは、最大1,000件です。

トレース分析の詳細は、次の文書を参照してください。

照会基準の変更

TXマップのチャート領域、左上のボタンを選択してデータ照会基準を変更することができます。

- **経過時間**: トランザクションが実行された時間を基準にデータを表示
- **HTTPコール時間**: HTTP呼び出しに対する応答時間を基準にデータを表示
- **SQL時間**: SQL実行時間を基準にデータを表示
- **CPU使用時間**: CPU使用時間を基準にデータを表示

参照結果フィルタリング

TXマップのチャート領域、右上ではTotal、Error、Readフレーズを選択して結果を区分できます。

Total: 全体トランザクション表示

Error: エラーが発生したトランザクションのみ表示

Read: サーバーで読み込んだトランザクションの数を表示

例えば、エラーを選択した後、チャートのトランザクションをドラッグすると、エラー状態のトランザクションリストが表示されている**トレース分析**画面が開きます。 もう一度全体のトランザクションを照会するにはTotalをクリックします。

Y軸の調整

- チャートの左上の ∧ または ∨ ボタンを選択すると、Y軸の最大値を簡単に変更できます。 また、キーボードの上または下方向のボタンを押して、同じ機能を利用できます。

  Y軸の区間によって ∧ または ∨ ボタン、キーボードボタンを利用する際の動作方法については、以下を参照してください。

  - 10秒から50秒の区間は10秒間隔に変更
  - 50秒から300秒は50秒間隔に変更
  - 300秒以上は100秒間隔に変更
- Y軸の最大値を直接入力して変更することができます。 ✎ ボタンを選択し、希望する値を入力して**適用**ボタンを選択します。

TXマップのチャートに表示されるトランザクションの最大数(**Limit**)は、2,000,000個です。

フィルターを適用して照会する時は、Total(画面に表示される個数)よりReadの値が大きい場合があります。

過去のデータ参照

TXマップは、リアルタイムモニタリングを基本的に提供します。 既定値として5分間のデータをリアルタイムで照会で

きます。 タイムセレクターの緑のボタンをクリックして、希望の照会時間を選択します。

過去のデータを照会するには、タイムセレクターで一時停止(⏸)ボタンをクリックします。 期間照会モードに変更され、最大1時間までのトランザクションデータを照会できます。 ユーザーが設定した日付と時刻を基準に**TXマップ**のチャートデータを更新します。

過去のデータを照会している間はいつでも照会を停止できます。 _ フィルターを変更したい場合は、まず照会を停止してからフィルターを変更し、再度照会します。 _ フィルターを変えずにもう一度照会ボタンを押すと、停止した部分から照会を続けます。

過去のデータを照会してからリアルタイムモードに切り替えるには、▶ ボタンをクリックします。

> ⓘ • 過去の照会範囲は、トランザクションの量によって異なる場合があります。

フィルター適用

過去のデータを照会する場合、フィルターを適用して希望するデータだけを照会できます。

1. タイムセレクターで一時停止(⏸)ボタンをクリックすると、フィルター入力ボックスが表示されます。
2. フィルター入力ボックスをクリックすると、以下のように選択できるフィルター項目の一覧が表示されます。
   - 一般フィルター

項目	説明
ログインID	ユーザーのログイン識別子(ID)で照会
WClientID	クライアント別の固有識別子で照会 (注意: 完全一致で照会)
TXID	個別トランザクションの固有識別子 (注意: 完全一致で照会)
リファラー	トランザクションリクエスト時に渡されたReferer(参照ページアドレス)で照会

項目	説明
マルチTXID	複数のトランザクションを束ねる固有識別子 (注意: 完全一致で照会)
カスタムTID	ユーザーが直接定義したトランザクション識別子(Custom Transaction ID)で照会
エラー(クラス、段階、メッセージ)	トランザクションで発生したエラーの種類、段階、メッセージで照会(部分一致を含む)
都市	トランザクションが発生したユーザーの都市情報で照会
国/地域	トランザクションが発生したユーザーの国家情報で照会
HTTPメソッド	HTTPリクエスト方式(GET、POSTなど) (注意: 完全一致で照会)
ユーザーエージェント	トランザクションリクエストに使用されたブラウザまたはクライアント情報(User Agent)で照会

- ステップ(STEP)フィルター
  STEPとは、トランザクションを実行するときに収集されるプロファイルデータを意味します。 ヒットマップのドラッグ時、トランザクションLISTでトランザクションをクリックして見るTable ViewやTree Viewに表示される順次データをSTEPといいます。

```
| 項目    | 説明                                                
```

```
| SQL     | 実行されたクエリ情報が保存されているSTEP
|
| SQL_BIND | 実際には存在せず、SQLステップに含まれるデータ。SQL呼び出し時にバインド変数と置換変数を照会できるSTEP (注意: 敏感な情報照会権限と復号化キーがなければ照会できない) |
| ALL     | 上記のすべてのSTEP情報を照会するステップ (注意: 敏感な情報照会権限と復号化キーがなければ照会できない)                                    |
```

- 数値フィルター

項目	説明
経過時間	トランザクションが実行されるのにかかった全体時間
HTTPコール時間	HTTPリクエストに対する応答時間
HTTPコール件数	トランザクション中に発生したHTTPリクエストの回数
SQL時間	トランザクションで実行されたSQLクエリの総実行時間
SQL件数	トランザクションで実行されたSQLクエリの数
状態	トランザクションの状態(例: 正常、エラーなど)

3. フィルタリング項目に希望する値を入力します。 フィルタリングには、フィルターキー、演算子、値のすべてを入力する必要があります。 入力が完了していないフィルターが残っている場合、検索はできません。
4. フィルター入力完了後 ボタンをクリックします。

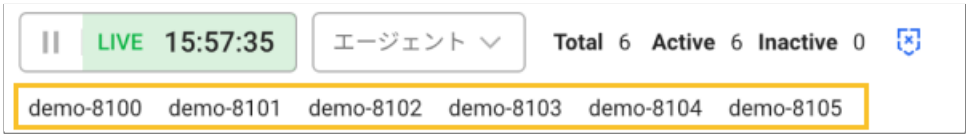
Top 30一覧と**TXマップ**チャートにフィルタリングしたデータを更新します。

- 複数の条件を設定してフィルタリングすることができます。 追加した条件は、`&&` を基準として適用されます。 例えば、トランザクションが `/account/save/employee/seoul` でありながら経過時間が3秒以上の場合は、次のように適用できます。

  `TX` `=` `/account/save/employee/seoul` `経過時間 (ms)` `>=` `3`

エージェント確認

エージェント接続状態を確認する



画面の左上のタイムピッカーの右側には、プロジェクトに関連したエージェントの状態を確認できる情報が表示されます。 この機能で、モニタリング対象サーバーの動作有無がすぐに確認できます。

- Total: プロジェクトに関連付けられているすべてのエージェントの数
- Active: 活性化されたエージェントの数
- Inactive: 非活性化状態のエージェント数
- : 非活性化されたエージェントを表示または非表示します。

エージェント別モニタリング

基本的に、チャートにはすべてのエージェントから収集した指標を表示しますが、エージェントごとのデータ照会もできます。 タイムピッカーの下にあるエージェントを1つ以上選択します。 選択したエージェントのトランザクションデータをチャートに更新します。

> エージェントを1つ以上選択した状態で再度すべてのエージェントを選択するには、選択を解除するか、Totalをクリックします。

> プロジェクトに接続されたエージェントの数が多い場合は、エージェント名を短く設定することをお勧めします。 エージェント名の設定に関する詳細は、次の文書を参照してください。

分類別エージェントモニタリング

エージェント設定で分類した単位で複数のエージェントを簡単にモニタリングできます。

- **エージェント**: 個別エージェント
- **種類別**: エージェントの種類(例: ウェブサーバー、データベースなど)に分類されたグループ、`whatap.okind`に設定
- **サーバー別**: サーバ(ノード)別に分類されたグループ、`whatap.onode`に設定
- **種類別エージェント**: `whatap.okind`に分類されたエージェントの一覧
- **サーバー別エージェント**: `whatap.onode`に分類されたエージェントの一覧

> ⓘ エージェント設定の詳細については次の文書を参照してください。

エージェントを検索する

プロジェクトに接続されたエージェントの一覧が多く、見つけるのが難しい場合は、エージェント検索機能を利用します。 画面の左上にある🔍**検索**ボタンをクリックすると、文字列を入力するボックスが表示されます。

入力した文字列に一致するエージェントのみが、エージェント一覧に表示されます。 検索したエージェントに基づいてダッシュボードのデータをフィルタリングするには、文字入力ボックスの右側にある**選択する**ボタンをクリックします。

選択したエージェント項目を初期化するには、文字入力ボックスの右側の**閉じる**ボタンをクリックします。

上位一覧を確認する

Top 30セクションでは、**TX**および**ドメイン**、**IP**、**エージェント**を基準としてトランザクションの実行数の多い上位30のリストを表示します。

- **TX**：トランザクションURLに基づいて集計された上位30個のリスト
- **ドメイン**：クライアントが接続したIPアドレスに指定されたインターネットアドレスに基づいて集計された上位30のリスト
- **IP**：クライアントのIPアドレスに基づいて集計された上位30のリスト
- **エージェント**：当該プロジェクトに含まれるエージェントに基づいて集計された上位30のリスト

リストで個別項目を選択するとTXマップチャートに選択した項目に基づいてデータを反映します。 複数の項目を複数選択するには、`Ctrl` (Windows/Linux) または `CMD` (Mac) キーを押したまま、目的の項目をクリックします。

Top 30セクションのリストで、キーボードの上または下方向のボタンを押して項目を移動できます。

ユーザーIPアドレスを追加

TOP 30セクションの**IP**基準でトランザクションの実行件数を照会する時、特定のIPアドレスを追加して常時確認できる機能を提供します。

1. **TOP 30**セクションで**IP**を選択します。
2. 一覧の一番上の入力欄に特定のIPアドレスを入力します。
3. ＋ボタンをクリックします。



一覧の一番上に入力したIPアドレスが固定されて表示されます。

追加したIPアドレスを削除するには、ボタンをクリックします。

ユーザーカスタム項目のモニタリング

特定のトランザクション、ドメイン、IPを**カスタム**タブに追加して、応答時間の分布図チャートで選択した項目だけを確認できる機能を提供します。 これにより、ユーザーは問題が発生する特定のアイテムを効率的にモニタリングして分析できます。

1. **カスタム**タブをクリックします。

2. **TX**または**ドメイン**、**IP**項目は直接入力、**Top 30**一覧内の選択、項目検索が可能です。
3. 追加した項目をクリックします。



選択した項目の応答時間の分布図が、右のチャートにリアルタイムでフィルタリングされることを確認できます。

> ⓘ 追加したカスタム項目を削除するには、🗑ボタンをクリックします。

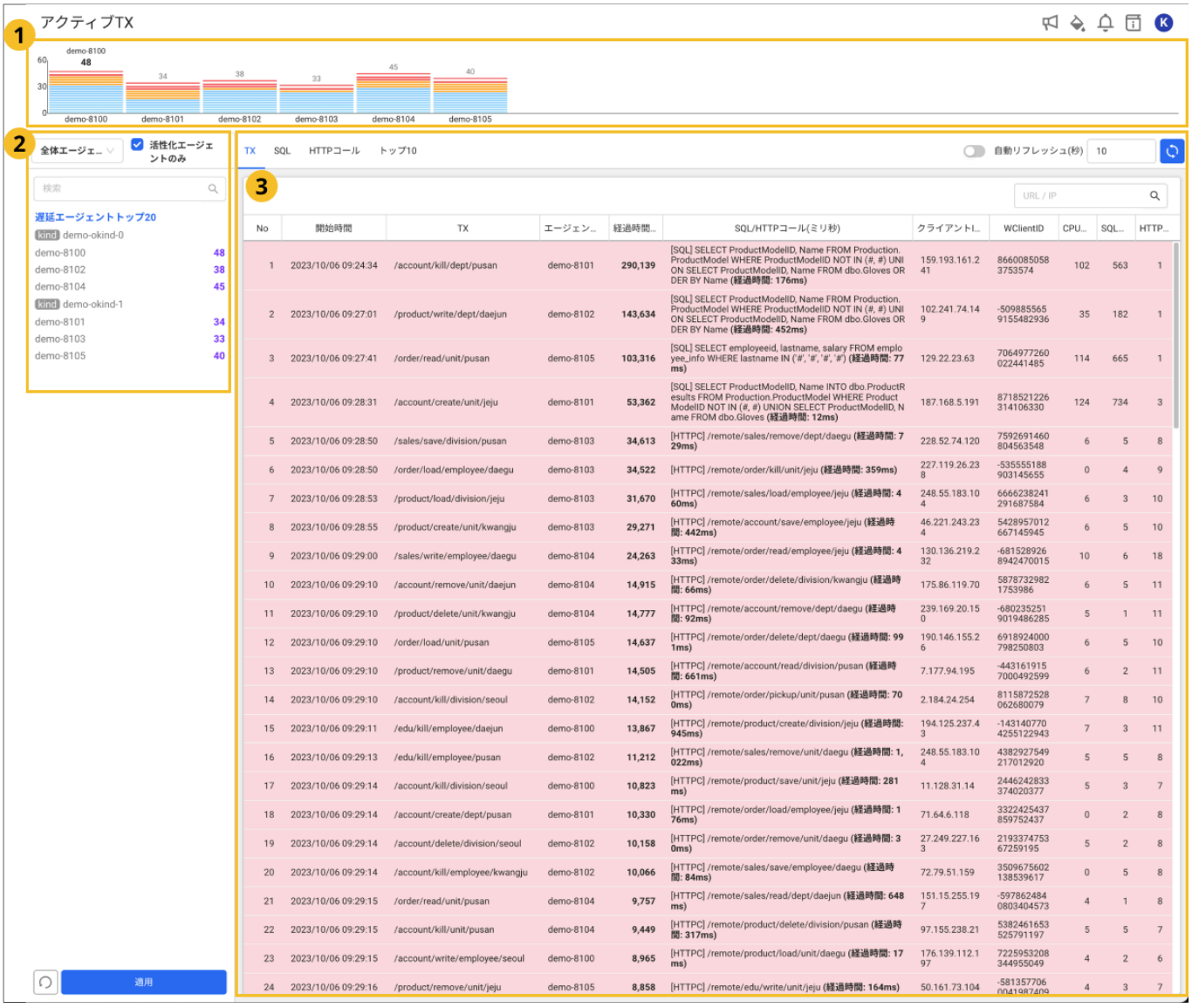
アクティブTXダッシュボード

ホーム画面 > プロジェクト選択 > **ダッシュボード** > **アクティブTX**

実行中のトランザクションのURL、SQL、HTTP呼び出し情報から異常トランザクションをすばやく特定し、詳細なトレースを通じて遅延部分を詳細に分析できます。

> **連携プロジェクト管理**メニューを通じて、データベースプロジェクトを追加すると、現在進行中のトランザクションと連携したデータベースのSQL性能を照会できます。**連携プロジェクト管理**メニューと連携プロジェクトの分析についての詳細は、次の文書を参照してください。

画面構成



1 エージェント別のアクティブTX

すべてのアプリケーションサーバーの一覧と稼働中のトランザクションの合計数を表示します。チャート内の各グラフにカーソルを合わせると、トランザクションステータスが表示されます。トランザクションステータスの速度Normal、Slow、Very Slowに分類します。

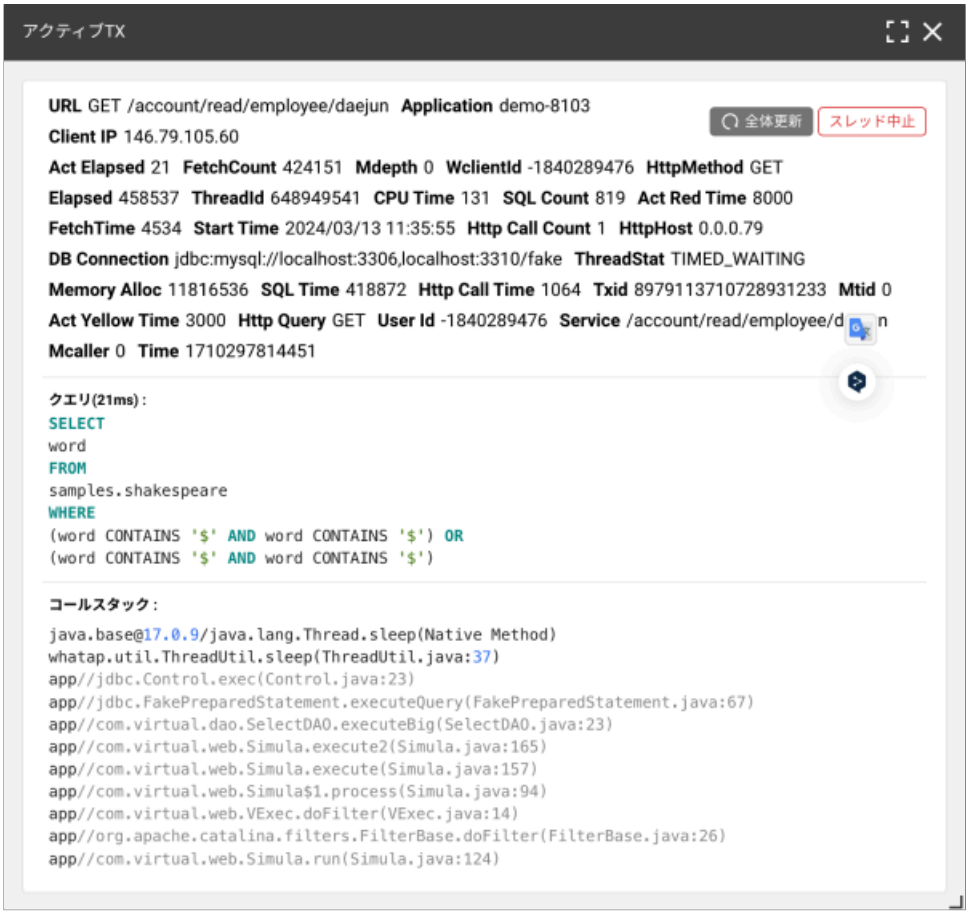
2 エージェント一覧

アプリケーションサーバーの一覧と進行中のトランザクションの合計数を表示します。該当アプリケーション サーバーを選択すると3トランザクション一覧に詳細が表示されます。

3 トランザクション一覧

進行中のトランザクションのURL、SQL、HTTP呼び出し情報の詳細を表示します。Top 10項目タブを選択すると、各項目別の経過時間が高い順にソートされている表を確認できます。同様に、トランザクションステータスの速度はNormal、Slow、Very Slowに分類します。

トランザクション一覧から項目を選択すると、選択したトランザクションに関する詳細情報が確認できる**アクティブTX**画面が表示されます。



- 実行中のトランザクションに関する詳細情報や外部呼び出し、クエリ、コールスタックなどを確認できます。
  - **全体更新**：進行中のトランザクションの情報を更新できます。

- **スレッド中止**：進行中のトランザクションを停止できます。スレッド中止のためには、**シークレットキー**が必要です。**シークレットキー**についての詳細は、次の文書を参考にしてください。
- **連携DBセッション**: 進行中のトランザクションに連携されたデータベースのSQL性能を照会できます。選択したトランザクションで連携プロジェクトへSQLを実行したトランザクションである場合にのみ表示されます。連携プロジェクトの追加と管理についての詳細は、次の文書を参照してください。

> **スレッド中止**機能は、アプリケーションで実行されているスレッドを強制的に終了する機能です。ただし、この機能は動作しない場合があります。例えば、スレッドがIOタスクを実行している場合、スレッドをすぐに中止できません。
>
> **スレッド中止**機能を使用すると、スレッドがIOタスクを実行中であっても終了を試みます。しかし、スレッドがIOタスクを完了するまでは中止されません。つまり、スレッドは中止要求を受け入れますが、作業が完了して次の作業に移るまで中止されません。
>
> スレッドを強制終了させると、ビジネスロジックに問題が発生する恐れがあり、予期しない動作を引き起こすため、お勧めしません。それでもスレッドを中止させる必要がある場合は、アプリケーション全体を中止してから再起動する必要があります。

- すでに終了したトランザクションであれば**トレース表示**ボタンが表示されます。ボタンを選択すると**トランザクション情報**画面が表示されます。**トランザクション情報**の詳細については、次の文書を参照してください。

エージェント一覧のフィルタリング

2 **エージェント一覧**でフィルタリング対象を選択または検索して、トランザクション一覧で個々の対象を確認できます。

- 活性化されたエージェントを確認するには、**活性化エージェントのみ**チェックボックスを選択します。
- **全体エージェント**一覧ボックスをクリックすると、エージェントをカテゴリごとに分類して確認できます。
- 実行中のトランザクション情報をエージェントごとに確認する場合は、エージェントを選択し、**適用**ボタンをクリックします。選択を取り消すには、↻ボタンをクリックします。

自動更新設定

実行中のトランザクション情報を設定した間隔で自動更新できます。**自動リフレッシュ(秒)**トグルボタンでオプションを有効にし、右側の入力欄に更新間隔を入力します。

自動更新設定を無効にするには、**自動リフレッシュ(秒)**トグルボタンをもう一度選択します。

- 時間は秒単位で更新します。
- 自動更新設定を無効した状態でトランザクション一覧を更新するには、ボタンを選択します。

トランザクション一覧のフィルタリング

3 **トランザクション一覧**でURLまたはIPアドレスを検索してフィルタリングできます。右上のURL / IPの入力欄にフィルタリングする対象を入力し、Enterキーまたはボタンを選択します。トランザクション一覧は、入力値でフィルタリングされます。

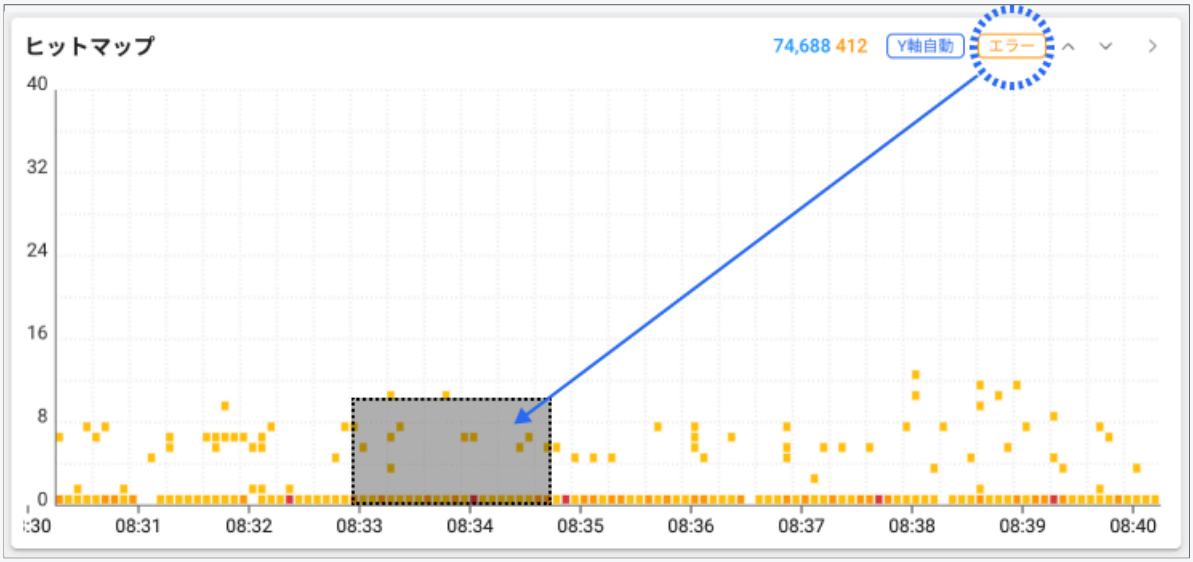
フィルタリングした一覧を初期化するには右上のURL / IPテキスト入力ボックスのテキストを削除し、Enterキーまたはボタンを選択します。

アクティブTX設定の詳細については、次の文書を参照してください。

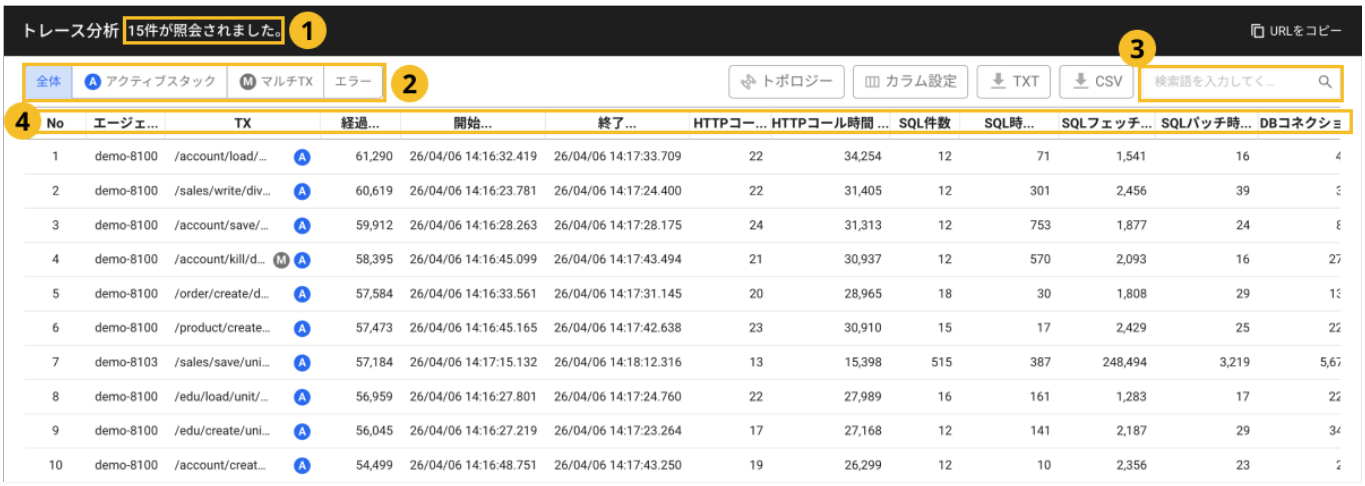
ヒットマップ/トレース分析

アプリケーションの**ヒットマップ**ウィジェットから**トレース分析**画面を開くと、トランザクション一覧と各トランザクションの下位ステップ情報が一緒に確認できます。 ヒットマップウィジェットで照会する領域をドラッグします。 選択した領域のトランザクション一覧と下位ステップ情報が**トレース分析**画面に表示されます。

エラーが発生したトランザクション情報のみを分類して確認するには、**アプリケーション**の**ヒットマップ**ウィジェットの右上にある**エラー**ボタンをクリックし、チャート領域をドラッグしてください。

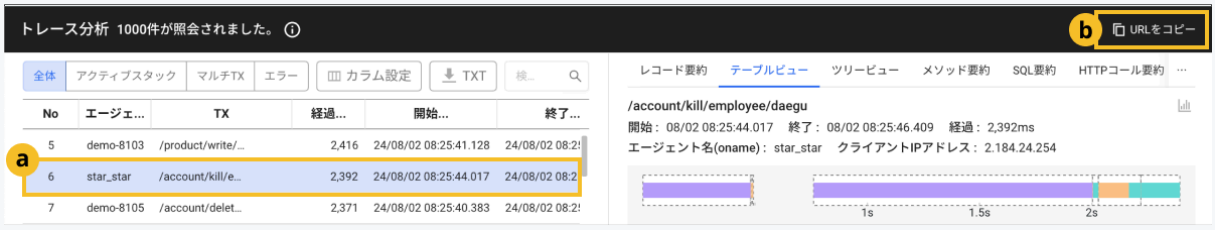


基本画面ガイド



- 1 **トランザクション照会件数**:ヒットマップウィジェットからドラッグした領域のトランザクション一覧の数を確認できます。 最大1000件まで照会できます。
- 2 **トランザクションの分類**：A **アクティブスタック**とM **マルチTX**、**エラー**項目を区分して一覧を確認できます。
- 3 **トランザクション検索**：トランザクション名、エージェント名、エラー メッセージに基づいてトランザクション一覧を検索できます。
- 4 **一覧ソート**：テーブルのヘッダーカラム項目をクリックすると、選択した項目に基づいてトランザクション一覧をソートできます。
- エラーが発生したトランザクション項目は、一覧に赤色で表示されます。

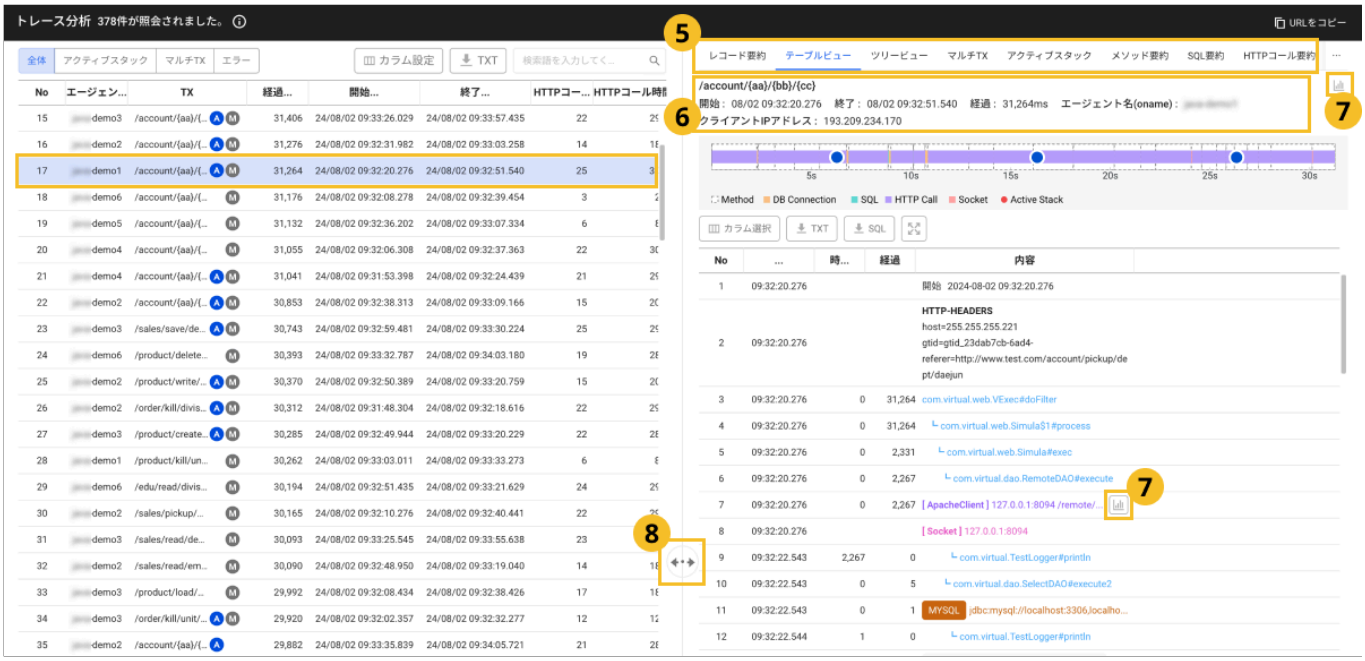
- テーブルでよく参照するカラムを前方に配置したり、カラムの追加および非表示などができます。 詳細は次の文書を参照してください。
- **ヒットマップ**ウィジェットから別の領域をドラッグすると、新たなトレースの分析画面を表示できます。既存のトレース分析画面と比較して確認できます。
- 表示されているトランザクション項目を他のユーザーに共有できます。 トランザクション項目を選択した状態(a)で、トレース分析画面b **URLをコピー** ボタンを選択します。 コピーしたURLアドレスを他のユーザーに共有します。 URLアドレスを受け取ったユーザーは、ブラウザアドレスバーにURLアドレスを入力すると、同じ項目をすばやく確認できます。



トレース分析

トランザクション一覧から詳細内容を確認する項目を選択します。 画面右側の選択したトランザクションに関しての詳細分析内容を表示します。 この状態で別のトランザクション項目を選択すると、選択したトランザクション項目の詳細分析内容に変更されます。

WhaTapは、トランザクションの性能を分析するために、名前やクライアント情報などの属性、トランザクションの処理性能、そして各区間別の詳細な実行履歴を収集して表示します。



- 5領域では、ステップ情報をさまざまな形式で確認できます。 希望する形式のタブを選択します。
- 6領域では、トランザクションに関する基本情報を確認できます。 左一覧のカラム項目と一致する情報です。

- 7 📊：該当URLまたは各実行区間の統計データの画面が表示されます。 統計データ画面のグラフチャートで、希望する時間をクリックすると**統計**または**TX検索**メニューに移動します。 選択した時間に基づいて統計データを照会することができます。
- 8 ↔左右にドラッグして、画面のサイズを調整できます。 より多くの情報を確認する際に便利な機能です。

> ⓘ
> - **統計**メニューの詳細は次の文書を参照してください。
> - **TX検索**メニューの詳細は、次の文書を参照してください。
> - ◀または▶ボタンをクリックすると、トランザクション一覧または詳細な分析画面をフルサイズに変更できます。

収集する**ステップ**(**Step**)の種類は次のとおりです。 ダイアグラムとテキストの色を区分して表示します。

- DB Connectionステップ START-END

  RDBへの接続のパフォーマンスが含まれます。 ステップ情報には、名前、レスポンス時間、エラーが含まれます。

- SQLステップ START-END

  JDBC SQLのパフォーマンスが含まれます。 ステップ情報には、接続情報、SQL文、エラーが含まれています。

- HTTP Callステップ START-END

  外部HTTPサービス呼び出しのパフォーマンスが含まれます。 ステップ情報には、url、host、port、レスポンス時間、エラーが含まれます。

- Messageステップ ADD

  トレースを収集する過程で、非構造化のすべての間隔の履歴を収集する際にメッセージステップを使用します。

- Socketステップ ADD

  Socketオープンを表現するステップです。

- Methodステップ START-END

  メソッド応答時間を追跡します。

- Active Stackステップ START-END

  アクティブスタックに関する情報を含みます。 別のスレッドが作成し、トレースに追加する方法で収集します。

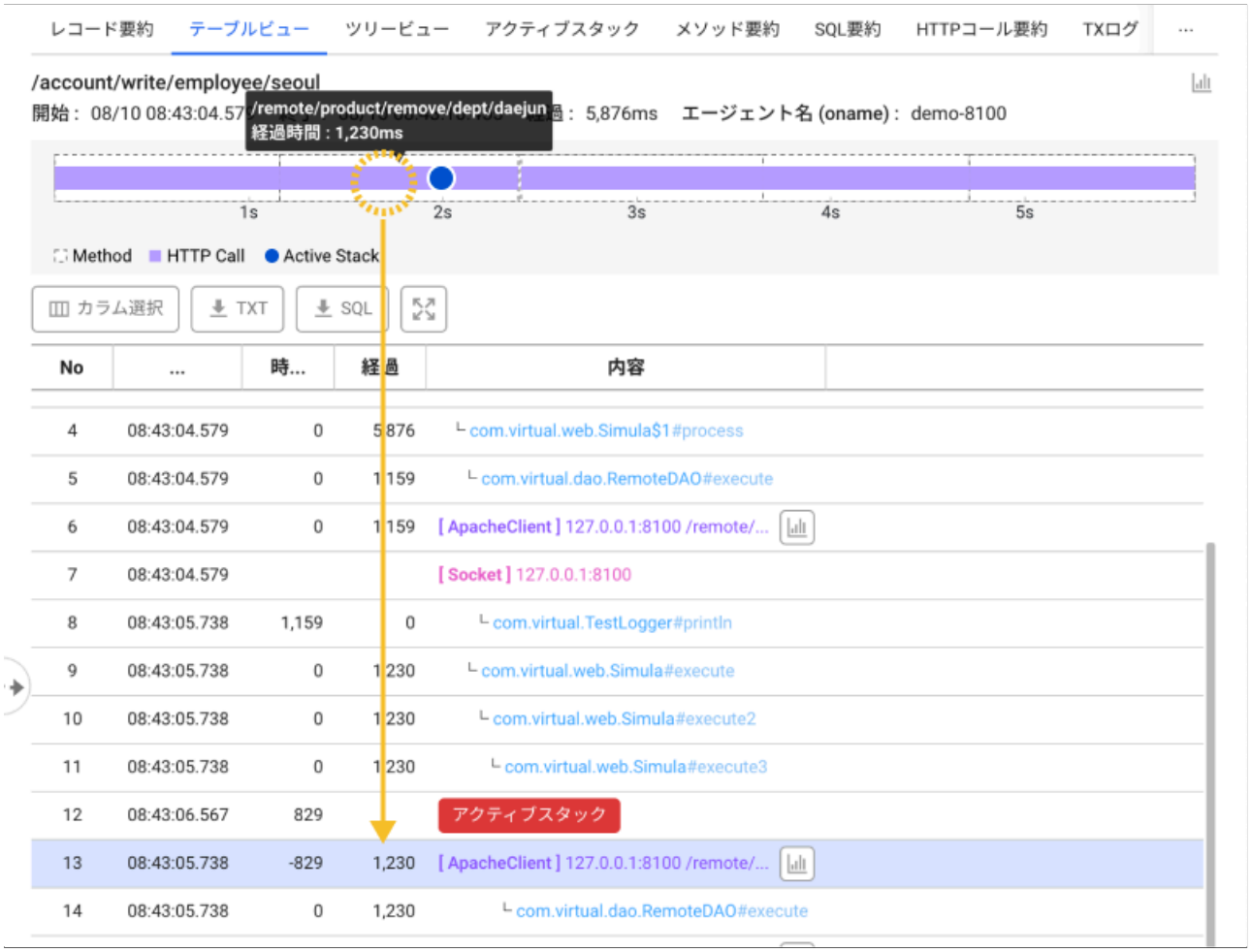
連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、連携したプロジェクトの性能メトリクスを照会できます。 連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

- WhaTapモニタリングサービスでは、**ステップ**(**Step**)は**スパン**(**Span**)と同じ意味で使用されます。
- 2023年7月25日にリリースされた**トレース分析**機能の変更点を確認するには 次の文書を参照してください。

テーブルビュー

テーブルビュータブでは、トランザクションの実行過程を時間順に確認できます。

- トランザクションを実行区間別に分類し、ダイアグラムで確認できます。 全経過時間のうち、各区間別に所要された時間、最も長くかかった区間を素早く把握できます。
- ダイアグラムの各区間を選択すると、そのステップが位置するテーブルの一覧に移動します。



- 青い丸(●)が位置する領域は、アクティブスタックが収集された時点です。 青い丸を選択すると、**アクティブスタック**ボタンがあるテーブル一覧に移動できます。 **アクティブスタック**ボタンをクリックすると、その区間で行われたステップ情報を確認できます。

> ⓘ ◦ トランザクション一覧からⒶアイコンが表示された項目で確認できます。
> ◦ アクティブスタックの詳細については、次の文書を参照してください。

- **カラム選択**：テーブル一覧にカラムでメモリ累積情報およびCPU累積情報を追加または非表示できます。 カラム情報は次のとおりです。
  - **No**：ステップの実行順序
  - **時間**：各ステップの開始時刻

  - **時間差**：直前ステップの開始時刻から現在のステップに移るまでの待ち時間、外部要因により遅延される場合、経過時間と異なる場合があります。
  - **経過**：各メソッドの開始から終了までの所要時間
  - **内容**：該当ステップの詳細な実行内容
- **TXT**：トランザクションの基本情報および区間ごとの実行情報をTXTファイルでダウンロードできます。
- **SQL**：トランザクションの基本情報およびSQL実行情報をTXTファイルでダウンロードできます。
  - **置換値を含むダウンロード**: バインド変数値を元の値に置き換え、SQL統計データをダウンロードします。 セキュリティキー(`paramkey`)が設定されている場合は、セキュリティキー入力画面が表示されます。 セキュリティキーを入力しないとダウンロードできません。 セキュリティキーに関するエージェント設定の詳細については、次の文書を参照します。
  - **置換値を含まないダウンロード**: バインド変数値を置き換えない状態でSQL統計データをダウンロードします。
- :トランザクションの要約情報とダイアグラムを非表示にし、テーブル一覧のみを確認できます。 ボタンをクリックすると、隠された情報を再表示します。 テーブル一覧が長い場合は、この機能を利用すると便利です。
- SQLステップを選択すると、パラメータを照会できる**SQL**画面が表示されます。 詳細については、次の文書を参照してください。

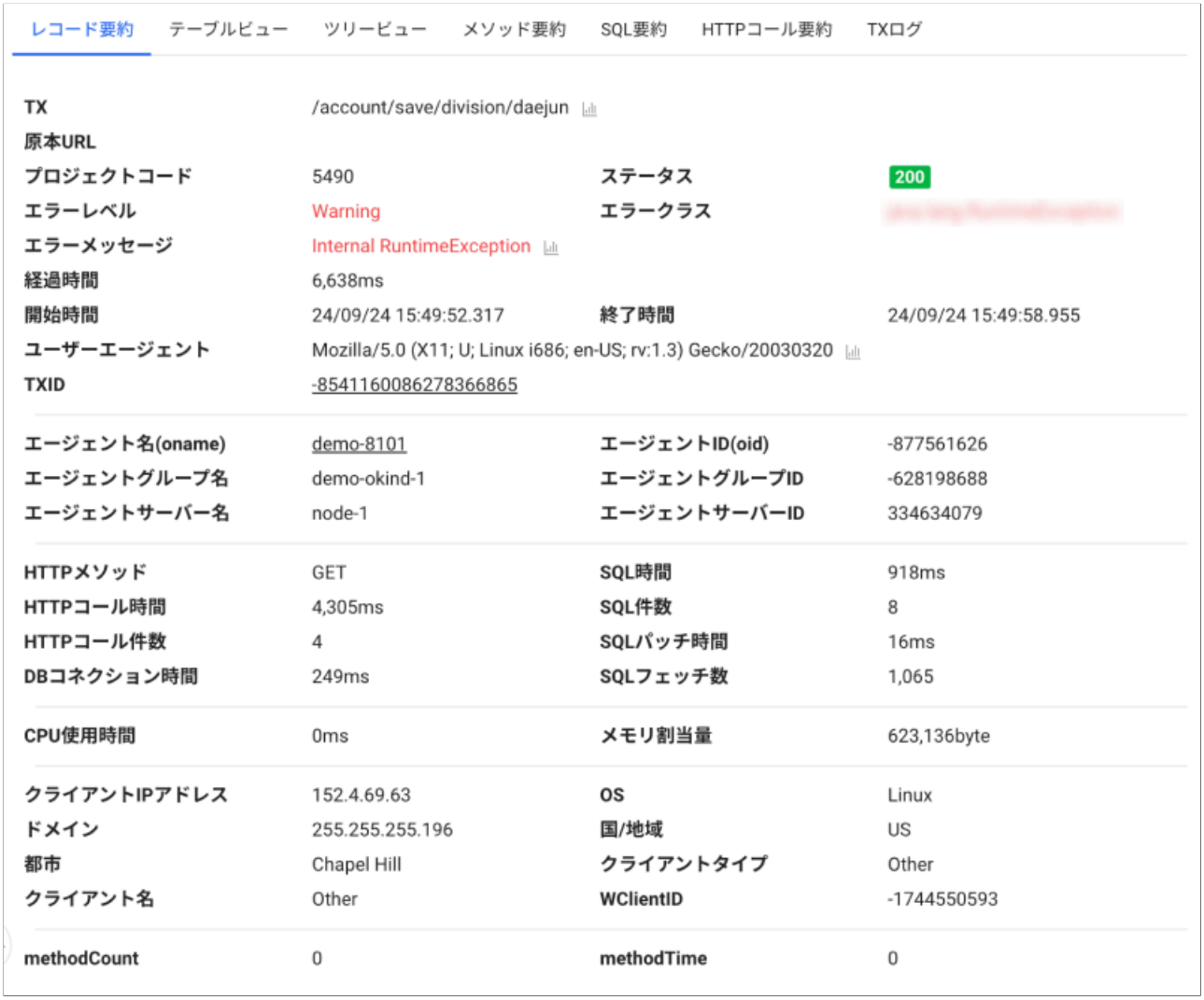
連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、**DB Connectionステップ**で連携したプロジェクトの性能メトリクスを照会できます。



連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

レコード要約

トランザクションの基本情報を確認するには、**レコード要約**タブを選択します。 トランザクションの基本情報およびそのトランザクションが実行されたエージェント関連情報、メモリ割り当て使用量、クライアント関連情報を確認できます。



分類	プロパティ	説明
トランザクション	TX	アプリケーションで実行されたAPIまたはURL呼び出し、そのトランザクションのパスです。 正規表現式を使用して一定の形式に変換します。

分類	プロパティ	説明
	原本URL	実際に呼び出されたURLパスです。 変数を含まない具体的なリクエストパスです。 元のURLがある場合は元のURL表示、元のURLがない場合は表示しません。
	プロジェクトコード	WhaTapモニタリングサービスに登録したプロジェクトの識別コードです。
	ステータス	そのトランザクションのHTTP応答ステータスのコードです。
	エラーレベル	そのトランザクションで発生したエラー レベルです。 WarningまたはCriticalと表示されます。
	エラークラス	そのトランザクションで発生したエラー関連クラスです。
	エラーメッセージ	そのトランザクションで発生したエラーメッセージです。
	経過時間	トランザクションの実行時間です。
	開始時間	トランザクションの開始時刻です。
	終了時間	トランザクションの終了時刻です。
	ユーザーエージェント	クライアント関連情報の抽出に利用するブラウザ情報です。
	リファラー	クライアントがブラウザで以前に利用したページアドレスまたは流入パスです。
	TXID	該当トランザクションの固有識別子(ID)です。
	マルチTXID	マルチトランザクションの固有識別子(ID)です。 他のアプリケーションのトランザクションと並列に実行された場合に表示されます。
エージェント	エージェント名(oname)	エージェント名です。
	エージェント	エージェントの固有識別子(ID)です。

分類	プロパティ	説明
	ID(oid)	
	エージェントグループ名	`okind`オプションで設定されたエージェントグループ名です。
	エージェントグループID	エージェントグループの固有識別子(ID)です。
	エージェントサーバー名	`ondoe`オプションで設定したエージェントサーバー名です。
	エージェントサーバーID	エージェント サーバーの固有識別子(ID)です。
ステップ (Step)	HTTPメソッド	HTTPメソッドです。 (GET、POST、PUT、HEADなど)
	HTTPコール時間	外部Http Call時間です。
	HTTPコール件数	外部HTTP Call件数です。
	DBコネクション時間	データベースに接続するまでにかかる時間です。
	SQL時間	SQLの実行時間です。
	SQL件数	SQLの実行件数です。
	SQLパッチ時間	レコードの照会にかかった時間です。 ネストが発生するか、他のロジックが含まれる場合があります。
	SQLフェッチ数	SQLでデータを照会したレコードの件数です。

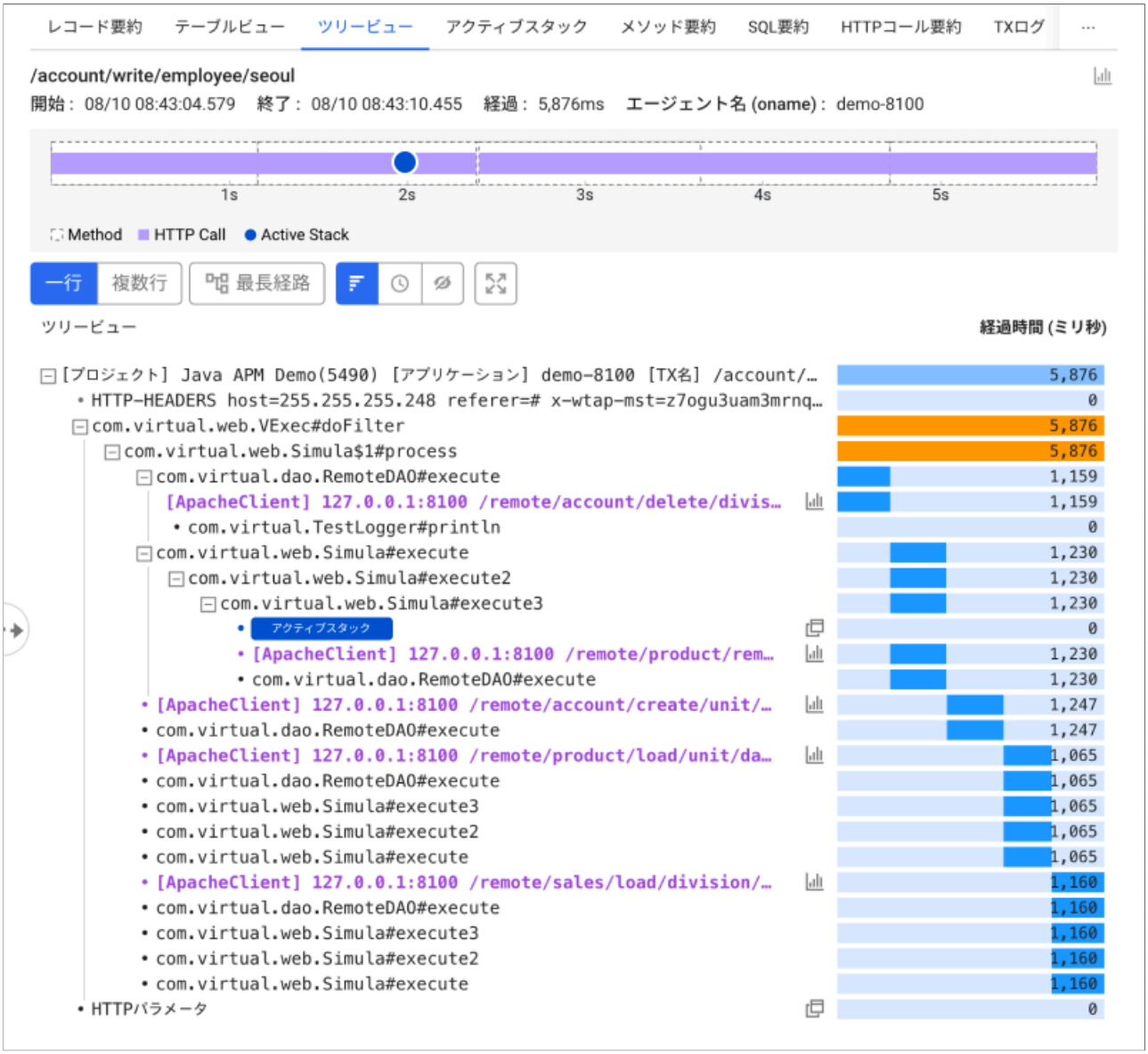
分類	プロパティ	説明
リソース (Resource)	CPU使用時間	トランザクションの処理に使用されたCPU時間です。
	メモリ割当量	そのトランザクションで使用されたメモリ量です。
コンテナ	Pod名	該当トランザクションを処理したKubernetes Podの名前です。
	コンテナID	Pod内で実行されている特定のコンテナの固有識別子です。
クライアント	クライアントIPアドレス	クライアントのIPアドレスです。
	OS	ブラウザが実行されるOS環境です。
	ドメイン	クライアントが接続したIPアドレスに指定されたインターネットアドレスです。
	国/地域	クライアントの国の情報です。
	都市	クライアントの国内の都市情報です。
	クライアントタイプ	クライアントが利用したブラウザの種類です。
	クライアント名	クライアントが利用したデバイス名です。
	WClientID	クライアントの固有識別子です。
Method	methodCount	トランザクションで呼び出されたメソッドの数です。
	methodTime	メソッドの実行にかかった時間です。

- アプリケーションの種類や設定、ステップの種類によって収集する情報は異なる場合があります。 **コンテナ**項目は、アプリケーションがコンテナ環境で実行されている場合にのみ表示されます。
- エラー関連項目は、そのトランザクションでエラーが発生した場合に表示されます。

- トランザクション属性の一部は、統計機能を提供します。 ボタンを選択すると、時系列チャートを通じて照会時間中に発生したトランザクション状況を把握できる画面が表示されます。 チャート内の棒グラフを選択すると、**分析** > **TX検索**メニューに移動して詳細照会できます。
- WhaTapは、クライアントに関する情報をデフォルトで保存します。 ユーザーデータの収集に関するエージェント設定の詳細については、次の文書を参照してください。

ツリービュー

トランザクションの実行過程をツリー形式で確認するには、**ツリービュー**タブを選択します。 各トランザクションとそれに属するトレースの詳細、トレースの開始時刻と所要時間、呼び出し関係を確認できます。 ダイアグラムの各区間を選択すると、そのステップが位置するツリービューに移動します。



- **一行**：各区間別の実行情報に表示されたテキストを一行で表示し、ツリー形式をシンプルに整理できます。
- **複数行**：各区間別の実行情報に表示されたテキストを改行してすべて表示します。
- **最長経路**：最長の経路で移動できます。
- **タイムバーの表示**：経過時間を棒チャートで表示します。
- **時間を表示する**：各区間ごとのタイムスタンプ、ギャップ、経過時間をテキスト形式で表示します。
  - 8秒以上：超過遅延状態で赤色で表現します。

- 3秒以上8秒未満：遅延状態でオレンジで表現します。
- 3秒未満：正常な状態で青色で表示します。

- **時間を表示しない**：時間情報を非表示にします。
- ：トランザクションの要約情報とダイアグラムを非表示にし、ツリービューのみを確認できます。 ボタンをクリックすると、隠された情報を再表示します。 ツリービュー一覧が長い場合は、この機能を利用すると便利です。
- ：SQL変数とHTTPクエリを照会できる画面が表示されます。 詳細については、次の文書を参照してください。

> 開始および所要時間の場合、トランザクション呼び出し環境によって生じる時差を上位トランザクションのトレースやマッピングで修正および表示することで、実際に収集された時間データとの差異が生じる可能性があります。

アクティブスタック



アクティブスタックタブを選択すると、該当トランザクションから収集したアクティブスタックの一覧を確認できます。

- 左側の**アクティブスタック**一覧から収集時点（No、時間）を選択すると、右側に該当時点のスタック情報を表示し

ます。

- **スタック**：アクティブスタックの詳細情報をTXTファイルでダウンロードできます。

> - トランザクション一覧でAアイコンが表示された項目からアクティブスタックを確認できます。
> - アクティブスタックの詳細については、次の文書を参照してください。

AIアクティブスタック分析

Beta

アクティブスタック分析ボタンをクリックすると、収集したアクティブスタックをAIが自動的に分析します。進行中のトランザクションのスタックダンプを基にパフォーマンスの問題を特定し、改善方法を案内します。

分析結果は次の4つのセクションで構成されます。

- **全体要約**：分析したスタック数と全体の実行フローを要約します。
- **共通パターン**：繰り返し現れる実行パターンを表示します。パターン名、発生回数、説明を含みます。
- **発見されたイシュー**：特定したパフォーマンスの問題を深刻度（**High**、**Medium**、**Low**）とともに表示し、判断の根拠となるスタック情報も提供します。

深刻度	基準
High	デッドロック、完全ブロッキング、同一ブロッキング4回以上繰り返し
Medium	I/O待機、2〜3回のロック競合、遅い外部呼び出し
Low	単一発生の待機、軽微な最適化の機会

- **推奨事項**：優先順位に基づいて具体的な改善方法を案内します。

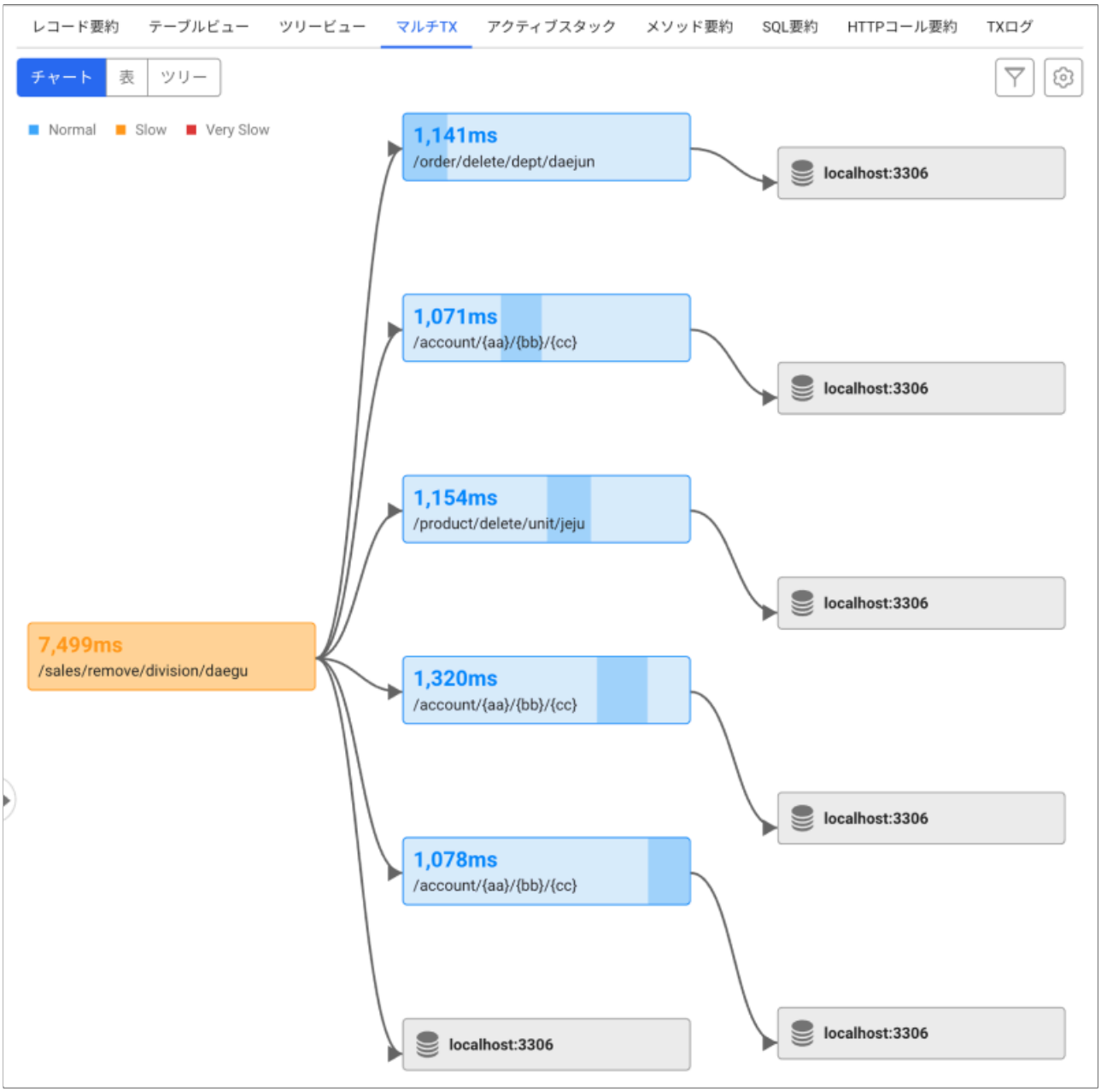
各分析項目には収集したスタック番号（ No.N ）が表示されます。スタック番号をクリックすると、該当スタックの原本を確認できます。

> - AIが生成した分析結果のため、実際の状況と異なる場合があります。参考資料としてご活用ください。
> - プロジェクト**メンバー**権限以上が必要です。

マルチトランザクション

マルチトランザクションとは、他のエージェントやプロジェクトに関連付けられたトランザクションを意味します。**マルチTX**タブでは、WhaTapモニタリングサービスに登録したアプリケーション間の呼び出し関係を確認できます。

> マルチトランザクションを追跡するには、**管理** > **エージェント設定**メニューで `mtrace_enabled` オプションを `true` に設定します。 エージェント設定の詳細については、次の文書を参照してください。



- **チャート**：各トランザクションの呼び出し関係をフローチャート形式で提供します。 トランザクションノードを選択すると、そのトランザクションノードのトレース分析情報を確認できます。 **チャート**モードではマウスを利用して位置を移動したり、スクロールして拡大または縮小できます。
  - ⚙**チャートビュー設定**：チャートに表示するアイテムを表示または非表示します。
- **表**：テーブル形式で、マルチトランザクション内の各トランザクションごとの情報を確認できます。 カラム選択

アイコンを選択して、テーブルヘッダーカラムを編集できます。 各トランザクション項目を選択すると、トレース分析情報を確認できます。

- **ツリー**：ツリー形式でトランザクション間の呼び出し関係を把握できます。 関連した付加機能は、**ツリービュー**タブの機能と同じです。
- **プロジェクト選択**：チャートに表示されるプロジェクトを選択または解除できます。

> ⓘ
> - トランザクション一覧からⓂアイコンがある項目で確認できます。
> - マルチトランザクションの詳細については、次の文書を参照してください。

メソッド要約



No	ク...	メソ...	パラメータ/戻...	数	合計 (ms)	最大 (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

メソッド情報のみを確認するには、**メソッド要約**タブを選択しますて。 エージェントに追跡が設定されたメソッド名と所要時間を表示します。 不要なロジックが繰り返し実行される場合、低い件数に比べて実行時間が長くかかる場合を把握し、メソッドロジック改善のための分析情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。

- **メソッド**: メソッドに関する詳細情報をTXTファイルにダウンロードできます。

> メソッド(method)と関連するエージェント設定オプションの詳細については、次の文書を参照してください。 ご利用中の製品によって、メソッド追跡サポートの可否は異なる場合があります。

SQL要約



No	データベ...	SQL	数	合計 (ms)	最大 (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文についての情報を確認するには、**SQL要約**タブを選択します。 不要に繰り返し実行される場合、低い件数に比べて実行時間が長くかかる場合を把握し、クエリ性能改善のための分析情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **SQL**: SQLに関する詳細情報をTXTファイルにダウンロードできます。

HTTPコール要約



No	リモートサー...	URL	数	合計 (ms)	最大 (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呼び出しのリクエスト件数、合計時間、平均時間などを確認するには、**HTTPコール要約**タブを選択します。 不要な外部への呼び出しが繰り返して実行される場合や、低い件数だが比較的に実行時間が長くかかる場合などを把握するなど、トランザクションの遅延要因が外部か内部かを分析する情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **HTTP**: HTTP Callに関する詳細情報をTXTファイルにダウンロードできます。

TXログ

トランザクションに関するログ情報を確認するには、**TXログ**タブを選択します。



- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **ログ検索機**: ログ検索に移動して、目的のログデータを簡単に照会できます。
- **キーワード検索**：検索バーに検索するテキストを入力し、エンターキーを入力するか、ボタンを選択します。 入力したテキストと一致するキーワードをハイライトで表示します。
- **キーワードハイライト**：キーワードと色を設定すると、自動的にログ一覧でキーワードと一致するテキストをハイライトします。 詳細については、次の文書を参照してください。
- **カラム設定** : ログリストテーブルに表示されるカラムの追加や順序変更ができます。 詳細については、次の文書を参照してください。
- **ログ表示詳細設定**: ログリストに表示される内容を設定できます。 詳細については次の文書を参照してください。

> - **TXログ**タブを有効にするには、ログに関連するエージェント設定オプションを適用する必要があります。 ログ設定の詳細については、次の文書を参照してください。
> - **TXログ**タブは**ログ読取り**権限を持つメンバーのみアクセスできます。 メンバー権限の詳細については、次の文書を参照してください。

キーワードハイライトの設定

1. ハイライトで表示するキーワードを追加するにはボタンを選択します。



2. ご希望のキーワードを入力します。
3. 色アイコンを選択し、ご希望の色を選択します。
4. エンターを入力します。 追加したキーワードが一覧に作成されます。

キーワードハイライト削除

ハイライト一覧から利用しない項目を削除するには、—ボタンを選択します。

テーブルカラムの表示または非表示

1. テーブルの右上の⊞ボタンを選択します。
2. **カラム設定**ウィンドウが表示されたら、左側のリストで目的の項目を選択・解除します。
   右のリストから×ボタンを選択してカラムを非表示にすることもできます。
3. すべての設定を完了した後**確認**ボタンを選択します。 設定した内容がテーブルに適用されます。

テーブルカラムの表示順番の変更

1. テーブルの右上の⊞ボタンを選択します。
2. **カラム設定**ウィンドウが表示されたら、右側のリストで目的の項目をドラッグして順序を変更します。
3. すべての設定を完了した後**確認**ボタンを選択します。 設定した内容がテーブルに適用されます。

テーブルコンテンツの表示設定

1. テーブル右上の⚙ボタンを選択します。
2. **ログ表示詳細設定**ウィンドウが表示されたら、目的のオプションを設定します。
   - **content**または**タグ**オプションを選択して、ログにコンテンツの表示有無を選択します。
   - **タグ管理**: タグは、追加した順にログに表示されます。 タグの順序はドラッグで変更できます。 非活性化されているタグはログに表示されません。
3. すべての設定を完了して✕ボタンを選択します。 設定した内容がテーブルに適用されます。

> ⓘ **ログ表示詳細設定**の詳細については次の文書を参照してください。

追加機能

HTTPパラメータ照会

テーブルビュー タブでそのトランザクションのHTTPパラメータを照会することができます。

1. ページの下にスクロールして、トランザクションの実行の最後のステップに移動してください。
2. 🔒 **HTTPパラメータ**項目を選択します。
3. **HTTP-PARAMETERS**画面が表示されると、**パスワード** ボタンを選択します。
4. 設定したParam Keyを入力します。隠されたパラメータを確認できます。

> ⓘ
> - HTTPパラメータに関するエージェント設定は、次の文書を参考にしてください。
> - **パスワード**：復号化されたパラメータの値を確認できます。 パスワードは `WHATAP_HOME /security.conf (paramkey.txt)`ファイル内の6桁の文字列です。 別の文字列に変更できます。 SQL変数の照会、HTTPクエリの照会、Threadの中止に必要です。 詳細については、次の文書を参照してください。

SQLパラメータの照会

テーブルビュータブでSQLステップを選択するか、**ツリービュー**タブでボタンを選択します。 パラメータを照会できる**SQL**画面が表示されます。



- **パスワード**：復号化されたパラメータの値を確認できます。 パスワードは* `WHATAP_HOME` /security.conf*ファイル内の6桁の文字列(`paramkey`)です。 別の文字列に変更できます。

  > **パスワード**は、SQL変数の照会、HTTPクエリの照会、Threadの中止に必要です。 詳細については、次の文書を参照してください。

- **SQLフォーマット**：SQL文をインデントまたはフォーマット化して可読性を高めることができます。
- ：SQL文をクリップボードにコピーできます。
- **統計**: **統計** > **SQL**メニューに移動します。

> SQL変数とHTTPクエリを照会するには、次のオプションをエージェント設定に追加します。
>
> - SQLパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
> - HTTPパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
>
> whatap.conf

```
# SQLパラメータ照会オプション：オプションが適用されるとSQLパラメータを暗号化して収集します。
profile_sql_param_enabled=true

# HTTPパラメータ照会オプション：オプションが適用されると、HTTPクエリパラメータを暗号化して収集します。
profile_http_parameter_enabled=true
```

連携プロジェクトの性能メトリクスを照会する

連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、**DB Connectionステップ**及び**SQLステップ**で連携したプロジェクトの性能メトリクスを照会できます。

DB Connectionステップで確認する

Correlated project



1. ダッシュボードで**ヒットマップ**ウィジェットのチャート領域をドラッグします。
2. **トレース分析**画面が表示されたら、照会したいトランザクション項目を選択します。
3. トランザクションの詳細情報が右側に表示されると、**テーブルビュー**タブを選択します。
4. DB Connectionステップでボタンを選択します。
5. 連携プロジェクトの**インスタンスダッシュボード**画面が表示されたら、関連するメトリクスを確認します。

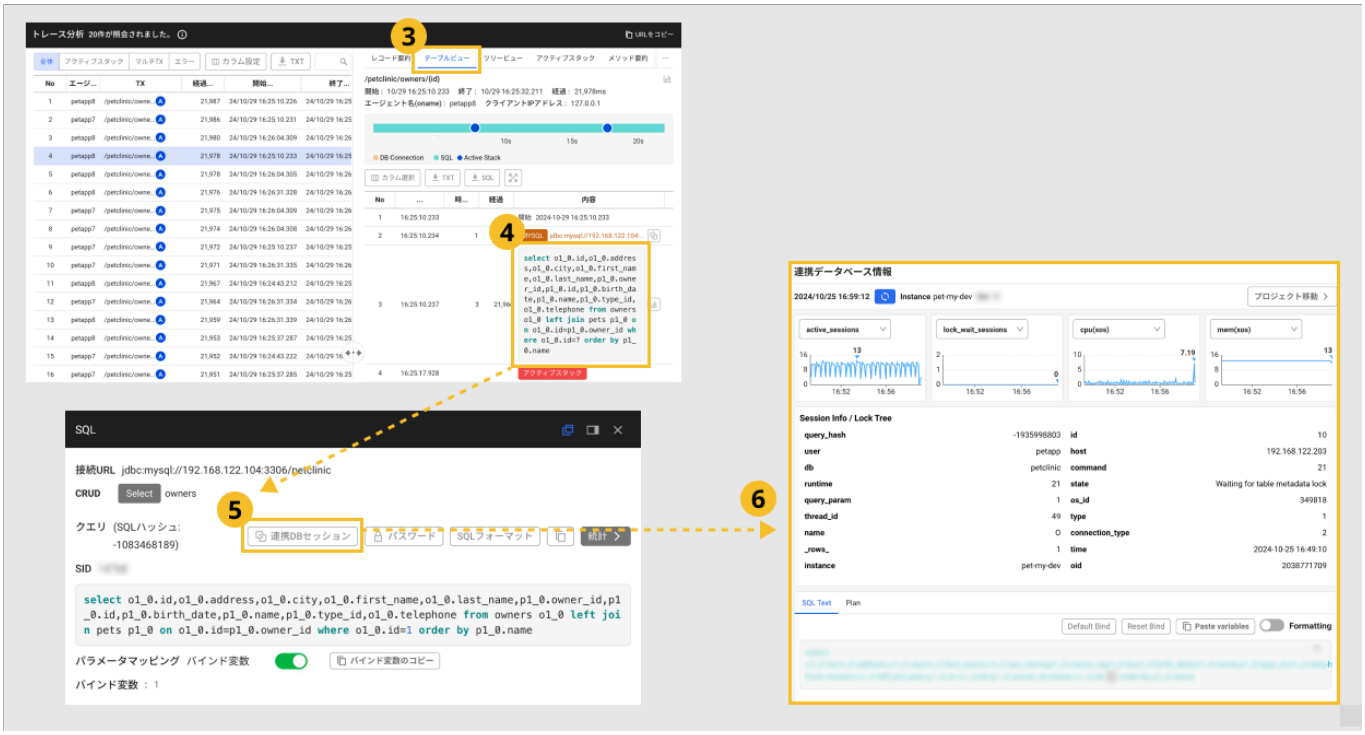
- 連携分析のために登録したデータベースプロジェクトのDBが現在選択したトランザクションのSQLを実

行した場合のみ、**テーブルビュー**タブでボタンが表示されます。

- 連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。
- **インスタンスダッシュボード**画面についての詳細は、次の文書を参照してください。

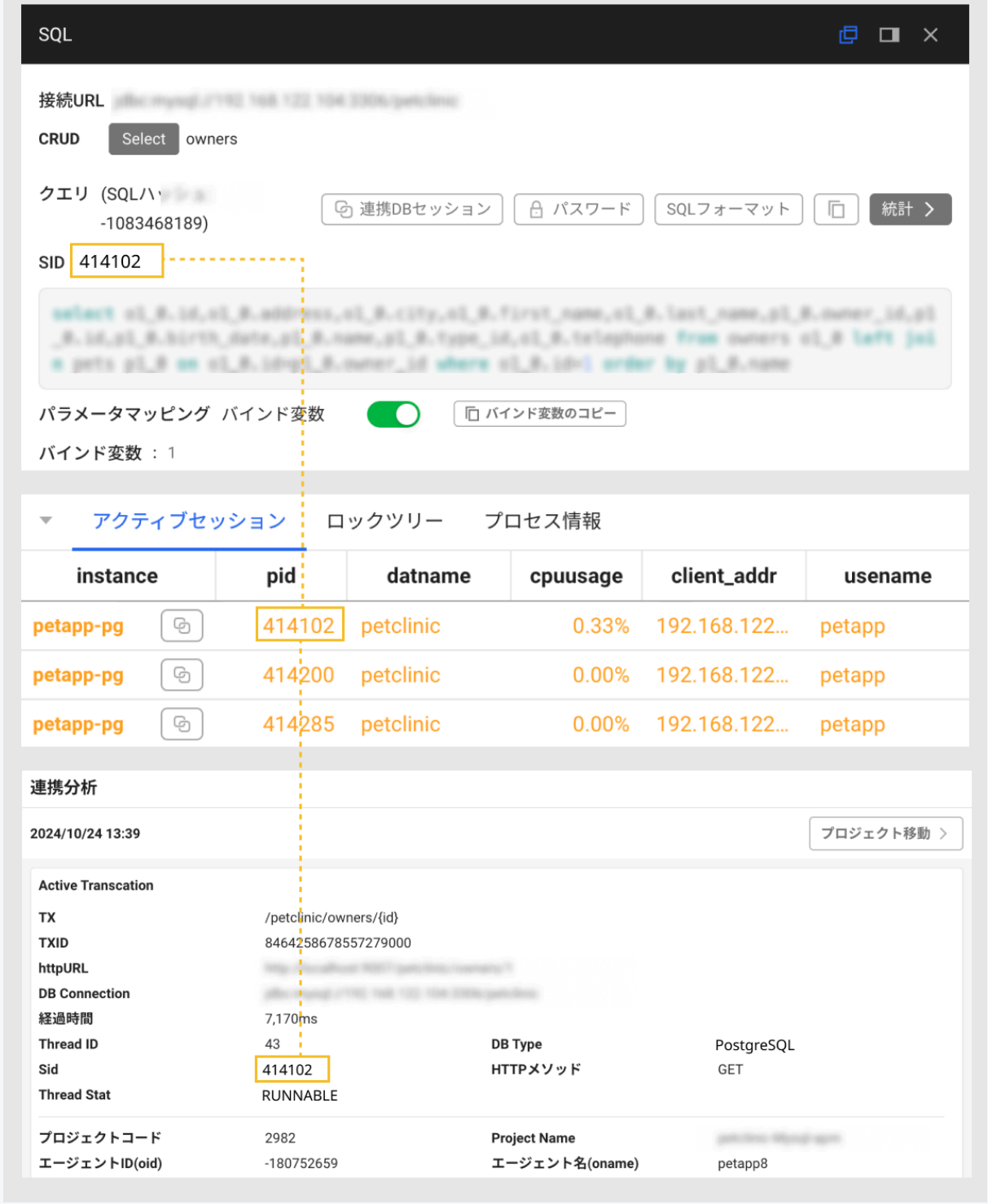
SQLステップで確認する

Correlated project



1. ダッシュボードで**ヒットマップ**ウィジェットのチャート領域をドラッグします。
2. **トレース分析**画面が表示されたら、照会したいトランザクション項目を選択します。
3. トランザクションの詳細情報が右側に表示されると、**テーブルビュー**タブを選択します。
4. SQLステップでSQL文を選択します。
5. **SQL**画面が表示されたら、**連携DBセッション**ボタンを選択します。
6. **連携データベース情報**画面が表示されたら、関連するメトリクスを確認します。

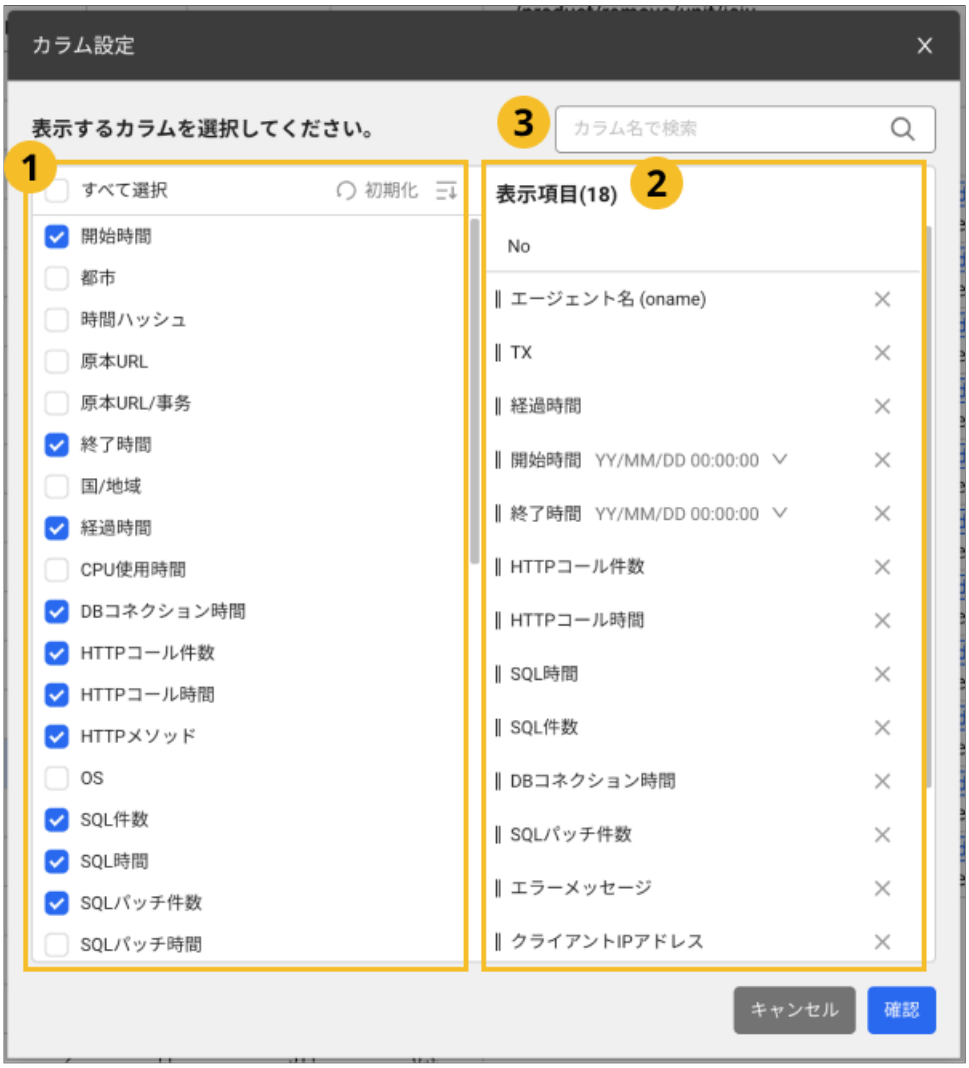
- 連携分析のために登録したデータベースプロジェクトのDBが現在選択したトランザクションのSQLを実行した場合のみ、**SQL**画面に**連携DBセッション**ボタンが表示されます。
- **バインド変数のコピー**機能を利用すると変数をコピーして**連携データベース情報**ウィンドウの**SQL Text**タブに変数を貼り付けることができます。 この機能は**パスワード**ボタンを選択してパスワードを入力した後、バインド変数の値が表示されるときに利用できます。
- **連携データベース情報**ウィンドウの詳細については次の文書を参照してください。
- この機能をサポートするデータベースは、**PostgreSQL**、**Oracle**、**Oracle Pro**、**MySQL**プラットフォームです。
- **PostgreSQL**プラットホームで**アクティブセッション**テーブルの**PID**項目は、**SID**項目と同じです。


SQL
接続URL
CRUD
Select
owners
クエリ (SQLハッシュ
-1083468189)
連携DBセッション
パスワード
SQLフォーマット
統計
SID
414102
パラメータマッピング バインド変数
バインド変数のコピー
バインド変数 : 1
アクティブセッション
ロックツリー
プロセス情報
instance
pid
datname
cpuusage
client_addr
usename
petapp-pg
414102
petclinic
0.33%
192.168.122...
petapp
petapp-pg
414200
petclinic
0.00%
192.168.122...
petapp
petapp-pg
414285
petclinic
0.00%
192.168.122...
petapp
連携分析
2024/10/24 13:39
プロジェクト移動
Active Transcation
TX
/petclinic/owners/{id}
TXID
8464258678557279000
httpURL
DB Connection
経過時間
7,170ms
Thread ID
43
DB Type
PostgreSQL
Sid
414102
HTTPメソッド
GET
Thread Stat
RUNNABLE
プロジェクトコード
2982
Project Name
エージェントID(oid)
-180752659
エージェント名(oname)
petapp8

> **Javaエージェント2.2.40**以前のバージョンでは**アクティブTX**ウィンドウで**ThreadStat**値が数字値形式で表示されます。 文字形式で表示するにはJavaエージェント2.2.40バージョン以降をインストールします。

テーブルカラムの設定

トランザクション一覧でテーブルヘッダーカラムを非表示にしたり、希望する項目を追加できます。 カラムの順序を変更することもできます。 **カラム設定**ボタンを選択します。



- 設定を完了した後は、**確認**ボタンを選択すると、設定事項がテーブルに反映されます。
- 3検索バーにテキストを入力し、カラム項目を検索できます。 入力したテキストと一致するカラム項目が表示されます。

カラムの追加

1一覧でテーブルヘッダーカラムに追加する項目を選択します。 すべての項目を追加するには、**すべて選択**を選択します。

カラムを削除する

1一覧から削除するカラム項目のチェックボックスのチェックを解除します。 または2一覧から削除する項目の右側の×ボタンを選択します。

カラム順序の変更

2一覧で順序を変更するアイテムをドラッグして、希望する位置に移動できます。

設定事項の初期化

変更事項をすべてキャンセルし、初期化するには、**初期化**ボタンを選択します。

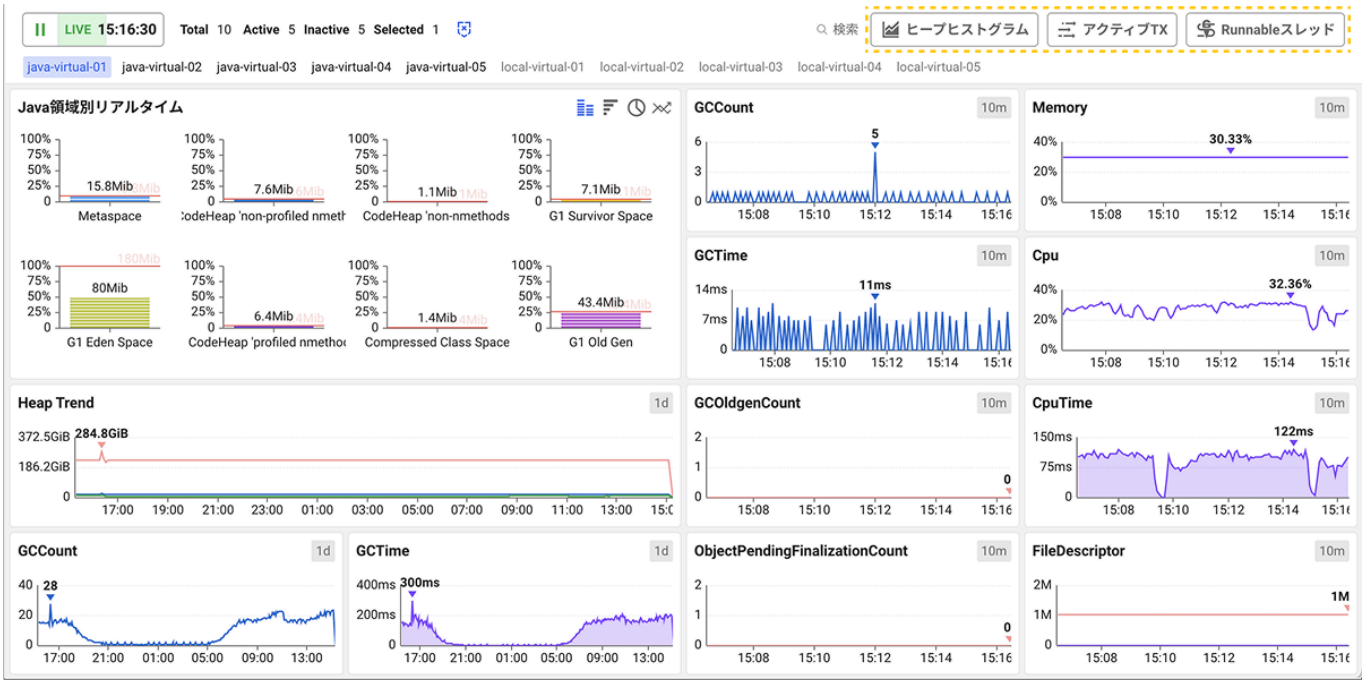
照会データのダウンロード

テーブルに出力されたデータをtxtファイル形式でダウンロードするにはTXTボタンを選択します。

エージェントリソースダッシュボード設定

> **要求事項**
>
> WhaTap Javaエージェントの2.2.39バージョン以降が必要です。

エージェントリソースダッシュボードメニューで個別のJavaエージェントのメモリ、トランザクション、スレッドなどの主要指標をリアルタイムでモニタリングし、最大3時間前の過去の時点まで照会できます。 これにより、メモリ使用量の変化、活性化されたトランザクション、実行中のスレッド情報を把握し、リソースの問題を引き起こす原因を分析できます。



基本画面

エージェントリソースダッシュボードにより、現在実行中のエージェントのメモリプール状態と関連指標の変化の推移を一目で確認できます。 ダッシュボードで確認できるリアルタイムのJavaメモリ領域及び主な指標は次のとおりです。

ウィジェット案内

リアルタイムJavaメモリ領域ウィジェット

Java領域別リアルタイムウィジェットを通じてJVMの主要メモリプールの状態をリアルタイムで確認できます。 各Javaメモリ領域チャートは、使用中の容量(MB)と比率(%)を表示します。

- **Metaspace**

  クラスメタデータを保存する領域です。

- **CodeHeap non-profiled nmethod**, **CodeHeap profiled nmethod**, **CodeHeap non-nmethods**

  JIT(Just-In-Time)コンパイラが生成したマシンコードを保存する領域であり、**CodeHeap non-profiled nmethod**、**CodeHeap profiled nmethod**、**CodeHeap non-nmethods**は、それぞれコンパイルされたコードを用途に応じて絞り込んで保管します。

- **G1 Survivor Space**

  G1 GCで生き残ったオブジェクトが一時的に保存される領域です。

- **G1 Eden Space**

  新しく作成されたオブジェクトが最初に割り当てられる領域です。

- **G1 Old Gen**

  Young GCを経て、生き残ったオブジェクトが移動する領域です。 メモリが増加した場合、Old Genを中心にリークが発生しているかを確認できます。

- **Compressed Class Space**

  クラス関連のポインタを圧縮して保存する領域で、Metaspaceとは別に管理されます。

> - 個別チャート: 各指標を別の棒形式で表示し、指標ごとに最大値を明確に比較できます。
> - 統合チャート: 全体に対する各指標の相対的な大きさを一目で把握できます。
> - ドーナツチャート: 全体を円形の割合で表現し、各指標が占める割合を確認できます。
> - マルチラインチャート: 全指標の推移を確認できます。

主要指標ウィジェット

- **Memory**

  全体メモリ使用率(%)を表示します。 Eden、Old Gen、Metaspaceなど内部構成を総合した総使用量です。

- **CpuTime**

  JVMがアプリケーションコードの実行に費やされたCPU時間をミリ秒(ms)単位で示します。

- **GCCount, GCOldgenCount**

  GCCountは、一定周期内に発生したGC回数、**GCOldgenCount**は、Old領域を対象としたGC回数のみ別途集計します。

- **GCTime**

  Garbage Collectionに費やされた時間をms単位で表示します。

- **Cpu**

  JVMプロセスのCPU使用率(%)を示します。

- **TPS**

  1秒あたりのトランザクション処理回数を表示します。

- **FileDescriptor**

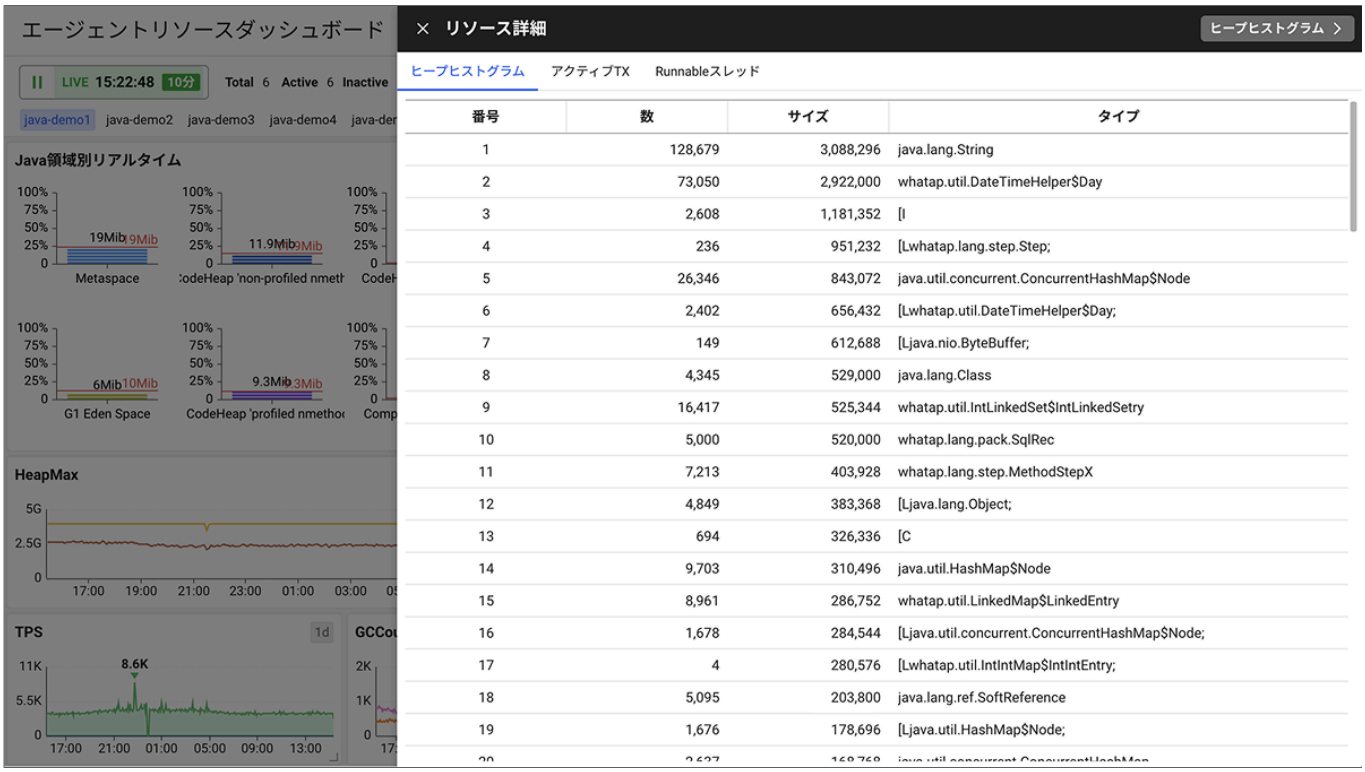
  開いているファイルやソケットなど、現在使用中のファイルディスクリプターの数を示します。

- **ObjectPendingFinalizationCount**

  GCが処理すべきオブジェクトが `finalize()` 待機状態のまま残っている個数を表示します。

リアルタイム照会

リアルタイム照会時、上段右側の**ヒープヒストグラム**、**アクティブTX**、**Runnableスレッド**ボタンを通して**リソース詳細**パネルを確認できます。



ヒープヒストグラム

ヒープヒストグラムタブで現在ヒープメモリに常駐しているJavaクラスの情報を分析し、メモリを最も多く占めるクラスTOP100を照会できます。 過度にメモリを占有するクラスがある場合、メモリリークの可能性が疑われます。

アクティブTX

アクティブTXタブで実行中のトランザクションのリストを照会できます。 開始時刻、トランザクション経過時間、SQL/HTTP呼び出しなどの詳細情報を確認できます。 特定のトランザクションが時間がかかるか、リソースを大量に使用しているかを確認することで、ボトルネック区間を特定できます。

Runnableスレッド

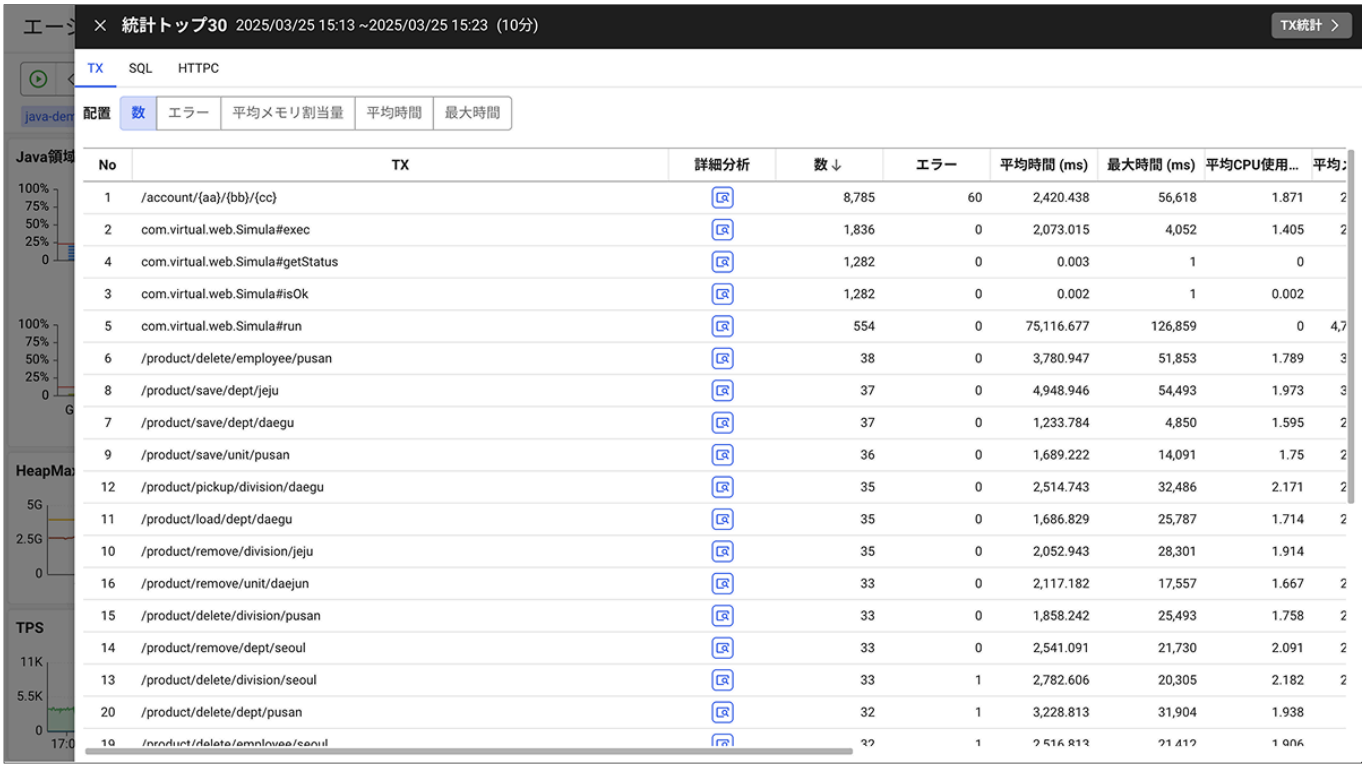
Runnableスレッドタブで現在JVMでRunnable状態のスレッドリストをリアルタイムで確認できます。 CPU使用量が高くなったり、レスポンスが遅れたりする場合、どのスレッドが問題を引き起こすかを把握するのに役立ちます。

過去時点照会

タイムセレクターは、過去の特定時点を選択した後、その時点のリソースの状態を確認できます。 最大3時間前のリソース状態を照会し、その時点で発生したトランザクション、SQL、HTTP呼び出し情報を分析できます。 これにより、CPUやメモリ使用量が急騰したり、TPSが急減した時点の原因を効果的に追跡できます。



過去の時点を照会する際、上段右側の**TX**、**SQL統計**、**HTTPC統計**ボタンをクリックすると、各上位30個の統計情報が確認できる**統計Top 30**パネルを照会できます。



TX統計

TXタブで特定時点に行われたトランザクションをレスポンスタイム、実行回数などの条件でソートし、上位30個のリストを照会できます。 レスポンスタイムが長いか、実行回数が多い項目を優先的に確認できます。

平均メモリ割当量照会設定

過去時点のトランザクション統計で**平均メモリ割当量**項目が表示されない場合は、次のオプションを活性化有無を確認します。 ただし、このオプションは平均TPSが高いアプリケーションでは性能の影響が発生する可能性があるため、**平均TPS 300 以下の環境**でのみ適用をお勧めします。

- **trace_malloc_enabled** Boolean

  既定値 `false`

  トランザクションごとの平均メモリ割当量を収集します。

SQL統計

SQLタブで特定の時点で実行されたSQLクエリを実行回数、全体時間などのソート条件によって上位30個のリストを照会できます。 DB負荷が増加した場合、同一時点で実行されたSQLクエリの中で高いコストを占めるクエリを把握できます。

HTTPC統計

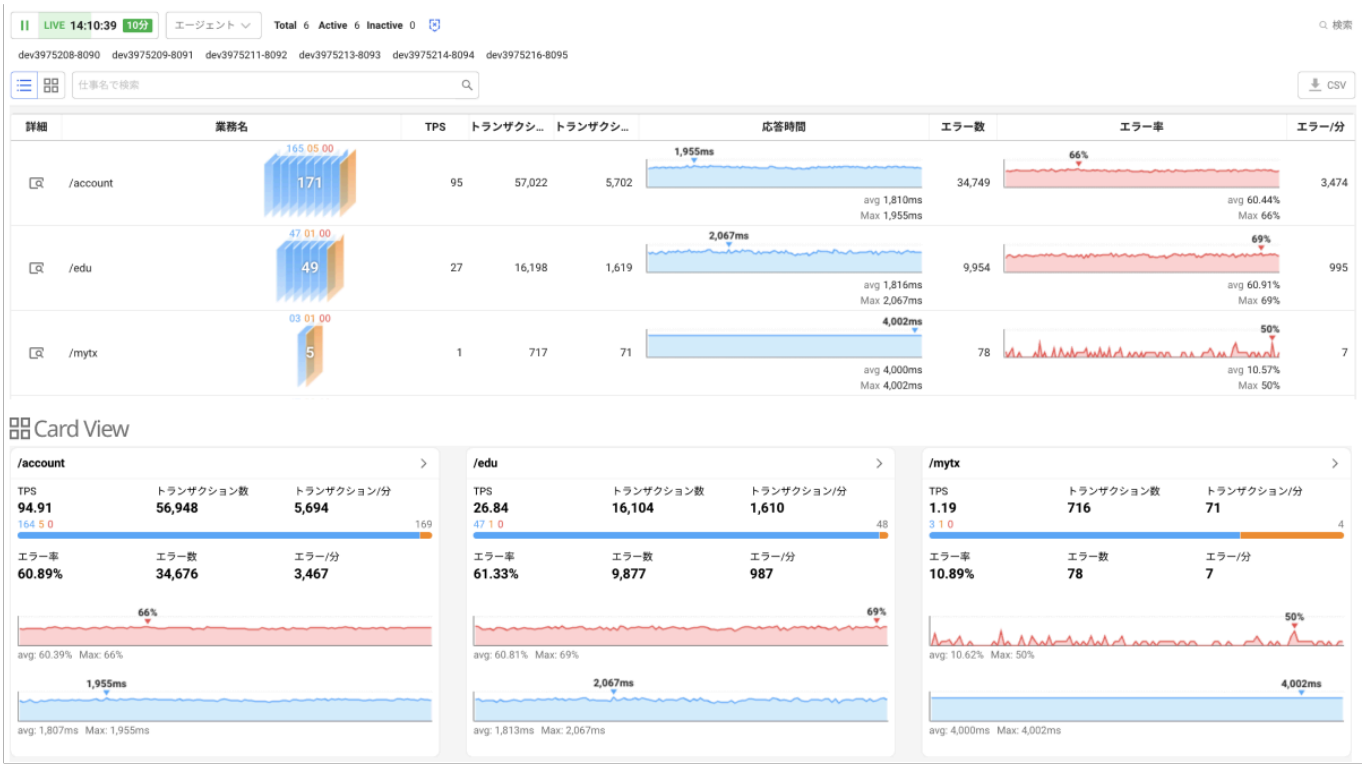
HTTPCタブで外部HTTP呼び出し情報をソートして上位30個のリストを照会できます。 外部APIや内部HTTPリクエストが多い時点でボトルネック発生有無を確認できます。

ビジネスダッシュボード

ホーム画面 > プロジェクト選択 > **ダッシュボード** > **ビジネスダッシュボード**

ビジネスダッシュボードは、アプリケーションのビジネス単位（App Context）ごとのパフォーマンスをモニタリングできるダッシュボードです。
URL パターンやパスに基づいて定義された各ビジネス項目に対し、リアルタイムおよび過去のパフォーマンスデータを確認・分析できます。
TPS、応答時間、エラー率などの主要なパフォーマンス指標をビジネス単位で把握することで、特定業務やサービスの状態をすばやく把握できます。

画面構成



時間選択

画面左上のタイムセレクターを使用して、表示する時間範囲を設定します。 画面の左上のタイムピッカーの右側には、プロジェクトに関連したエージェントの状態を確認できる情報が表示されます。 この機能で、モニタリング対象サーバーの動作有無がすぐに確認できます。

- **リアルタイムモード**：5秒ごとに自動更新され、直近10分のデータを表示
- **過去モード**：最大1ヶ月前までのデータを表示可能
- **データ集計単位**：
  - 3時間未満：Rawデータを使用
  - 3時間〜3日：5分単位の統計データ
  - 3日以上：1時間単位の統計データ

エージェント選択

時間選択の下には、プロジェクトに所属するエージェントリストが表示されます。
エージェントを選択すると、青いタグが付き、選択したエージェントまたはグループに関するデータのみがフィルタリングされます。

> ⓘ エージェント選択に関する詳細は、次のドキュメントを参照してください。

ビューモード変更

ビジネスダッシュボードは、ユーザーの好みに応じて2種類のビューモードで表示を切り替えられます。
画面左上の ☰（テーブルビュー）または ⊞（カードビュー）をクリックして切り替えます。

- ☰ **テーブルビュー**

  すべてのビジネス項目を表形式で一覧表示し、比較や並べ替えがしやすくなります。
  各項目の主要なパフォーマンス指標（TPS、応答時間、エラー率など）を一目で確認できるため、全体的な状況を迅速に把握するのに便利です。

- ⊞ **カードビュー**

各ビジネス項目をカード形式で視覚的に表示します。
ビューの列数は1列から最大4列まで選択でき、モニターサイズや作業環境に応じて柔軟に調整できます。
カードビューは特定のビジネスの状態を個別に集中して確認したり、ダッシュボードとして使用するのに適しています。

並び替えオプション

テーブルビューとカードビューの両方で、業務別データを並び替える指標を選択できます。

業務名検索

検索エリアに業務名を入力して、特定のビジネスを素早く検索できます。

業務名の別名設定

ボタンをクリックすると、URLプレフィックスとして表示される業務名に別名を設定できるドロワーが開きます。

- 左の列に別名を設定したい業務名（URLプレフィックス）を入力
- 右の列に対応する別名を入力
- **項目追加**ボタンで項目を追加、**削除**ボタンで項目を削除
- 下部の**適用**ボタンをクリックすると設定が反映されます

CSVダウンロード

画面右上の検索エリア横にある  をクリックすると、現在表示中のビジネスのパフォーマンスデータを CSV ファイルとしてダウンロードできます。

ビジネスパフォーマンス概要

各ビジネス単位ごとに以下の主要なパフォーマンス指標が確認できます。
リアルタイム状態と設定した期間の累積統計が提供されます。
アクティブTX、**応答時間**、**エラー率** はグラフ形式で視覚的に表示されます。

- **アクティブTX**：現在処理中のトランザクション数をスピードメーター形式で表示

- **TPS (Transactions Per Second)**：1秒あたりの処理数（性能指標）
- **トランザクション数**：集計期間中に処理された総トランザクション数
- **トランザクション/min**：1分あたりの平均トランザクション数
- **応答時間**：平均および最大応答時間
- **エラー数**：発生したエラーの総数
- **エラー率（%）**：全トランザクション中でエラーが発生した比率
- **エラー/min**：1分あたりの平均エラー数

ビジネス詳細分析

Top トランザクション統計表示

各ビジネスの主要トランザクションに関する統計を確認できます。
テーブルビューまたはカードビューで、詳細表示アイコンをクリックすると、右側に上位30件のトランザクション統計がドロワーで表示されます。

> **詳細表示ボタンの案内**
>
> - テーブルビュー：テーブル左端の
> - カードビュー：カード右上の矢印(>)

1. テーブルまたはカードから詳細表示アイコンをクリック。
2. 右側にドロワーが開き、Top 30 トランザクションが表示されます。
3. トランザクション一覧では以下を確認できます:
    - トランザクション名
    - 詳細分析（ クリックで詳細画面へ）
    - 呼び出し回数
    - エラー回数
    - 平均時間（ms）

- 最大時間（ms）
- 平均 CPU 使用時間
- 平均メモリ割当量
- SQL 件数
- 平均 SQL 実行時間（ms）
- SQL フェッチ件数
- 平均 SQL フェッチ件数
- HTTP 呼び出し件数
- 平均 HTTP 呼び出し時間（ms）

4. 上部の検索窓で特定のトランザクションを検索可能。
5. 各トランザクションの詳細分析 をクリックして、さらに詳しい情報を確認できます。
6. 選択した時間範囲内でURL単位のトランザクションイベントが発生した場合、設定されたレベルが表示されます。
   レベルが（警告）または（重大）の場合は、Health列のアイコンをクリックしてイベント履歴を確認できます。

イベント履歴

選択した時間範囲内で**業務別**のトランザクションイベントが発生した場合、設定されたレベルが表示されます。アイコンをクリックすると、イベント履歴を確認できます。

- イベント名、発生時刻、メッセージなどを確認できます。
- 検索機能を使って特定のイベントを探せます。

エージェント設定

ビジネスダッシュボードを使用するには、エージェントの設定が必要です。

対応エージェントバージョン

- **Java**: 2.2.40 以上

必須設定

```
# 機能全体の有効化
app_context_enabled=true
```

選択設定

```
# パーサーの選択 (default/prefix/match)
# default: 最初のパスセグメントを抽出
# prefix: 接頭辞マッチングパーサー（最も柔軟）
# match: 完全一致のみのパーサー
app_context_parser=default

# パーサーの内部キャッシュと設定を強制的にリセットするカウンター
app_context_parser_reset=0

# URLから抽出するパスセグメントの深さを指定（default パーサーで 1 または 2 を使用）
app_context_path_depth=1

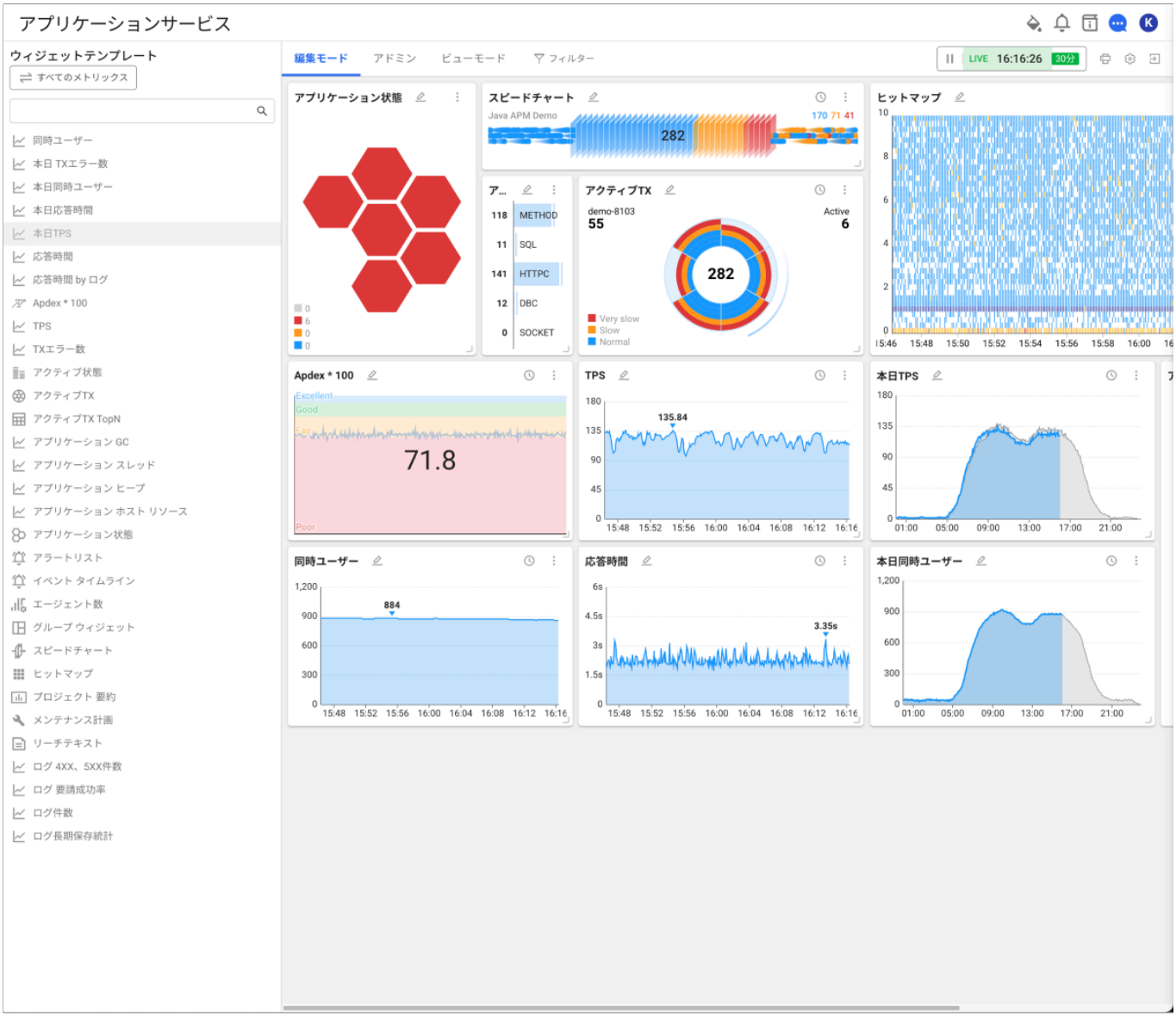
# 監視対象のパスとその名前を定義するマッピング設定（カンマ区切りで使用）
app_context_path_set=name1@url1,name2@url2,url3,name4@url4
```

- ビジネスダッシュボードにデータが表示されない場合は、エージェントの設定を確認してください。
- エージェントのバージョンがサポートバージョンよりも古い場合は、エージェントのアップデートが必要です。
- ビジネスコンテキストは、URLパターンを基準に自動または手動で定義できます。

Flexボード

Flexボードはユーザー定義型の統合ダッシュボードです。アプリケーション、サーバー、データベース、コンテナなどのプロジェクトのデータを使用して、リアルタイムのダッシュボードが作成できます。

事前構成されたテンプレートで初期設定を迅速に完了し、希望する形式のダッシュボードを作成できます。データウィジェットを追加したりプロパティを修正したりして、必要な形式で表示することができ、フィルタリング機能によって監視対象を絞り込むこともできます。時間範囲を指定すれば特定の時点のデータを確認でき、補助チャートを活用することで多様な方法で分析できます。ダッシュボードはお気に入りに登録して簡単にアクセスでき、カスタマイズしたダッシュボードは他のアカウントにコピーして再利用することも可能です。



Flexボードは、次のアクセスパスによってさまざまな用途があります。

- ホーム画面 > **統合Flexボード**
  - ウィジェットを作成すると、表示可能なすべてのプロジェクトを選択オプションとして提供します。
  - ユーザーアカウントにダッシュボードが保存され、他のユーザーにコピーする機能を使用して共有できます。
  - 個人アカウントダッシュボードで、権限による影響はありませんが、読み取り専用として共有されたダッシュボードの場合は修正できません。

- ホーム画面 > プロジェクト選択 > **ダッシュボード** > **Flexボード**
  - ウィジェットを作成すると対応するプロジェクトの情報が自動入力されます。
  - プロジェクトメンバーで作成したFlexボードは、自動共有されます。
  - プロジェクト**修正**権限、または**プロジェクトのFlexボードの修正**権限、**Site Admin**権限を持つユーザーが修正できます。

Flexボードの編集権限を持つユーザーは、以下の機能を利用できます。

- ダッシュボードをJSONファイルとしてエクスポート/インポート
- ダッシュボード内のデータリクエストおよびレスポンス内容を確認
- ウィジェット設定オプションをJSON形式で参照および修正

プロジェクト内の**Flexボード**メニューでは、ダッシュボードの変更権限のあるユーザーのみが**編集モード**および**アドミン**モード、**フィルター**機能にアクセスできます。

- アクセス可能なメンバー権限
  - プロジェクト**修正**権限
  - **プロジェクトのFlexボードの修正**権限
  - **Site Admin**権限
- **ビューモード**および**フィルター**機能の詳細については、次の文書を参照してください。
- メンバー権限の詳細については、次の文書を参照してください。

Flexボード使用ガイド

> ダッシュボードはメトリクスウィジェットを含むさまざまなウィジェットテンプレートを活用して構成できます。
> 各テンプレートの特徴と使用方法については、ウィジェットテンプレート ドキュメントで詳しく確認できます。

事前テンプレートを適用する

事前構成テンプレートで初期設定を素早く進め、必要に応じて修正して活用してください。

1. **Flexボード** メニューを選択します。
2. 左側の **テンプレート** 一覧から使用するテンプレートを選択します。
3. **Flexボード管理** ウィンドウが表示されたら、**名前変更** に希望の名前を入力します。
   - **レイアウト** でウィジェットを選択し、ドラッグして希望の位置へ移動できます。
4. **保存** ボタンをクリックします。選択したテンプレートを基準に Flex ボードを作成します。
   - 作成したボードは **Flexボード** のボード一覧で確認できます。

Flex ボードを作成する

Flex ボードを作成し、ウィジェットを配置して自分だけのダッシュボードを作れます。

1. **Flexボード** で右上の **ダッシュボード生成** ボタンをクリックします。
2. **ダッシュボード生成** ウィンドウが表示されたら、入力ボックスにダッシュボードの **名前** を入力します。
3. ウィジェットの配置方式を選択し、**ダッシュボード生成** ボタンをクリックします。
   - **固定レイアウトFlexボード**: ピクセル基準の自由配置
   - **レスポンシブFlexボード** : ブラウザサイズ基準のグリッドレスポンシブレイアウト

（グリッドを設定してから **ダッシュボード生成** ボタンをクリック）

4. **ウィジェットテンプレート** 選択画面で、ダッシュボードに配置するウィジェットを選択します。
   - メトリクスウィジェットを追加する場合は、**ウィジェットテンプレート** の右側にある **すべてのメトリクス** ボタンをクリックします。

   > ⓘ
   > - **ウィジェットテンプレート**: 一般的な監視状況で重要となる指標を要約し、事前定義されたウィジェット一覧から選択できます。
   > - **すべてのメトリクス**: ユーザーのプロジェクトで収集中のすべてのメトリクスデータを基準にウィジェットを作成します。メトリクスウィジェットの詳細は次のドキュメントを参照してください。

5. 配置したウィジェットの位置を移動したりサイズを調整します。
   - ウィジェット上部で十字カーソルが表示されたら、クリックしたままドラッグして位置を移動します。
   - ウィジェット右下で矢印カーソルが表示されたら、クリックしたままドラッグしてサイズを調整します。
6. 追加するウィジェットの配置が完了したら、**ビューモード** を選択してレイアウトを確認します。レイアウトを再編集する場合は、**編集モード** を選択してウィジェット配置を変更します。
7. すべての工程が完了したら、右上の（**リストへ**）アイコンをクリックします。
   - **Flexボード** の **ダッシュボードリスト** で作成したボードを確認できます。作成したボードを選択して新しい Flex ボードを確認してください。

> ⓘ お客様のご要望を反映してダッシュボードテンプレートやウィジェットの追加を検討しています。必要なテンプレートまたはウィジェットがあれば support@whatap.io までご連絡ください。

Flex ボードを編集する

> ⓘ **Flex ボード編集権限**

- **ホーム** > **統合Flexボード**: 個人アカウントのダッシュボードで権限の影響はありませんが、閲覧専用で共有されたダッシュボードは編集できません。
- **ホーム** > **プロジェクト選択** > **ダッシュボード** > **Flexボード**: プロジェクトの編集権限、プロジェクトのフレックスボード編集権限、または Site Admin 権限のユーザーが編集できます。

メンバー権限の詳細は次のドキュメントを参照してください。

1. **Flexボード** の **ダッシュボードリスト** で、編集アイコンをクリックします。画面右側に **Flex ボード管理** ウィンドウが表示されます。
2. **Flex ボード管理** ウィンドウで、名前、プロジェクト、レイアウト、widget.json、option.json を設定します。
   - **名前変更**: ダッシュボード名を編集します。
   - **プロジェクト**: ダッシュボードのウィジェットがデータを取得するプロジェクトを選択します。

     プロジェクト オプションは **ホーム** > **統合Flexボード** メニューでのみ設定できます。
     統合Flexボード から入った場合は、ウィジェットごとにデータ取得元のプロジェクトを選択できます。

     1. ウィジェット右上の ＋ ボタンをクリックし、目的のプロジェクトを検索して単一または複数選択します。
     2. プロジェクトを選択したら **適用** ボタンをクリックします。

   - **レイアウト**
     - ウィジェットをドラッグして位置を移動します。
     - ウィジェット右下をマウスでクリックしたままドラッグしてサイズを調整します。
     - ウィジェットを削除するには、ウィジェット右上の アイコンをクリックします。
   - **ウィジェット.json**: ダッシュボードのウィジェット設定を JSON 形式で読み込みます。 アイコンをクリックすると JSON をコピーします。
   - **オプション.json**: ダッシュボードのオプション設定を JSON 形式で読み込みます。 アイコンをクリックすると JSON をコピーします。

Flex ボードを削除する

1. **Flexボード** > **ダッシュボードリスト** で、削除する項目の アイコンをクリックします。
2. 確認メッセージで **削除** ボタンをクリックします。

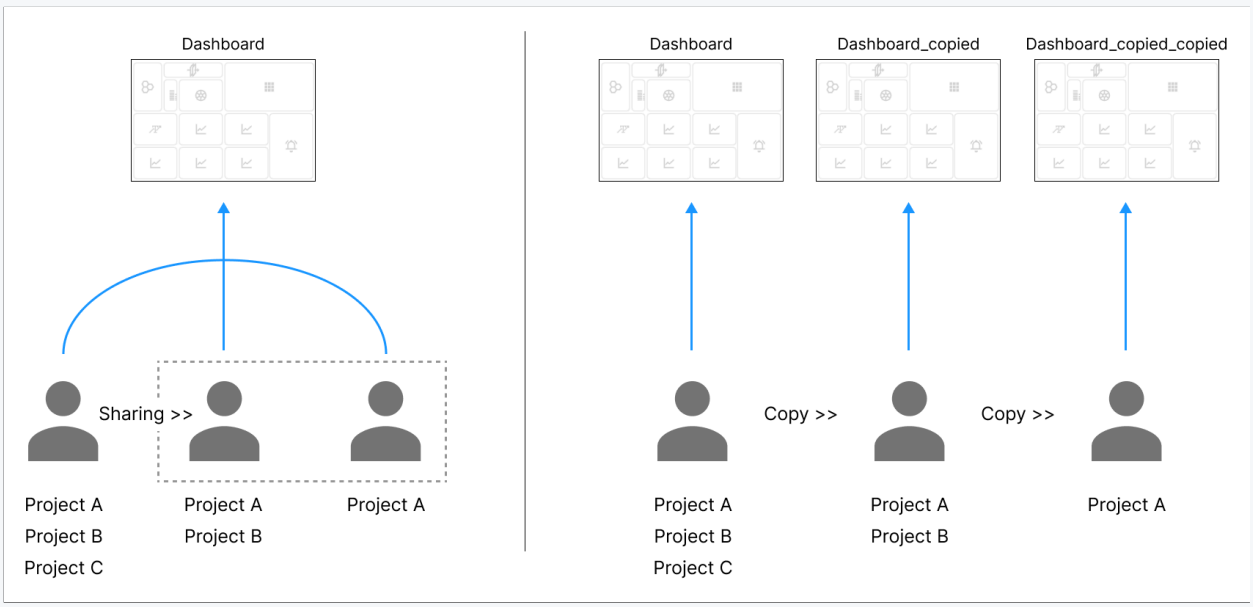
ダッシュボードの削除は所有者と共有を受けたユーザーで動作が異なります。

- ダッシュボード所有者がダッシュボードを削除すると、共有されたすべてのユーザーアカウントでもそのダッシュボードが削除されます。
- 共有を受けたユーザーがダッシュボードを削除すると、そのユーザーの **ダッシュボードリスト** のみから削除されます。元のダッシュボードは維持されます。

Flex ボードを共有する

Flex ボードの共有機能は **ホーム** > **統合Flexボード** メニューでのみ使用できます。
プロジェクトの **Flexボード** メニューでは提供されません。

共有とコピーの違い



- 共有: 共有されたメンバーは同じダッシュボードを閲覧します。
- コピー: ダッシュボードを他のメンバーまたは自分にコピーすると、コピーされた別個のダッシュボードを閲覧します。
- **読取り専用** で共有/コピーされたダッシュボードは編集できません。**編集モード** で共有/コピーされた場合のみ編集できます。
- **読取り専用** で共有を受けたメンバーも、そのダッシュボードを **閲覧専用で再共有** したり **コピー** できます。
- ダッシュボードに含まれるプロジェクトのうち **少なくとも1つ** のプロジェクトに所属しているメンバーにのみ共有できます。共有またはコピー先のメンバーがプロジェクトに所属しているか確認してください。



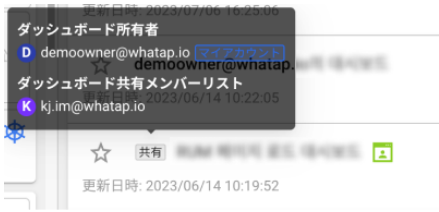
統合Flexボード で作成した Flex ボードを他のメンバーと共有できます。所有者が共有されたダッシュボードを編集すると、共有を受けたすべてのメンバーに同じ変更が反映されます。

1. **統合Flexボード** の **ダッシュボードリスト** で、共有するダッシュボードの アイコンをクリックします。
2. **ダッシュボード共有** ウィンドウが表示されたら、共有するメンバーを1名または複数名選択するか、アカウントを直接入力します。
    - **読取り専用** を選択すると、共有を受けたメンバーはダッシュボードを編集できません。

- 共有対象は、ダッシュボードに含まれるプロジェクトのうち少なくとも1つのプロジェクトに所属している必要があります。

3. **共有** ボタンをクリックします。

- 共有された項目には **共有** タグが表示されます。**共有** タグにマウスを載せると、共有したメンバー情報を確認できます。
- 閲覧専用で共有された項目には **読取り専用** タグが表示されます。

ダッシュボードを共有で受け取った場合	ダッシュボードを共有した場合（ダッシュボード
	

ダッシュボード共有の条件

- ダッシュボード共有機能は **ホーム** > **統合Flexボード** のみで提供します。
- だれでも自分のダッシュボードを共有でき、共有で受けたダッシュボードを再共有やコピーできます。
- **読取り専用** で受けたダッシュボードは編集できませんが、**編集モード** で受けた場合は編集できます。
- **所有者** がダッシュボードを削除すると、共有を受けたすべてのアカウントで該当ダッシュボードが削除されます。
- **共有を受けたユーザー** が削除した場合は、そのユーザーの **ダッシュボードリスト** のみから削除され、元のダッシュボードは維持されます。
- 共有で受けたダッシュボードに含まれるプロジェクトのうち、**ビュー** 権限がないプロジェクトはデータを参照できません。

Flex ボードをコピーする

ユーザーは Flex ボードをコピーして他のメンバーに渡せます。

1. **統合Flexボード** の **ダッシュボードリスト** で、 アイコンをクリックします。
2. コピーオプションを選択します。
   - **自分にコピー**: 自分のアカウントにコピーします。

     a. `_copied` 接尾辞が付いた項目が **ダッシュボードリスト** に追加されます。

   - **他人にコピー**: 他のメンバーのアカウントにコピーします。

     a. ＋ **アカウント** をクリックします。
     b. コピー先メンバーのメールを選択し、**ダッシュボードコピー** ボタンをクリックします。
     c. そのメンバーの **ダッシュボードリスト** に `_copied` 接尾辞が付いた項目が追加されます。

JSON ファイルで共有する

Flex ボードの設定を JSON ファイルとして他のメンバーに渡したり、他のメンバーの設定を取り込めます。

エクスポート

1. **統合Flexボード** またはプロジェクトの **Flexボード** メニューに移動します。
2. **ダッシュボードリスト** で ボタンをクリックします。
3. エクスポートオプションを選択します。
   - **プロジェクト情報を含む**: ダッシュボードに設定されたプロジェクト情報を JSON ファイルに含めて共有
   - **プロジェクト情報を含まない**: ダッシュボードに設定されたプロジェクト情報を JSON ファイルに含めずに共有
4. JSON ファイルをダウンロードし、共有先のメンバーに渡します。

インポート

1. **統合Flexボード** またはプロジェクトの **Flexボード** メニューに移動します。
2. 画面右上の **インポート** ボタンをクリックします。
3. ダウンロードした JSON ファイルを選択します。

> この機能は、プロジェクトの **編集** 権限を持つメンバーのみ利用できます。

Flex ボードの画面モード

Flex ボードの画面モードに応じて提供機能が異なります。

> **アクセス可能なメンバー権限**
>
> プロジェクト内の **Flexボード** メニューでは、**編集権限** のあるユーザーのみが **編集モード**、**アドミン**、**フィルター** 機能にアクセスできます。メンバー権限の詳細は次のドキュメントを参照してください。
>
> - **プロジェクト編集** 権限
> - **プロジェクトのFlexボードの修正** 権限
> - **Site Admin** 権限

モード種類

Flexボード は画面モードにより設定できる機能が異なります。

- **編集モード**: ウィジェットテンプレート一覧からレイアウトにウィジェットを配置し、ウィジェットの編集、削除、位置調整ができます。
- **アドミン**: ウィジェットの詳細プロパティを JSON 形式で管理できます。レイアウトに配置したウィジェット右上の 🔧 ボタンを選択します。
- **ビューモード**: ウィジェットの配置、サイズ調整、詳細オプションなどの設定を完了後、ダッシュボードのように各ウィジェットのデータをリアルタイムで監視できます。

共通機能

- データフィルタリング
- リアルタイムデータの参照範囲設定
- 印刷モード
- Flex ボード管理
- データ結合オプション

データフィルタリング

データフィルタリングで監視対象を絞り、効率的な Flex ボードを構成できます。

1. 画面上部の **フィルター** タブをクリックします。
2. 画面右に表示される **フィルター** ウィンドウでフィルタリング対象を選択します。すべて選択するには **すべて選択** を選択します。
   - **エージェント** オプション: エージェント種別またはアプリケーション一覧を選択、**検索** 機能あり
   - **メトリクス** オプション: プロジェクトまたはエージェント種別を選択、**+ And** 条件の追加や、**含む**、**除外** による文字列ベースの自動選択が可能
3. フィルタリング対象を選択したら、**適用** ボタンをクリックします。
4. よく使う条件は **フィルター保存** ボタンで保存します。

   a. フィルター名を入力し、**保存** を選択します。**保存フィルター** は **フィルター** ウィンドウ下部に表示されます。

フィルタリング条件

フィルタリング条件はエージェントタイプとメトリクスタイプに分かれます。
エージェントは基本的な監視対象のみ選択肢として提供し、メトリクスはプロジェクトで収集したすべてのメトリクスデータのタグを選択肢として提供します。メトリクスタイプは、非エージェント形式でデータを収集し、その後フィルタリングを適用する場合に有用です。メトリクスの詳細は次のドキュメントを参照してください。

タイプ	エージェント	メトリクス
選択オプション	WhaTap プロジェクトの監視対象 (アプリケーション、サーバー、データベース、 コンテナ、Kubernetes の POD と NODE)	エージェントを含むすべてのメトリクスタグ
適用範囲	すべてのウィジェット	メトリクスデータを使用するすべてのウィジェット
フィルタリング方式	チェックボックス	**一致, 含む, 除外**

参照時間を設定する

Flexボード の画面右上の時間セレクターで、ウィジェットのデータ参照時間を設定できます。

- ||: リアルタイムモードを停止して特定の参照時間に設定
- ⊙: リアルタイムモードに戻す

印刷する

Flex ボード画面を PDF で保存して印刷できます。

1. Flex ボード画面の右上にある 🖶 アイコンをクリックして印刷モードに切り替えます。
2. オプションを設定します。
   - **フィルターオプション表示**: 画面のフィルターオプションを表示/非表示に切り替え
   - **{mode}方向に切換え / {mode}方向に切換え**: 画面を横向きまたは縦向きに変更
3. 完了後、**PDFダウンロード** を選択します。
4. 印刷モードを終了するには、右上の ✕ **印刷モード終了** をクリックします。

Flex ボード管理

Flex ボードのレイアウト変更、ウィジェットの位置移動やサイズ調整、名称変更などの管理機能を提供します。 **Flex ボード管理** 機能は、**ダッシュボード** > **Flexボード** メニューの **ダッシュボードリスト** で ✎ アイコンをクリックして入

ることもできます。

1. Flex ボード画面の右上の ⚙ アイコンをクリックすると、**Flexボード管理** ウィンドウが開きます。
   - **名前変更**: Flex ボード名の変更
   - **レイアウト**: ウィジェットの位置・サイズ調整および削除
   - **他のユーザーの修正を許可しない**: 変更制限オプション
     - 編集権限のあるメンバーは編集・削除・変更が可能で、オプションを有効化すると **他のユーザーの修正を許可しない: On** に変更されます
   - **ウィジェット.json**: Flex ボードに含まれるウィジェットを JSON 形式で編集・管理
   - **オプション.json**: Flex ボードの設定を JSON 形式で編集・管理（プロジェクト情報、Flex ボードフィルター、時間選択などの設定値）

     > ⚠ **ウィジェット.json** と **オプション.json** 機能は、ダッシュボード管理者権限のユーザーにのみ提供します。

2. すべての設定を完了し、**保存** ボタンをクリックします。

ダッシュボードをお気に入りに登録する

作成した Flex ボードはお気に入りに登録して素早くアクセスできます。**ダッシュボード** > **Flexボード** の **ダッシュボードリスト** で、お気に入り登録するダッシュボードの ☆ ボタンをクリックすると、★ に変わります。

Flexボードウィジェット管理

Flexボードに配置したウィジェットを管理して、ダッシュボードをユーザーがカスタマイズします。

> ダッシュボードはメトリクスウィジェットを含むさまざまなウィジェットテンプレートを活用して構成できます。
> 各テンプレートの特徴と使用方法については、ウィジェットテンプレート ドキュメントで詳しく確認できます。

ウィジェット追加

作成済みのFlexボードにウィジェットを追加できます。

1. **Flexボード**メニューの**ダッシュボードリスト**で、ウィジェットを追加するダッシュボードを選択します。
2. 画面左上の**編集モード**を選択します。
3. 左側の**ウィジェットテンプレート**一覧から目的のウィジェットを選択します。メトリクスウィジェットを追加するには、**すべてのメトリクス**を選択してから、目的のウィジェットをクリックします。

- ホーム画面の**統合Flexボード**を選択し、ウィジェットを追加した場合は、**プロジェクト選択**ボタンをクリックして、収集対象のプロジェクトを選択します。
- 追加するウィジェットがある場合は、同じプロセスを繰り返してください。

4. すべての手順が完了したら、画面上部の**ビューモード**を選択してレイアウトを確認し、(**リストへ**)ボタンをクリックします。

ウィジェットの属性変更

Flexボードメニューの**ダッシュボードリスト**で、ウィジェットのプロパティを変更するボードを選択します。**編集モード**に入ります。

ウィジェット名の変更

1. 名前を変更するウィジェットの名前の右側の✎ボタンをクリックします。
2. ウィジェットの**題名**、**フォントサイズ**、**色**を設定し、**保存**ボタンを選択します。
   - 保存する際に、**すべてのウィジェットに適用**トグルボタンの機能を活性化すると、フォントサイズと色が他のウィジェットにも適用されます。
   - ウィジェットのタイトルをロックするには👁ボタンを選択し、**保存**ボタンをクリックします。

時間設定

1. 時間を設定するウィジェットの右上の🕒ボタンを選択します。
2. **時間選択**ウィンドウで時間を設定してください。
   - ダッシュボード画面で設定した時間を適用するには、**ダッシュボード設定に従う**をチェックします。ダッシュボードの時間設定は右上にあります。
   - 特定の時間を設定する場合は、**ダッシュボード設定に従う**のチェックを外します。〈または〉ボタンを使用して照会する時間を設定します。
3. 時間設定が終了したら✕ボタンをクリックします。

> ⓘ ウィジェットのデータタイプによっては、時間設定機能をサポートしていない場合があります。

出力データオプションの設定

チャート型ウィジェットの出力データオプションを設定して、ユーザーが必要なデータを選択できます。

1. ウィジェットの右上にある ⋮ アイコンをクリックして、出力データのオプションを設定します。
   - プロジェクト、エージェント、アプリケーションなどを選択してデータをフィルタリングできます。
   - 個々のデータまたはマージされたデータを選択して、チャートに表示されるグラフのタイプを変更できます。
   - ⋮ > ⚙を選択します。**チャート設定**画面で詳細オプションを変更できます。設定できるオプションは次

のとおりです。

- **チャートタイプ**：集計する現在の数値をチャートに表示するオプションです。
- **y軸 チャート最大値**：Y軸にチャートの最大値を**自動**または**固定**で選択できます。
- **水平線設定**：**追加**ボタンを選択し、しきい値または範囲を水平線として表示できます。複数を追加して適用できます。水平線を追加した後は、**適用**ボタンを選択してチャートに反映してください。
- **チャートの最大値を表示**：チャートに表示されるグラフの最大値を表示します。
- **サブチャート**：ウィジェットのメインチャートにサブチャートを追加できます。サブチャートの種類を選択し、レイアウトとサイズ、データのマージ有無を選択します。他のウィジェットにも適用する場合は、**すべてのウィジェットに適用**ボタンを選択します。サブチャートは、データをさまざまな方法で確認するとき便利です。

2. オプション設定がすべて完了したら、**チャート設定**画面の左上の×ボタンをクリックするか、**チャート設定**領域の外側をクリックします。

> ⓘ ウィジェットのデータタイプによって選択できるオプションは異なる場合があります。

メトリクスウィジェットデータの変更

メトリクスウィジェットのデータ照会の条件を変更できます。

1. **Flexボード** > **ダッシュボードリスト**で、ウィジェットデータを変更するダッシュボードを選択します。
2. **編集モード**で、変更したいウィジェットの右上にある ⋮ アイコンをクリックします。
3. <> アイコンをクリックすると、データ設定編集ウィンドウが表示されます。
4. 次を参照して設定を変更するには、適用アイコン**ウィジェットに適用**ボタンをクリックします。
   - ウィジェットを作成すると、デフォルトで生成されるメトリクスのオプションがタブに表示されます。
   - メトリクス オプションには、**カテゴリ**と**フィールド**、**タグ**が含まれます。
   - **カテゴリ**項目は現在、変更を加える機能を制限しています。
   - **フィールド**、**タグ**項目は複数選択できます。選択したタグオプションでチャートデータをグループ化します。
   - 🔍ボタンをクリックすると、元のデータを表示できます。

- **オブジェクトのマージ**および**時間マージ**項目の詳細については、データマージオプションを参照してください。

データマージオプション

Flexボードの**すべてのメトリクス**一覧に配置されたマトリクスウィジェットでは、データのマージオプションを提供します。 データのマージは、**オブジェクトのマージ**と**時間マージ**方法を提供します。

- **オブジェクトのマージ**

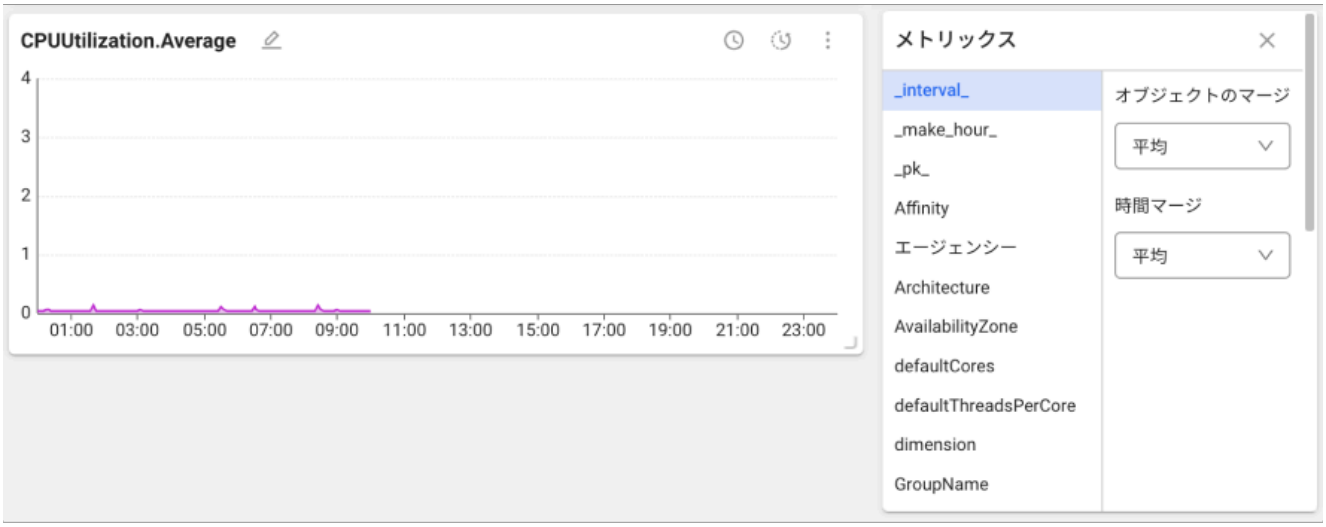
  左側で選択したタグ(例：エージェント名)に基づいてチャートデータをグループ化するために使用します。この場合、異なるフィールド(例：CPU)値を持つデータ間でタグが一致する場合、そのデータは結合されます。

- **時間マージ**

  主に検索時間が長い場合に使用します。ソースデータとフィールド値が同じデータを時間範囲に基づいて一定の時間間隔(5分または1時間)でデータを結合します。この結合方法は、時間結合と呼ばれます。

- **AWS CloudWatchメトリクス**

  AWSモニタリング指標の場合、統計手法はフィールド名の後にサフィックス(接尾辞) として追加されます。データマージの既定値は、このサフィックスを参照しています。



1. **編集モード**で**すべてのメトリクス**ボタンを選択し、**ウィジェットテンプレート**一覧を**すべてのメトリクス**一覧に変更します。
2. メトリクスウィジェットをレイアウトに配置します。

3. メトリクス ウィジェットの右上にある ⋮ ボタンを選択します。
4. タグ(例: エージェント名)オプションを選択すると、データ結合オプションが表示されます。ウィジェットのサイズによっては、メトリクス オプションの位置が異なる場合があります。

ウィジェットのコピー

1. ウィジェットの右上にある ⋮ ボタンをクリックします。
2. ボタンをクリックします。
3. **コピー**画面が表示されたら、＋ **ウィジェット追加**を選択します。
4. 一覧にあるウィジェットの名前を変更します。
5. **適用**ボタンを選択します。
6. **編集モード**画面でコピーしたウィジェットで確認できます。

ウィジェットの削除

1. ウィジェットの右上にある ⋮ ボタンをクリックします。
2. ✕ ボタンをクリックします。
3. 確認メッセージの画面が表示されると、**削除**ボタンを選択します。削除を取り消すには**キャンセル**ボタンをクリックします。

分析レポート

アプリケーション環境を様々な観点から分析できるメニューを提供します。 負荷が高かったり性能が低下した区間を把握できるし、多様な性能指標を提供して日別状況を確認できます。 また、性能測定のためのトランザクションを追跡・分析できます。

WhaTapが提供する分析機能により、主な性能を分析しながら問題点を把握し、発生する可能性のある障害を予防できます。

日別APP現況

アプリケーションの主な性能指標の1日間の推移を時間単位のチャートで確認できます。 詳細については、次の文書を参照してください。

性能推移

アプリケーションの主な性能指標の推移を時間帯を特定してチャートで表示するため、負荷が高かったり、性能が低下した区間を素早く把握でき、問題になった時点を特定できます。 詳細については、次の文書を参照してください。

メトリクスチャート

可視化されたチャートにより、メトリクスデータを照会できます。 詳細については、次の文書を参照してください。

メトリクス照会

カテゴリ化されたアプリケーションの性能指標をタグとフィールドベースで構成されたデータで照会できます。 詳細については、次の文書を参照してください。

スタック

トランザクションのスタック情報を収集し、実行中のメソッドの使用量の統計と、多く使用されるスタックに関する統計を提供します。 実行中のトランザクションのスタック情報により、長時間実行されたメソッドや、短時間実行されるが頻繁に実行されるメソッドを把握できます。 詳細については、次の文書を参照してください。

キューブ

5分単位で作成された性能統計をキューブ(Cube)と呼びます。 事後分析のための統計分析ツールで、応答時間やエラー件数、処理量などが高い時間帯を特定できるため、同時間帯の問題要素を把握したり、サービス状況をモニタリングしたりすることができます。 トラフィックの地理的分布、リソースの使用量を確認できます。 詳細については、次の文書を参照してください。

ヒットマップ

時間の経過とともに、ユーザーのリクエストに対する応答時間を分布図の形で表現したチャートを提供します。 分布図チャートの形態によりどのような障害が発生したかを特定できます。 詳細については、次の文書を参照してください。

マルチTX追跡

他のエージェントやプロジェクトと関連するトランザクションの間での呼び出しを追跡します。 システム内またはシステム間で発生するさまざまな呼び出し関係を一目で確認でき、どのような問題が発生したかを識別して改善できるようにトランザクションとトレース情報を提供します。 詳細については、次の文書を参照してください。

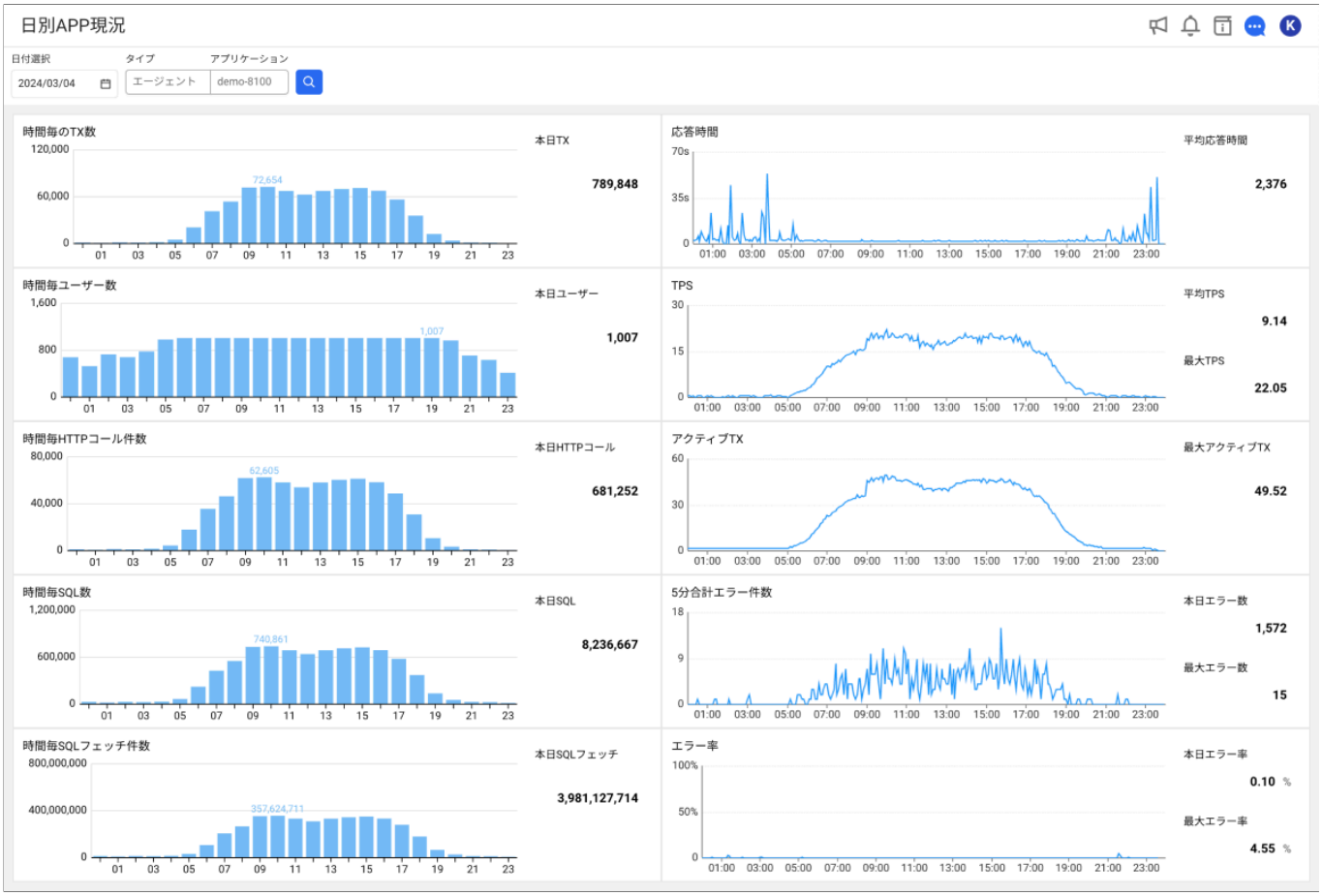
TX検索

特定の期間中の各トランザクションの性能属性に関する統計と実行状態を検索できます。 すべてのトランザクションを確認し、正常やエラー状態をすばやく確認できます。 提供される情報を通じて、どのようなトランザクションの使用量が多く、リソースを多く消費しているかがわかります。 トランザクションの属性に応じて様々なフィルタリングが可能で、特定の条件のトランザクションを見つけるのに役立ちます。 詳細については、次の文書を参照してください。

日別APP現況

ホーム画面 > プロジェクト選択 > **分析** > **日別APP現況**

アプリケーションの主な性能指標の1日間の推移を時間単位のチャートで確認できます。



1日間のアプリケーションの状態を確認できます。 検索条件(**日付選択**, **タイプ**, **アプリケーション**)を設定し 🔍 ボタンを選択します。 **タイプ**オプションの条件は次のとおりです。

分類	説明
エージェント	プロジェクトに追加されたアプリケーションエージェントの名前

分類	説明
エージェント種類	エージェント設定で `whatap.okind` オプションで分類されたグループ単位
エージェントノード	エージェント設定で `whatap.onode` オプションで分類されたグループ単位

提供するチャートは次のとおりです。 チャートの右側にある数値で、1日間の累積値、平均値または最大値を確認できます。

- **時間毎のTX数**
- **時間毎ユーザー数**

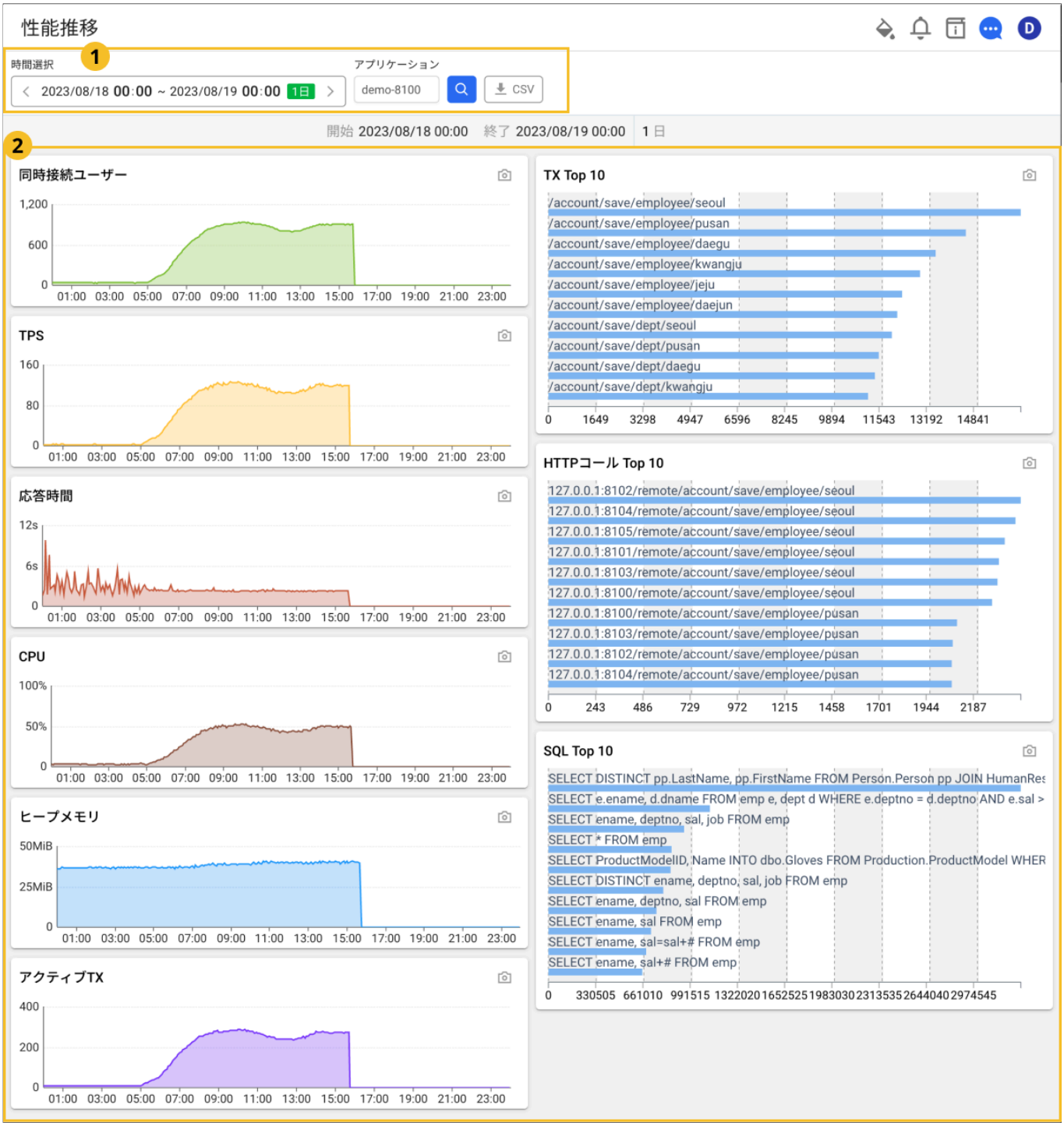
> ⓘ **本日ユーザー**は、一日間の累積されたユーザー数を重複なく合算した値です。 単に合算するのではなく**HyperLogLog**で算出します。 **HyperLogLog**は、非常に小さいメモリで集合の元素数を推定する確率的データ構造を意味します。

- **時間毎HTTPコール件数**
- **時間毎SQL数**
- **時間毎SQLフェッチ件数**
- **応答時間**
- **TPS**
- **アクティブTX**
- **5分合計エラー件数**
- **エラー率**

性能推移

ホーム画面 > プロジェクト選択 > **分析** > **性能推移**

WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**分析** > **性能推移**メニューを選択します。 照会を希望する特定の時間範囲内で、行われた性能に対する複数の指標を確認できます。


性能推移
1
時間選択
2023/08/18 00:00 ~ 2023/08/19 00:00 1日
アプリケーション
demo-8100
CSV
開始 2023/08/18 00:00
終了 2023/08/19 00:00
1 日
2
同時接続ユーザー
1,200
600
0
01:00 03:00 05:00 07:00 09:00 11:00 13:00 15:00 17:00 19:00 21:00 23:00
TPS
160
80
0
01:00 03:00 05:00 07:00 09:00 11:00 13:00 15:00 17:00 19:00 21:00 23:00
応答時間
12s
6s
0
01:00 03:00 05:00 07:00 09:00 11:00 13:00 15:00 17:00 19:00 21:00 23:00
CPU
100%
50%
0
01:00 03:00 05:00 07:00 09:00 11:00 13:00 15:00 17:00 19:00 21:00 23:00
ヒープメモリ
50MiB
25MiB
0
01:00 03:00 05:00 07:00 09:00 11:00 13:00 15:00 17:00 19:00 21:00 23:00
アクティブTX
400
200
0
01:00 03:00 05:00 07:00 09:00 11:00 13:00 15:00 17:00 19:00 21:00 23:00
TX Top 10
/account/save/employee/seoul
/account/save/employee/pusan
/account/save/employee/daegu
/account/save/employee/kwangju
/account/save/employee/jeju
/account/save/employee/daejun
/account/save/dept/seoul
/account/save/dept/pusan
/account/save/dept/daegu
/account/save/dept/kwangju
0 1649 3298 4947 6596 8245 9894 11543 13192 14841
HTTPコール Top 10
127.0.0.1:8102/remote/account/save/employee/seoul
127.0.0.1:8104/remote/account/save/employee/seoul
127.0.0.1:8105/remote/account/save/employee/seoul
127.0.0.1:8101/remote/account/save/employee/seoul
127.0.0.1:8103/remote/account/save/employee/seoul
127.0.0.1:8100/remote/account/save/employee/seoul
127.0.0.1:8100/remote/account/save/employee/pusan
127.0.0.1:8103/remote/account/save/employee/pusan
127.0.0.1:8102/remote/account/save/employee/pusan
127.0.0.1:8104/remote/account/save/employee/pusan
0 243 486 729 972 1215 1458 1701 1944 2187
SQL Top 10
SELECT DISTINCT pp.LastName, pp.FirstName FROM Person.Person pp JOIN HumanRes
SELECT e.ename, d.dname FROM emp e, dept d WHERE e.deptno = d.deptno AND e.sal >
SELECT ename, deptno, sal, job FROM emp
SELECT * FROM emp
SELECT ProductModelID, Name INTO dbo.Gloves FROM Production.ProductModel WHER
SELECT DISTINCT ename, deptno, sal, job FROM emp
SELECT ename, deptno, sal FROM emp
SELECT ename, sal FROM emp
SELECT ename, sal=sal+# FROM emp
SELECT ename, sal+# FROM emp
0 330505 661010 991515 1322020 1652525 1983030 2313535 2644040 2974545

1 性能推移の照会条件

- タイムピッカーを通じて、照会したい時間範囲を選択できます。 基本照会期間は1日です。
  - 照会範囲が3時間以下の場合、Rawデータに基づいて5秒単位の推移を確認できます。
  - 照会範囲が3日以内の場合、5分統計を使用して5分単位の推移を確認できます。
  - 照会範囲が3日を超える場合、1時間統計を使用して1時間単位の推移を確認できます。
- アプリケーション選択を通じて、全体または照会したい特定のアプリケーションを選択できます。
- CSVダウンロードアイコンを選択して、画面に表示された一覧データをCSVファイルでダウンロードできます。

2 性能推移チャート

- 指定した照会時間範囲をチャート上段で確認できます。
- チャート領域で**同時接続ユーザー**、**TPS**、**応答時間**、**CPU**、**ヒープメモリ**、**アクティブTX**、**TX Top 10**、**HTTPコール Top 10**、**SQL Top 10**などの情報を確認できます。
- **スナップショット**アイコンを選択して、ウィジェットのオプションを除いたチャートをスナップショットできます。

同時接続ユーザー

ブラウザIPを基準としたリアルタイムユーザー数を表示します。

- Rawデータを使用する場合、10秒間にリクエストを発生させたユニークなブラウザIP数を集計します。
- 5分統計を使用する場合、直近5分間にリクエストを送信したユニークなブラウザIP数を集計します。
- 1時間統計を使用する場合、直近1時間にリクエストを送信したユニークなブラウザIP数を集計します。

TPS

TPS(Tansaction Per Second)は、毎秒処理されたトランザクション数を意味します。

応答時間

リクエストに対する応答時間の平均を意味します。 平均が高い場合、特定の時点で遅いトランザクションが多く行われたか、またはトラブルが発生した可能性が高いです。

CPU

アプリケーションサーバーのCPU使用量を意味します。 使用量変化の推移が確認できます。

ヒープメモリ

モニタリング対象プロセスの**ヒープメモリ**使用量を意味します。 時間によるメモリ使用量の変化を確認できます。

アクティブTX

アクティブTXとは、進行中のトランザクションを意味します。 アクティブTXの詳細については、次の文書を参照してください。

TX Top 10

トランザクション呼び出し統計を使用して、トランザクション呼び出しが多い上位10の一覧を提供します。

HTTPコール Top 10

HTTPC統計を使用して、HTTP Call URLの件数が多い上位10の一覧を提供します。

SQL Top 10

SQL統計を使用して、SQL呼び出しの件数が多い上位10の一覧を提供します。

スタック

ホーム画面 > プロジェクト選択 > **分析** > **スタック**

特定の時間帯の呼び出しスタックの頻度とメソッド、トランザクションに関する詳細情報が確認できます。**トップスタック**, **ユニークスタック**, **アクティブスタック**が照会できます。**トップスタック**は、呼び出し頻度の高いメソッドの使用量を示し、**ユニークスタック**は、同じ呼び出しスタックの頻度を分析します。**アクティブスタック**は、実行中のトランザクションのスタック情報を定期的に収集し、リアルタイムのパフォーマンス分析ができます。

> スタック分析機能は**Java**及び**Python**、**.NET**のモニタリングで提供します。

主な機能の案内

次の主な機能を利用できます。スタックの照会により、アプリケーションのパフォーマンスの問題をすばやく識別し、解決できます。特定メソッドの呼び出し頻度、呼び出し経路、応答時間などを確認し、性能向上のための改善策が用意できます。



- **スタックタイプの選択**

  トップスタック, **ユニークスタック**, **アクティブスタック**中から希望するスタックを選択して照会できます。

  - **トップスタック**：呼び出し頻度が最も高いスタック情報を提供します。詳細については、次の文書を参照してください。
  - **ユニークスタック**：実行されたメソッドのセットが同じ場合の統計情報を提供します。詳細については、次の文書を参照してください。
  - **アクティブスタック**:実行中のトランザクションのスタック情報を提供します。詳細については、次の文書を参照してください。

- **時間選択 / 開始日**：照会する時間を設定します。特定の時間帯のスタックを照会したり、目的の期間を設定できます。スタックの種類によっては、照会できる期間が制限されます。

  > ◦ **トップスタック**及び**ユニークスタック**を照会できる最大期間は3日です。

- **アプリケーション**：照会するエージェントを選択できます。特定のエージェントを選択でき、何も選択しないとプロジェクトに含まれるすべてのエージェントを対象に照会します。
- **カラム選択**：希望するカラム項目を表示または非表示します。
- **比較する**：同じフィルター条件で1日以内に照会する際、選択したスタックを別の時間帯と比較できます。この機能で、時間による呼び出しスタックの変化を把握し、性能のトレンドを確認できます。詳細については、次の文書を参照してください。
- ：各スタック タイプのテーブル リストからスタックまたはトランザクション項目を拡張すると、一番右側にスナップショット アイコンが表示されます。このアイコンを選択すると、スタックまたはトランザクション情報を画像ファイルに保存できます。

基本的な使い方

1. 画面から照会するスタックタイプを選択します。
2. **時間選択**または**開始日**で希望する照会時間を設定します。
3. **アプリケーション**で、照会するエージェントを選択します。何も選択しないと、エージェント全体を対象に照会します。
4. ボタンを選択します。

照会結果をスタックテーブルの一覧から確認できます。

トップスタック

トップスタックでは、呼び出し頻度が最も高いメソッドの情報を提供します。スタックの最上段に位置するメソッドの呼び出し頻度により、サービスに最も影響を与えるメソッドを素早く確認できます。メソッドの呼び出し頻度を把握し、CPUまたはメモリに負荷がかかる原因を分析できます。この機能で性能改善に必要な対策を講じられます。

WhaTapは、10秒(既定値) 間隔で収集したThread Stackを使用して、メソッドレベルの性能遅延を分析します。

st1

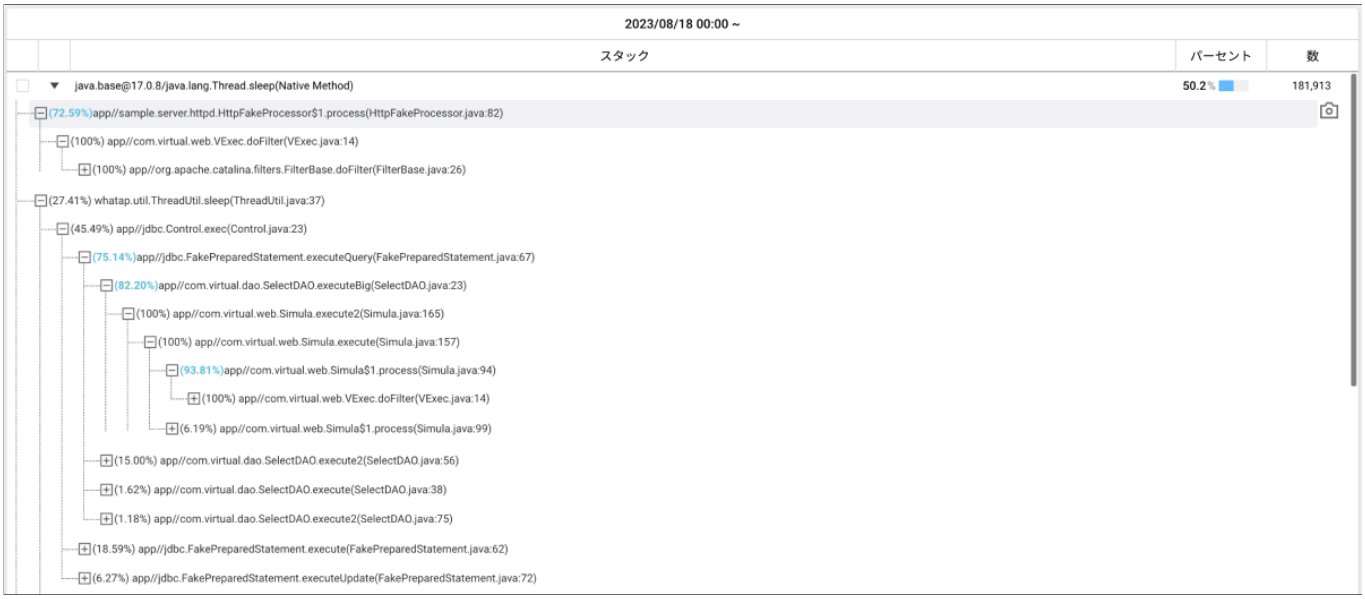
例）スタックの**トップライン**は socketRead0 です。

```
java.net.SocketInputStream.socketRead0(Native Method)
```

トップラインは、実行中のスレッドが該当メソッドを実行していることを意味します。途中でキャプチャされた可能性もありますが、確率的には各モジュールの処理時間の合計の比率でスタックに現れます。この**トップライン**メソッドの頻度を計算してメソッドレベルのパフォーマンスを判断できます。WhaTapは、**トップラインの頻度統計**を**トップスタック**(Top Stack)と呼びます。



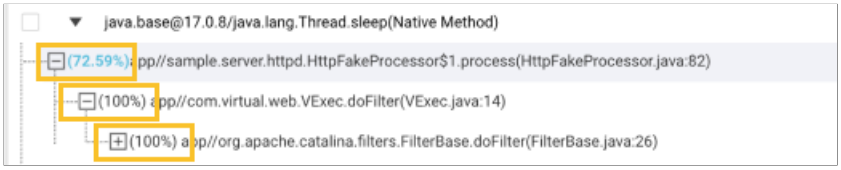
トップスタック分析で現れたメソッドをどのメソッドが呼び出したかを頻度で分析できます。Stack Traceの各ステップで、ステップとステップの間で呼び出した比率をパーセンテージで分析した情報を提供します。最上位ステップから呼び出したステップの頻度をパーセンテージで計算し、降順に並べ替えた結果を表示します。



各ステップ項目の一番左にある  ボタンを選択すると、そのステップを呼び出す上位ステップの呼び出し頻度をパーセンテージで提供します。

トップスタック統計は充分に多いデータで判断する必要があります。収集したスタックの数が10個未満の場合、統計的な意味を持つには不十分です。

トップスタックは、把握しづらい部分のチューニングポイントを見つけるのに役立ちます。呼び出し頻度が最も高いスタックは、アプリケーションサーバーで最も多くの応答遅延が発生していると判断できます。左に表示する比率は、アプリケーションサーバーのパフォーマンスへの影響度です。



安定したアプリケーションサーバーであっても、呼び出し頻度の高いスタックは性能の低下を引き起こす可能性があるため、そのクラスは注意深く見ることをお勧めします。

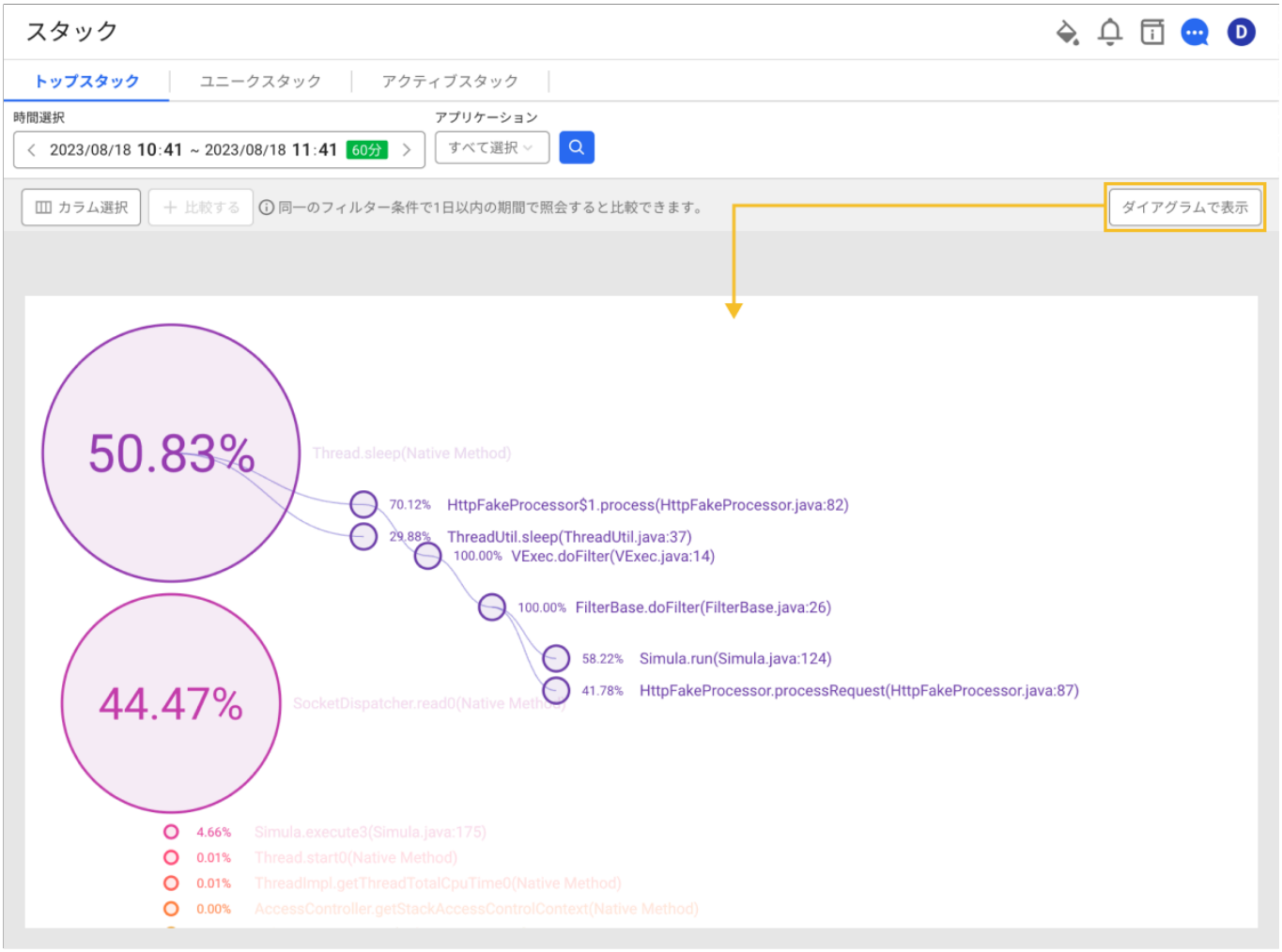
トップスタックをクリックすると、その最上位スタックに対する呼び出し頻度が確認できます。**トップスタック**の呼び出し関係は1対1なので、**トップスタック**のdepthが下がるほど情報の精度が低下する可能性があります。チューニング作業を行う際には、下位depthの情報は参照レベルで考えてください。

アプリケーションのパフォーマンス改善のため、最上位ステップへの呼び出し比率が高いモジュールのボトルネックを調べる必要があります。呼び出し比率が高いモジュールの場合、小さな性能改善でも、アプリケーション全体に大きな

改善効果がある場合があります。

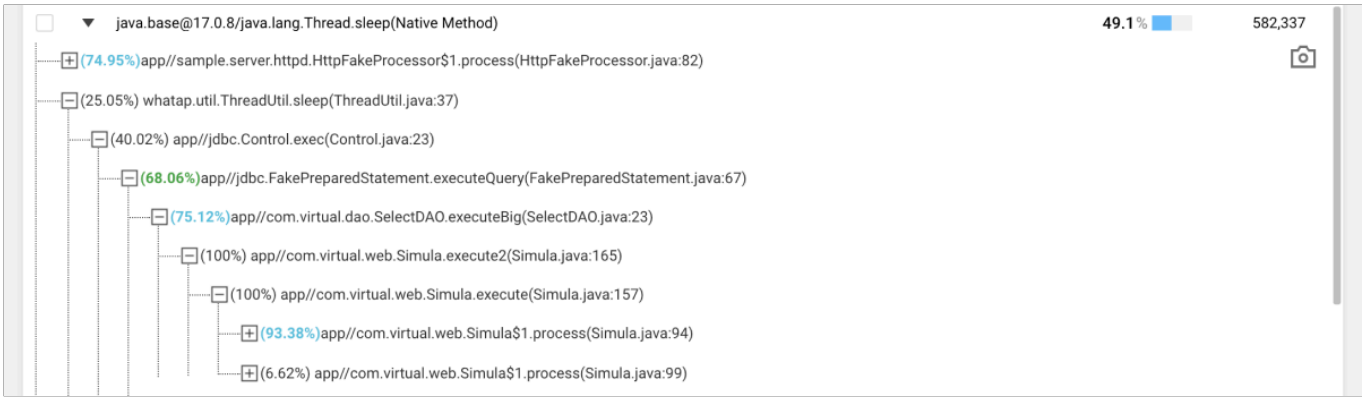
ダイアグラムで表示

複雑なスタックの呼び出しステップを簡単に理解できるため、ダイアグラムチャートを提供します。スタックテーブルの右上にある**ダイアグラムで表示**ボタンを選択すると、各ステップの呼び出し頻度と実行経路を一目で把握できます。



- 各ノードを選択すると、下位のdepthのスタックが表示されます。
- マウススクロールで拡大/縮小したり、ドラッグで動かします。
- スタックテーブルリストに戻るには**テーブルで表示**ボタンを選択します。

注意事項



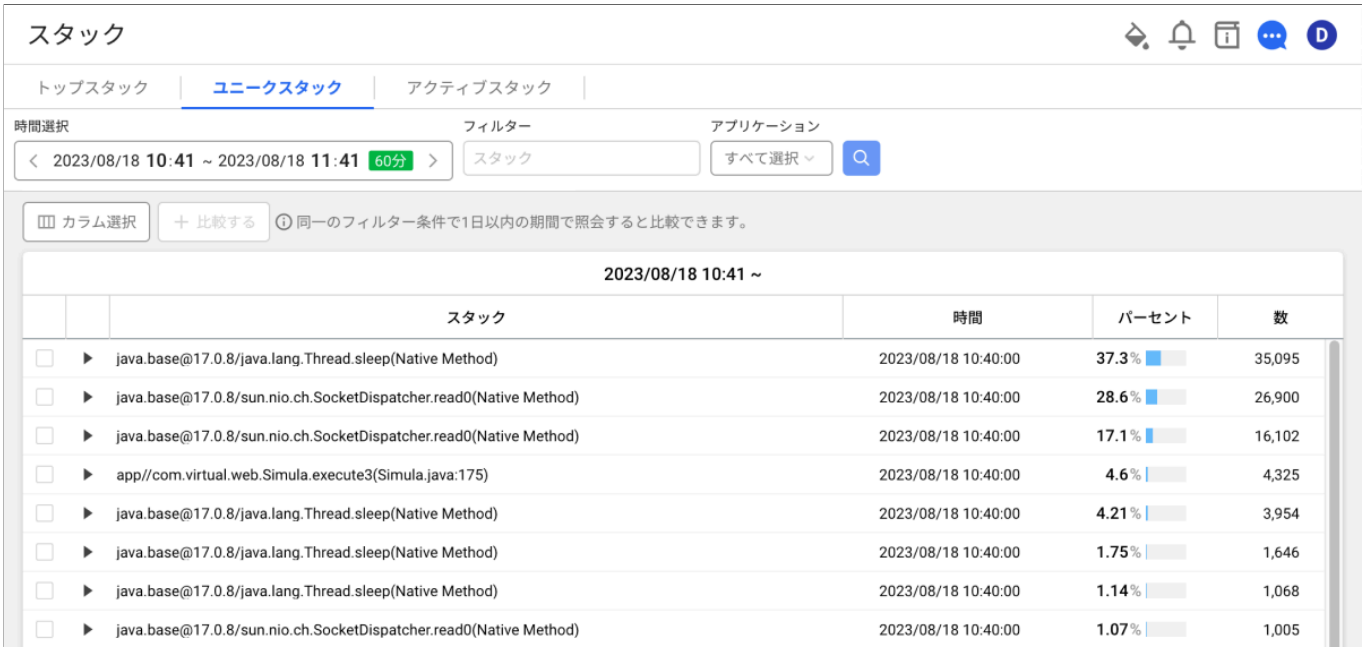

```
whatap.util.ThreadUtil.sleep
// jdbc.Control.execの呼び出し比率は40.02%
jdbc.Control.exec
/// jdbc.FakePreparedStatement.executeQuery呼び出し比率は68.06%
```

`whatap.util.ThreadUtil.sleep` ← `jdbc.Control.exec` ← `jdbc.FakePreparedStatement.executeQuery` の呼び出し比率が 40.02% * 68.06% を意味するものではありません。`jdbc.Control.exec` で他モジュールの呼び出す可能性があるためです。

トップスタックで呼び出し比率を確認する際には、各ステップ間の呼び出し比率を合わせて、全体の呼び出し比率を判断してはいけません。**トップスタック**の呼び出し比率はStack Trace上に表示される情報からステップ間の呼び出し比率を算出した結果であるため、ステップ間の呼び出し比率から全体の呼び出し比率を算出すると、歪んだ計算結果になる恐れがあります。

ユニークスタック

ユニークスタックは、Stack Trace全体の正確な呼び出し情報に基づいて実行されたメソッドのセットが同じ場合の統計情報を提供します。

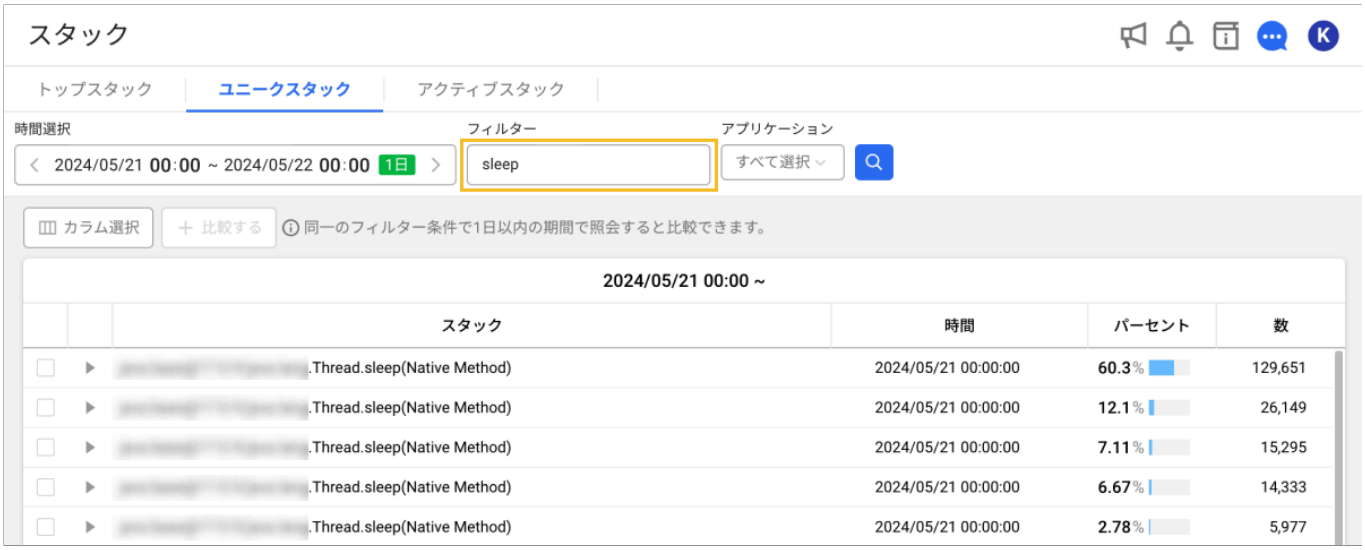


最もよく使われるスタックの情報が把握できます。同じスタックが何度も繰り替えても、重複を排除し、最も高い呼び出し頻度のスタックが確認できます。例えば呼び出す比率が高いStack Traceを識別できます。詳細な呼び出しステップのチェックにより、呼び出し経路上で異常モジュールの存在有無も把握できます。



各スタック項目では、メソッドの呼び出し経路(**スタック**)と当該メソッドの占める割合(**パーセント**)、全体の呼び出し件数(**数**)を表示します。各ステップ項目の左側に ▶ ボタンを選択すると、その呼び出しスタックのメソッド名、ファイル名、ライン番号などの分析に役立つ情報を提供します。

フィルタリング

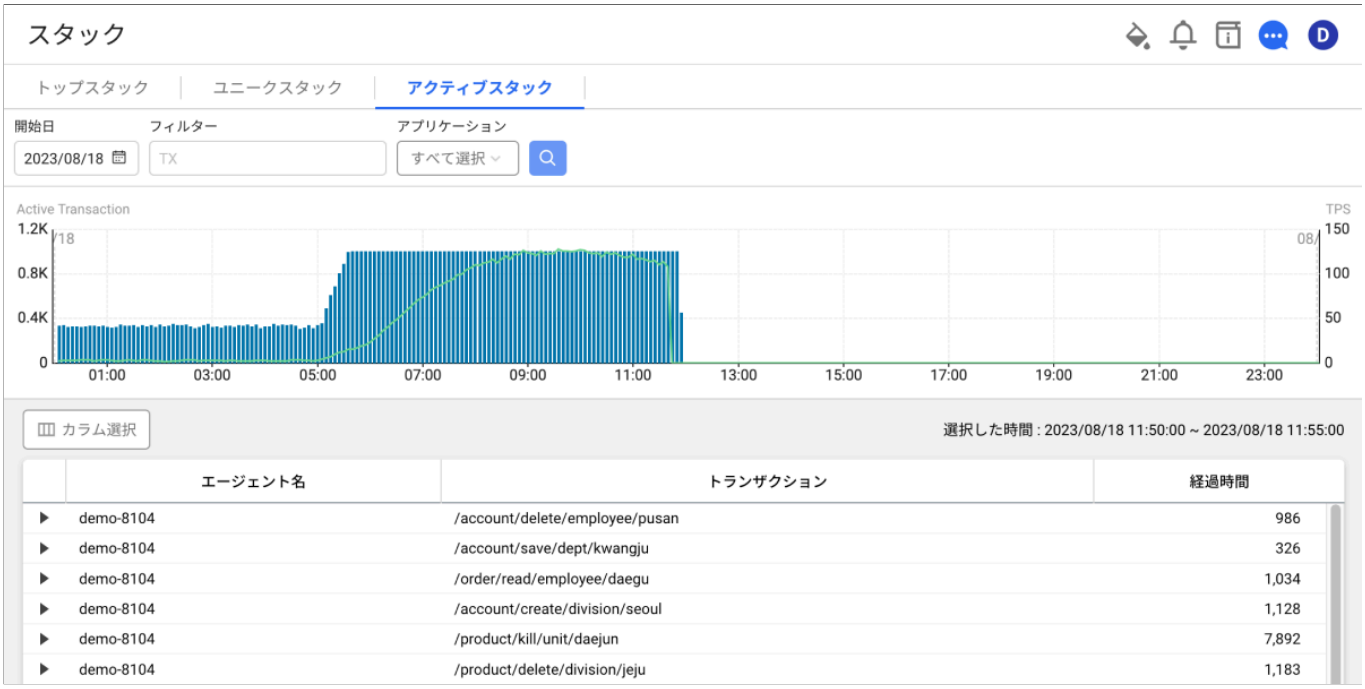


フィルター項目にメソッド名を入力して、照会結果をフィルタリングできます。文字列を入力した後、ボタンを選択します。

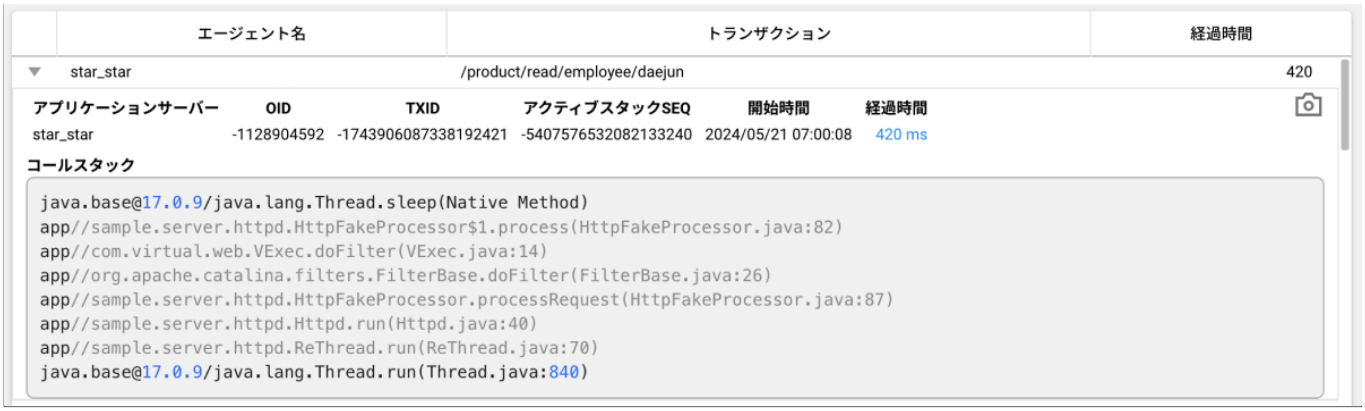
アクティブスタック

進行中のトランザクションをアクティブTXと呼びます。**アクティブTXから定期的にダンプされたスタック**を**アクティブスタック**と呼びます。

アクティブスタックでは、収集された**アクティブスタック**を時系列チャートで確認できます。長時間実行されるメソッドと短時間で実行されるが、頻繁に実行されるメソッドの比率により、実行中のトランザクションの状態をリアルタイムで把握できます。



チャートは5分間の単位統計データを活用して、**Active Transaction**の数を棒グラフで、**TPS**を折れ線グラフで表示します。特定の時間帯の棒グラフをクリックすると、その時間帯の**Active Transaction**のリストを確認できます。



トランザクションリストの左側にある ▶ ボタンを選択すると、そのトランザクションの**アクティブスタック**情報を確認できます。

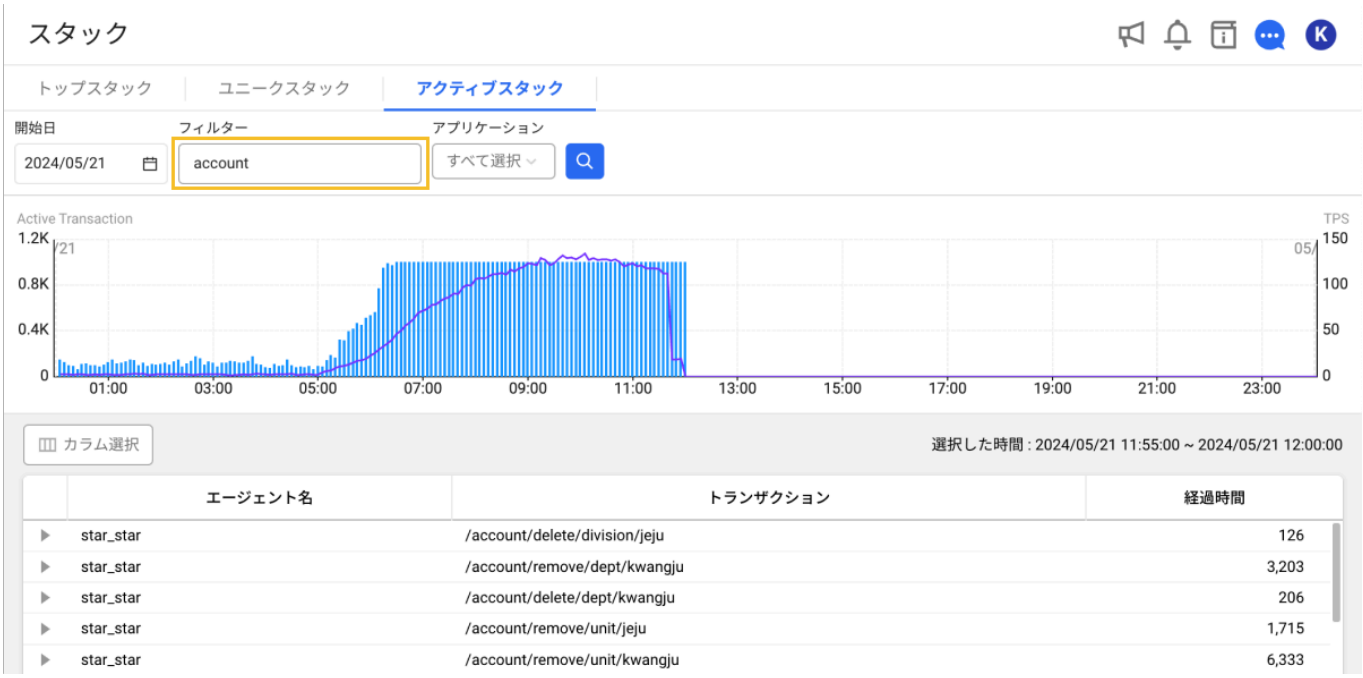
> ⓘ
> - **アクティブTX**収集の詳細は、次の文書を参照してください。その他のトランザクション収集に関連するエージェントオプションは、次の文書を参照してください。
> - WhaTapエージェントは、**アクティブTX**から10秒(変更可)ごとにダンプした**アクティブスタック**をサーバ

ーに転送します。収集間隔のエージェントオプションについては、次の例を参照してください。

whatap.conf

```
active_stack_second=10
```

フィルタリング



フィルター項目にトランザクション名を入力して、照会結果をフィルタリングできます。文字列を入力した後、ボタンを選択します。

スタックの比較

トップスタックと**ユニークスタック**では、一定期間を基準にして、時間帯ごとでの比率の変化と収集件数に対する履歴情報を提供します。

時間選択に照会する時間を設定した上で**比較する**ボタンを選択します。同じフィルター条件から**1日以内**期間での照会

時にスタックを時間帯ごとで比較できます。この機能で、時間帯ごとでの呼び出しスタックの変化を把握し、性能のトレンドを確認できます。また、障害時点の状況把握や性能改善前後の比較などに有用なデータとして活用できます。



- **パーセント**：照会期間中に選択されたスタックの比率の変化を確認できます。
- **数**：収集されるスタックの数は、アクティブTXの数に比例します。例えば、特定の区間で収集量が増加していれば、サービスの遅延や急激な流入量の増加があったと推測できます。

> ⓘ **比較する**ボタンが活性化にならない場合は、照会できる時間範囲を超えたか、比較する時間帯のデータが不十分な場合です。**時間選択**で照会できる時間範囲に設定すると、**比較する**ボタンが活性化されます。

キューブ

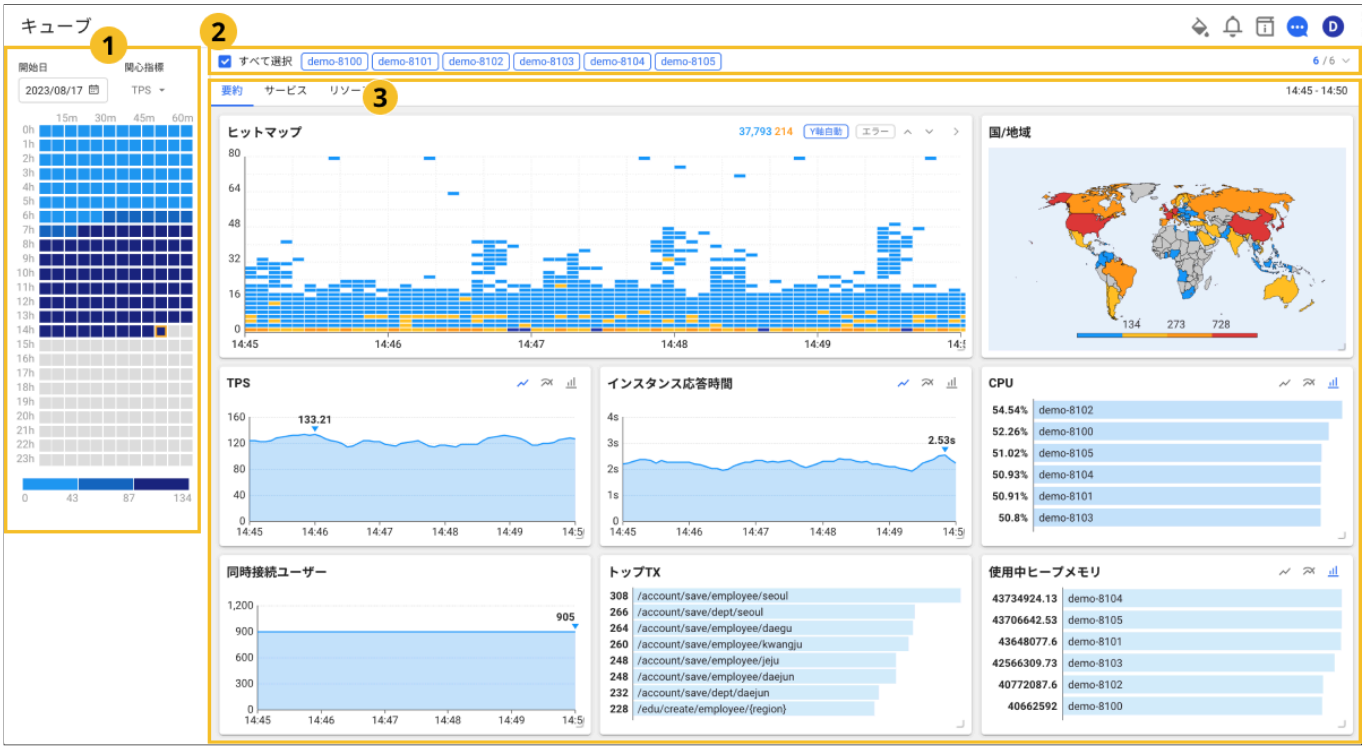
ホーム画面 > プロジェクト選択 > **分析** > **キューブ**

国別接続、ヒットマップ、トランザクション、TOPトランザクション、TPS、応答時間、リソース使用情報を提供します。

- 時間の経過に伴うアプリケーションの性能推移を把握できます。5分間隔で処理された統計データを使用して、アプリケーションの性能変化をすばやく把握できます。
- トラフィックの地理的分布を把握できます。アプリケーションへのトラフィックデータにより、トラフィックの地理的分布を把握し、地域ごとの性能を比較できます。
- リソースの使用量を把握できます。アプリケーションが使用するリソースの使用量を把握し、性能低下の原因を分析できます。

キューブとは？

WhaTapは5分単位で作成されたパフォーマンス統計を**キューブ**と呼びます。キューブ分析は、キューブに保存された5分単位のパフォーマンスデータを利用する分析機能です。



1 キューブのセレクトパネル

左側1領域のキューブセレクトパネルで、特定の時間に基づいてキューブを選択します。 キューブは5分単位で保存されるため、特定の時間を選択する必要があります。

- **パネル関心メトリクス**
  - **TPS**
  - **応答時間**
  - **TXエラー件数**
  - **アクティブTX数**
  - **同時接続ユーザー**
- **キューブ表示色**

  キューブは、選択された関心指標の数値に応じて異なる色で表示されます。

  - 紺色：高い数値

- 水色：低い数値

2 アプリケーション選択

上段2のアプリケーション選択ボタンを使用して、特定のアプリケーションデータのみ表示できます。右側のボタンを選択すると、次のように選択可能なエージェントオプションを選択できます。



- **すべて選択**：すべてのアプリケーションを表示します。既定値は**すべて選択**と**活性アプリケーション**が選択されています。
- **活性アプリケーション**：アプリケーション間でアクティブなアプリケーションを指定および表示します。
- **選択のみ表示**：アプリケーションからユーザーが指定したアプリケーションのみを表示します。
- **検索**：アプリケーションが多数の場合は、入力ウィンドウからそのアプリケーションを検索できます。
- 数字/数字：指定されたアプリケーション数/全体のアプリケーション数を表示します。

3 キューブデータ

3領域で該当するキューブデータを**ヒットマップ**、**国/地域**、**TPS**、**インスタンス応答時間**、**CPU**、**同時接続ユーザー**、**トップTX**、**使用中ヒープメモリ**などのチャートから確認できます。

- **ヒットマップ**

  キューブ時間の経過に伴うトランザクション分布を示します。トレースの詳細分析が可能です。

- **国/地域**

  クライアントIPに基づいて国をマッピングし、どの国からトランザクションが来ているかを示します。マップの色は、トランザクションの量を相対的に表示します。

> WhaTapは、クライアントに関する情報を保存します。ユーザーデータの収集に関するエージェント設定の詳細については、次の文書を参考にしてください。

- **TPS**

  Tansaction Per Second(TPS)は、毎秒処理されるトランザクション数を意味します。

- **インスタンス応答時間**

  トランザクションの平均応答時間です。

- **CPU**

  プロジェクトのアプリケーション(エージェント)をIPでソートし、使用率の高いサーバーのCPUをチャートで表示します。

- **同時接続ユーザー**

  リアルタイムブラウザのユーザー数を表示します。

- **トップTX**

  呼び出し件数の多い5つのトランザクションが表示されます。 トランザクション統計は、より詳細なデータを見つけることができます。

- **使用中ヒープメモリ**

  アプリケーション(インスタンス) のヒープメモリ使用量が高い順にインスタンスを表示します。

> - チャート右上の↗アイコンを選択すると、統合された折れ線チャートを提供します。
> - チャート右上の↝アイコンを選択すると、個々の折れ線チャートを提供します。
> - チャート右上の.ılアイコンを選択すると、バーチャートが表示されます。

> WhaTap uses the IP2Location LITE database for IP geolocation.

ヒットマップ

トランザクショントレーシングとは？

トランザクション性能とは、トランザクションの開始と終了までの要約指標や属性を意味します。一方、トランザクショントレースは、トランザクションが実行されているステップのトレースです。 トランザクションが遅いかエラーがある場合は、その原因を追跡するために実行履歴をステップごとに追跡する必要があります。 これを**トランザクショントレーシング**と言います。

トランザクションのパフォーマンス追跡のために収集するステップの種類は次のとおりです。

- **DB接続ステップ** START-END

  RDBへの接続のパフォーマンスが含まれます。 ステップ情報には、名前、レスポンス時間、エラーが含まれます。

- **SQLステップ** START-END

  JDBC SQLのパフォーマンスが含まれます。 ステップ情報には、接続情報、SQL文、エラーが含まれています。

- **HTTP Callステップ** START-END

  外部HTTPサービス呼び出しのパフォーマンスが含まれます。 ステップ情報には、url、host、port、レスポンス時間、エラーが含まれます。

- **Messageステップ** ADD

  トレースを収集する過程で、非構造化のすべての間隔の履歴を収集する際にメッセージステップを使用します。

- **SOCKETステップ** ADD

  Socketオープンを表現するステップです。

- **METHODステップ** START-END

  メソッド応答時間を追跡します。

- **ACTIVE STACK ステップ** START-END

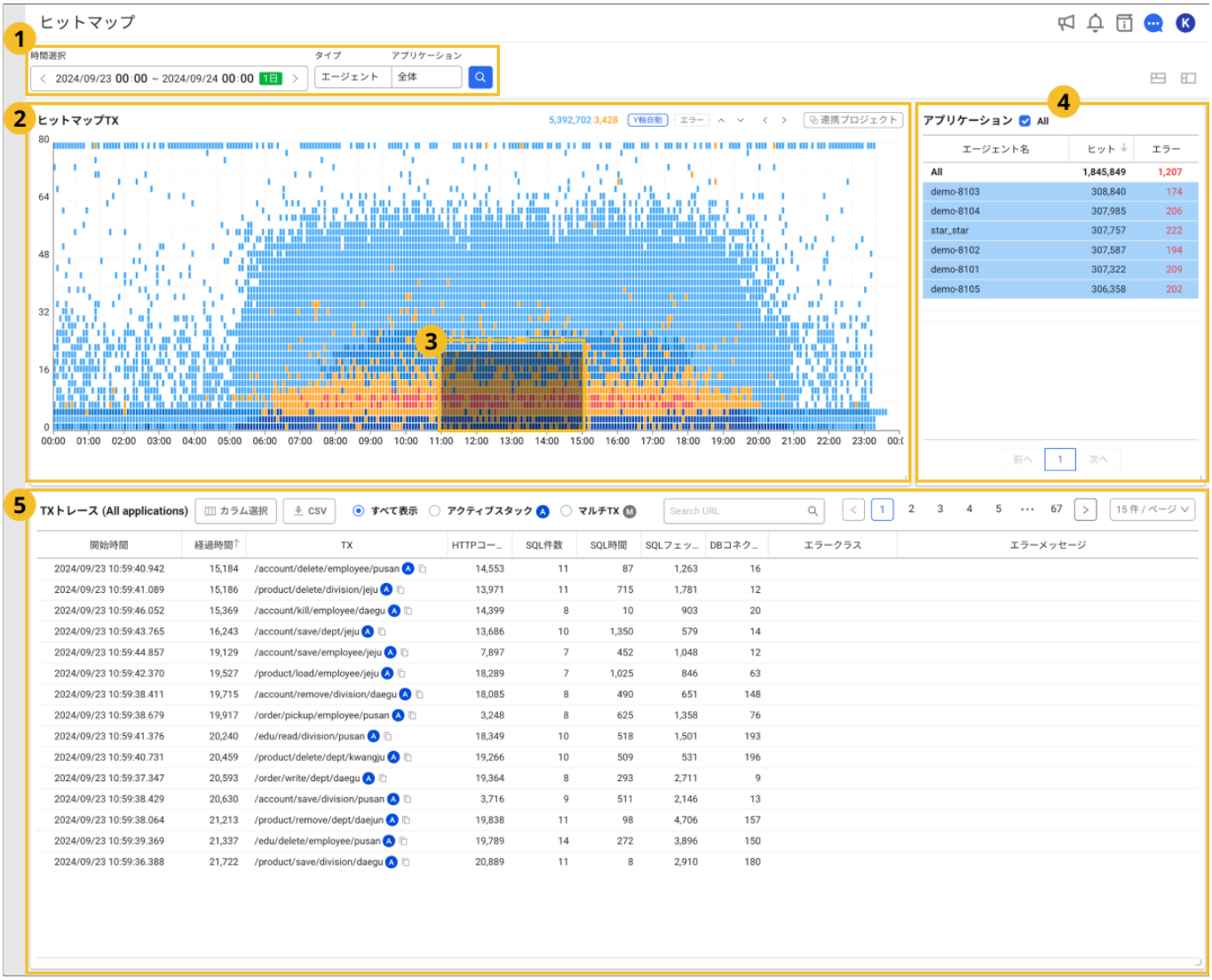
  アクティブスタックに関する情報を含みます。 別のスレッドが作成し、トレースに追加する方法で収集します。

連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、連携したプロジェクトの性能メトリクスを照会できます。 連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

WhaTapモニタリングサービスでは、**ステップ**(**Step**)は**スパン**(**Span**)と同じ意味で使用されます。

詳細分析

ホーム画面 > プロジェクト選択 > **分析** > **ヒットマップ**

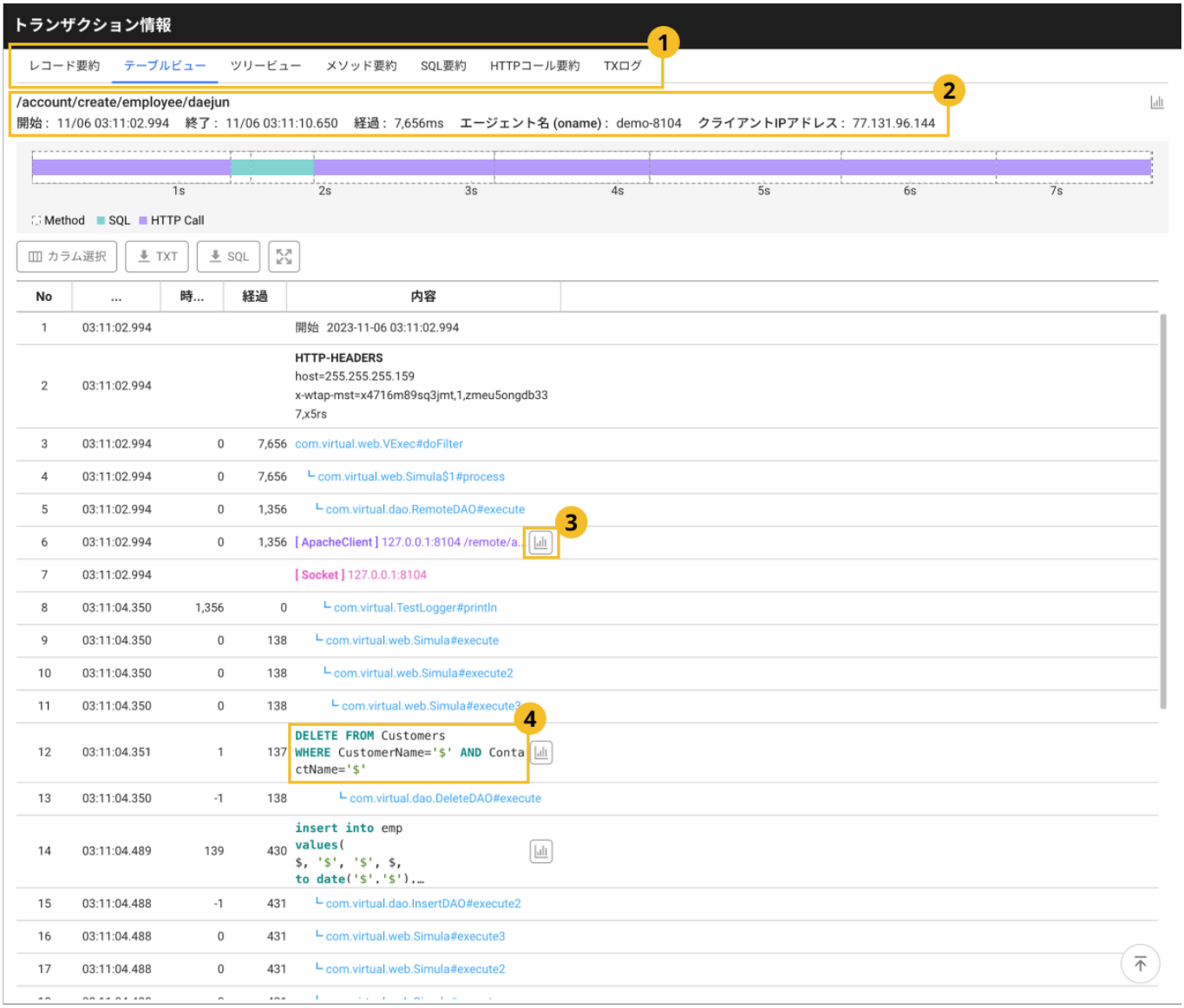


1. 1 **時間選択**で、ヒットマップを照会する期間を選択します。
2. 1 **タイプ**、**アプリケーション**から照会する対象を選択します。
3. ボタンを選択します。
4. 2 **ヒットマップTX**から照会する領域を3のようにドラッグします。
5. 5 **TXトレース**に一覧が表示されたら、分析するトランザクションを選択します。

トランザクションのパフォーマンス分析のためのクライアント情報などの属性、トランザクションの処理性能、各区間別の詳細な実行履歴などを確認できる**トランザクション情報**画面が表示されます。

- **ヒットマップ**メニューの機能については、次の文書を参照してください。
- アプリケーションダッシュボードのヒットマップウィジェットの詳細については、次の文書を参照してください。

トランザクション情報画面の案内



- 1領域では、ステップ情報をさまざまな形式で確認できます。 希望する形式のタブを選択します。

- 2領域では、トランザクションに関する基本情報を確認できます。
- 3：該当URLまたは各実行区間の統計データ画面が表示されます。 統計データ画面のグラフチャートで、希望する時間をクリックすると**統計**または**TX検索**メニューに移動します。 選択した時間に基づいて統計データを照会することができます。

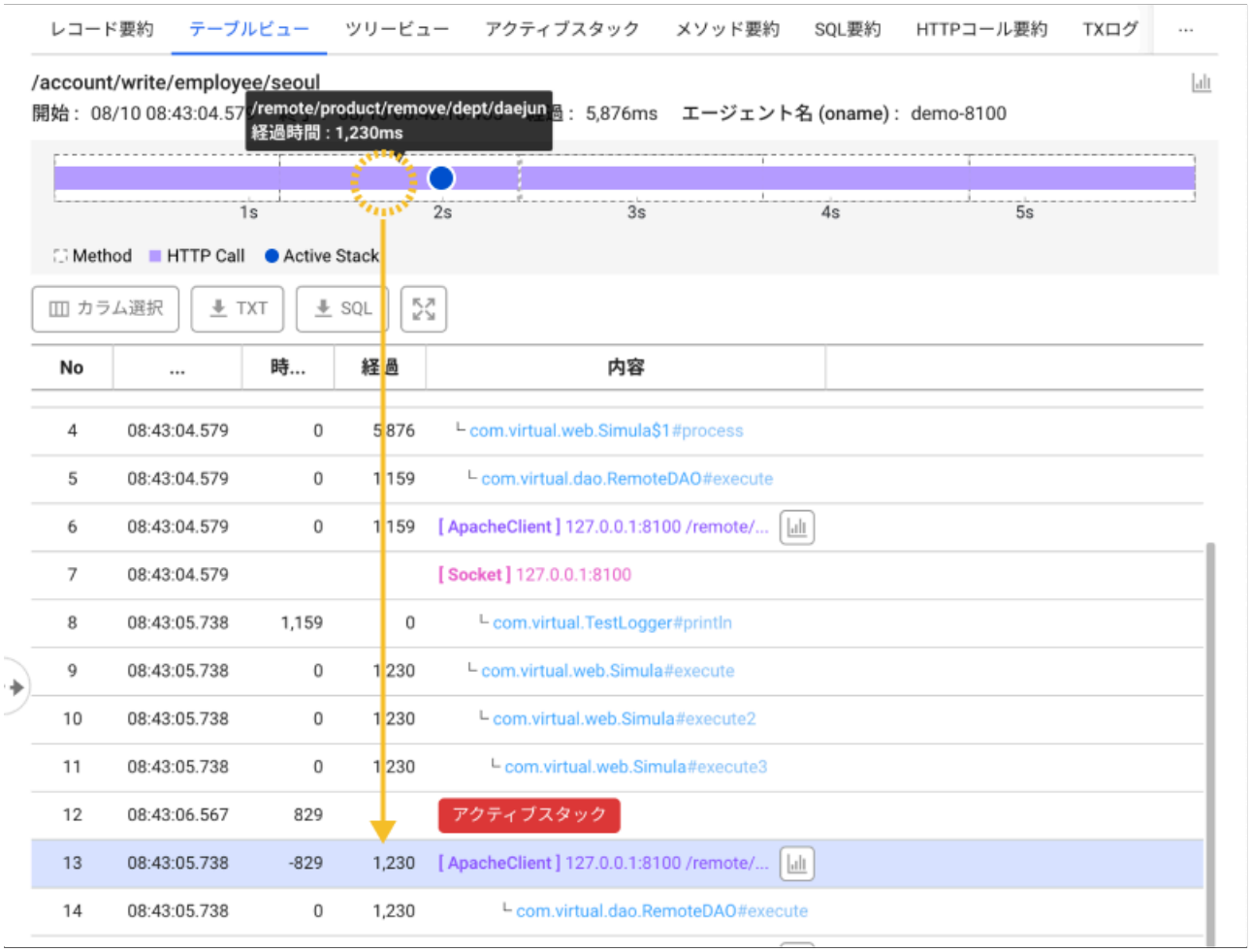
  > ◦ **統計**メニューの詳細については、次の文書を参照してください。
  > ◦ **TX検索**メニューの詳細は次の文書を参照してください。

- 4SQLステップを選択すると、SQL変数とHTTPクエリを照会することができます。 詳細については、次の文書を参照してください。

テーブルビュー

テーブルビュータブでは、トランザクションの実行過程を時間順に確認できます。

- トランザクションを実行区間別に分類し、ダイアグラムで確認できます。 全経過時間のうち、各区間別に所要された時間、最も長くかかった区間を素早く把握できます。
- ダイアグラムの各区間を選択すると、そのステップが位置するテーブルの一覧に移動します。



- 青い丸(●)が位置する領域は、アクティブスタックが収集された時点です。 青い丸を選択すると、**アクティブスタック**ボタンがあるテーブル一覧に移動できます。 **アクティブスタック**ボタンをクリックすると、その区間で行われたステップ情報を確認できます。

> ⓘ
> - トランザクション一覧からⒶアイコンが表示された項目で確認できます。
> - アクティブスタックの詳細については、次の文書を参照してください。

- **カラム選択**：テーブル一覧にカラムでメモリ累積情報およびCPU累積情報を追加または非表示できます。 カラム情報は次のとおりです。
  - **No**：ステップの実行順序
  - **時間**：各ステップの開始時刻

  - **時間差**：直前ステップの開始時刻から現在のステップに移るまでの待ち時間、外部要因により遅延される場合、経過時間と異なる場合があります。
  - **経過**：各メソッドの開始から終了までの所要時間
  - **内容**：該当ステップの詳細な実行内容
- **TXT**：トランザクションの基本情報および区間ごとの実行情報をTXTファイルでダウンロードできます。
- **SQL**：トランザクションの基本情報およびSQL実行情報をTXTファイルでダウンロードできます。
  - **置換値を含むダウンロード**: バインド変数値を元の値に置き換え、SQL統計データをダウンロードします。 セキュリティキー(`paramkey`)が設定されている場合は、セキュリティキー入力画面が表示されます。 セキュリティキーを入力しないとダウンロードできません。 セキュリティキーに関するエージェント設定の詳細については、次の文書を参照します。
  - **置換値を含まないダウンロード**: バインド変数値を置き換えない状態でSQL統計データをダウンロードします。
- :トランザクションの要約情報とダイアグラムを非表示にし、テーブル一覧のみを確認できます。 ボタンをクリックすると、隠された情報を再表示します。 テーブル一覧が長い場合は、この機能を利用すると便利です。
- SQLステップを選択すると、パラメータを照会できる**SQL**画面が表示されます。 詳細については、次の文書を参照してください。

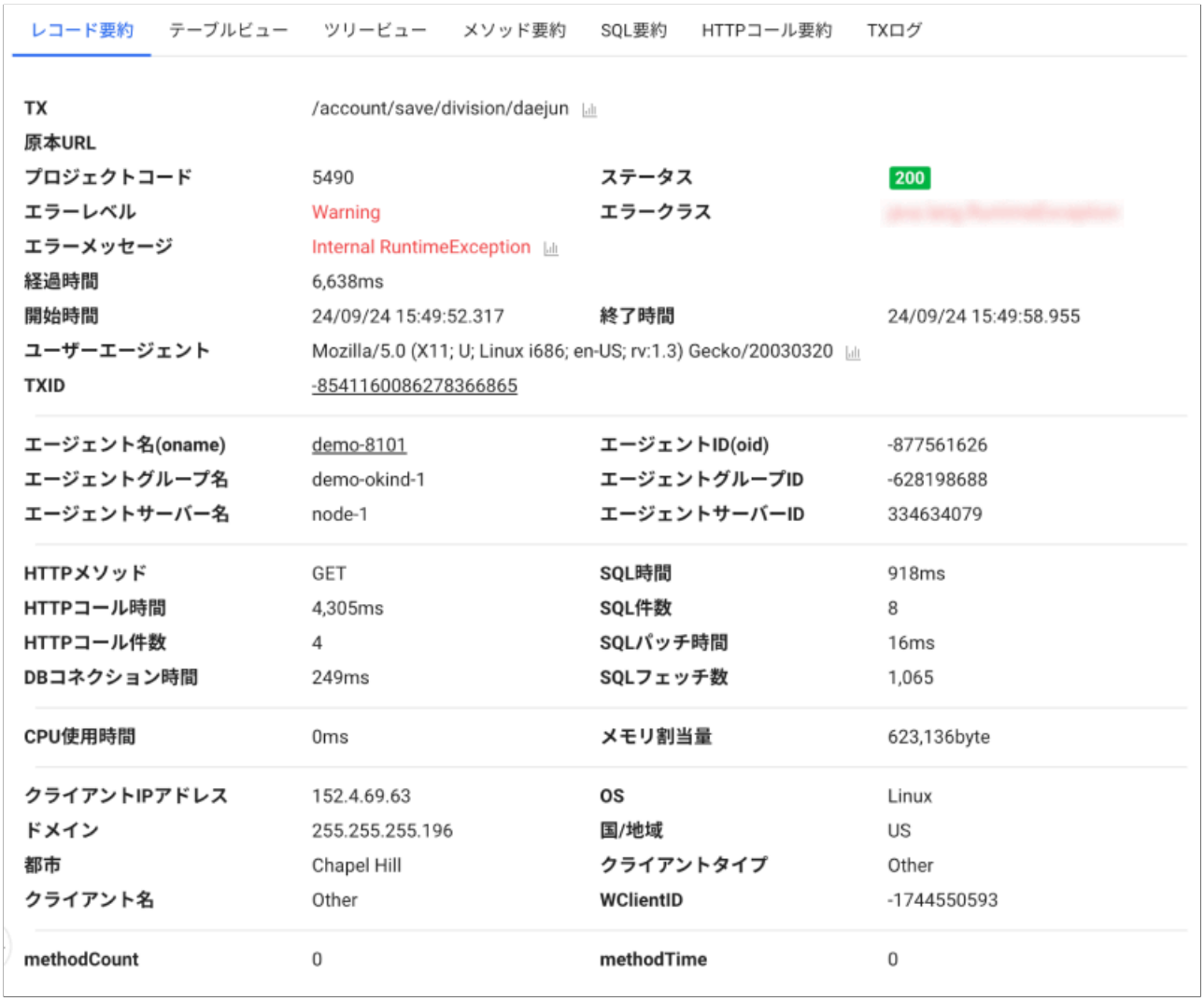
連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、**DB Connectionステップ**で連携したプロジェクトの性能メトリクスを照会できます。



連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

レコード要約

トランザクションの基本情報を確認するには、**レコード要約**タブを選択します。 トランザクションの基本情報およびそのトランザクションが実行されたエージェント関連情報、メモリ割り当て使用量、クライアント関連情報を確認できます。



分類	プロパティ	説明
トランザクション	TX	アプリケーションで実行されたAPIまたはURL呼び出し、そのトランザクションのパスです。 正規表現式を使用して一定の形式に変換します。

分類	プロパティ	説明
	原本URL	実際に呼び出されたURLパスです。 変数を含まない具体的なリクエストパスです。 元のURLがある場合は元のURL表示、元のURLがない場合は表示しません。
	プロジェクトコード	WhaTapモニタリングサービスに登録したプロジェクトの識別コードです。
	ステータス	そのトランザクションのHTTP応答ステータスのコードです。
	エラーレベル	そのトランザクションで発生したエラー レベルです。 WarningまたはCriticalと表示されます。
	エラークラス	そのトランザクションで発生したエラー関連クラスです。
	エラーメッセージ	そのトランザクションで発生したエラーメッセージです。
	経過時間	トランザクションの実行時間です。
	開始時間	トランザクションの開始時刻です。
	終了時間	トランザクションの終了時刻です。
	ユーザーエージェント	クライアント関連情報の抽出に利用するブラウザ情報です。
	リファラー	クライアントがブラウザで以前に利用したページアドレスまたは流入パスです。
	TXID	該当トランザクションの固有識別子(ID)です。
	マルチTXID	マルチトランザクションの固有識別子(ID)です。 他のアプリケーションのトランザクションと並列に実行された場合に表示されます。
エージェント	エージェント名(oname)	エージェント名です。
	エージェント	エージェントの固有識別子(ID)です。

分類	プロパティ	説明
	ID(oid)	
	エージェントグループ名	`okind`オプションで設定されたエージェントグループ名です。
	エージェントグループID	エージェントグループの固有識別子(ID)です。
	エージェントサーバー名	`ondoe`オプションで設定したエージェントサーバー名です。
	エージェントサーバーID	エージェント サーバーの固有識別子(ID)です。
ステップ (Step)	HTTPメソッド	HTTPメソッドです。 (GET、POST、PUT、HEADなど)
	HTTPコール時間	外部Http Call時間です。
	HTTPコール件数	外部HTTP Call件数です。
	DBコネクション時間	データベースに接続するまでにかかる時間です。
	SQL時間	SQLの実行時間です。
	SQL件数	SQLの実行件数です。
	SQLパッチ時間	レコードの照会にかかった時間です。 ネストが発生するか、他のロジックが含まれる場合があります。
	SQLフェッチ数	SQLでデータを照会したレコードの件数です。

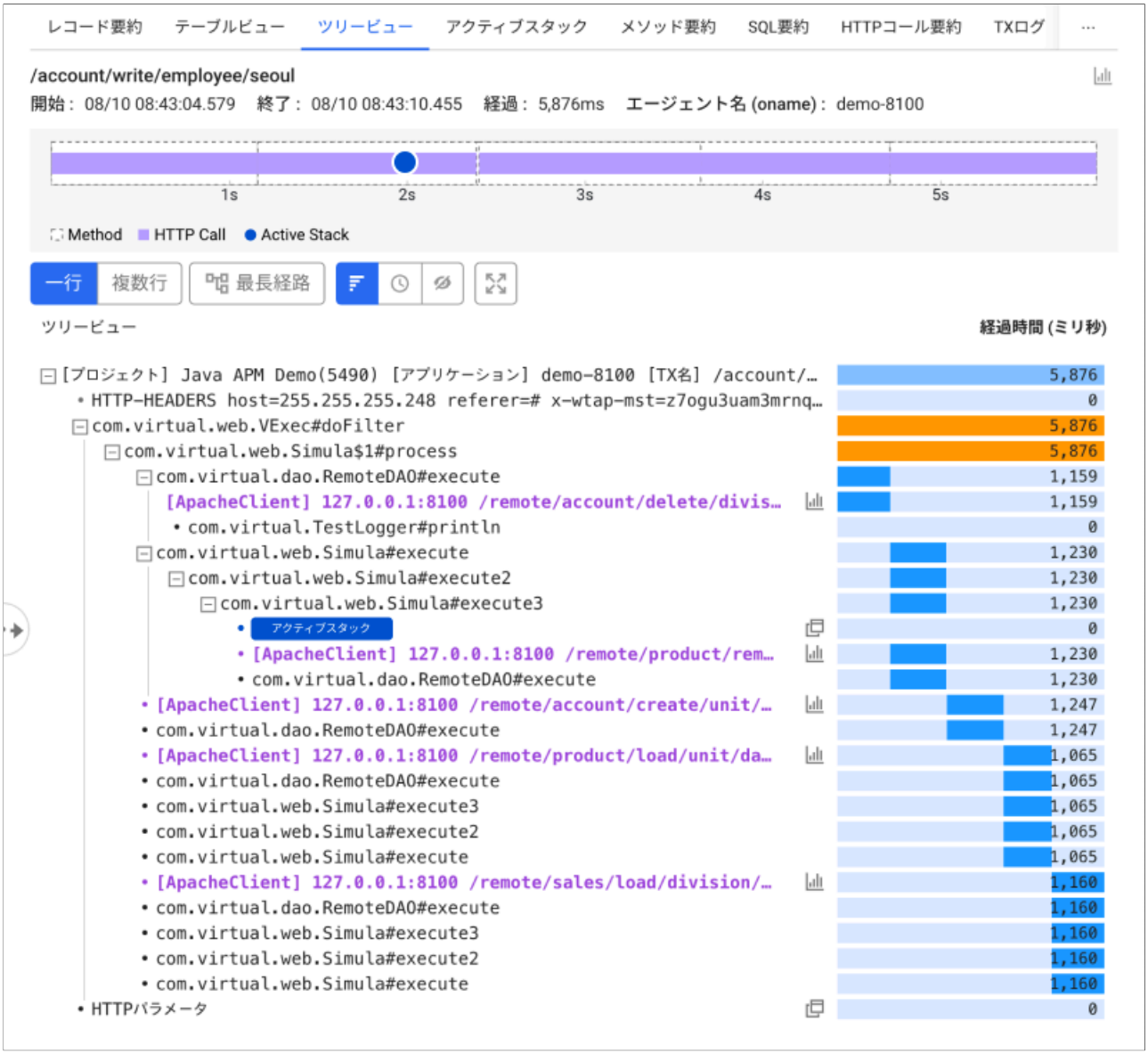
分類	プロパティ	説明
リソース (Resource)	CPU使用時間	トランザクションの処理に使用されたCPU時間です。
	メモリ割当量	そのトランザクションで使用されたメモリ量です。
コンテナ	Pod名	該当トランザクションを処理したKubernetes Podの名前です。
	コンテナID	Pod内で実行されている特定のコンテナの固有識別子です。
クライアント	クライアントIPアドレス	クライアントのIPアドレスです。
	OS	ブラウザが実行されるOS環境です。
	ドメイン	クライアントが接続したIPアドレスに指定されたインターネットアドレスです。
	国/地域	クライアントの国の情報です。
	都市	クライアントの国内の都市情報です。
	クライアントタイプ	クライアントが利用したブラウザの種類です。
	クライアント名	クライアントが利用したデバイス名です。
	WClientID	クライアントの固有識別子です。
Method	methodCount	トランザクションで呼び出されたメソッドの数です。
	methodTime	メソッドの実行にかかった時間です。

- アプリケーションの種類や設定、ステップの種類によって収集する情報は異なる場合があります。 **コンテナ**項目は、アプリケーションがコンテナ環境で実行されている場合にのみ表示されます。
- エラー関連項目は、そのトランザクションでエラーが発生した場合に表示されます。

- トランザクション属性の一部は、統計機能を提供します。 ボタンを選択すると、時系列チャートを通じて照会時間中に発生したトランザクション状況を把握できる画面が表示されます。 チャート内の棒グラフを選択すると、**分析** > **TX検索**メニューに移動して詳細照会できます。
- WhaTapは、クライアントに関する情報をデフォルトで保存します。 ユーザーデータの収集に関するエージェント設定の詳細については、次の文書を参照してください。

ツリービュー

トランザクションの実行過程をツリー形式で確認するには、**ツリービュー**タブを選択します。 各トランザクションとそれに属するトレースの詳細、トレースの開始時刻と所要時間、呼び出し関係を確認できます。 ダイアグラムの各区間を選択すると、そのステップが位置するツリービューに移動します。



- **一行**：各区間別の実行情報に表示されたテキストを一行で表示し、ツリー形式をシンプルに整理できます。
- **複数行**：各区間別の実行情報に表示されたテキストを改行してすべて表示します。
- **最長経路**：最長の経路で移動できます。
- **タイムバーの表示**：経過時間を棒チャートで表示します。
- **時間を表示する**：各区間ごとのタイムスタンプ、ギャップ、経過時間をテキスト形式で表示します。
  - 8秒以上：超過遅延状態で赤色で表現します。

- 3秒以上8秒未満：遅延状態でオレンジで表現します。
- 3秒未満：正常な状態で青色で表示します。

- **時間を表示しない**：時間情報を非表示にします。
- ：トランザクションの要約情報とダイアグラムを非表示にし、ツリービューのみを確認できます。 ボタンをクリックすると、隠された情報を再表示します。 ツリービュー一覧が長い場合は、この機能を利用すると便利です。
- ：SQL変数とHTTPクエリを照会できる画面が表示されます。 詳細については、次の文書を参照してください。

> 開始および所要時間の場合、トランザクション呼び出し環境によって生じる時差を上位トランザクションのトレースやマッピングで修正および表示することで、実際に収集された時間データとの差異が生じる可能性があります。

アクティブスタック



アクティブスタックタブを選択すると、該当トランザクションから収集したアクティブスタックの一覧を確認できます。

- 左側の**アクティブスタック**一覧から収集時点（No、時間）を選択すると、右側に該当時点のスタック情報を表示し

ます。

- **スタック**：アクティブスタックの詳細情報をTXTファイルでダウンロードできます。

> - トランザクション一覧でAアイコンが表示された項目からアクティブスタックを確認できます。
> - アクティブスタックの詳細については、次の文書を参照してください。

AIアクティブスタック分析

Beta

アクティブスタック分析ボタンをクリックすると、収集したアクティブスタックをAIが自動的に分析します。進行中のトランザクションのスタックダンプを基にパフォーマンスの問題を特定し、改善方法を案内します。

分析結果は次の4つのセクションで構成されます。

- **全体要約**：分析したスタック数と全体の実行フローを要約します。
- **共通パターン**：繰り返し現れる実行パターンを表示します。パターン名、発生回数、説明を含みます。
- **発見されたイシュー**：特定したパフォーマンスの問題を深刻度（**High**、**Medium**、**Low**）とともに表示し、判断の根拠となるスタック情報も提供します。

深刻度	基準
High	デッドロック、完全ブロッキング、同一ブロッキング4回以上繰り返し
Medium	I/O待機、2〜3回のロック競合、遅い外部呼び出し
Low	単一発生の待機、軽微な最適化の機会

- **推奨事項**：優先順位に基づいて具体的な改善方法を案内します。

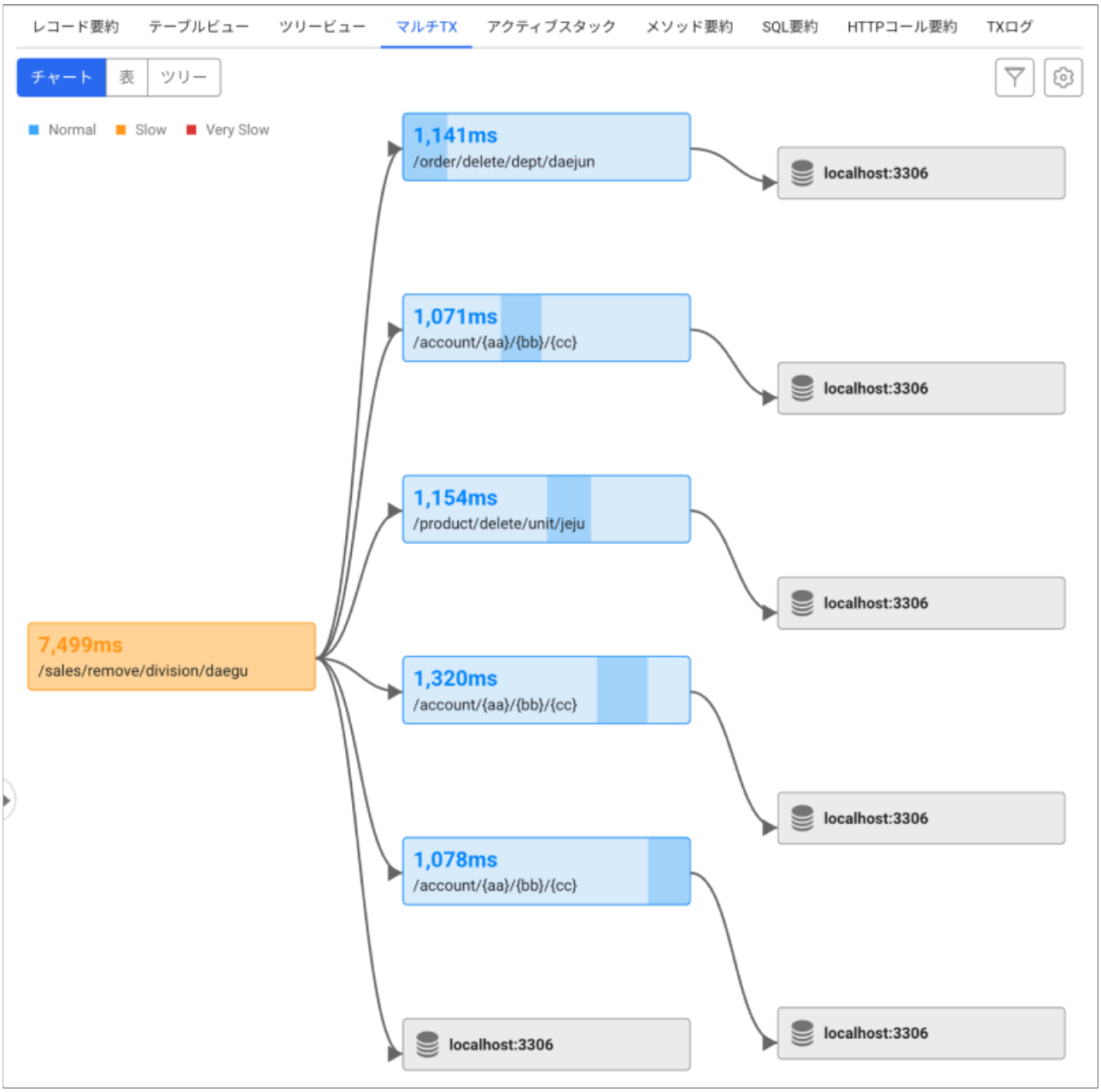
各分析項目には収集したスタック番号（ No.N ）が表示されます。スタック番号をクリックすると、該当スタックの原本を確認できます。

> - AIが生成した分析結果のため、実際の状況と異なる場合があります。参考資料としてご活用ください。
> - プロジェクト**メンバー**権限以上が必要です。

マルチトランザクション

マルチトランザクションとは、他のエージェントやプロジェクトに関連付けられたトランザクションを意味します。**マルチTX**タブでは、WhaTapモニタリングサービスに登録したアプリケーション間の呼び出し関係を確認できます。

> マルチトランザクションを追跡するには、**管理** > **エージェント設定**メニューで `mtrace_enabled` オプションを `true` に設定します。 エージェント設定の詳細については、次の文書を参照してください。



- **チャート**：各トランザクションの呼び出し関係をフローチャート形式で提供します。 トランザクションノードを選択すると、そのトランザクションノードのトレース分析情報を確認できます。 **チャート**モードではマウスを利用して位置を移動したり、スクロールして拡大または縮小できます。
  - **チャートビュー設定**：チャートに表示するアイテムを表示または非表示します。
- **表**：テーブル形式で、マルチトランザクション内の各トランザクションごとの情報を確認できます。 カラム選択

アイコンを選択して、テーブルヘッダーカラムを編集できます。 各トランザクション項目を選択すると、トレース分析情報を確認できます。

- **ツリー**：ツリー形式でトランザクション間の呼び出し関係を把握できます。 関連した付加機能は、**ツリービュー**タブの機能と同じです。
- **プロジェクト選択**：チャートに表示されるプロジェクトを選択または解除できます。

> ⓘ
> - トランザクション一覧からⓂアイコンがある項目で確認できます。
> - マルチトランザクションの詳細については、次の文書を参照してください。

メソッド要約



No	ク...	メソ...	パラメータ/戻...	数	合計 (ms)	最大 (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

メソッド情報のみを確認するには、**メソッド要約**タブを選択しますて。 エージェントに追跡が設定されたメソッド名と所要時間を表示します。 不要なロジックが繰り返し実行される場合、低い件数に比べて実行時間が長くかかる場合を把握し、メソッドロジック改善のための分析情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。

- **メソッド**: メソッドに関する詳細情報をTXTファイルにダウンロードできます。

> メソッド(method)と関連するエージェント設定オプションの詳細については、次の文書を参照してください。 ご利用中の製品によって、メソッド追跡サポートの可否は異なる場合があります。

SQL要約



No	データベ...	SQL	数	合計 (ms)	最大 (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文についての情報を確認するには、**SQL要約**タブを選択します。 不要に繰り返し実行される場合、低い件数に比べて実行時間が長くかかる場合を把握し、クエリ性能改善のための分析情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **SQL**: SQLに関する詳細情報をTXTファイルにダウンロードできます。

HTTPコール要約



No	リモートサー...	URL	数	合計 (ms)	最大 (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呼び出しのリクエスト件数、合計時間、平均時間などを確認するには、**HTTPコール要約**タブを選択します。 不要な外部への呼び出しが繰り返して実行される場合や、低い件数だが比較的に実行時間が長くかかる場合などを把握するなど、トランザクションの遅延要因が外部か内部かを分析する情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **HTTP**: HTTP Callに関する詳細情報をTXTファイルにダウンロードできます。

TXログ

トランザクションに関するログ情報を確認するには、**TXログ**タブを選択します。



- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **ログ検索機**: ログ検索に移動して、目的のログデータを簡単に照会できます。
- **キーワード検索**：検索バーに検索するテキストを入力し、エンターキーを入力するか、ボタンを選択します。 入力したテキストと一致するキーワードをハイライトで表示します。
- **キーワードハイライト**：キーワードと色を設定すると、自動的にログ一覧でキーワードと一致するテキストをハイライトします。 詳細については、次の文書を参照してください。
- **カラム設定** : ログリストテーブルに表示されるカラムの追加や順序変更ができます。 詳細については、次の文書を参照してください。
- **ログ表示詳細設定**: ログリストに表示される内容を設定できます。 詳細については次の文書を参照してください。

> - **TXログ**タブを有効にするには、ログに関連するエージェント設定オプションを適用する必要があります。 ログ設定の詳細については、次の文書を参照してください。
> - **TXログ**タブは**ログ読取り**権限を持つメンバーのみアクセスできます。 メンバー権限の詳細については、次の文書を参照してください。

キーワードハイライトの設定

1. ハイライトで表示するキーワードを追加するにはボタンを選択します。



2. ご希望のキーワードを入力します。
3. 色アイコンを選択し、ご希望の色を選択します。
4. エンターを入力します。 追加したキーワードが一覧に作成されます。

キーワードハイライト削除

ハイライト一覧から利用しない項目を削除するには、—ボタンを選択します。

テーブルカラムの表示または非表示

1. テーブルの右上の▥ボタンを選択します。
2. **カラム設定**ウィンドウが表示されたら、左側のリストで目的の項目を選択・解除します。
   右のリストから×ボタンを選択してカラムを非表示にすることもできます。
3. すべての設定を完了した後**確認**ボタンを選択します。 設定した内容がテーブルに適用されます。

テーブルカラムの表示順番の変更

1. テーブルの右上の▥ボタンを選択します。
2. **カラム設定**ウィンドウが表示されたら、右側のリストで目的の項目をドラッグして順序を変更します。
3. すべての設定を完了した後**確認**ボタンを選択します。 設定した内容がテーブルに適用されます。

テーブルコンテンツの表示設定

1. テーブル右上の⚙ボタンを選択します。
2. **ログ表示詳細設定**ウィンドウが表示されたら、目的のオプションを設定します。
   - **content**または**タグ**オプションを選択して、ログにコンテンツの表示有無を選択します。
   - **タグ管理**: タグは、追加した順にログに表示されます。 タグの順序はドラッグで変更できます。 非活性化されているタグはログに表示されません。
3. すべての設定を完了して✕ボタンを選択します。 設定した内容がテーブルに適用されます。

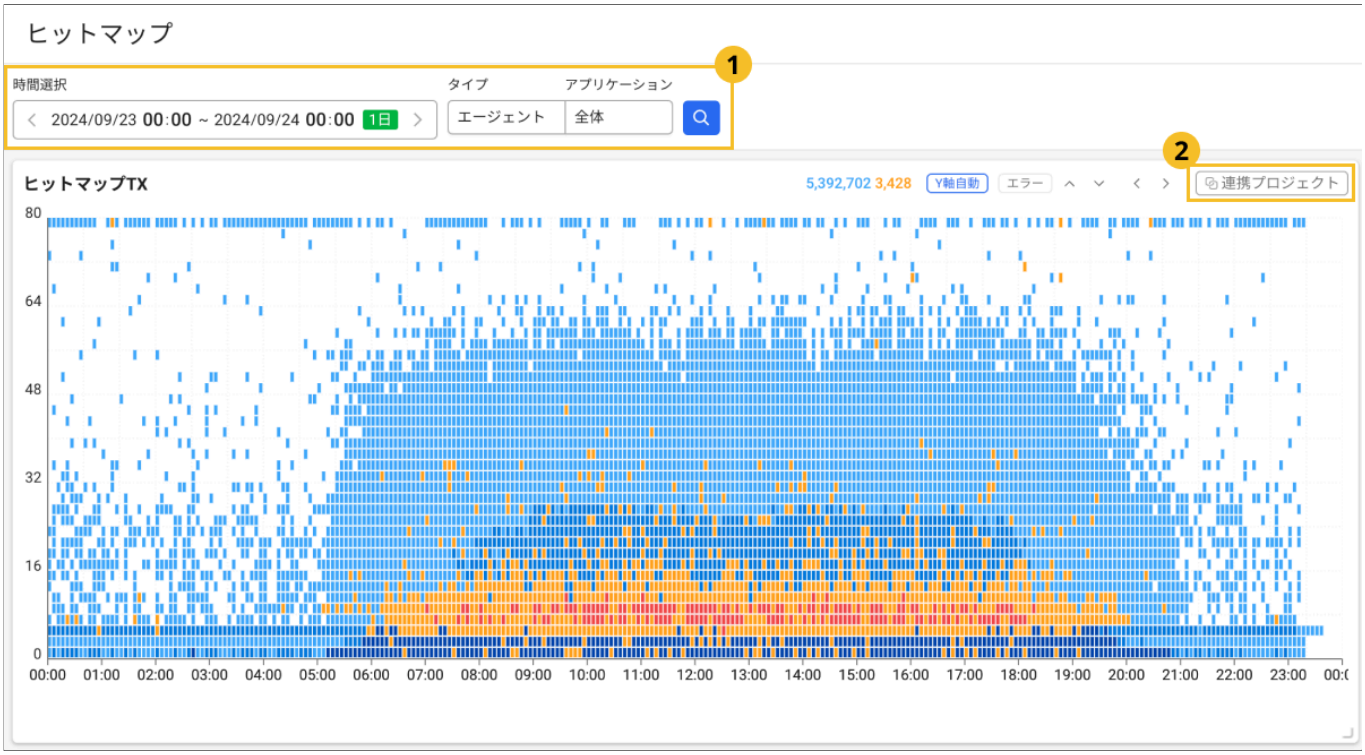
> ⓘ **ログ表示詳細設定**の詳細については次の文書を参照してください。

追加機能

連携プロジェクトメトリクスの確認

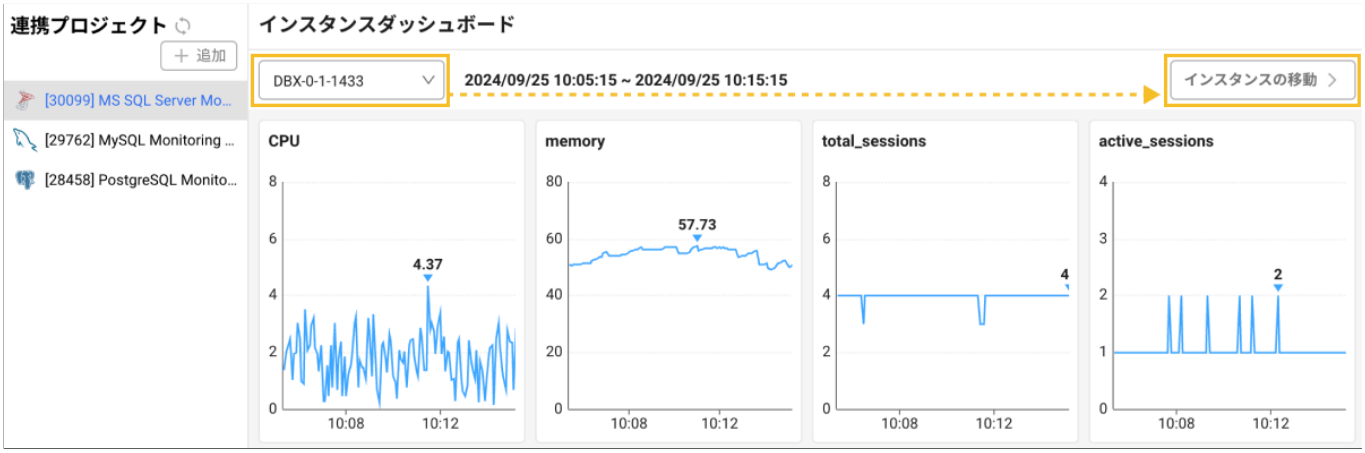
データベースモニタリング(DPM) プロジェクトとアプリケーションのモニタリング(APM) プロジェクトを生成した場合、2つのプロジェクトを連携してDPMで収集したモニタリングデータをAPMプロジェクトで確認できます。 **ヒットマップTX**セクションの右上にある**連携プロジェクト**機能で、トランザクションの照会中に連携したプロジェクトの性能メトリクスを一緒に確認できます。

連携プロジェクト機能は、APMプロジェクトでトランザクションを分析する際に、遅いSQLがある場合、類似な複数のDBを使用する環境から、正確に問題が発生したDBを把握しにくい状況に役立ちます。



1. トランザクションを照会する時間と**アプリケーション**を設定した後 ボタンを選択します。
2. **ヒットマップTX**セクションで右上にある**連携プロジェクト**ボタンを選択します。
3. 連携プロジェクトのメトリクスを確認できる新しい画面が表示されます。

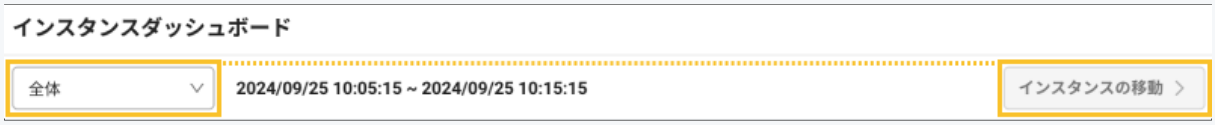
左の一覧で連携プロジェクトを選択し、**インスタンスダッシュボード**セクションのメトリクスを確認します。



選択したプロジェクトのダッシュボードメニューに移動するには、インスタンスを選択し、右上の**インスタンスの移動**

ボタンをクリックします。 **ダッシュボード** > **インスタンスモニタリング**メニューに移動します。

- **ヒットマップTX**セクションの**連携プロジェクト**ボタンは、連携されたプロジェクトがある場合にのみ表示されます。 連携プロジェクトの追加についての詳細は、次の文書を参照してください。
- 左の**連携プロジェクト**一覧の上にある**追加**ボタンを選択するとプロジェクトを追加できます。
- **インスタンスの移動**ボタンは、**全体**を選択した状態では無効になります。



- データベース商品の種類によって**分析** > **カウント推移**メニューに移動できます。

HTTPパラメータ照会

テーブルビュー タブでそのトランザクションのHTTPパラメータを照会することができます。

1. ページの下にスクロールして、トランザクションの実行の最後のステップに移動してください。
2. **HTTPパラメータ**項目を選択します。
3. **HTTP-PARAMETERS**画面が表示されると、**パスワード** ボタンを選択します。
4. 設定したParam Keyを入力します。隠されたパラメータを確認できます。

- HTTPパラメータに関するエージェント設定は、次の文書を参考にしてください。
- **パスワード**：復号化されたパラメータの値を確認できます。 パスワードは `WHATAP_HOME /security.conf (paramkey.txt)`ファイル内の6桁の文字列です。 別の文字列に変更できます。 SQL変数の照会、HTTPクエリの照会、Threadの中止に必要です。 詳細については、次の文書を参照してください。

SQLパラメータの照会

テーブルビュータブでSQLステップを選択するか、**ツリービュー**タブで  ボタンを選択します。 パラメータを照会できる**SQL**画面が表示されます。



- **パスワード**：復号化されたパラメータの値を確認できます。 パスワードは*`WHATAP_HOME`/security.conf*ファイル内の6桁の文字列(`paramkey`)です。 別の文字列に変更できます。

  > **パスワード**は、SQL変数の照会、HTTPクエリの照会、Threadの中止に必要です。 詳細については、次の文書を参照してください。

- **SQLフォーマット**：SQL文をインデントまたはフォーマット化して可読性を高めることができます。
- ：SQL文をクリップボードにコピーできます。
- **統計**: **統計** > **SQL**メニューに移動します。

SQL変数とHTTPクエリを照会するには、次のオプションをエージェント設定に追加します。

- SQLパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
- HTTPパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。

whatap.conf

```
# SQLパラメータ照会オプション：オプションが適用されるとSQLパラメータを暗号化して収集します。
profile_sql_param_enabled=true

# HTTPパラメータ照会オプション：オプションが適用されると、HTTPクエリパラメータを暗号化して収集します。
profile_http_parameter_enabled=true
```

トレース一覧カラムを設定する

TXトレースセクションのカラム設定により、照会された結果の詳細項目を追加で確認できます。

1. **TXトレース** エリアで **カラム選択** ボタンをクリックします。
2. 表示したいカラム項目を選択または解除します。設定内容はテーブルに即時反映されます。
   - **すべて選択**: すべてのカラム項目を選択します。
   - **すべて解除**: すべてのカラム項目を解除します。
   - **初期化(既定値)**: カラム項目をデフォルトに設定します。
3. 設定が完了したら、右上の × アイコンをクリックします。

トレース一覧をフィルタリングする

TXトレースセクションで照会された結果を次のオプションでフィルタリングできます。



- **すべて表示**: 照会された結果からすべてのトランザクションが確認できます。
- **アクティブスタック**: 照会された結果から、アクティブスタックがあるトランザクションのみを確認できます。
- **マルチTX**: 照会された結果から、マルチトランザクションのトランザクションのみを確認できます。
- **Search URL**: 入力した文字列と一致する項目を含むトランザクションのみが表示されます。 文字列を入力した後、キーボードの `Enter` キーを押します。
  - **トランザクション**
  - **クライアントIPアドレス**
  - **ドメイン**
  - **エラークラス**
  - **エラーメッセージ**

照会一覧をダウンロードする

TXトレースセクションで照会された結果をCSV形式のファイルにダウンロードできます。 ダウンロードしたCSVファイ

ルの名前は、profile {project_code} YYYYMMDD .csv形式です。

1. トランザクションを照会した後に CSV ボタンを選択します。
2. **最大CSV行数**オプションに希望する値を入力します。
3. **ダウンロード**ボタンを選択します。

マルチTX

マルチトランザクションは、他のエージェントやプロジェクトに関連付けられたトランザクションを意味します。WhaTapプロジェクトに登録されたアプリケーションサービス間の呼び出しを追跡するのが、**マルチトランザクション追跡**です。

> Javaエージェントは、3つのHTTPヘッダーキー値(`x-wtap-po` 、 `x-wtap-mst` 、 `x-wtap-sp1`)でマルチトランザクションを追跡します。 ゲートウェイを通過するHTTPトランザクションの連携追跡ができない場合は、HTTPヘッダー条件を確認します。

> **マルチトランザクションの活性化**
>
> マルチトランザクションを追跡するには、**管理** > **エージェント設定**メニューで `mtrace_enabled` オプションを `true` に設定します。 エージェント設定の詳細については、次の文書を参照してください。

マルチトランザクションIDを確認する

マルチTX追跡メニューを利用するには、**MTID**(Multi Transaction ID)が必要です。 次の手順で**MTID**値が確認できます。

1. **分析** > **ヒットマップ**メニューでチャート領域をドラッグします。
2. ドラッグしたチャート領域のトランザクション情報が以下のように下段の**TXトレース**一覧に表示されます。



3. Ⓜ アイコンが表示されたトレースを選択すると**トランザクション情報**画面が表示されます。
4. **レコード要約**タブで**マルチTXID**値を確認できます。



- トランザクションで外部呼び出しをする場合でも、同じ**マルチTXID**が生成されます。 サービスごとにプロジェクトが分離されていても、最初に発行した**マルチTXID**を通じてアプリケーション間のすべてのトランザクションが確認できます。 **トランザクション情報**画面でトランザクショントレースを詳細に分析する方法は、次の文書を参照してください。
- **トランザクション情報**画面で**マルチTXID**を選択すると**マルチTX**タブに移動します。 システム内またはシステム間で発生するさまざまな呼び出し関係を一目で確認でき、どのような問題が発生したかを識別して改善できるようにトランザクションとトレース情報を提供します。

マルチトランザクション追跡機能を利用する

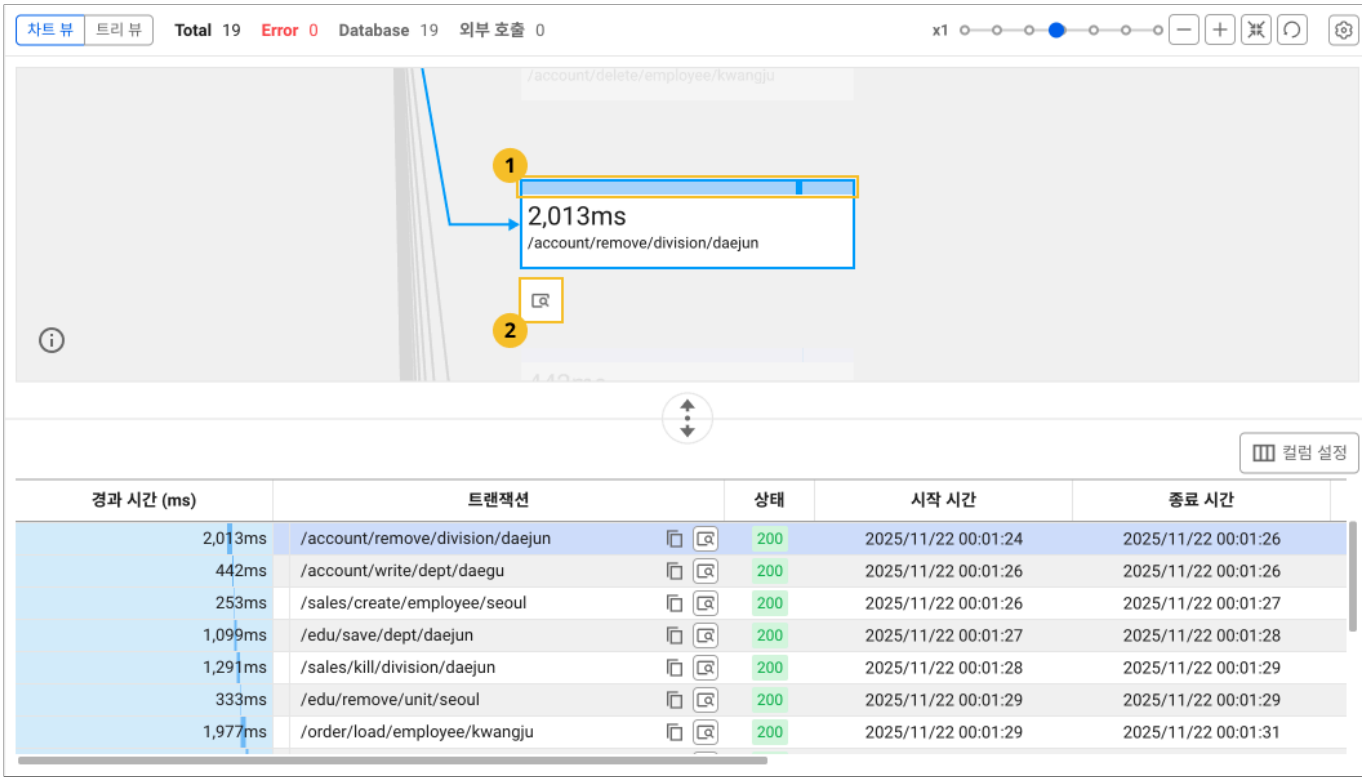
1. **分析** > **マルチTX追跡** メニューに移動します。
2. **トランザクション情報** ウィンドウで確認した **MTID** の値を、**MTID/CUSTID照会** 項目に入力します。
3. 照会する日付とプロジェクトを選択します。
4. 画面下部の **適用** ボタンをクリックします。
   - 次の場合、**適用** ボタンは非活性化されます:
     - 前回の検索と同じフィルタ値（MTID/CUSTID、日付、選択プロジェクト）が設定されている場合
     - フィルタ値（MTID/CUSTID、日付、選択プロジェクト）が一つも入力されていない場合
5. 右側の **チャート** タブに、各トランザクションの呼び出し関係を把握できるダイアグラムとテーブルが表示されます。

ダイアグラムとテーブル領域は上下に分割されており、その比率を自由に調整できます。

- 上矢印をクリックすると、テーブルが全領域を占有します。ただし、ダイアグラム領域が全領域を占有していた場合は、ダイアグラムとテーブルが半分ずつの領域に分割されます。
- 下矢印をクリックすると、チャート（ダイアグラム）領域が全領域を占有します。ただし、テーブル領

チャート

チャート は、各トランザクション間の呼び出し関係を素早く、かつ明確に把握できるビューを提供します。
同じ **マルチTXID** を持つトランザクションサービスの個別実行時間を確認できます。
トランザクションノード上部に表示される所要時間
(1 **タイムバー**) を通じて、トランザクション間の呼び出し関係を把握できます。



チャートはマウスで自由に移動・拡大・縮小できます。

- チャート領域をドラッグ → 好きな位置に移動
- チャート領域でスクロール
  - 上下スクロール：上下移動
  - 左右スクロール：左右移動
- `Ctrl` + 上下スクロール：拡大・縮小
  - タッチパッドのピンチイン/ピンチアウトにも対応

トランザクションを選択すると、接続されているノードだけが強調表示され、関連のないノードは薄く表示されます。
選択したトランザクションの下にある **詳細を見る** ボタン
(2) が有効化され、クリックすると
トランザクション情報 ウィンドウが開き、詳細情報を確認できます。
トランザクション情報 を利用した詳細なトランザクショントレース分析方法は
こちらのドキュメント を参照してください。

- **拡大**
- **縮小**
- **画面に合わせる**：すべてのノードを画面に収まるように表示
- **拡大/縮小をリセット**：初期状態の表示位置とズームに戻す
- **チャートビュー設定**：表示する要素を切り替え
  - **詳細**：アプリ名、プロジェクト情報、IP、ステータスコードの表示/非表示
  - **データベース / 外部呼出し**：当該トランザクションで発生した DB 接続や外部 HTTP 呼び出しをノードとして表示可能

表

表タブでは、マルチトランザクションに含まれている各トランザクションごとの情報をテーブル形式で確認できます。

各トランザクション項目を選択すると**チャート**タブのノード選択と同様に、選択したトランザクションに対する**トランザクション情報**画面が表示されます。 トランザクショントレースを使用して、トランザクションの詳細を確認できます。 **トランザクション情報**画面でトランザクショントレースを詳細に分析する方法は、次の文書を参照してください。

カラム設定：テーブルのカラムを編集できます。

ツリー

ツリータブでは、各トランザクションとそれに属しているトレースの詳細を確認できます。 全体のトランザクション所要時間内での各下位トランザクションやトレースの開始および所要時間を可視化し、トランザクションの呼び出し関係をツリー形式で提供します。

- **一行**：各区間ごとの実行情報に表示されたテキストを一行で表示し、ツリー形式を簡潔にまとめられます。
- **複数行**：各区間ごとの実行情報に表示されたテキストを改行してすべて表示します。
- **最長経路**：最長の経路で移動できます。
- **タイムバーの表示**：経過時間を棒チャートで表示します。
- **時間を表示する**：各区間ごとのタイムスタンプ、ギャップ、経過時間をテキスト形式で表示します。
  - 8秒以上：超過遅延状態で赤色で表現します。
  - 3秒以上8秒未満：遅延状態でオレンジで表現します。
  - 3秒未満：正常な状態で青色で表示します。

  > 開始および所要時間の場合、トランザクション呼び出し環境によって生じる時差を上位トランザクションのトレースやマッピングで修正および表示することで、実際に収集された時間データとの差異が生じる可能性があります。

- **時間を表示しない**：時間情報を非表示にします。

> 該当トレースのボタンまたはボタンを選択すると**HTTPコール統計**、**アクティブスタック**などの情報を確認できます。

> SQL変数とHTTPクエリを照会するには、次のオプションをエージェント設定に追加します。
>
> - SQLパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
> - HTTPパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
>
> whatap.conf

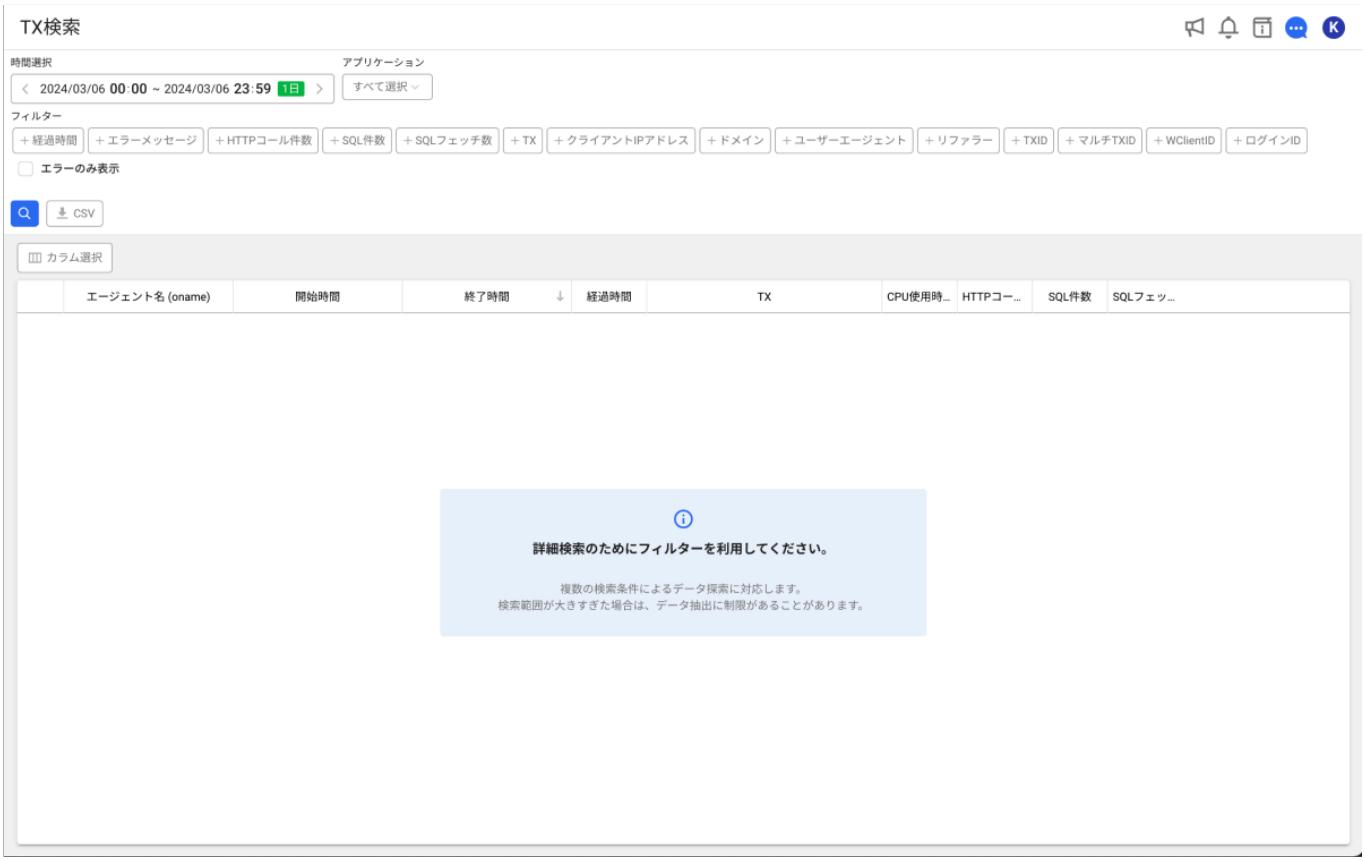
```
# SQLパラメータ照会オプション：オプションが適用されるとSQLパラメータを暗号化して収集します。
profile_sql_param_enabled=true

# HTTPパラメータ照会オプション：オプションが適用されると、HTTPクエリパラメータを暗号化して収集します。
profile_http_parameter_enabled=true
```

TX検索

ホーム画面 > プロジェクト選択 > **分析** > **TX検索**

特定の期間中の各トランザクションの性能属性に関する統計と実行状態を検索できます。 すべてのトランザクションを確認し、正常やエラー状態をすばやく確認できます。 提供される情報を通じて、どのようなトランザクションの使用量が多く、リソースを多く消費しているかがわかります。 トランザクションの属性に応じて様々なフィルタリングが可能で、特定の条件のトランザクションを見つけるのに役立ちます。



照会時間を設定した上で**アプリケーション**オプションで照会対象を選択します。 ボタンを選択すると、検索結果がテーブルに表示されます。 結果リストから希望する項目を選択すると**トランザクション情報**画面が表示されます。 **トランザクション情報**画面を使用したトランザクショントレースの分析に関する詳細は、次の文書を参照してください。

- エラー状態のトランザクションのみを検索するには、**エラーのみ表示**をクリックしボタンを選択しま

> す。
>
> - 検索条件によってレスポンス遅延が発生する可能性があるため、検索時間をできるだけ短く設定することをお勧めします。

カラムを編集する



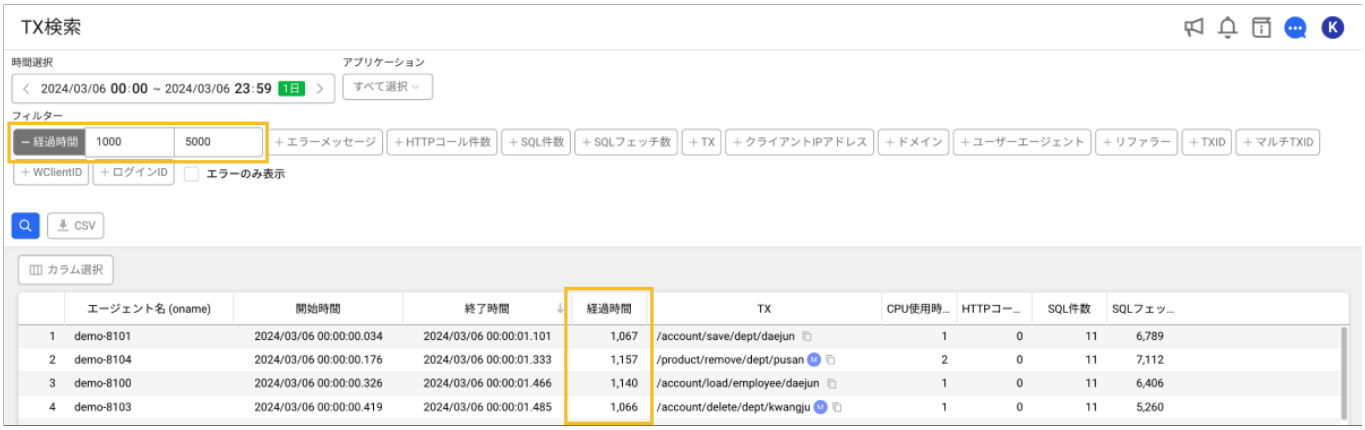
結果照会時、 **カラム選択**ボタンをクリックすると、確認が必要なカラムを適切に選択できます。

フィルター適用

詳細検索のためにフィルターを利用してください。 複数の条件によるデータの検索方法を提供します。 フィルターを設定して ボタンを選択します。

> 一部のフィルター項目は、製品によっては対応できない場合があります。

経過時間



トランザクションの応答時間を基準に検索します。 基準から以上と未満の2つの値を入力して検索します。

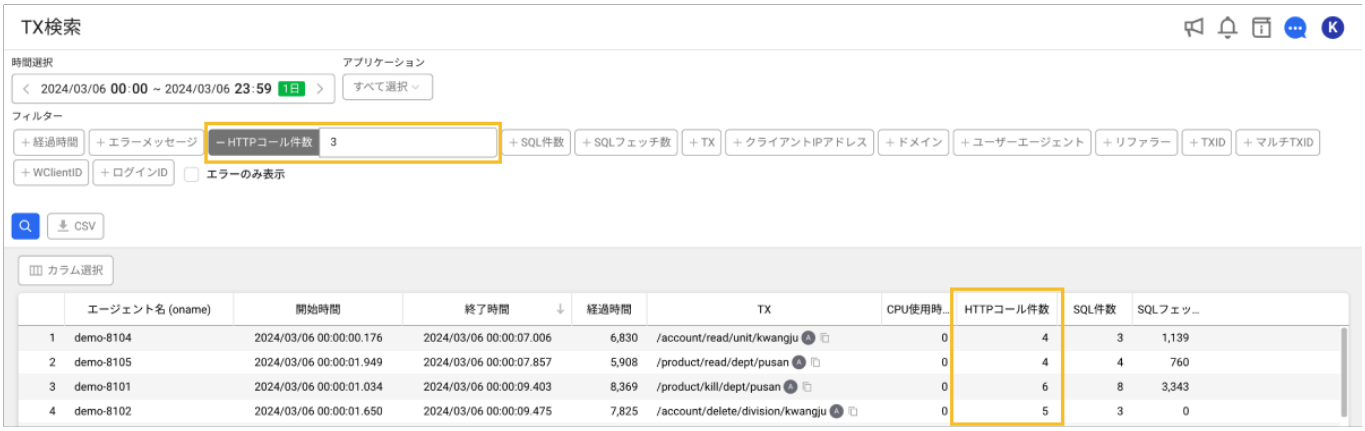
エラーメッセージ



トランザクションのエラーメッセージで検索します。 デフォルト状態ではエラー関連のカラムが表示されない場合があります。 カラムを選択します。

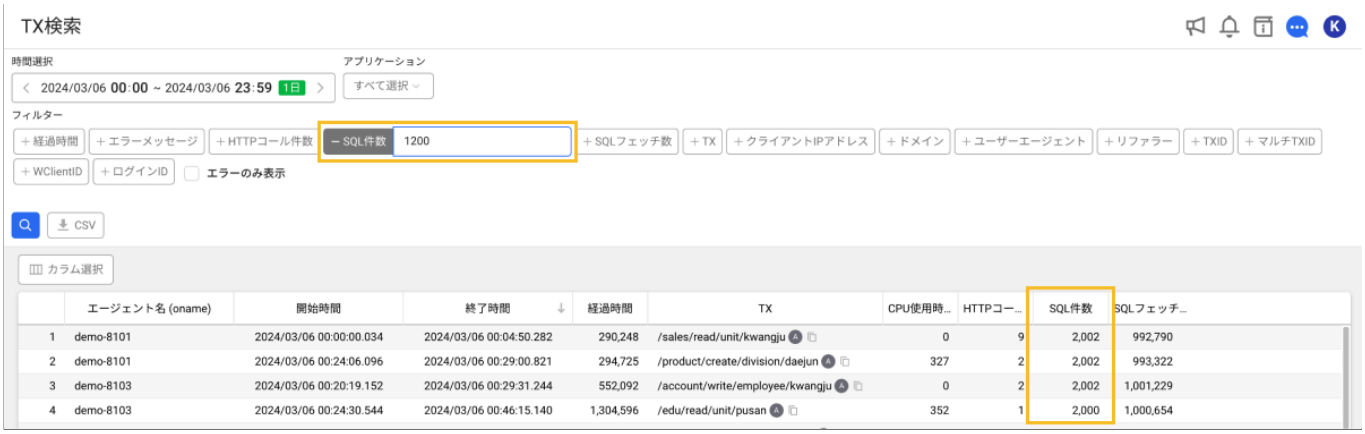
> ⓘ エラー状態のトランザクションは赤で表示されます。

HTTPコール件数



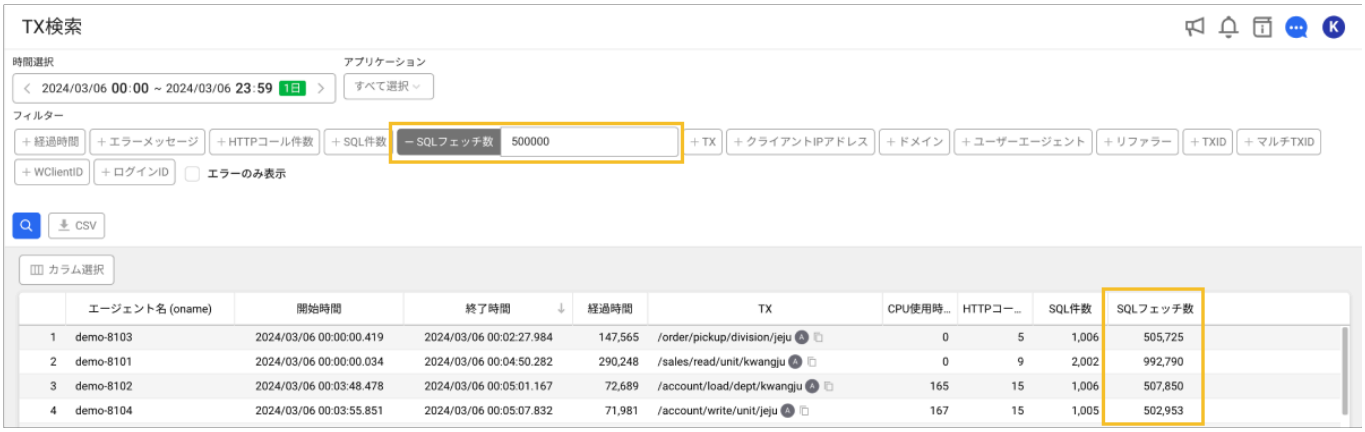
トランザクションのうち、外部HTTPサービスの呼び出し件数に基づいて入力した自然数以上に該当するトランザクションを検索します。

SQL件数



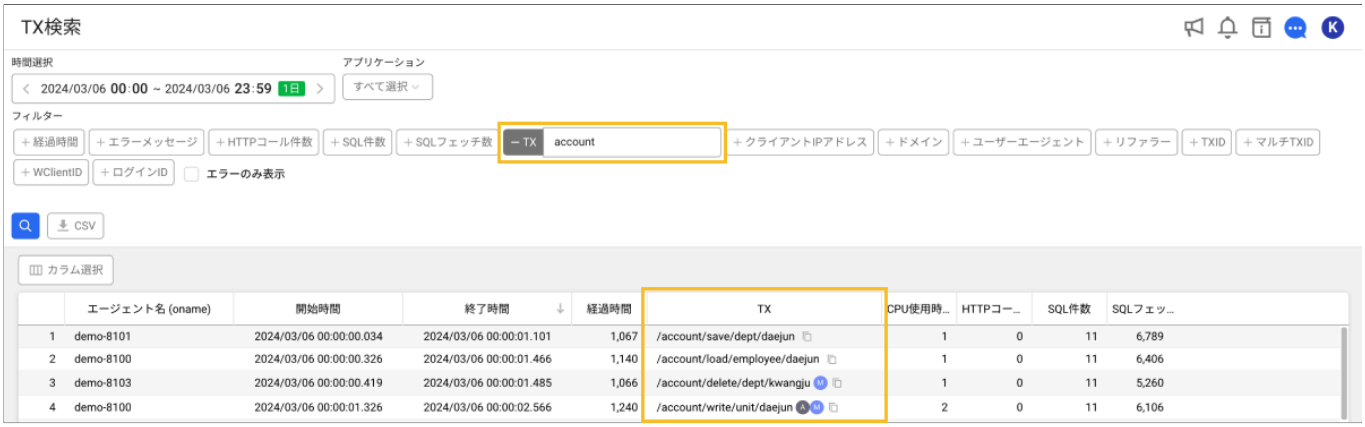
トランザクションごとのSQL実行件数が基準で入力した自然数以上に該当するトランザクションを検索します。 例外的に多い数のSQLを呼び出すトランザクションを検索するときに利用します。

SQLフェッチ数



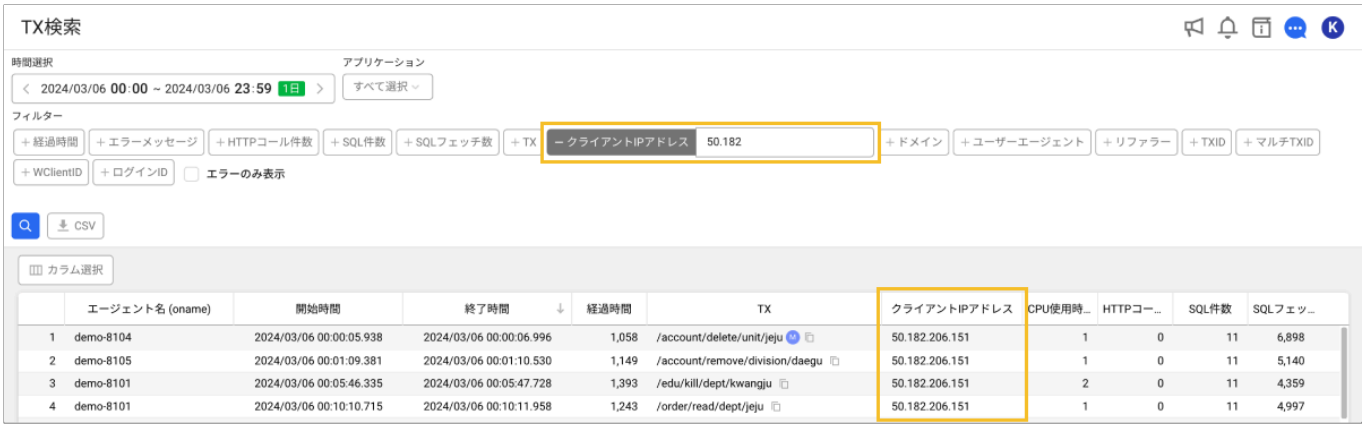
SQL実行後、DBフェッチ件数を基準に入力した自然数以上に該当するトランザクションを検索します。 過度なデータフェッチを実行するトランザクションを検索するときに利用します。

TX



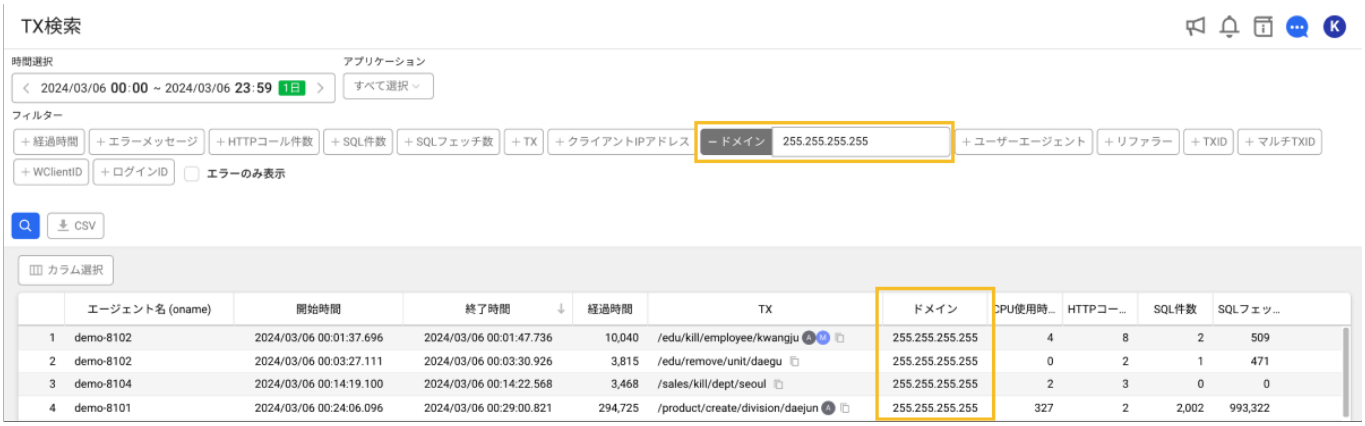
トランザクションに基づいて検索します。 入力した検索文字列は、含まれているか(部分一致)で比較し、検索します。

クライアントIPアドレス



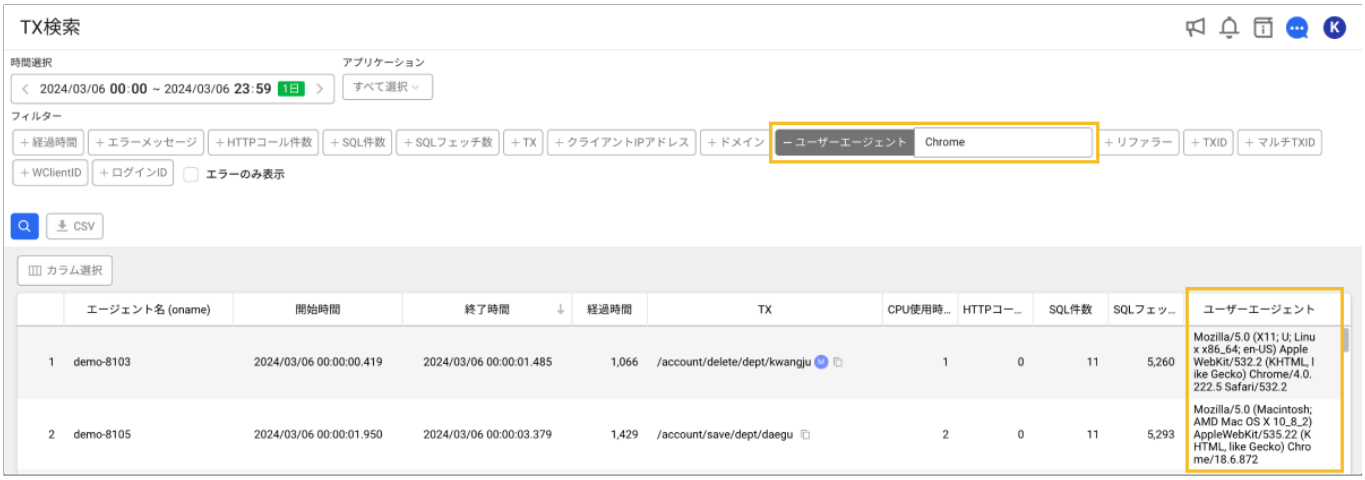
クライアントIPに基づいてトランザクションを検索します。 ただし、IPは、PREFIX比較(前方一致)で検索します。

ドメイン



HTTPサービスドメイン(host:port)ベースでトランザクションを検索します。 検索条件は、ドメインが完全に一致(完全一致)する必要があります。

ユーザーエージェント



ユーザーエージェントベースでトランザクションを検索(部分一致)します。

リファラー

Referer文字列ベースでトランザクションを検索します。

> ⓘ デフォルト設定状態のエージェントは、Refer情報を収集しません。 Referer設定が必要な場合は、次の文書を参照して関連する設定を追加します。

TXID



トランザクションIDベースでトランザクションを検索(完全一致)します。 トランザクションIDの場合、検索インデックスが作成されているため、高速で検索できます。

- トランザクションIDは、**トランザクション情報**ウィンドウの**レコード要約**タブで確認できます。 詳細については、次の文書を参照してください。
- 最新バージョンのエージェントを使用していない場合、インデックスを使用しないフルスキャンが発生する可能性があります。

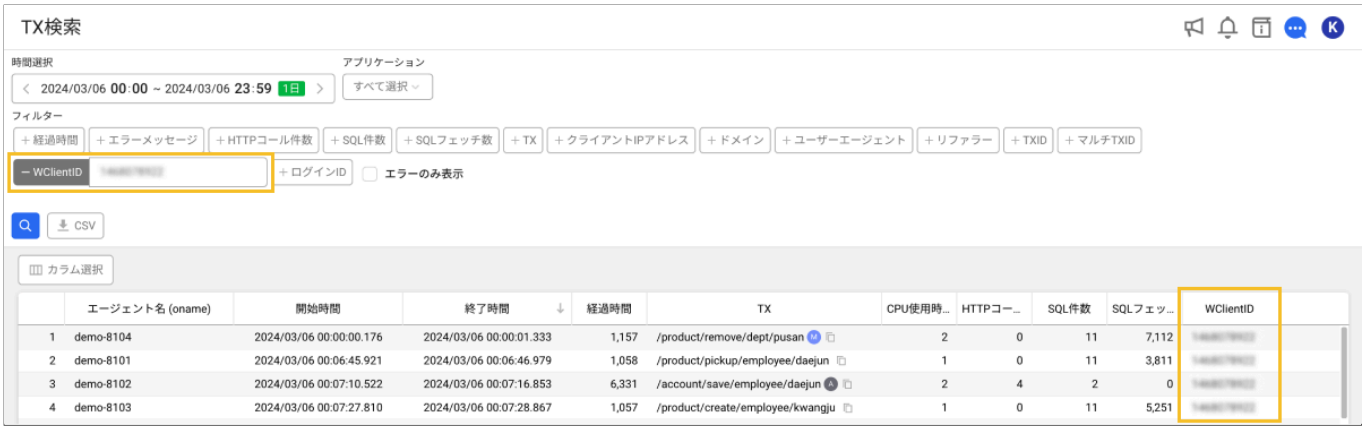
マルチTXID



マルチTXIDベースでトランザクションを検索(完全一致)します。

- マルチトランザクションIDは、**トランザクション情報**ウィンドウの**レコード要約**タブで確認できます。 詳細については、次の文書を参照してください。
- 最新バージョンのエージェントを使用していない場合、インデックスを使用しないフルスキャンが発生する可能性があります。
- **マルチTXID**フィルターを適用して検索する場合、**TXID**フィルターのみ追加できます。

WClientID



クライアントID(`wClientId`)は、WhaTapエージェントがユーザーを区分するために付与するIDです。 このID値に基づいてトランザクションを検索(完全一致)します。 検索のためのインデックスはありません。

> ⓘ WhaTapは、クライアントに関する情報をデフォルトで保存します。

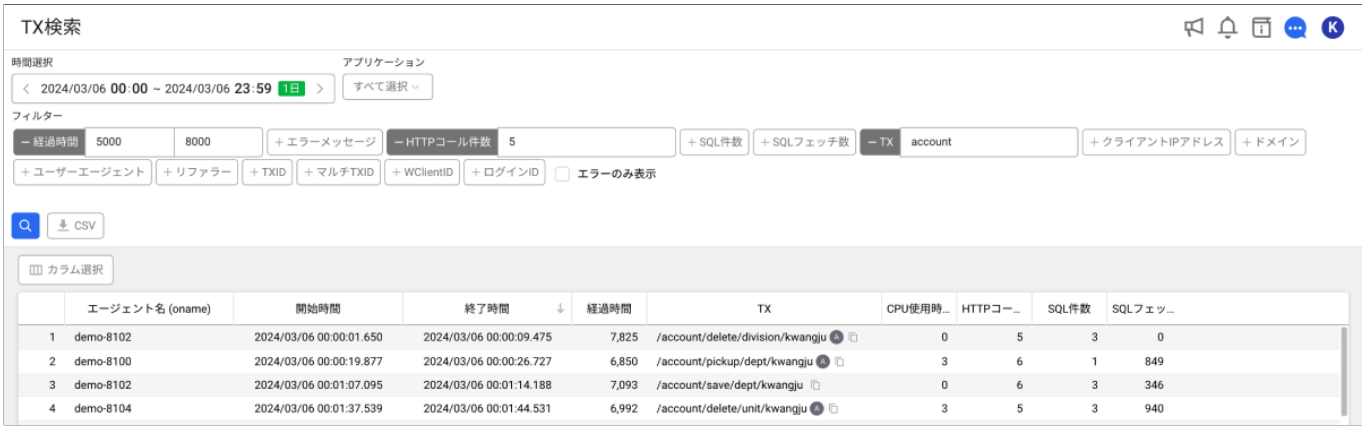
ログインID

ログインIDに基づいてトランザクションを検索します。

> ⓘ 基本設定状態のエージェントでは、ログインID情報を収集しません。 ログインIDの設定が必要な場合は、次の文書を参照して設定を追加します。

複数フィルター適用

複数のフィルター項目を適用して、目的のトランザクション一覧を確認できます。 次の事例は、経過時間が5~8秒の間であり、HTTP呼び出し件数が5件以上、トランザクション経路に'account'が含まれている場合です。



> ⓘ **マルチTXID**フィルターを適用して検索する場合、**TXID**フィルターのみ追加できます。

検索結果のダウンロード

ユーザーが設定したフィルター条件に従って検索された結果をCSVファイルにダウンロードできます。

1. ボタンをクリックして検索結果を照会します。
2. CSVボタンをクリックします。
3. 最大CSV行数を入力します。 入力した行数だけ、照会された結果を保存することができます。
4. **ダウンロード**ボタンを選択するとCSVファイルでダウンロードします。

> ⓘ CSVファイル名は、profile_ {pcode} _ YYMMDD .csv形式です。

MSA分析

MSA分析とは？

MSA分析機能は、WhaTapの**マルチTX追跡**機能を使用して収集されたトランザクション間の呼び出しの割合を分析します。 これによって、マイクロサービス間の依存性をインスタンス中心ではなくURLレベルで確認できます。



例えば、A、B、C、D、Eという5つのマイクロサービスがあります。ユーザーがAを呼び出すと、ロジックに従って次のような3つの呼び出しパターンが発生すると仮定します。

1. AからBとCを呼び出し
2. AからBとCを呼び出し + またBからDを呼び出し
3. AからBとCを呼び出し + またBからDとEを呼び出し

この3つのパターンが一度ずつ実行された場合、MSA分析では基準URLに従って関連度分析のデータが確認できます。

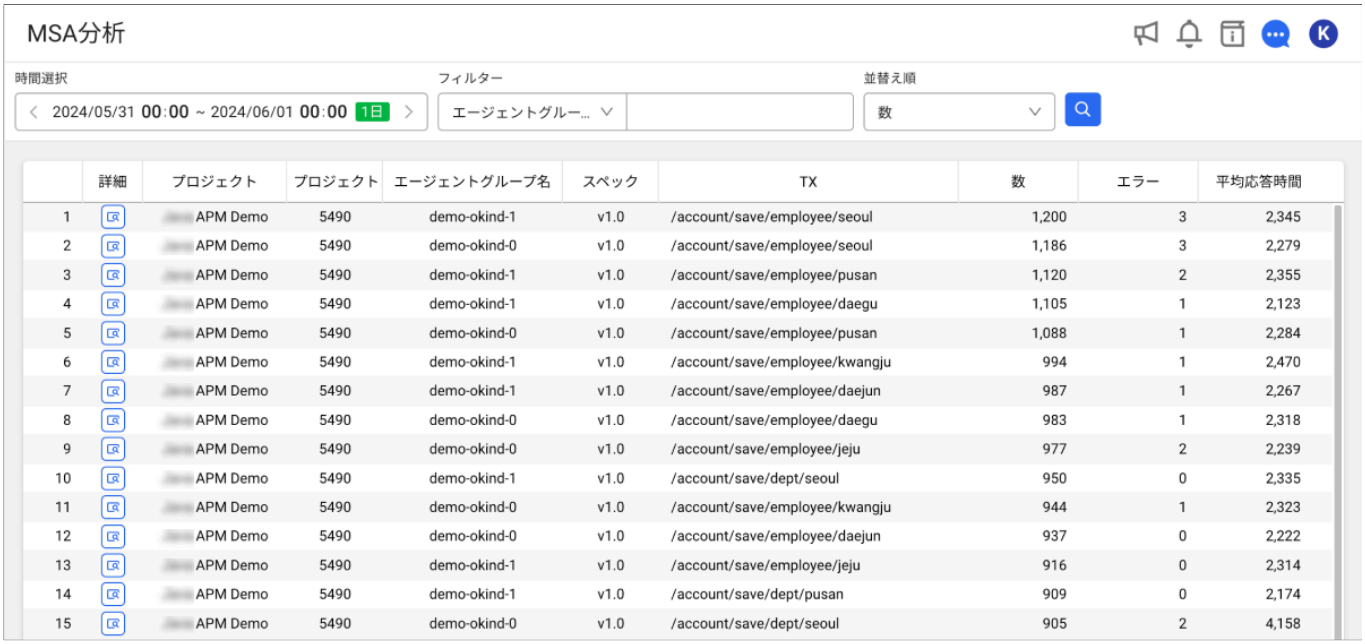


> **マルチトランザクション追跡**に関する詳細は、次の文書を参照してください。

MSAトランザクションの統計

ホーム画面 > プロジェクト選択 > **サイトマップ** > **分析** > **MSA分析**

MSA分析メニューは、マイクロサービス環境で各サービスの性能と安全性を効果的に分析できるツールです。 トランザクションの詳細な呼び出し関係を分析し、これを基づいてシステムの性能を最適化することで、潜在的な問題を事前に識別できます。



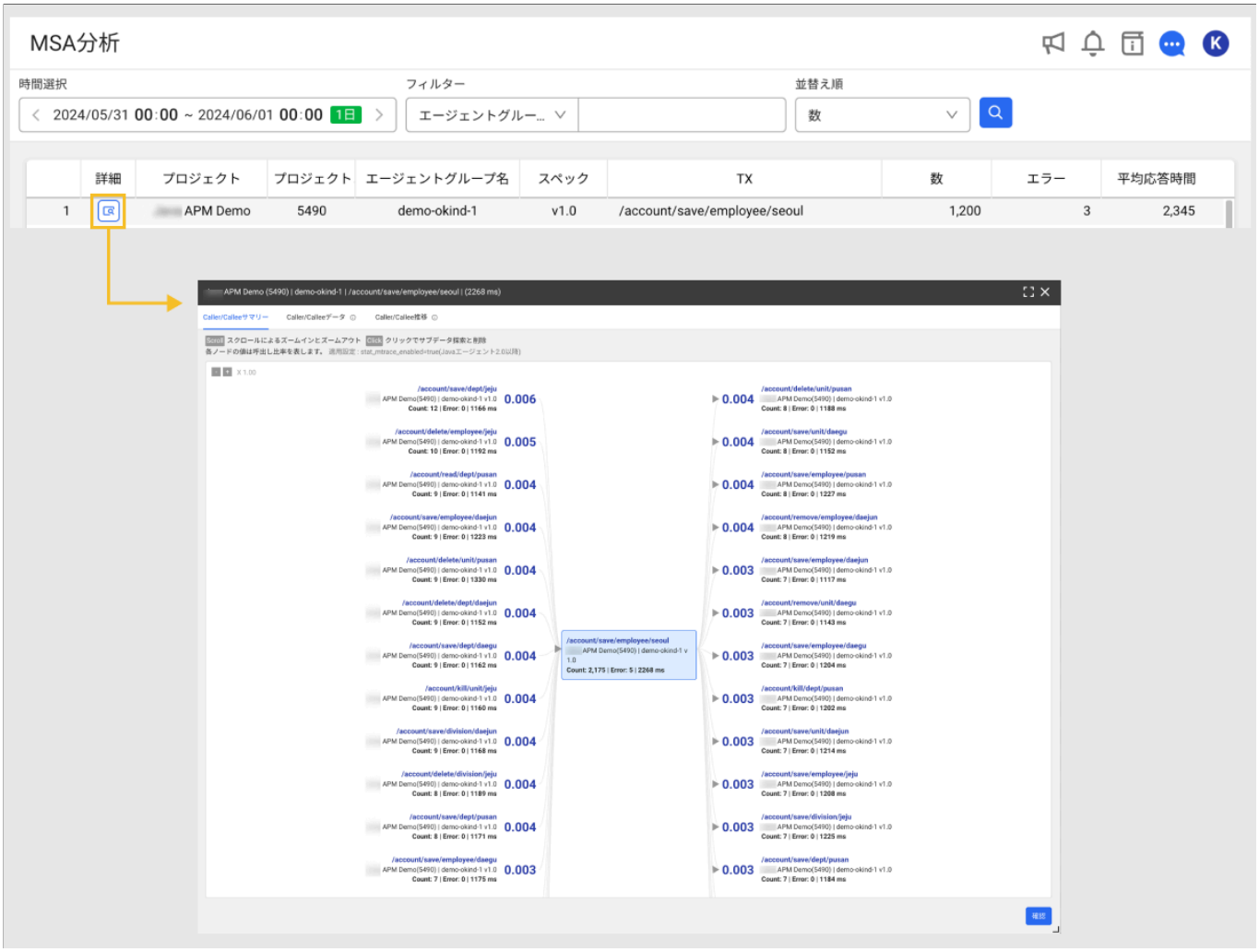
	詳細	プロジェクト	プロジェクト	エージェントグループ名	スペック	TX	数	エラー	平均応答時間
1		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/seoul	1,200	3	2,345
2		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/seoul	1,186	3	2,279
3		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/pusan	1,120	2	2,355
4		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/daegu	1,105	1	2,123
5		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/pusan	1,088	1	2,284
6		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/kwangju	994	1	2,470
7		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/daejun	987	1	2,267
8		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/daegu	983	1	2,318
9		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/jeju	977	2	2,239
10		APM Demo	5490	demo-okind-1	v1.0	/account/save/dept/seoul	950	0	2,335
11		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/kwangju	944	1	2,323
12		APM Demo	5490	demo-okind-0	v1.0	/account/save/employee/daejun	937	0	2,222
13		APM Demo	5490	demo-okind-1	v1.0	/account/save/employee/jeju	916	0	2,314
14		APM Demo	5490	demo-okind-1	v1.0	/account/save/dept/pusan	909	0	2,174
15		APM Demo	5490	demo-okind-0	v1.0	/account/save/dept/seoul	905	2	4,158

- **時間選択**: 照会期間を選択します。 基本1日を基準に、Caller/Calleeを含むトランザクションの一覧を照会できます。
- **フィルター**: **エージェントグループ名**, **スペック**, **TX** 基準でフィルタリングし、検索できます。
- **並替え順**: **数**, **エラー**, **平均時間**(平均の応答時間) 基準で一覧をソートできます。

> ⓘ
> - **時間選択**で照会できる時間範囲は、最大1日(24時間)となります。 **時間選択**オプションの利用方法の詳細については。
> - **エージェントグループ名**は、エージェント設定オプションの'whatap.okind'に分類されたグループ名です。
> - 条件を設定してからトランザクション一覧を更新するには、ボタンを選択します。

詳細分析する

トランザクション一覧で、照会する項目のボタンを選択すると、詳細表示画面が表示されます。 詳細表示画面で、該当トランザクションに対する呼び出し元(Caller)、呼び出し先(Callee) 情報を照会できます。



各タブでは、次の分析機能を提供します。

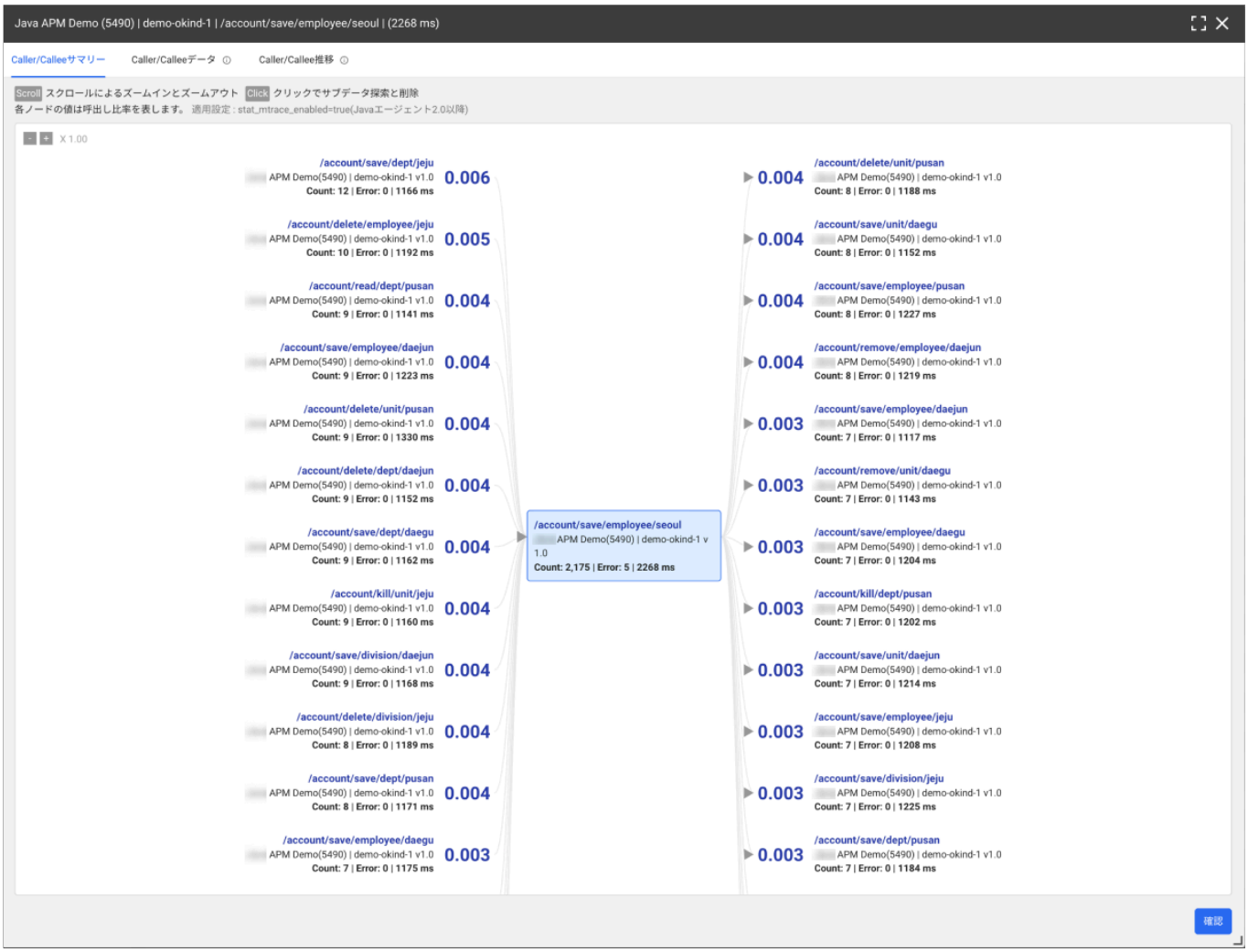
- **Caller/Calleeサマリー**: トランザクションの呼び出し元と呼び出し先の間の関係を、視覚化されたダイアグラムで提供します。
- **Caller/Calleeデータ**: トランザクションの呼び出し回数、エラー数、応答時間を詳細に提供します。
- **Caller/Callee推移**: 時間によるトランザクションの呼び出し回数とレートの性能変化を、時系列チャートで提供します。

> ⓘ
> - 詳細画面の右上にある[ ]ボタンを選択すると、全画面で確認できます。
> - 詳細画面を閉じるには、右下にある**確認**ボタンを選択するか、ESCキーを押します。

Caller/Calleeサマリー

Caller/Calleeサマリータブでは、トランザクション一覧で選択したトランザクションのCaller、Callee間の依存性情報を視覚化されたダイアグラムで表示します。 サービス間の呼び出し関係を、直観的に理解できます。 呼び出し経路と従属性が視覚的に把握でき、問題の原因分析に便利です。

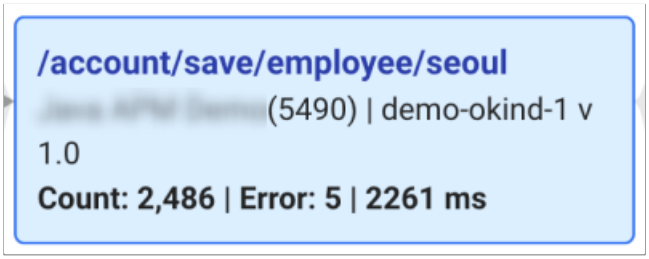
呼び出し関係の分析



CallerとCalleeノードには、最大0.004、0.003など最大1人の数字情報を表示しますが、これは全体のCallerの中で占める割合を意味します。 同様に各Calleeの比重は、右側に一緒に表示します。

上の画面から_/account/save/employee/seoul_というトランザクションを基準に、CallerとCallee間の依存関係を分析できます。 トランザクションがどのサービスから呼び出され、どのサービスに呼び出されるかを視覚的に確認できます。

性能指標の確認

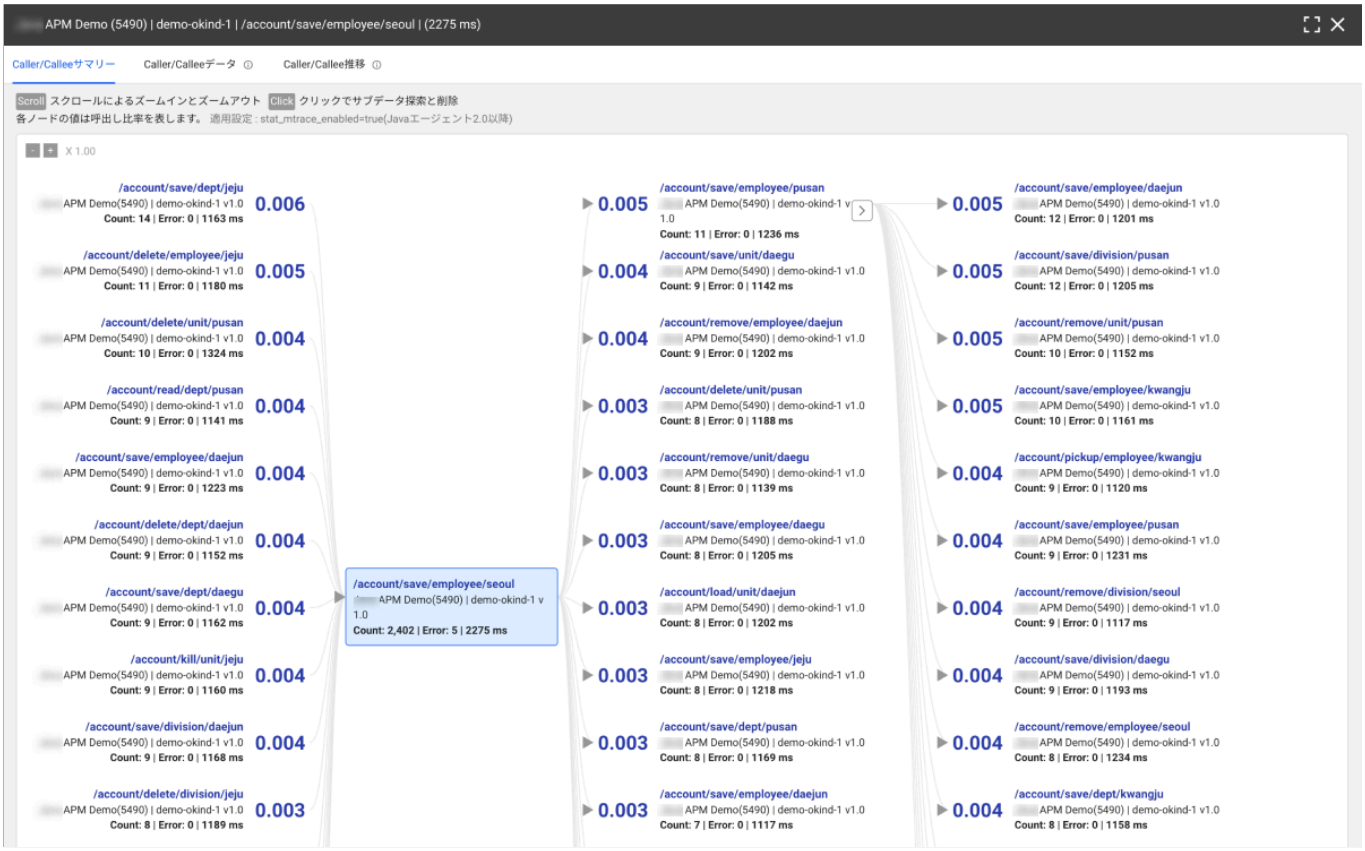


各ノードはトランザクションURL、プロジェクト名、エージェントのグループ名、そして次の性能情報で構成されています。

- **Count**: 当該トランザクションを呼び出した回数
- **Error**: 呼び出し中に発生したエラー数
- 平均応答時間: 当該トランザクションの平均の応答時間、ミリ秒(ms) 単位

トランザクションの流れを把握

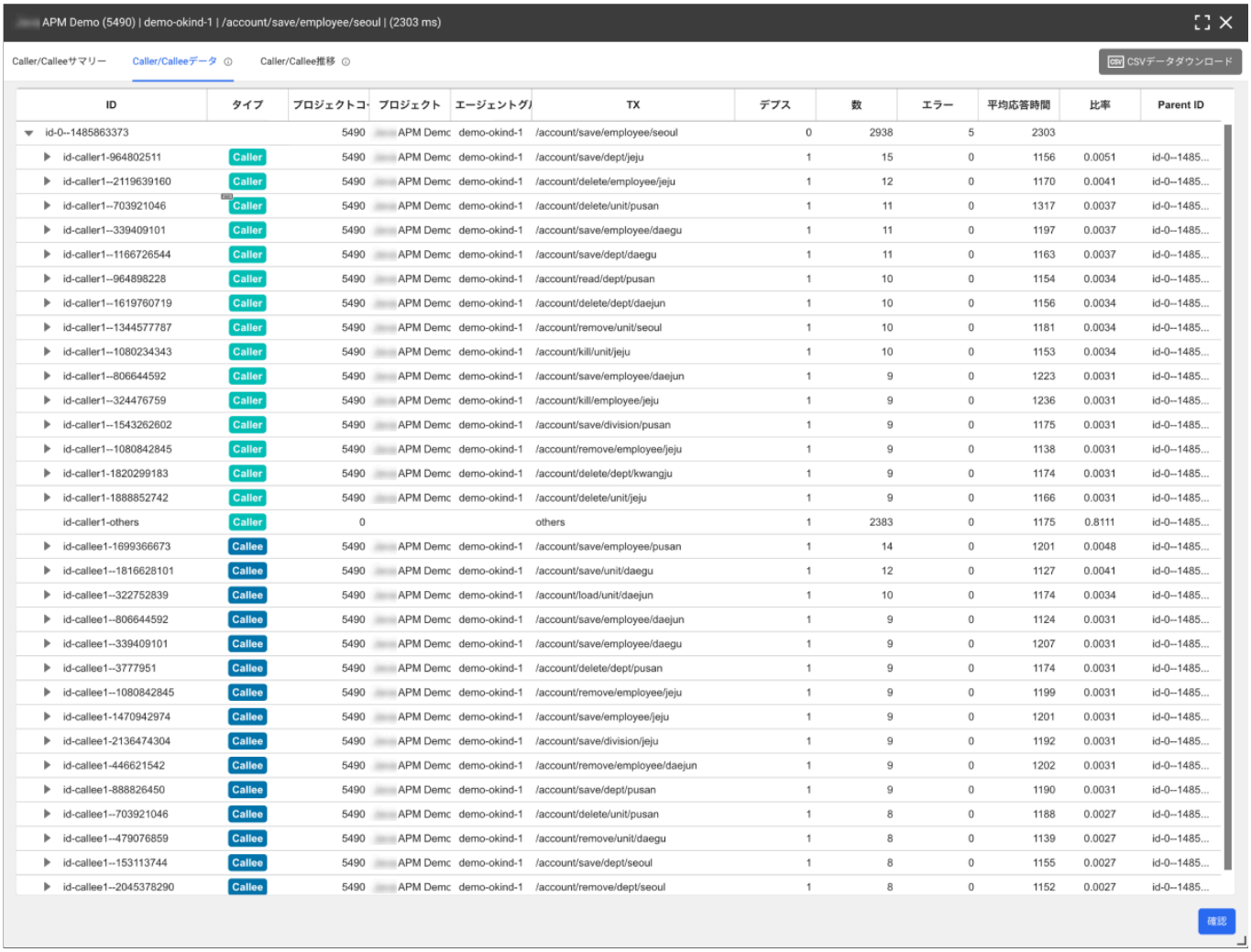
CallerやCalleeノードにマウスをオーバーすると、 > ボタンが表示されます。 このボタンを選択すると、下位ノードの詳細に展開できます。 トランザクションの全体的な流れを把握して、問題の発生元を見つけるのに役に立ちます。



- 下位ノードを閉じるには、もう一度 > ボタンを選択します。
- マウスのスクロール機能により、画面を拡大または縮小できます。 または画面左上の+ / -ボタンをクリックします。

Caller/Calleeデータ

Caller/Calleeデータタブでは、基準URL(例: /account/save/employee/seoul)に対する呼び出し元(Caller)と呼び出し先(Callee)の相関関係を、表の形式で確認できます。 各トランザクションの詳細情報(呼び出し回数、エラー数、平均応答時間など)が具体的に確認でき、データ基盤で精密に分析できます。

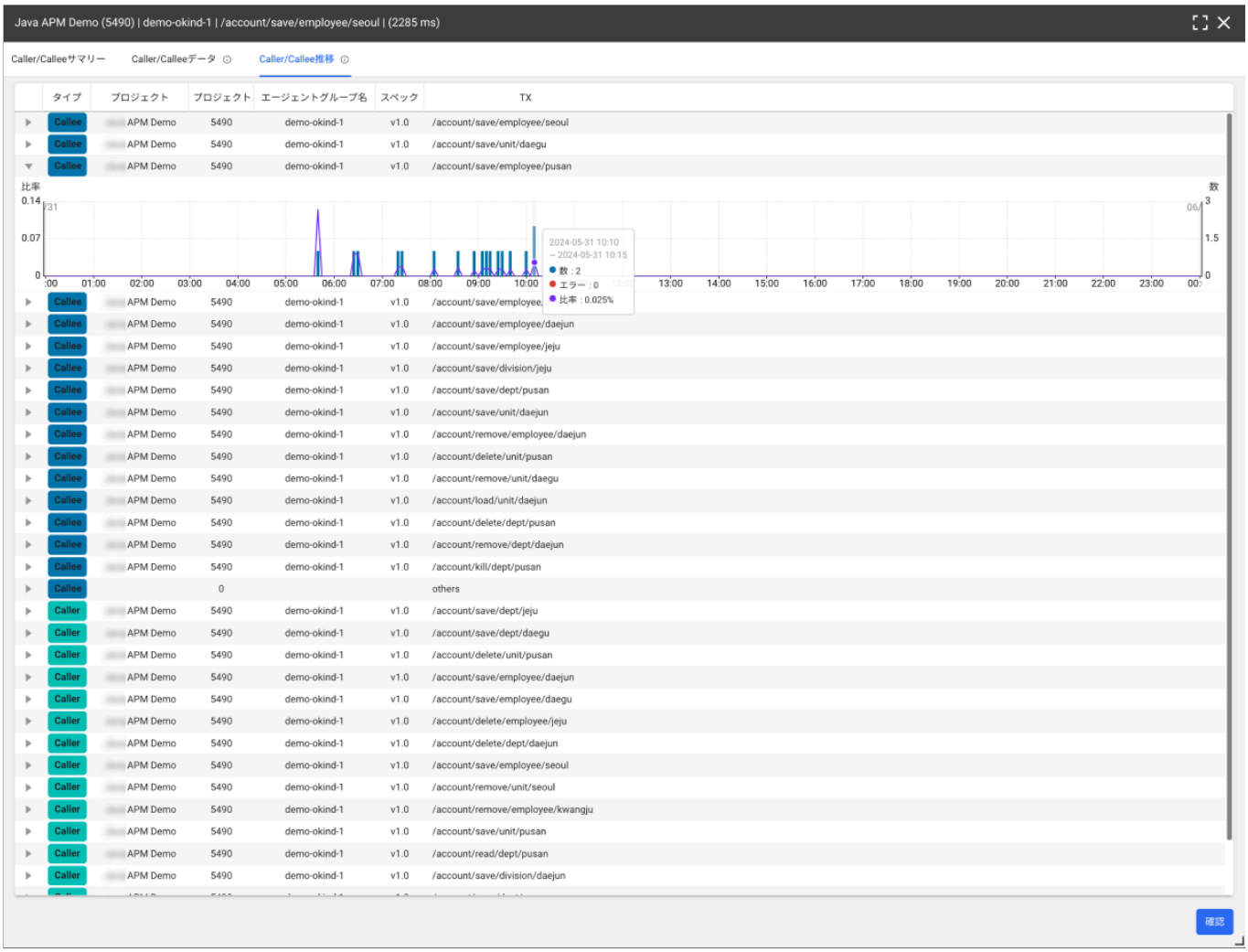


- 各トランザクションの呼び出し回数と割合、エラー数、平均応答時間などを確認し、性能問題を分析するのに活用できます。
- **タイプ**カラムを通じてCallerとCalleeを簡単に区別することができ、呼び出し関係を把握できます。
- 各トランザクションの ▶ ボタンを選択すると、そのトランザクションに関連するトランザクションを、追加で確認できます。
- データ一覧をCSV形式で保存するには、右上の**CSVデータダウンロード**ボタンを選択します。

Caller/Callee推移

Caller/Callee推移タブでは、基準URL(/account/save/employee/seoul)の呼び出し関係と性能データを時間の経過と

ともに視覚的に分析できます。 これにより、トランザクションの性能変動パターンを把握できます。



- 特定のトランザクションの呼び出し回数と割合を、時間帯別グラフで可視化し、性能の変化を一目で把握できます。
- **タイプ**カラムを通じてCallerとCalleeを簡単に区別することができ、呼び出し関係を把握できます。
- 時系列チャートにマウスをオーバーすると、ツールチップを通じて数値情報を確認できます。
- チャートの特定の時間帯をクリックすると、**分析** > **TX検索**メニューへ移動します。 選択した時間帯のトランザクション一覧を照会できます。

Javaメトリクス

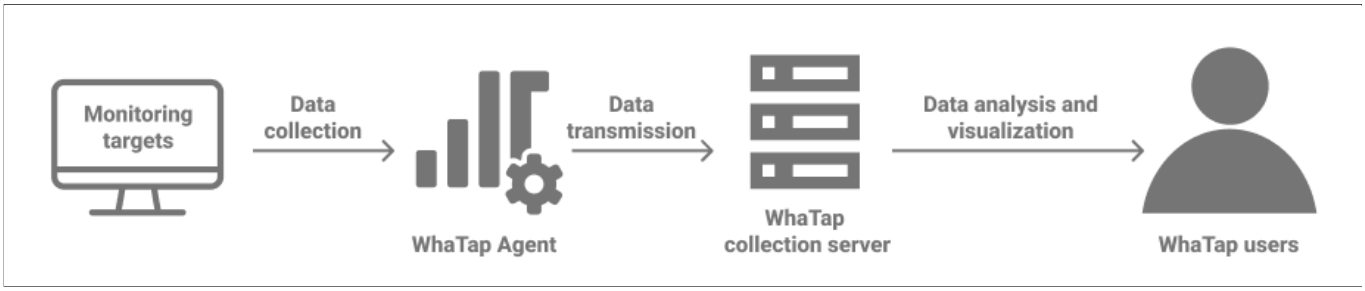
メトリクスとは?

メトリクスはモニタリング対象の状態を数値で表現した時系列データです。WhaTapエージェントがモニタリング対象から収集した性能データを加工した結果です。

メトリクスを通じてサーバー別メモリ使用率、DB接続時間などの主要な性能指標を一目で把握できます。問題を発見した後はログとトレースを通じて詳細分析ができます。

リソース使用量の統計を基に必要なリソース量を算定でき、コスト効率化にも活用できます。

WhaTapのメトリクス収集方式



メトリクスを収集するにはモニタリング対象にWhaTapエージェントをインストールする必要があります。エージェントが送信したメトリクスはWhaTap収集サーバーに保存されます。

例えば**Javaアプリケーション**をモニタリングするにはまずエージェントをインストールしてください。収集可能なメトリクスはアプリケーションメトリクスで確認できます。

WhaTapのメトリクス構成要素

WhaTapの**メトリクス**は以下の要素で構成されています。

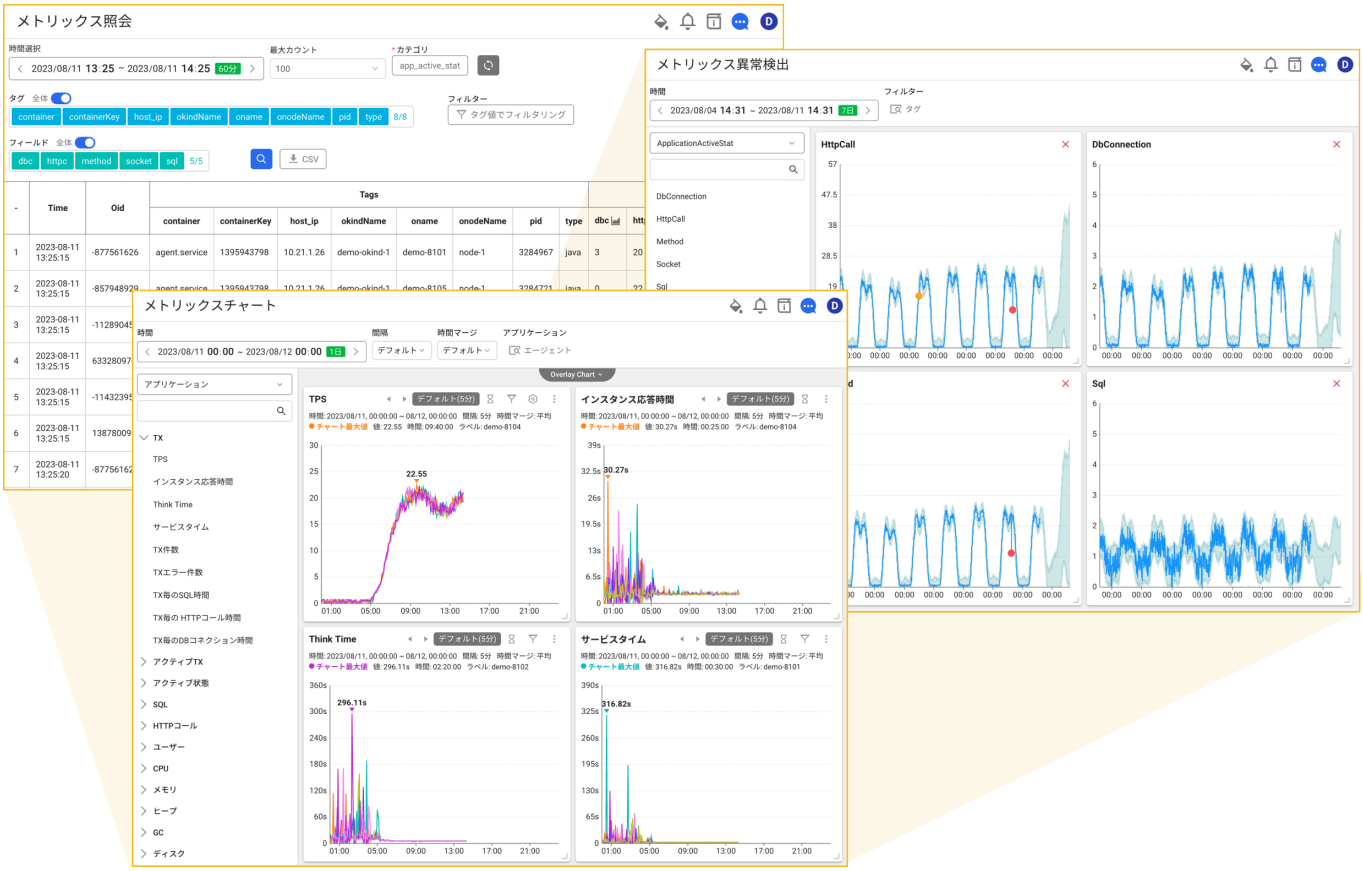
- **Category**: 関連メトリクスを1つのグループにまとめる名前です。

- **Tags**: 収集対象を識別するラベルです。IP、Oname、Hostなど変更の少ない固有情報を保存します。1つのメトリクスに複数のタグが含まれることがあります。
- **Fields**: 実際の測定値を保存します。1つのメトリクスに複数のフィールドが含まれることがあります。
- **Time**: メトリクスを収集した時間です。
- **Oid**: メトリクスを収集したエージェントの固有番号です。
- **Oname**: メトリクスを収集したエージェントの名前です。

メトリクスデータの照会と可視化

WhaTapは収集したメトリクスをさまざまな方式で照会できるメニューを提供します。

- **メトリクス照会**: メトリクスの原本データを照会します。
- **メトリクスチャート**: 可視化したチャートを通じてメトリクスデータを照会します。
- **メトリクス異常検出**: AIが学習したメトリクスパターンと比較して予想パターンを外れた異常を探知します。


メトリックス照会
メトリックス異常検出
メトリックスチャート
TPS
インスタンス応答時間
Think Time
サービスタイム
HttpCall
DbConnection
Sql

アプリケーションメトリクス

ホーム画面 > プロジェクト選択 > **分析** > **メトリクス照会**

メトリクス照会メニューから検索できるメトリクスです。メトリクスは以下の情報を含みます。一部のメトリクスにはタグ情報がない場合があります。

- **Tags**: 収集対象を識別する情報（エージェント名、IPなど）
- **Fields**: 実際の測定値（CPU使用率、応答時間など）

---

agent_status_summary

エージェントの状態に関するメトリクスを10秒間隔で収集したカテゴリです。

Fields

Field	Type	Unit	Description
inActTime	-	ミリ秒(ms)	エージェントが無効になったままの時間
isActive	Boolean	-	現在のエージェントの活性化状態
isRestart	Boolean	-	エージェントが最近再起動したかどうか (`true` / `false`)
lastActTime	-	ミリ秒(ms)	最後にエージェントが活性化された状態の時刻 (`0`: 非活性化された場合)
oid	-	-	プロジェクトに含まれる各エージェントの固有の識別子
startTime	-	ミリ秒(ms)	エージェントが開始された時点でのタイムスタンプ

app_active_stat

- アクティブTX区間(Step)関連メトリクスを収集
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Tags

Tag	Type	Unit	Description
container	-	-	コンテナ名(固有値)
containerKey	-	-	コンテナキー値 (固有値)
host_ip	-	-	ホストIPアドレス(固有値)
okindName	-	-	whatap.okindに分類されたグループ名
oname	-	-	エージェント名 (固有値)
onodeName	-	-	whatap.onodeに分類されたグループ名
pid	-	-	プロセスの固有識別子(ID)
type	-	-	プラットフォームタイプ(アプリケーションの開発言語)

Fields

Field	Type	Unit	Description
dbc	-	件数	Database Connectionが実行されているトランザクションの件数
httpc	-	件数	HTTP Callが行われたトランザクションの件数

Field	Type	Unit	Description
method	-	件数	Methodロジックによって実行されているトランザクションの件数
socket	-	件数	Socket接続が実行されているトランザクションの件数
sql	-	件数	SQLクエリが実行されているトランザクションの件数

---

app_actx_meter

- アクティブTXのHTTPリクエスト、SQLクエリ、データ転送など様々な活動メトリクスを収集
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Tags

Tag	Type	Unit	Description
alias	-	-	エージェントの別称
container	-	-	コンテナ名
containerKey	-	-	コンテナキーの値
host_ip	-	-	ホストのIPアドレス
okindName	-	-	whatap.okindに分類されたグループ名
oname	-	-	エージェント名
onodeName	-	-	whatap.onodeに分類されたグループ名

Fields

Field	Type	Unit	Description
active_httpc	-	件数	実行中のHTTPクライアントのリクエスト数
active_sql	-	件数	実行中のSQLクエリ数
data	-	バイト	データ伝送量
txcaller_group	-	Int	トランザクション呼び出し元のグループを識別するための定数
txcaller_oid	-	Int	トランザクション呼び出し元のエージェントIDを識別するための定数
txcaller_unknown	-	Int	不明なトランザクション呼び出し元を識別するための定数

---

app_counter

- アプリケーションの様々な性能メトリクスおよび統計を収集
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Tags

Tag	Type	Unit	Description
alias	-	-	アプリケーション別称(固有値)
container	-	-	コンテナ名(固有値)
containerKey	-	-	コンテナキー値 (固有値)
host_ip	-	-	ホストIPアドレス(固有値)

Tag	Type	Unit	Description
okindName	-	-	whatap.okindに分類されたグループ名
oname	-	-	エージェント名 (固有値)
onodeName	-	-	`whatap.onode`に分類されたグループ名
pid	-	-	プロセスの固有識別子(ID)
type	-	-	プラットフォームタイプ(アプリケーションの開発言語)

Fields

Field	Type	Unit	Description
active_tx_0	-	件数	3秒以下の間隔トランザクション数
active_tx_3	-	件数	3秒超過、8秒以下の間隔トランザクション数
active_tx_8	-	件数	8秒超過間隔のトランザクション数
active_tx_count	-	件数	実行されているトランザクションの合計数
apdex_satisfied	-	件数	APDEX満足数
apdex_tolerated	-	件数	APDEX許容数
apdex_total	-	件数	APDEXトランザクションの総数
arrival_rate	-	パーセント	トランザクションの要求率
httpc_count	-	件数	HTTP外部呼び出し数
httpc_error	-	件数	HTTP外部呼び出しのエラー数

Field	Type	Unit	Description
httpc_time	-	ミリ秒 (ms)	HTTP外部呼び出しの平均時間
metering	-	コア	アプリケーションが動作しているホストのコア数 (コンテナの場合limit cpu)
resp_time	-	ミリ秒 (ms)	平均応答時間
sql_count	-	件数	実行を終了したSQL件数
sql_error	-	件数	SQLエラーの件数
sql_fetch_count	-	件数	SQLFetchの件数
sql_fetch_time	-	ミリ秒 (ms)	SQLFetchの実行時間
sql_time	-	ミリ秒 (ms)	SQLの平均実行時間
tps	-	件数	一秒あたりのトランザクション処理数
tx_count	-	件数	トランザクションの処理件数
tx_dbc_time	-	ミリ秒 (ms)	DB平均接続時間
tx_error	-	件数	トランザクションのエラー件数
tx_httpc_time	-	ミリ秒	HTTPリクエストの平均時間

Field	Type	Unit	Description
		(ms)	
tx_sql_time	-	ミリ秒 (ms)	トランザクションあたりのSQL実行時間の合計の平均
tx_time	-	ミリ秒 (ms)	トランザクション実行時間
tx_time_sqr_sum	-	件数	トランザクション実行時間の平方和、トランザクション実行時間の分散や標準偏差といった統計的指標を求める時に使用

---

app_counter_okind

- `app_counter` メトリクスを `whatap.okind` グループ基準で集計
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Tags

Tag	Type	Unit	Description
okindName	-	-	whatap.okindに分類されたグループ名

Fields

Field	Type	Unit	Description
active_tx_0	-	件数	3秒以下の間隔トランザクション数

Field	Type	Unit	Description
active_tx_3	-	件数	3秒超過、8秒以下の間隔トランザクション数
active_tx_8	-	件数	8秒超過間隔のトランザクション数
active_tx_count	-	件数	実行されているトランザクションの合計数
apdex	-	number	APDEX(アプリケーションの顧客満足度)
apdex100	-	パーセント	apdexフィールドのパーセンテージ
apdex_satisfied	-	件数	APDEX満足数
apdex_tolerated	-	件数	APDEX許容数
apdex_total	-	件数	APDEXトランザクションの総数
arrival_rate	-	パーセント	トランザクションの要求率
num_of_agent	-	個数	グループ内で有効になったエージェントの数
realtime_user	-	件数	同時接続ユーザー数
sql_count	-	件数	実行を終了したSQL件数
sql_error	-	件数	SQLエラーの件数
sql_fetch_count	-	件数	SQLFetchの件数
sql_fetch_time	-	ミリ秒(ms)	SQLFetchの実行時間
sql_time	-	ミリ秒(ms)	SQLの平均実行時間
tps	-	件数	一秒あたりのトランザクション処理数
tx_count	-	件数	トランザクションの処理件数

Field	Type	Unit	Description
tx_dbc_time	-	ミリ秒(ms)	DB平均接続時間
tx_error	-	件数	トランザクションのエラー件数
tx_httpc_time	-	ミリ秒(ms)	HTTPリクエストの平均時間
tx_sql_time	-	ミリ秒(ms)	トランザクションあたりのSQL実行時間の合計の平均
tx_time	-	ミリ秒(ms)	トランザクション実行時間

---

app_counter_project

- `app_counter` メトリクスをプロジェクト基準で集計
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Fields

Field	Type	Unit	Description
active_tx_0	-	件数	3秒以下の間隔トランザクション数
active_tx_3	-	件数	3秒超過、8秒以下の間隔トランザクション数
active_tx_8	-	件数	8秒超過間隔のトランザクション数
active_tx_count	-	件数	実行されているトランザクションの合計数
apdex	-	number	APDEX(アプリケーションの顧客満足度)
apdex100	-	パーセント	apdexフィールドのパーセンテージ

Field	Type	Unit	Description
apdex_satisfied	-	件数	APDEX満足数
apdex_tolerated	-	件数	APDEX許容数
apdex_total	-	件数	APDEXトランザクションの総数
arrival_rate	-	パーセント	トランザクションの要求率
num_of_agent	-	個数	プロジェクト内で有効になったエージェントの数
realtime_user	-	件数	同時接続ユーザー数
sql_count	-	件数	実行を終了したSQL件数
sql_error	-	件数	SQLエラーの件数
sql_fetch_count	-	件数	SQLFetchの件数
sql_fetch_time	-	ミリ秒(ms)	SQLFetchの実行時間
sql_time	-	ミリ秒(ms)	SQLの平均実行時間
tps	-	件数	一秒あたりのトランザクション処理数
tx_count	-	件数	トランザクションの処理件数
tx_dbc_time	-	ミリ秒(ms)	DB平均接続時間
tx_error	-	件数	トランザクションのエラー件数
tx_httpc_time	-	ミリ秒(ms)	HTTPリクエストの平均時間
tx_sql_time	-	ミリ秒(ms)	トランザクションあたりのSQL実行時間の合計の平均
tx_time	-	ミリ秒(ms)	トランザクション実行時間

app_host_resource

- アプリケーションサーバーのリソースメトリクスを収集
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Tags

Tag	Type	Unit	Description
alias	-	-	アプリケーションの別称
container	-	-	コンテナ名
containerKey	-	-	コンテナキーの値
host_ip	-	-	ホストのIPアドレス
okindName	-	-	whatap.okindに分類されたグループ名
oname	-	-	エージェント名
onodeName	-	-	whatap.onodeに分類されたグループ名
pid	-	-	特定プロセスの固有識別子(ID)
type	-	-	プラットフォームタイプ(アプリケーションの開発言語)

Fields

Field	Type	Unit	Description
cpu	-	percent	ホストCPU使用率

Field	Type	Unit	Description
cpu_cores	-	整数	ホストCPUコア数
cpu_irq	-	percent	ホストCPU IRQ使用率
cpu_proc	-	percent	JavaプロセスのCPU使用率
cpu_steal	-	percent	ホストCPU Steal使用率
cpu_sys	-	percent	ホストCPU SYS使用率
cpu_usr	-	percent	ホストCPU USER使用率
cpu_wait	-	percent	ホストCPU WAIT使用率
disk	-	percent	ルートファイルシステムのディスク使用率(JAVAは設定で指定したファイルシステムパスのディスク使用率)
mem	-	percent	ホストメモリの使用率
swap	-	percent	ホストSWAPの使用率

---

app_remote_active_stat

- DB接続とHTTP Callの応答時間範囲別トランザクション数を測定
- 収集間隔: 5秒

Tags

Tag	Type	Unit	Description
alias	-	一意の値	アプリケーションの別称

Tag	Type	Unit	Description
container	-	一意の値	コンテナ名
host_ip	-	一意の値	ホストのIPアドレス
okindName	-	-	whatap.okindに分類されたグループ名
oname	-	一意の値	エージェント名
onodeName	-	-	whatap.onodeに分類されたグループ名

Fields

Field	Type	Unit	Description
act0	-	count	応答時間が3秒以内のアクティブTXの数
act3	-	count	応答時間が3秒以上、8秒未満のアクティブTXの数
act8	-	count	応答時間が8秒以上のアクティブTXの数
actx	-	count	act0、act3、act8の合計
count	-	count	モニタリング期間中に収集したアクティブTXの総数
error	-	count	エラーが発生したトランザクションの数
id	Int32	-	アクティブTXを識別するための固有ID
timeSum	number	-	アクティブTXの応答時間の合計
type	string	-	db(DB接続) またはhttpc(HTTP Call)

app_user

- アプリケーションユーザー関連メトリクスを収集
- 収集間隔: 10秒
- 統計データ: 5分

Tags

Tag	Type	Unit	Description
host_ip	-	-	ホストIPアドレス(固有値)
oname	-	-	エージェント名 (固有値)
okindName	-	-	whatap.okindに分類されたグループ名
onodeName	-	-	whatap.onodeに分類されたグループ名

Fields

Field	Type	Unit	Description
logbits	-	バイト	ユーザーのログビット情報(ユーザー活動イベントで発生したデータ単位)
realtime_user	-	count	同時接続ユーザー数

db_pool_detail

- データベース接続プールの詳細状態を収集
- 収集間隔: 5秒
- 統計データ: 5分、1時間

Tags

Tag	Type	Unit	Description
$okind	-	-	whatap.okindに分類されたグループ名
$onode	-	-	whatap.onodeに分類されたグループ名
oname	-	-	エージェント名
pool	-	-	データベースプール(Pool)のタイプ
class	-	-	データベースプーリング(Pooling) のクラス

Fields

Field	Type	Unit	Description
act	-	count	現在有効なデータベースの接続数
idle	-	count	現在の非活性化状態のデータベース接続数

---

event_5sec

- 5秒ごとに発生するイベントを収集
- 収集間隔: 5秒

Tags

Tag	Type	Unit	Description
type	-	-	イベントタイプ

Tag	Type	Unit	Description
no_cache	-	-	イベント情報のキャッシュ有無

Fields

Field	Type	Unit	Description
curtime	-	ミリ秒(ms)	イベント発生時間(UNIX epoch time, millisecond)

---

event_cube_done

- キューブ作業完了時に発生するイベント情報を収集
- 収集間隔: 5秒

Tags

Tag	Type	Unit	Description
type	-	-	イベントタイプ
no_cache	-	-	イベント情報のキャッシュ有無

Fields

Field	Type	Unit	Description
cubetime	-	ミリ秒(ms)	キューブ作業が完了した時間(UNIX epoch time, millisecond)
date	-	date	cubetimeをyyyymmdd形式に変換

Field	Type	Unit	Description
date9	-	date	date値をGMT+9 の時間帯に変換
hhmmss	-	time	cubetimeをhhmmss形式に変換
hhmmss9	-	time	hhmmss値をGMT+9 の時間帯に変換

---

visitor

- アプリケーションユーザー統計を収集
- 収集間隔: 1時間
- 統計データ: 1時間

Tags

Tag	Type	Unit	Description
host_ip	-	一意の値	ホストのIPアドレス
oname	-	一意の値	エージェント名
okindName	-	-	whatap.okindに分類されたグループ名

Fields

Field	Type	Unit	Description
logbits	-	byte	ユーザーのログビット情報(ユーザー活動イベントで発生したデータ単位)
visit_user	-	count	照会した期間中の訪問者数

app_proc_counter

アプリケーションのプロセス指標を収集したデータのカテゴリです。

- 収集間隔：5秒
- 統計データ：5分、1時間

Tags

Field	Type	Unit	Description
alias	-	-	アプリケーション別称(固有値)
container	-	-	コンテナ名(固有値)
containerKey	-	-	コンテナキー値 (固有値)
host_ip	-	-	ホストIPアドレス(固有値)
okindName	-	-	whatap.okindに分類されたグループ名
oname	-	-	エージェント名 (固有値)
onodeName	-	-	whatap.onodeに分類されたグループ名
pid	-	-	特定プロセスの固有識別子(ID)
type	-	-	プラットフォームタイプ(アプリケーションの開発言語)

Fields

Field	Type	Unit	Description
cputime	-	ミリ秒(ms)	トランザクションのcpuの時間

Field	Type	Unit	Description
db_num_active	-	count	DB Connection Pool Active 数
db_num_idle	-	count	DB Connection Pool Idle数
gc_count	-	count	GC発生回数
gc_oldgen_count	-	count	Old Generation GCカウント
gc_time	-	ミリ秒(ms)	GCの実行時間
heap_max	-	バイト	ヒープメモリの最大サイズ(-Xmx値)
heap_perm	-	バイト	Perm領域のメモリ使用量
heap_tot	-	バイト	現在割り当てられているヒープメモリの総量
heap_use	-	バイト	現在使用中のヒープメモリの量
pending_finalization	-	count	GCでfinalizeの実行を待機しているオブジェクトの数
proc_fd	-	count	プロセスが開いているファイル記述子(file descriptors)の数
proc_fd_max	-	count	プロセスがオープンできる最大のファイルディスクリプタの数
thread_count	-	count	JVMで実行中のスレッドの総数
thread_daemon	-	count	JVMで実行中のデーモンスレッドの数
thread_peak_count	-	count	JVMが実行中に観察されたスレッド数の最大値
thread_total_started	-	count	JVMが開始以降に生成されたスレッドの合計数
threadpool_activeCount	-	count	現在実行中のスレッドプールのスレッド数

java_memory

アプリケーションの実行中にJVMのメモリ使用状態に対するメトリクスを収集したデータのカテゴリです。

- 収集間隔：5秒
- 統計データ：5分、1時間

Tags

Tag	Description	Remarks
host_ip	ホストのIPアドレス	一意の値
okindName	whatap.okind に分類されたグループ名	-
oname	エージェント名	一意の値
onodeName	whatap.onode に分類されたグループ名	-
pid	特定プロセスの固有識別子(ID)	-

Fields

Field	Unit	Description	Remarks
nonheap_committed	バイト	JVMで使用するために保証された非ヒープメモリサイズ	-
nonheap_init	バイト	JVMの起動時、OSからリクエストする初期非ヒープメモリサイズ	-
nonheap_max	バイト	メモリ管理に使用できる最大非ヒープメモリサイズ	-
nonheap_used	バイト	現在使用中の非ヒープメモリの量	-
objectPendingFinalizationCount	count	GC中にfinalizeの実行を待機しているオブジェクトの数	-

性能カウンター

WhaTapエージェントは、アプリケーションのパフォーマンスに関するさまざまな情報を収集します。情報は、大きく3つに分類できます。

- **User**：リアルタイムユーザーまたは訪問ユーザー
- **Service** ：トランザクション、SQL、外部呼び出し件数および応答、エラー率など
- **Resource**：システム、プロセスリソースの使用量

User Counter

ユーザーは、モニタリング対象システムを使用するクライアントを意味します。クライアントは通常、ブラウザに基づいてユーザー数を計算します。

ウェブシステムのパフォーマンスでは、ユーザーが負荷の始まりであるため重要です。ユーザーを追跡するためには、ユーザーを分類する方法とカウント方法を検討する必要があります。

ユーザー区分

WhaTapエージェントのユーザーを区別するためのさまざまなオプションを提供します。

- **Remote IP**

  既定値は、remote ipを使用してユーザーを区別します。remote ipは、実際のユーザーを区別するには制限があります。

- **Cookie**

  クッキーを使用してユーザーを区別します。すべての接続クライアントを対象に**WHATAP**というクッキーに、UUIDは保存されています。

  whatap.conf

```
trace_user_using_ip=false

# Java agent v2.2.0 or later
```

```
wclient_using_ip=false
```

- **Header Key**

  HTTP headerに渡される値でユーザーを区別できます。

  whatap.conf

```
user_header_ticket=USER

# Java agent v2.2.0 or later
wclient_header_ticket=USER
```

ユーザーカウント

ユーザーのカウント方法によって異なります。リアルタイムユーザーは、現在のシステムを使用しているユーザーの数を知るために測定します。毎日の訪問ユーザーは、日中にサービスに関心を持っているユーザーの数をビジネス管理のために測定されます。

- **リアルタイムユーザー**

  過去5分間のユーザー数をカウントします。 5秒ごとにshiftingすると、ユーザーをカウントします。 各サーバーでカウントされた数は、HyperLogLogアルゴリズムによってマージされます。

- **毎日の訪問者**(**DAU**, Daily Active User)

  1日にシステムに接続したユーザーをカウントします。24時間接続したユーザーをHyperLogLogで計算します。

> WhaTapでは、長期間ユーザーをカウントするために、ユーザーデータに対するbyte blockをサーバーで収集します。このデータをHyperloglogでマージすると、理論的には1ヶ月以上の訪問ユーザーを計算できます。

Service Counter

トランザクションによって使用されたSQLまたは外部呼び出しの件数、応答時間、エラー件数などのパフォーマンス指標が含まれます。

- **Transaction Counter**

  トランザクションを実行すると、測定するのはカウンターです。

  - **件数**
  - **応答時間**
  - **エラー件数**

- **Active Transaction Counter**

  進行中のトランザクションの数をカウントします。

  - **件数**
  - **Active Status**

    進行状況は、METHOD, SQL, HTTPC, DBC, SOCKETの5つの状態で固定されます。

    - METHOD - 一般関数を呼び出す状態
    - SQL - DB SQLの実行中の状態
    - HTTPC - 外部HTTP API(サービス)を呼び出しの状態
    - DBC - DB接続を要求された状態、通常はPoolから取得
    - SOCKET - TCPセッションに接続を要求している状態

- **SQL**

  SQLの実行状況をカウントします。

  - **件数**
  - **応答時間**
  - **エラー件数**
  - **フェッチ件数**

- **HTTP Call**

  外部HTTPリクエストの状況をカウントします。

  - **件数**
  - **応答時間**
  - **エラー件数**

Resource Counter

サーバーリソースまたはNodeプロセス内のリソース使用量をカウントします。

- **CPU** (sys, usr, wait, steal, irq, cores)

  CPU使用率%です。各種類別に収集されます。仮想環境のみ、Stealが意味を持ちます。Cpu Coreの数も収集されます。

- **Process CPU**

  Javaプロセスが使用するCPU%です。

- **Memory**

  システムメモリの使用率(%)です。

- **Swap**

  Swapメモリの使用率(%)です。

- **Disk**

  Diskは、Java ProcessのCurrentディレクトリの使用率(%)です。

- **Heap** (Total, Used, Perm)

  Java HeapメモリのTotal、Used、Perm量です。データ単位はKBytesです。

- **GC** (Count, Time)

  5秒間に発生したGCインシデント件数と時間の合計です。

- **ObjectPendingFinalizationCount**

  GC呼び出し中にfinalize() を実行するために待機しているオブジェクトの数です。この値が大きくなると、GCtimeが遅れる可能性があります。

- **Thread** (start Count, Count, Daemon, Peak Count)

  JVMが実行されてから開始されたスレッド数、現在実行中のスレッド、デーモン、最大スレッド数を収集します。例えば、スレッドプールが必要かどうかを判断できます。

- **DB Connection Count** (active,idle)

Connection Poolのidleとactiveカウントを収集します。

- **Tomcat Thread Pool** (active, queueSize)

  トムキャットのExecutorは、スレッドプールの設定時にパフォーマンス情報を収集します。

  XML

```
<Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
maxThreads="50" minSpareThreads="2" maxQueueSize="78"/>
```

DB Connection Poolカウンター

DB Connection Poolは、DataSourceと呼ばれます。WASによって提供される場合が一般的ですが、オープンソースPoolを使用することもあります。使用量の情報がJMXに公開されることがありますが、多くの場合は公開されません。

WhaTapは2つの方式を提供しています。１つのBCIによる直接問い合わせ方法とJMXを利用した方法です。デフォルトは、BCI方法を使用し、サブとしてJMXを使用します。

次の個別のオプションが既定値に関連してByteCodeがInjectionされるクラスです。

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
```

> BCI関連のオプションで変更する場合、**再起動**する必要があります。クラスがinstrumentされた後も、追跡テーブルに登録する必要があります。したがって、実行中のredefineは機能しません。

DataSourceの状況をJMXに読み込む

```
dbcp_pool_enabled==trueの場合、tomcat_ds_enabled=false
weblogic_pool_enabled== trueの場合、weblogic_ds_enabled=false
```

1. **オプション自動決定**の内容を確認してください。
2. 関連付けられたpoolのオプション値を `false` として宣言します。
3. DataSource関連オプション値を `true` に変更してください。既定値は `false` です。

   whatap.conf

   ```
   tomcat_ds_enabled=false
   weblogic_ds_enabled=false
   ```

ThreadPoolカウンター

ThreadPoolカウンターは、JMX Mbean情報から収集されます。Tomcatなどの使用事例が多数ある場合は問題ありませんが、特定の環境では、この指標が収集されない可能性があります。

収集には追加設定が必要です。関連する設定は次のとおりです。

whatap.conf

```
jmx_threadpool_objectname=Catalina:type=ThreadPool,name="http-bio-8080"
// JMX ThreadPool MBean名前を設定します。
jmx_threadpool_atter_activecount=connectionCount
// activecount数を表すattributeを設定します。
jmx_threadpool_atter_queuesize=maxThreads
// maxThreads数を表すattributeを設定します。
```

JMX ThreadPool ObjectNameやAttributeNameなどのMbean名は、JMXモニタリングツールを使用して確認または、各WASのMbean文書を参照してください。

> ⓘ 使用中のWAS、フレームワーク関連文書を参照してMbean情報を確認してください。

Jconsole使用時

- **ObjectName** : Catalina

  type : ThreadPool

  name : "http-bio-8080"

- **ActiveName** : connectionCount

  QueueName : maxThreads

結果確認

正常に設定された場合、カウンターで次のようにデータを検索することができます。

パフォーマンスカウンター - 正常状態

Apdex

Apdex(Appliccation Performance Index)は、オープン標準に従うアプリケーションのパフォーマンス指標です。

Apdexは応答時間に基づいており、全体の要求の中で満足と許容の割合で数値化します。 ダッシュボードにApdexグラフが追加されました。

Apdexはユーザー満足度の指標として使用でき、0~1の値です。

(満足回数+ (許容回数*0.5))／全体リクエスト数

状態	説明
満足 (Satisfied, **S**)	業務処理に全く問題なし≤ 1.2秒 (満足**S**既定値)
許容(Tolerating, **T**)	使用者が遅延を感じるが業務処理は可能≤ 4.8秒(満足**S** * 4)
不満(Frustrated, **F**)	業務処理が不可能> 4.8秒(許容**T**超過及びエラー)

- **whatap.apdex_time** millisecond

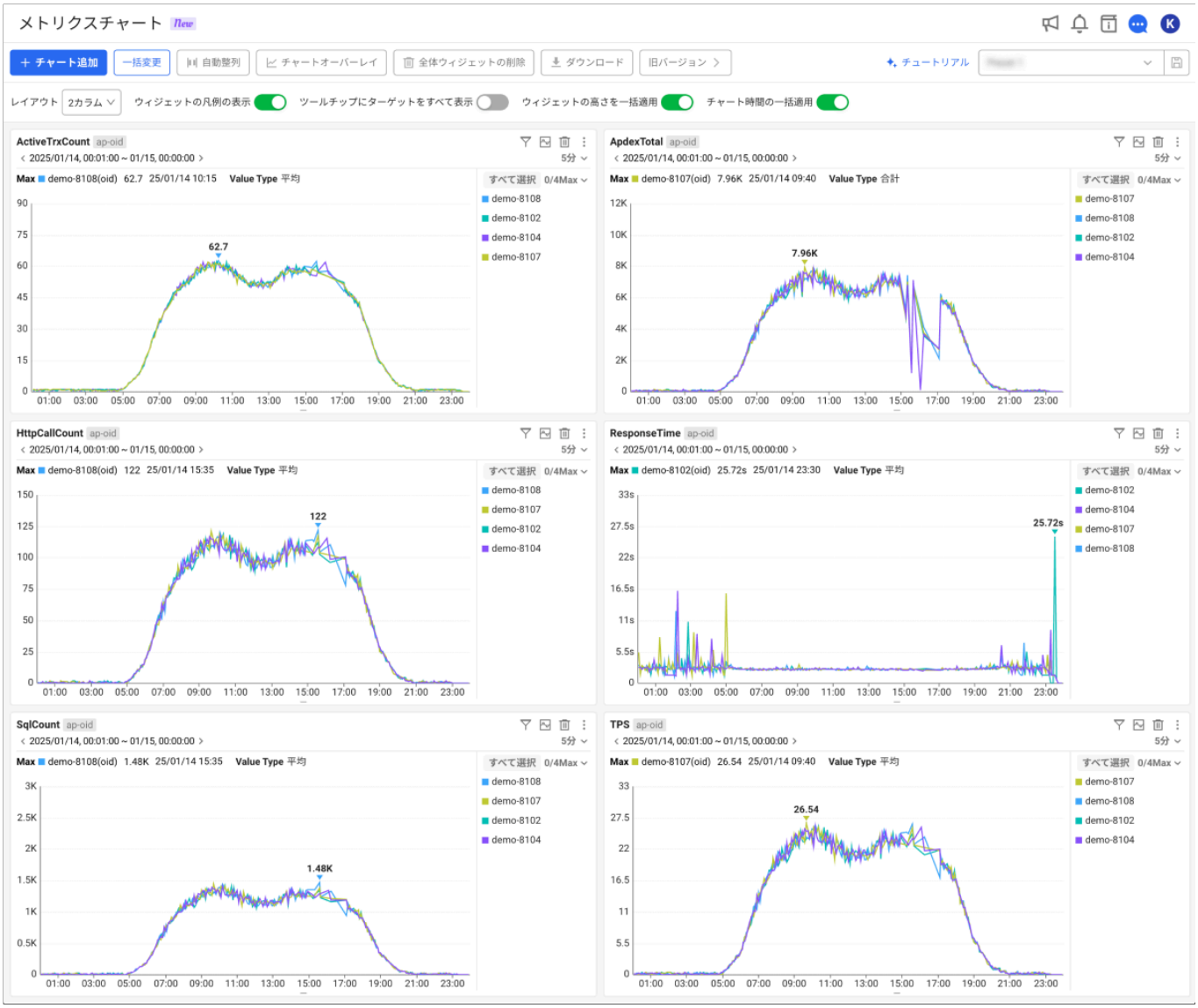
  既定値 `1200`

  満足 **S** 既定値は、エージェント設定メニューから変更できます。

メトリクスチャート v2

ホーム画面 > プロジェクト選択 > **分析** > **メトリクスチャート** *New*

メトリクスチャート *New* メニューは、WhaTapモニタリングソリューションの主要機能であり、様々なモニタリング対象から収集したデータを時系列チャートで視覚化して提供します。 これにより、ユーザーはシステムの性能を効果的にモニタリングし、異常をすばやく識別して、問題解決に必要なインサイトを確認できます。



- 様々なメトリクスを時系列チャートを通じて視覚的に確認でき、障害発生時点の様々なメトリクスを一度に分析できます。
- 特定の時間範囲を選択して過去のデータを照会し、その期間の性能を分析できます。
- 特定の時間帯や条件に合ったデータをフィルタリングすることで、より正確な分析ができます。
- ユーザーは、目的のチャートを画面に配置し、レイアウトを自由に調整できます。
- ユーザーが希望するメトリクスを配置し、プリセットとして保存し、必要に応じて読み込みできます。

> ⓘ **既存のメトリクスチャートメニューとの違い**
>
> Service 2.8.0 アップデート以降の変更点は以下の通りです。 アップデート後の変更事項を画面で確認するには、右上の**チュートリアル**ボタンを選択します。
>
> - ユーザーが時間、対象、インターバルオプションを設定した後、モニタリングする対象のメトリクスを特定し、チャートを追加できるようにユーザー経験(UX)を改善しました。
> - ダッシュボードに配置されたウィジェットを削除することなく、チャートの照会時間の範囲を変更できます。
> - ダッシュボードに配置したすべてのウィジェットのオプションを一括で適用できる**一括変更**機能を提供します。
> - プリセット機能により、ダッシュボードに配置されたウィジェットの情報を保存し、読み込みできます。
> - 提供する様々なオプションにより、ダッシュボードのレイアウトやウィジェットの機能を調整できます。

チャートウィジェットを追加する

ダッシュボードにメトリクスをウィジェットとして追加するには、画面左上の**チャート追加**ボタンを選択します。

照会時間の設定

時間でメトリクスを照会する日付と時間を設定します。

対象を選択する

プロジェクトに含まれるエージェントの中から、対象を選択します。 個々のエージェントを選択するか、エージェン

ト設定で分類されたグループ単位で選択できます。

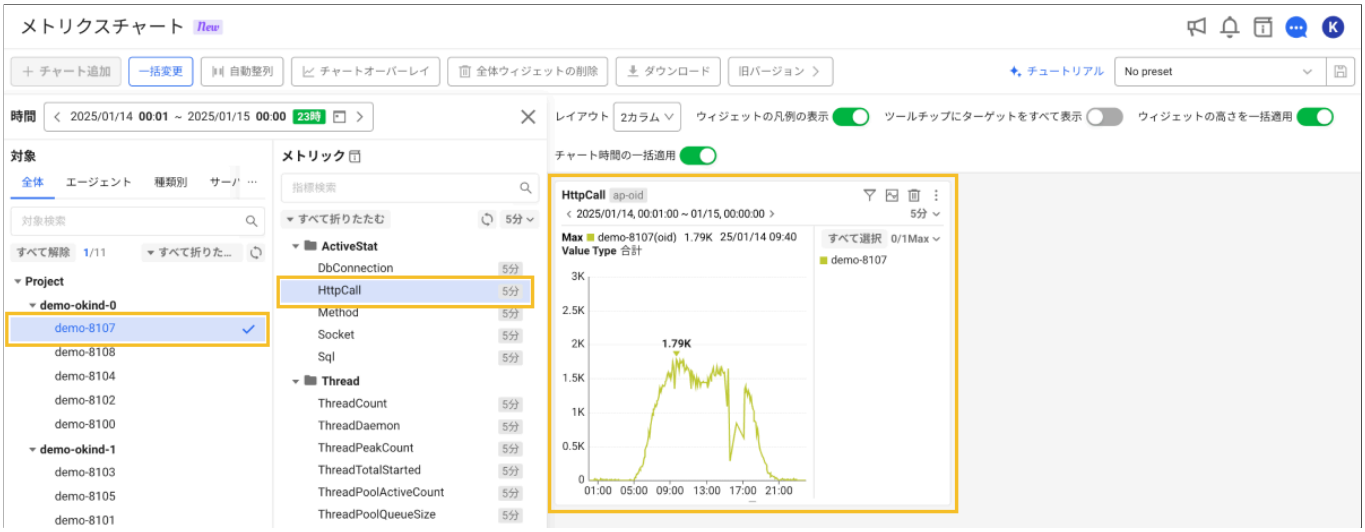


- **エージェント**: 個別エージェントまたはすべてを選択できます。
- **種類別**: エージェント設定で `whatap.okind` に分類されたグループ単位でエージェントをモニタリングします。
- **サーバー別**: エージェント設定で `whatap.onode` に分類されたグループ単位でエージェントをモニタリングします。

- 設定した時間と日付によって、選択できる対象が変更される場合があります。
- エージェントの一覧からマウスをクリックして、対象を選択または解除できます。 `Shift` キーを使用して複数のエージェントを選択できます。
- **すべて選択**または**すべて解除**ボタンを選択すると、関連する下位対象をすべて選択または解除できます。
- 希望のエージェントが一覧に表示されない場合は、**リフレッシュ**ボタンを選択します。
- **すべて折りたたむ**または**すべて開く**ボタンを選択してグループ単位で一覧を隠したり、展開したりできます。

指標を選択する

モニタリングする対象エージェントを選択した後、**メトリック**パネルに選択できる一覧が表示されます。 一覧からご希望のメトリクスを選択します。



選択したメトリクスに対応するチャートウィジェットが右側のダッシュボードに配置されます。 他のメトリクスを選択し、チャートウィジェットを追加できます。 チャート領域を拡大して見るには、**時間**オプションの右にある×ボタンを選択します。 **対象**と**メトリック**パネルが消え、チャート領域のみが画面に表示されます。

- **指標検索**入力欄に文字列を入力すると、希望のメトリクスをすばやく検索できます。
- メトリクスのカテゴリとフィールド名に基づいても検索できます。 カテゴリおよびフィールドの詳細については、次の文書を参照してください。
- **すべて折りたたむ**または**すべて開く**ボタンを選択して、カテゴリ単位でメトリクス一覧を非表示または表示できます。
- 希望するメトリクス項目が一覧に表示されない場合は、**リフレッシュ**ボタンを選択します。

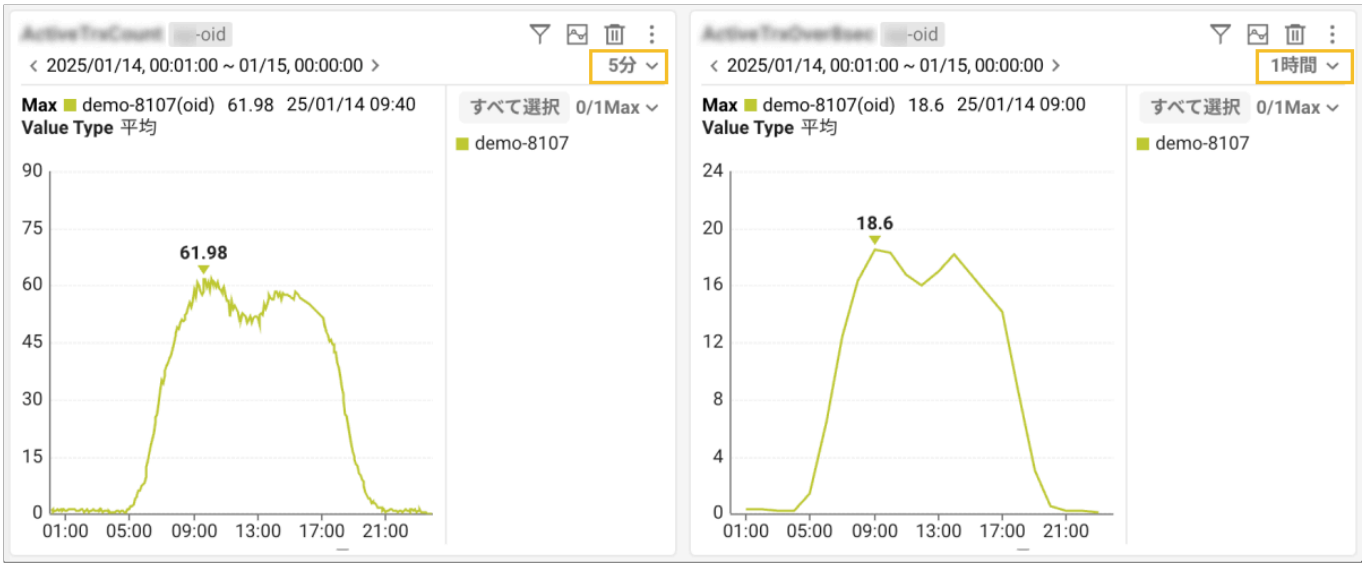
インターバルを設定する



チャートごとにインターバルを設定でき、データの集計単位を調整して、様々な時間間隔のデータを可視化できます。 **メトリック**パネルの右上にあるボタンを選択すると、インターバルオプションを選択できます。

メトリクスごとにデフォルトで設定されたインターバル時間が異なり、照会時間の範囲によってインターバル時間が変更されます。

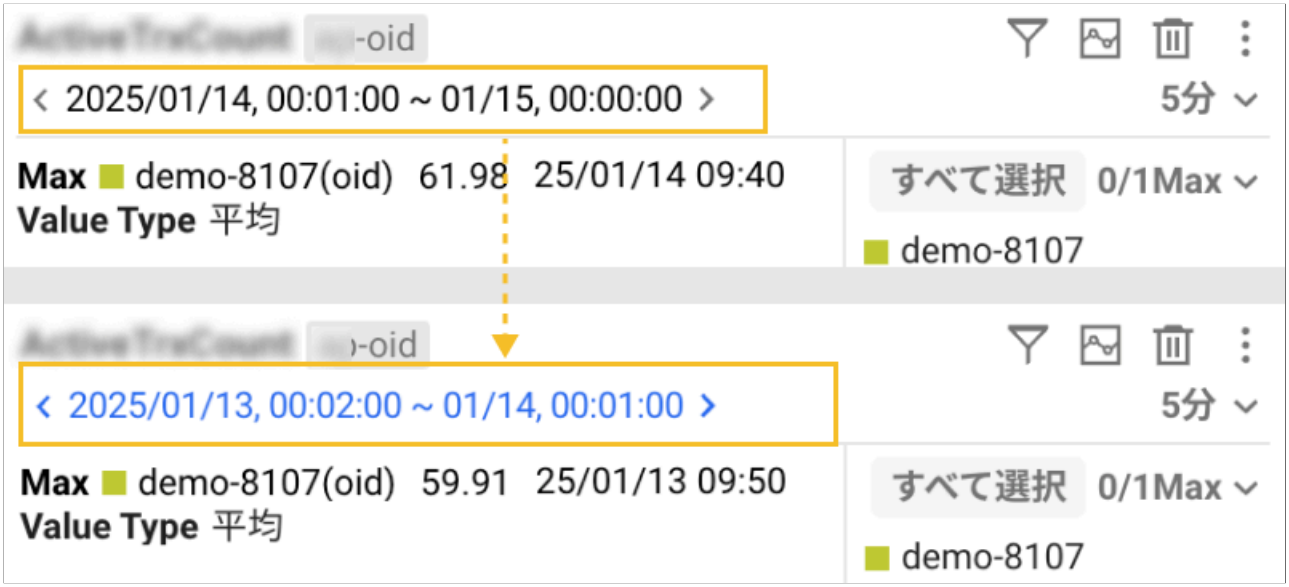
例えば、**5分間**インターバルは5分単位の平均値を表示し、**1時間**インターバルは、1時間単位の平均値を表示します。



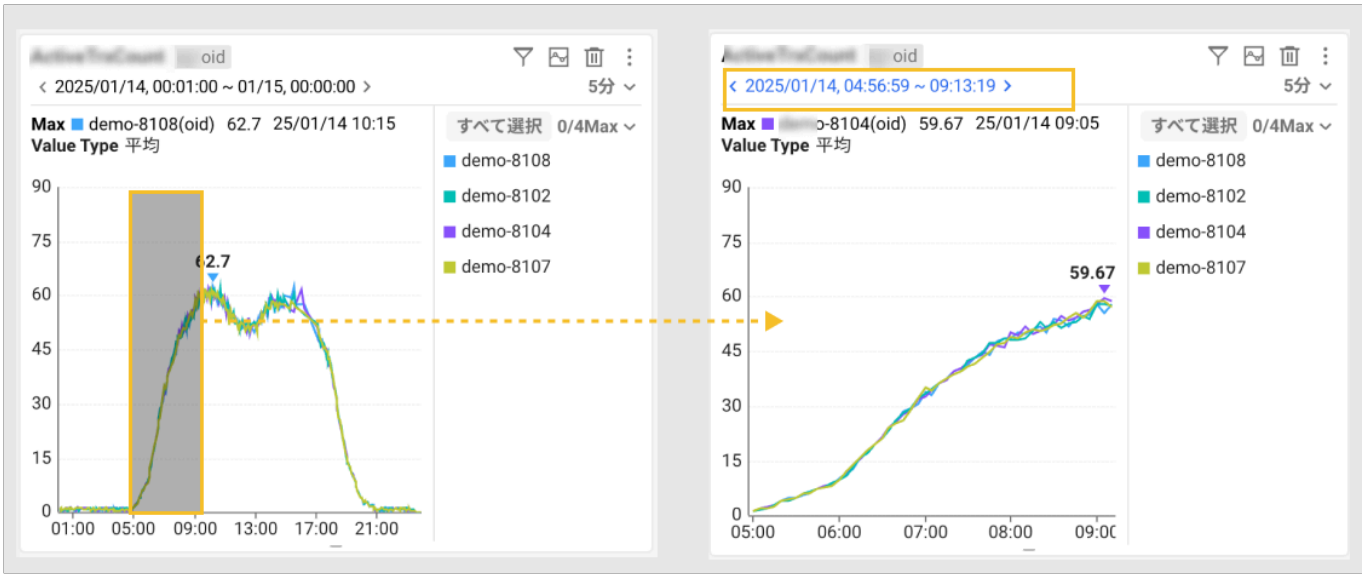
インターバルオプションを調整し、短い間隔の詳細データから長い間隔の全体的な推移まで、多様な観点からデータを分析できます。

照会時間を変更する

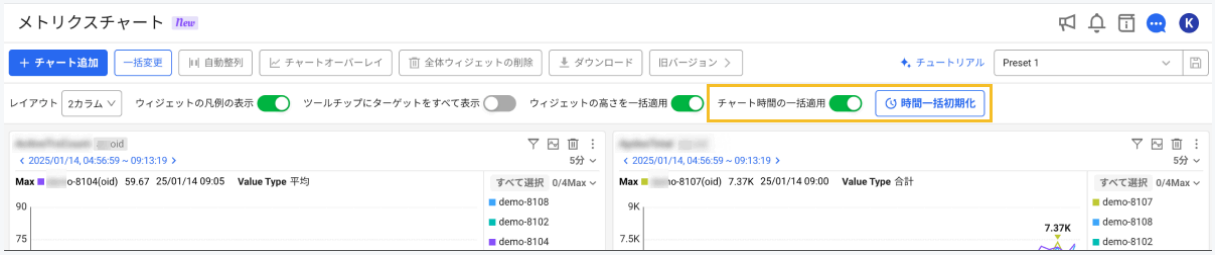
ウィジェットのチャート照会時間の範囲を変更するには、ウィジェット上部の < または > ボタンを選択します。 時間変更範囲は、**時間**オプションに設定された時間基準と同じです。 例えば、**時間**オプションで**1日間**に設定されている時、 < または > ボタンを選択すると、時間を1日単位で変更できます。



またはチャートの特定領域をドラッグすると、ドラッグした範囲に時間を変更して照会できます。



- 画面の上部にある**チャート時間の一括適用**オプションを有効にして、ウィジェットの時間を変更すると、すべてのウィジェットの時間を変更できます。



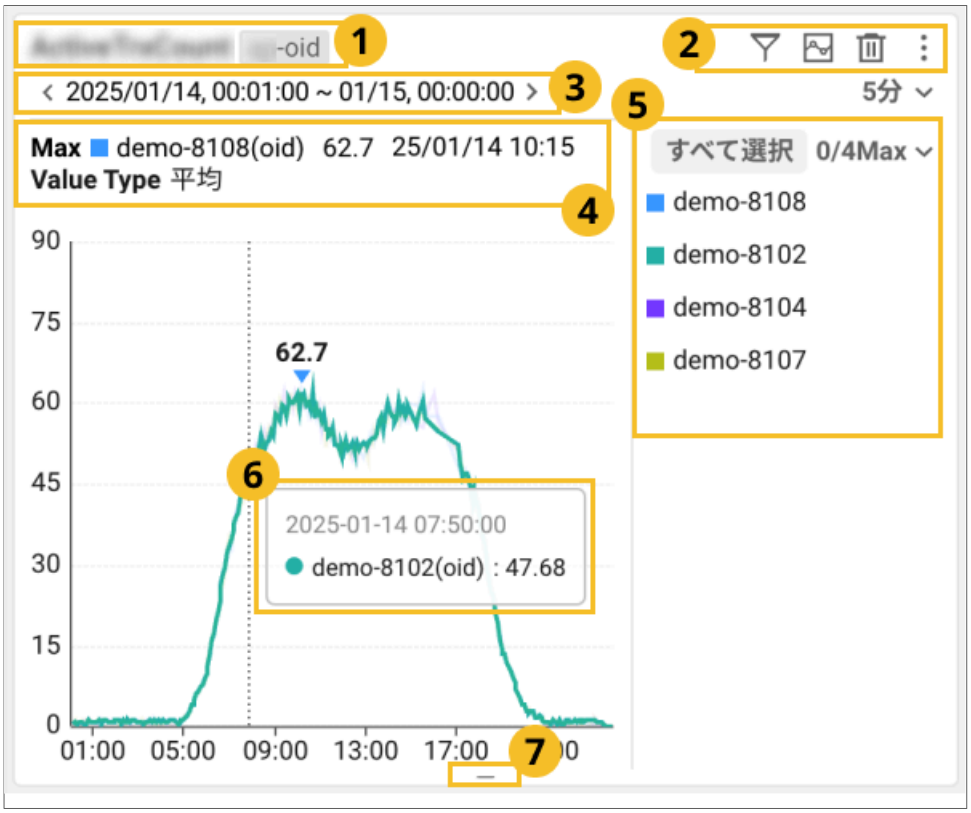
- 照会時間の範囲を変更した後、直近の時間に再度変更するには、ボタンを選択します。

  もし画面の上部にある**チャート時間の一括適用**オプションを有効にした場合、**時間バッチ初期化**ボタンが表示されます。 このボタンを選択すると、すべてのウィジェットの時間を初期化できます。

一括変更機能を利用してウィジェット全体の時間を変更し、モニタリング対象とインターバルも修正できます。 詳細については、次の文書を参照してください。

チャートウィジェットの詳細

チャートウィジェットに表示される各アイテムの詳細を確認します。



- 1 **メトリクス名とカテゴリ**: チャートウィジェットのメトリクス名を表示します。 メトリクスの右側には、照会対象のカテゴリをタグ形式で表示します。 ご利用の商品によりカテゴリ名称は異なる場合がございます。
  - *oid : 個々のエージェントを選択した時に表示されます。
  - project : プロジェクト単位で対象を選択した時に表示されます。
  - okind : エージェント設定で whatap.okind で分類されたグループを選択した時に表示されます。
  - onode : エージェント設定で、 whatap.onode で分類されたグループを選択した時に表示されます。
- 2 **ウィジェットオプション**: チャートウィジェットのオプションメニューです。 詳細については、次の文書を参照してください。 インターバルオプションの詳細については、次の文書を参照してください。
- 3 **時間**: チャートのデータを照会する時間範囲を表示します。 < または > ボタンを選択して照会時間範囲を調整できます。
- 4 **データ**: モニタリング対象のうち照会したデータの最大値(**Max**)とデータ結合方式(**Value Type**)を確認できます。
- 5 **凡例**: モニタリング対象を凡例として表示します。 凡例表示の詳細については次の文書を参照してください。
- 6 **チャート**: 照会したデータを時系列グラフで表示します。 マウスをグラフの上にかざすと、詳細な情報を確認できます。 チャートの特定領域をドラッグすると、ドラッグした範囲に時間を変更して照会できます。

- 7 **サイズ調整**: チャートウィジェットの真ん中の下部領域をドラッグすると、ウィジェットのサイズを調整できます。

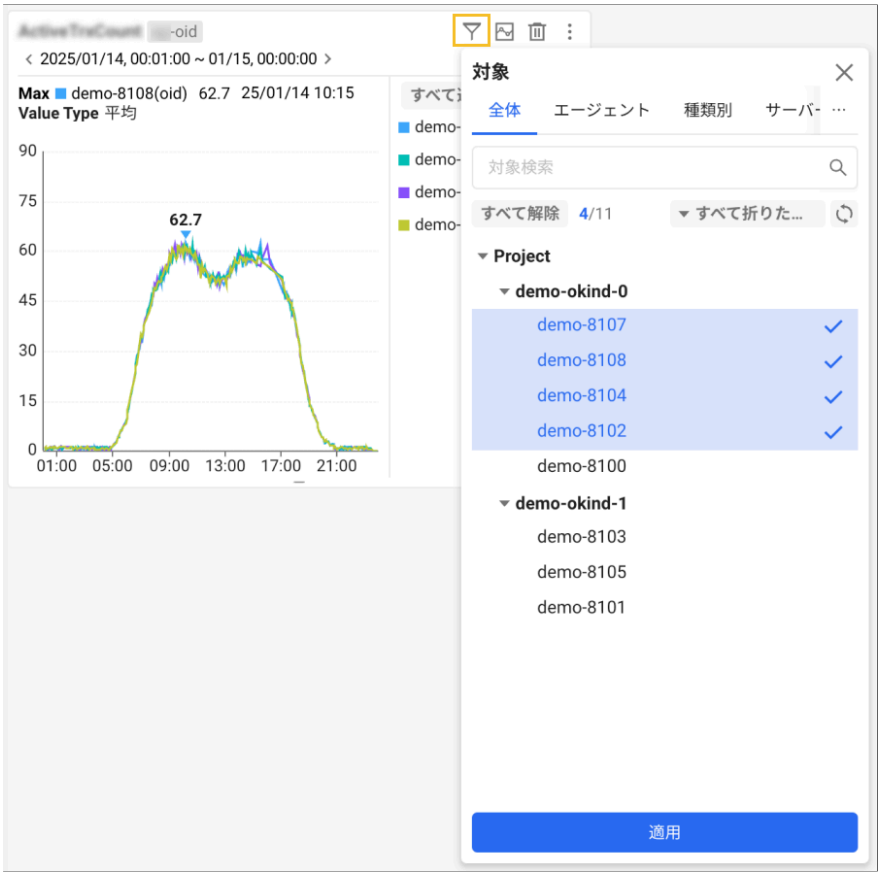
チャートウィジェットオプションを利用する

チャートウィジェットの右上にあるオプションボタンを使用すると、様々な機能が利用できます。

対象をフィルタリングする

画面に追加したチャートウィジェットに、モニタリング対象を追加または除外できます。

1. ボタンを選択すると、モニタリング対象を選択できる画面メニューが表示されます。



2. 画面の案内に従って希望する対象を選択します。

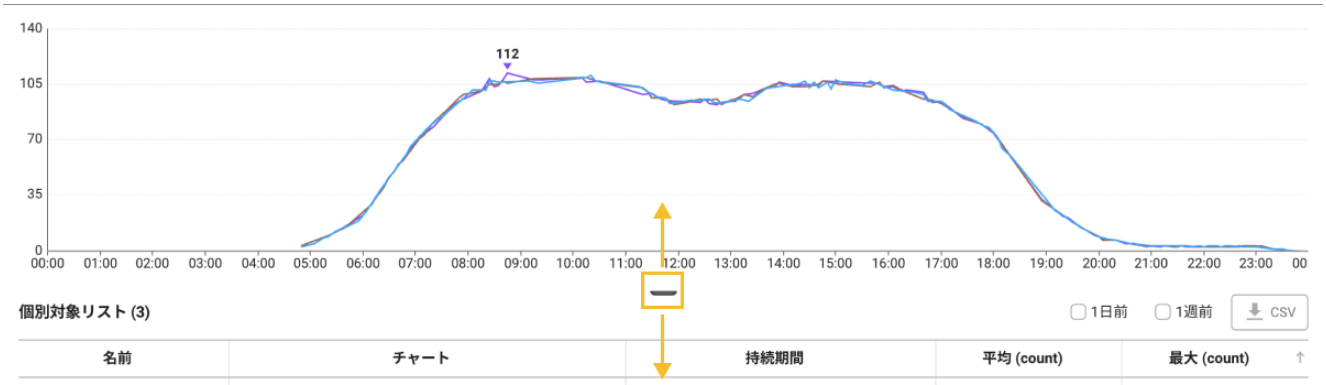
3. **適用**ボタンを選択します。

詳細を照会する

チャートのデータをより広い画面で詳細情報を確認するにはボタンを選択します。 **詳細検索**画面が表示されます。個々の対象に対するチャートと平均および最大値を照会できます。



- **1日前**: 1日前のデータと比較して確認できます。
- **1週前**: 1週間前のデータと比較して確認できます。
- **CSV**: 照会したデータをCSV形式のファイルとしてダウンロードできます。
- グラフチャートの下部にあるアイコンをドラッグすると、チャートのサイズを調整できます。

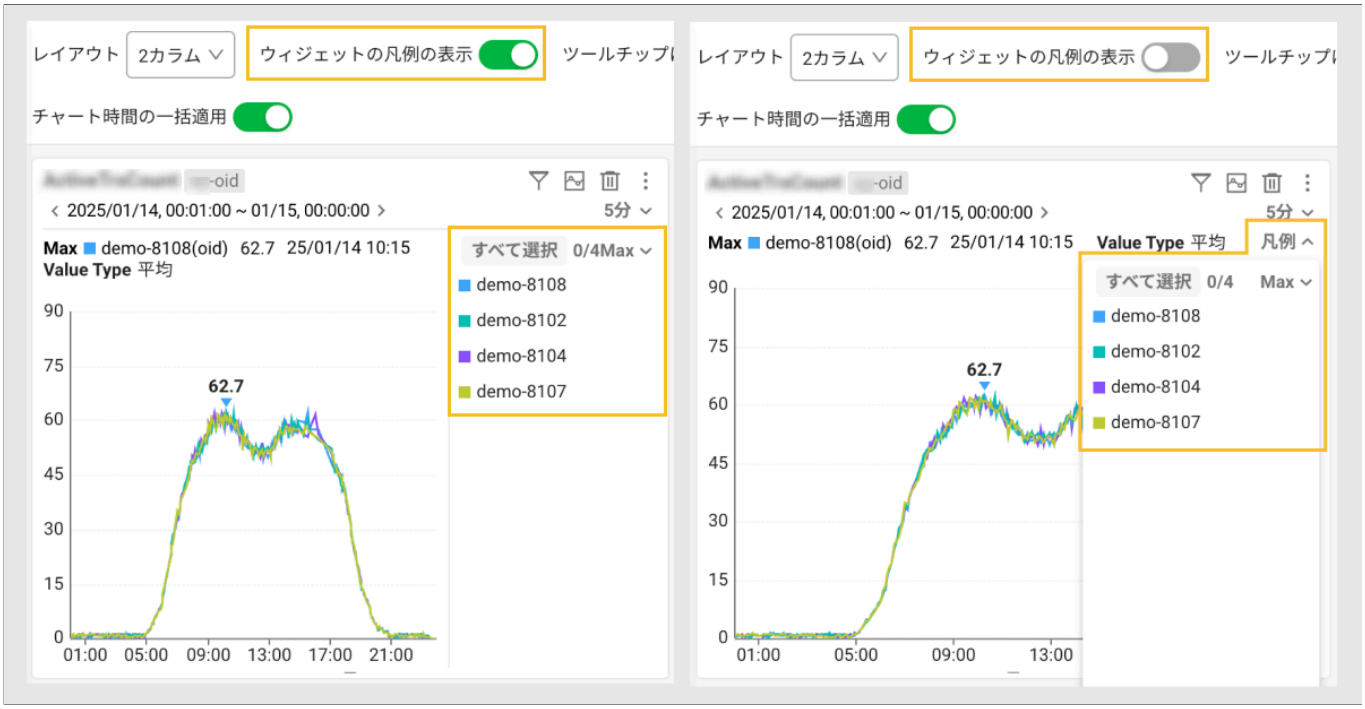


チャートウィジェットを削除する

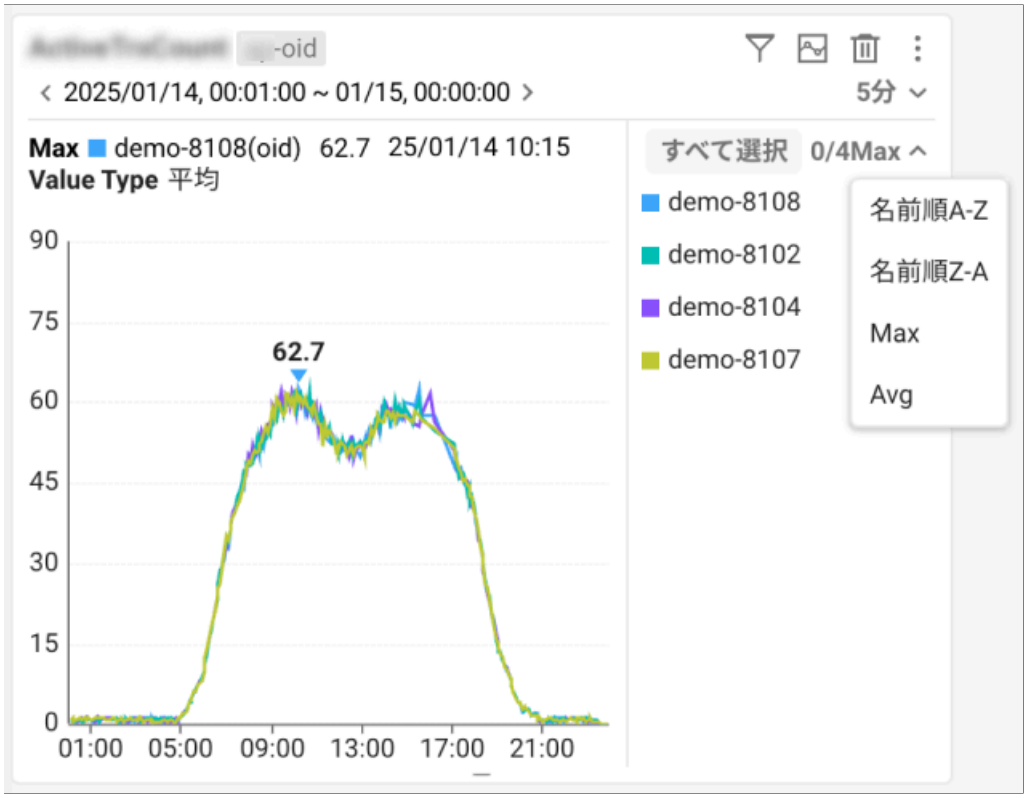
画面に配置したチャートウィジェットを削除するには、ボタンを選択します。

凡例を活用する

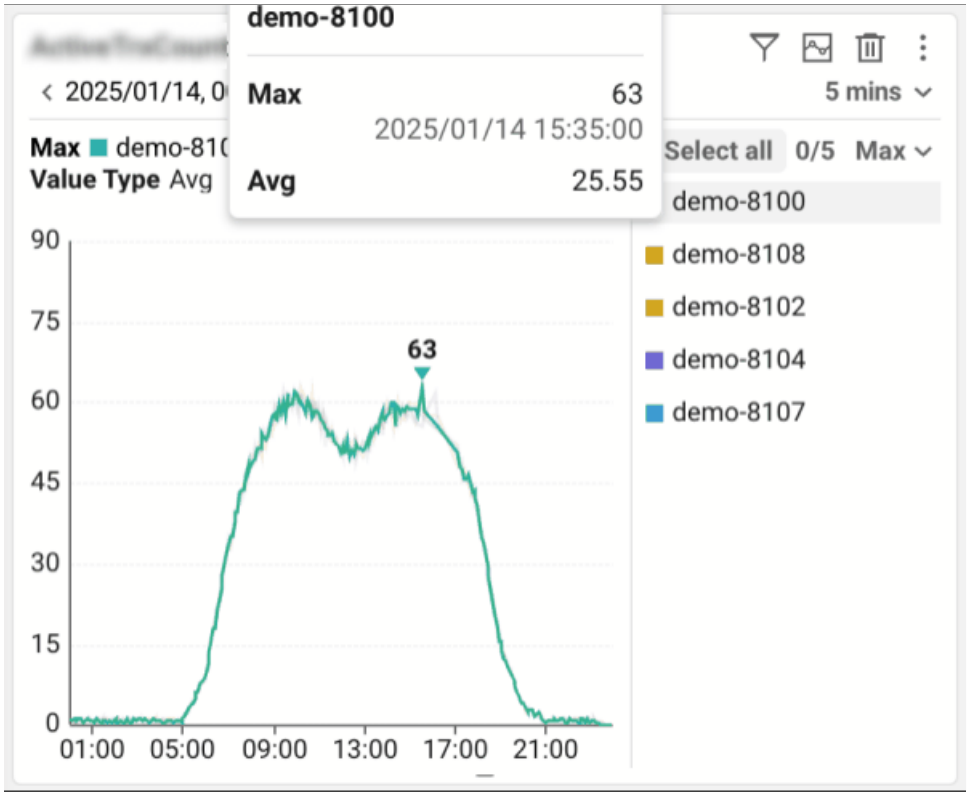
凡例を通じてチャートに表示されるモニタリング対象を確認し、希望する基準で整列したりフィルタリングできます。画面上部の**ウィジェットの凡例の表示**オプションをオンまたはオフにすると、ウィジェットのチャートの右側に凡例を表示・非表示できます。 オプションをオフにした状態では**凡例**をクリックしてモニタリング対象を確認して選択できます。



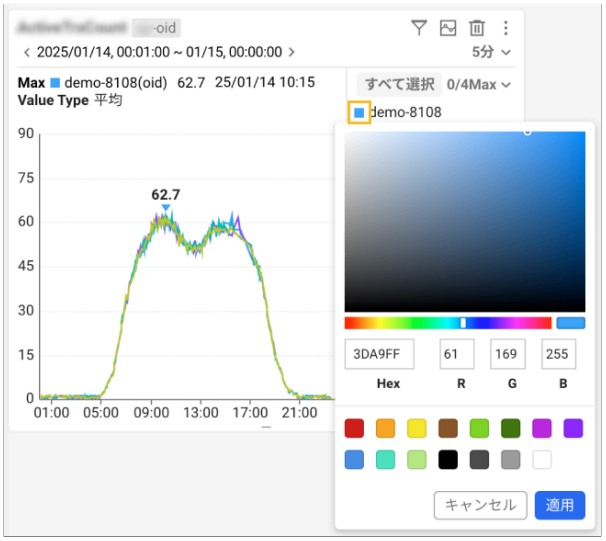
- **すべて選択/すべて解除**ボタンを選択してモニタリング対象を選択または解除できます。
- 凡例一覧では、モニタリング対象を名前順、最大値、平均値に基づいてソートできます。



- 凡例一覧の各項目にマウスをオーバーすると、そのモニタリング対象に対するグラフを強調して確認でき、Max値が記録された特定時間をツールチップで表示します。 または、凡例一覧の対象を選択してもフィルタリングできます。



凡例色を変更する



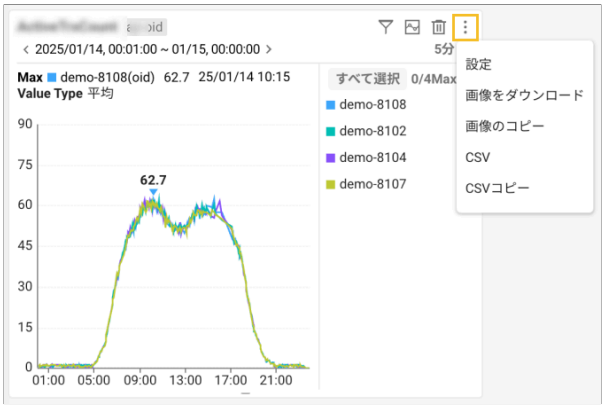
モニタリング対象に該当するグラフの色を変更できます。 凡例一覧から色を変更したい対象の左側にある四角い箱

を選択します。 希望する色を選択した後**適用**ボタンをクリックします。

> ⓘ 設定した色は、プリセット機能によって保存されません。 この機能は、今後のアップデートで提供される予定です。

追加オプションを利用する

チャートウィジェット右上にある ⋮ ボタンを選択すると、次の追加オプションが利用できます。



- **設定**
  - **自動Y軸を使用**: 照会したデータ範囲内でY軸の大きさを自動調整できます。
  - **最大値表示**: チャートのグラフ領域に最大値を表示できます。
- **画像をダウンロード** : チャートを画像形式のファイルとしてダウンロードできます。
- **画像のコピー**: チャートを画像形式でクリップボードにコピーできます。
- **CSV**: チャートで照会したデータをCSV形式のファイルにダウンロードできます。
- **CSVコピー**: チャートで照会したデータをCSV形式でクリップボードにコピーできます。

プリセットを設定する

ダッシュボードからユーザーが設定したウィジェットの設定とレイアウト状態を保存して読み込むことができます。ウィジェットのサイズを調整し、希望する場所に配置して新しいプリセットを作成できます。

プロジェクト修正権限のあるメンバーのみがプリセットを保存または変更できます。 メンバー権限の詳細については、次の文書を参照してください。

新しいプリセットを作成する

1. ダッシュボードから、希望する形にウィジェットを配置してみてください。 サイズを変更して、頻繁に確認するウィジェットのみを配置することもできます。
2. 画面の右上に保存ボタンを選択します。
3. 新しいプリセット名を入力します。



4. **保存**ボタンを選択します。

プリセット一覧から新しく保存したプリセットを確認できます。

- 新しく作成したプリセットに変更が加えられた場合は、再度プリセットを保存してください。 保存ボタンを選択し、同じ名前でプリセットを保存します。 既存のプリセットに変更を上書きします。
- ダッシュボードの変更を保存せずに別のメニューに移動すると、変更は保存されません。
- プリセットはプロジェクト単位で保存され、他のユーザーと共有できます。
- プリセットに保存される対象は、ダッシュボードのレイアウトとダッシュボードに配置したウィジェット、ウィジェットで修正したオプション(**対象**, **間隔**)です。 画面上部の**時間**オプションは保存されません。

- プリセットを読み込んでからウィジェットの照会時間を変更するには、**一括変更**機能を利用します。 詳細については、次の文書を参照してください。

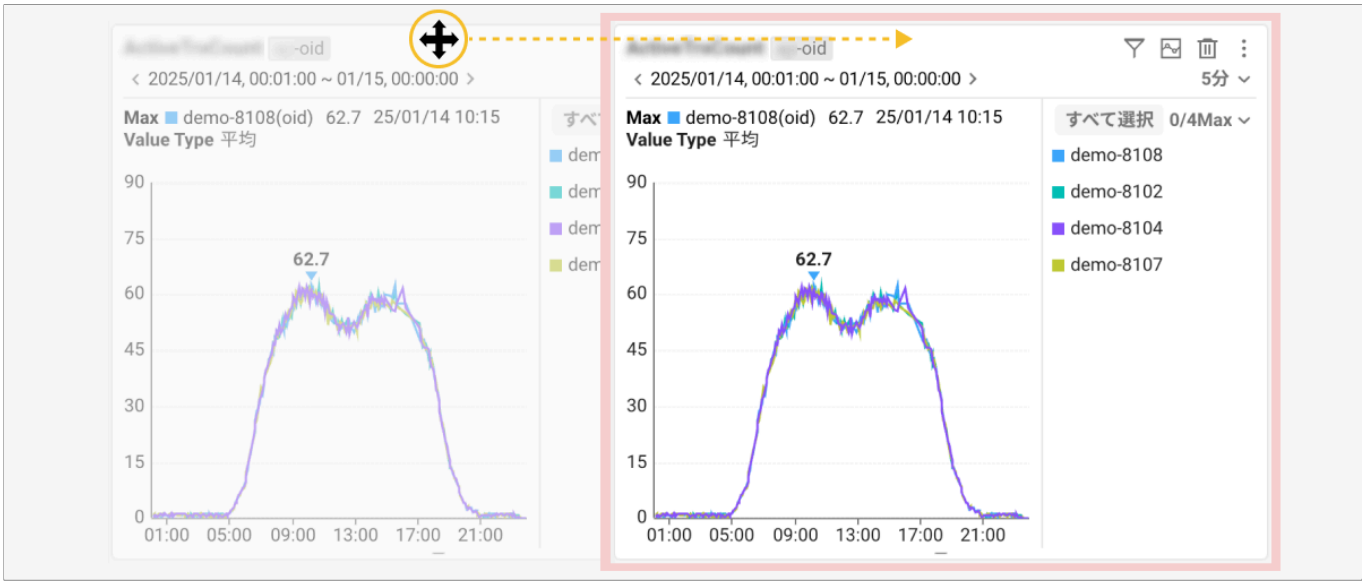
プリセットを削除する

使用していないプリセットは、プリセット一覧から削除できます。 プリセット一覧から削除する項目の右側にある ✕ ボタンを選択します。

ダッシュボードウィジェットの編集

ダッシュボードに配置したウィジェットをユーザーが希望する場所に配置したり、サイズを調整できます。

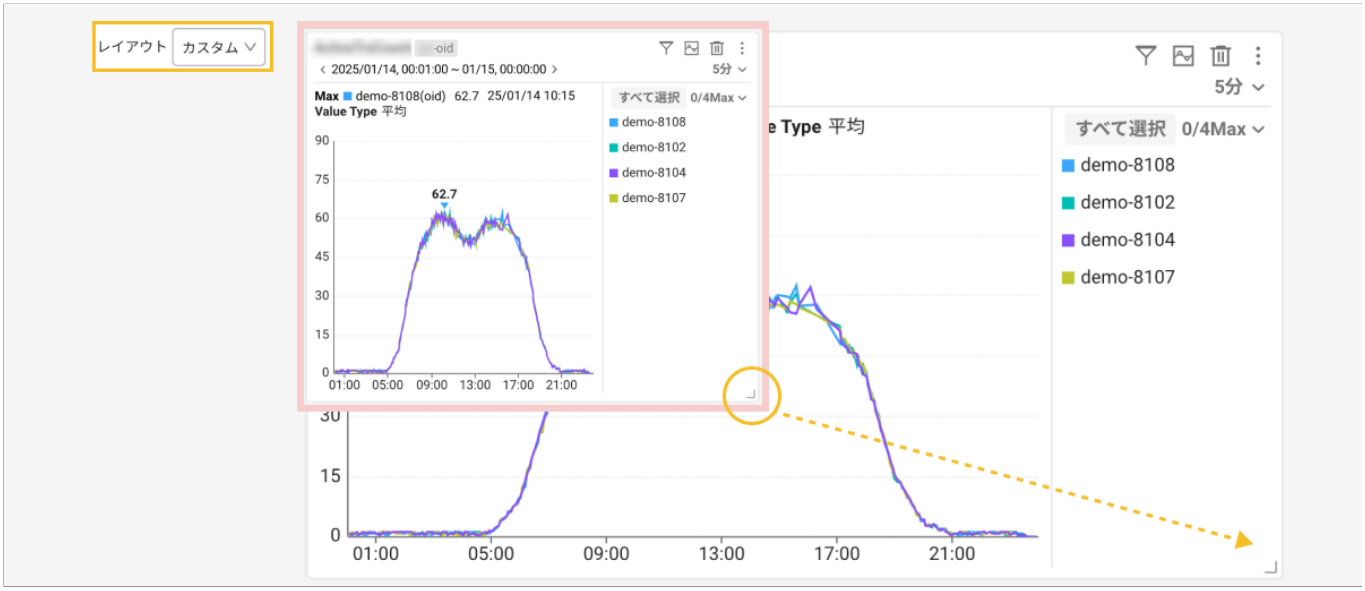
ウィジェットの移動



ウィジェットの上部にマウスをかざすと、✛に変更されます。 この時、クリックしたまま希望の位置にドラッグするとウィジェットを移動できます。

ウィジェットのサイズ変更

ダッシュボードオプションの**レイアウト**が**カスタム**に設定されていると、ウィジェットのサイズを調整できます。



ウィジェットの右下にある⌟をマウスでクリックした状態で希望するサイズにドラッグします。

> **レイアウト**が**カスタム**に設定されていない場合は、ウィジェットの高さだけを調整できます。 ウィジェットの真ん中の下部領域をマウスでクリックしたままでサイズを調整します。

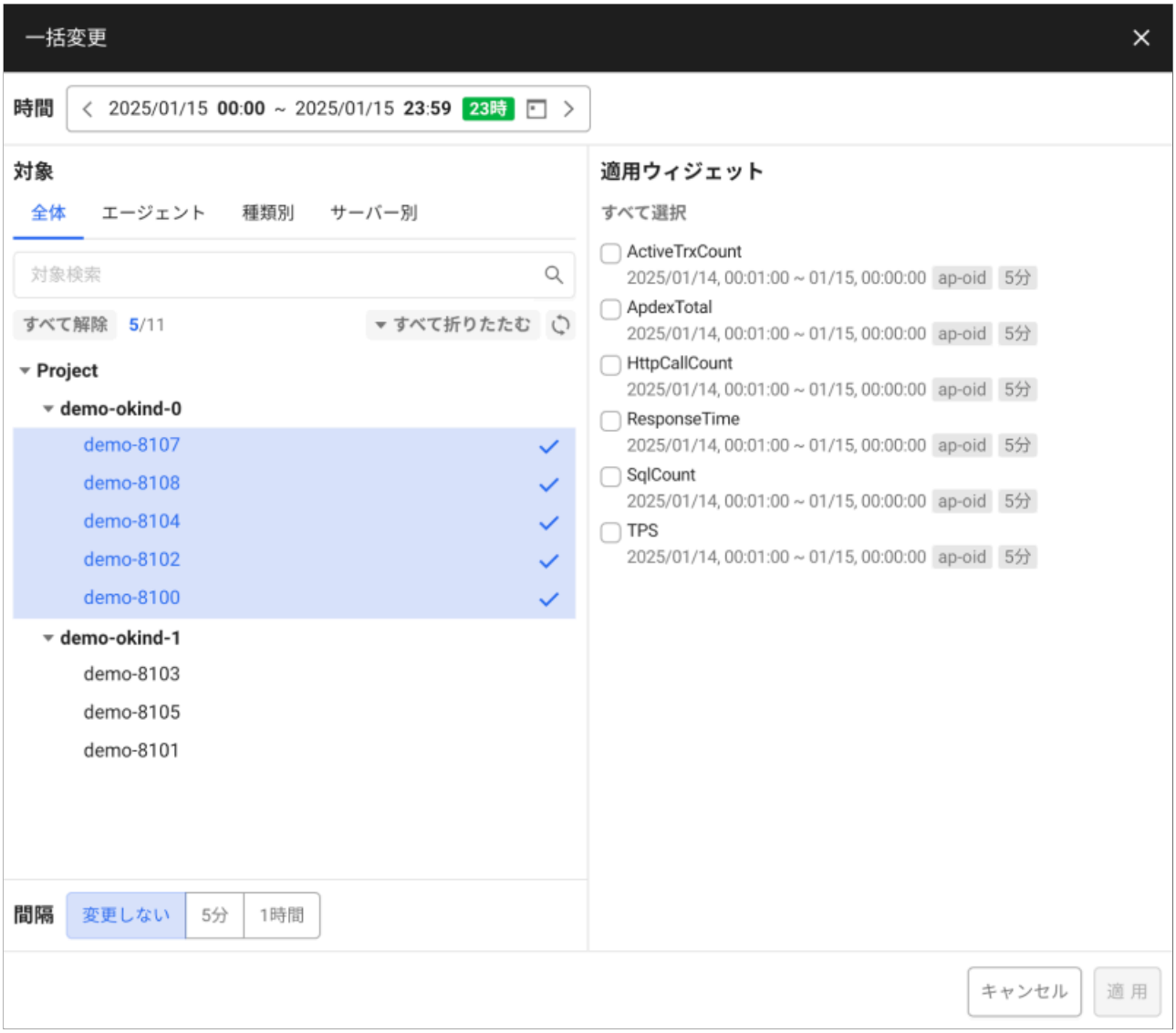
ダッシュボードオプションを利用する

画面上部にあるオプションメニューを利用して、チャートウィジェットに適用された様々な設定を一度に適用できます。 また、ダッシュボードのレイアウトを希望する形に設定できます。



チャートウィジェットオプションを一括変更する

画面に配置されたチャートウィジェットに、詳細オプションを一度に適用できる機能を提供します。 画面左上に**一括変更**ボタンを選択します。 **一括変更**画面が表示されたら、オプションを設定し、**適用**ボタンを選択します。



- **時間**: チャートのデータ照会範囲を設定できます。
- **対象**: モニタリング対象を選択します。 個々のエージェントを選択したり、エージェント設定を通じて分類したグループ単位で選択できます。

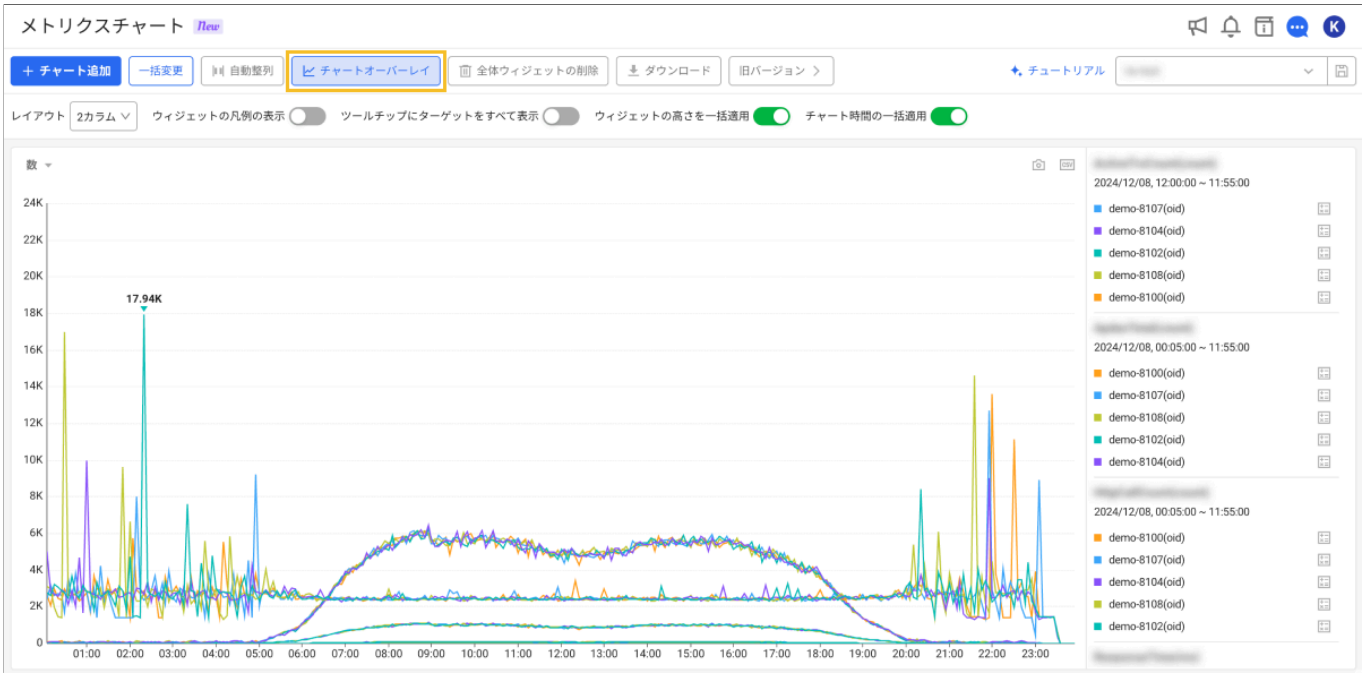
  - **エージェント**: 個別エージェントまたはすべてを選択できます。
  - **種類別**: エージェント設定で `whatap.okind` に分類されたグループ単位でエージェントをモニタリングします。
  - **サーバー別**: エージェント設定で `whatap.onode` に分類されたグループ単位でエージェントをモニタリングします。
- **間隔**: 様々な時間間隔でデータの集計単位を調整できます。
- **適用ウィジェット**: 変更するオプションを適用するチャートウィジェットを選択します。 選択したウィジェットに変更したオプションを一度に適用できます。

ダッシュボードを自動ソートする

ダッシュボードに配置したウィジェットの位置を自動的にソートするには**自動整列**ボタンを選択します。 ウィジェットの幅、高さが小さい順にウィジェットの配置を自動的にソートします。 幅が同じであれば、高さが小さい順にソートします。

複数のチャートを比較する

ダッシュボードに配置した複数のチャートデータを1つのチャートで確認できる機能です。 この機能により、同時間帯の様々なメトリクス情報を他のメトリクスと比較しながら確認できます。 **チャートオーバーレイ**ボタンを選択します。 画面にオーバーレイチャートを配置できます。



- オーバーレイチャート左上の**数**を選択すると、x軸チャートのデータ基準を変更できる一覧が表示されます。 希望する項目を選択して、チャートのデータを比較できます。
- : 照会中のデータに基づいて、チャートを画像形式のファイルとしてダウンロードできます。
- CSV: 照会中のデータに基づいて、チャートをCSV形式のファイルとしてダウンロードできます。
- 右側のセクションでは、メトリクスごとのエージェントの一覧を確認できます。 個々のエージェントの色は、チャートのグラフの色と一致します。

配置されたウィジェットを一括削除する

ダッシュボードに配置された複数のウィジェットを一度に削除できます。 **全体ウィジェットの削除**ボタンを選択します。 確認メッセージが表示されたら**削除**ボタンを選択します。

CSVファイルでダウンロードする

ダッシュボードに配置したウィジェットのデータをCSV形式のファイルとしてダウンロードできます。 **ダウンロード** > **CSV**を選択します。 時間帯のメトリクスのデータを確認できます。

イメージファイルでダウンロードする

ダッシュボードに配置されたすべてのウィジェットを1つのイメージファイルでダウンロードできます。 **ダウンロード** > **画像をダウンロード** を選択します。 ダッシュボード画面をそのままイメージで保存できます。

以前のバージョンを利用する

以前のバージョンの**メトリクスチャート**メニューを利用するには、**旧バージョン**ボタンを選択します。 または **サイトマップ** > **分析** > **メトリクスチャート** *Old* メニューパスで確認できます。

> **メトリクスチャート** *Old* メニューの詳細は、次の文書を参照してください。

その他のオプションを利用する

次のその他のオプションで、ダッシュボードのレイアウトを調整したり、ウィジェットに適用されたオプションを簡単に一括設定できます。



- **レイアウト**: ダッシュボードで列の数、つまり1行に配置できるウィジェットの数を設定できます。 最大4列まで設定できます。 **カスタム**を選択すると、列の数に関係なくウィジェットを配置できます。

  > **カスタム**では、ウィジェットの幅と高さを自由に調整できます。 他のオプションを選択すると、ウィジェットの高さだけを調整できます。

- **ウィジェットの凡例の表示**: オプションをオンまたはオフにすると、ウィジェットのチャートの右側に凡例を表示・非表示できます。 オプションをオフにした状態では**凡例**をクリックしてモニタリング対象を確認して選択できます。 凡例の活用についての詳細は次の文書を参照してください。
- **ツールチップにターゲットをすべて表示**: チャートにマウスかざすと、そのグラフの位置に該当する値を確認できます。 マウスが位置した時間帯のすべての対象のデータを確認するにはオプションをオンにします。
- **ウィジェットの高さを一括適用**: オプションをオンにすると、すべてのウィジェットの高さを同じく設定できます。

- **チャート時間の一括適用**: チャートウィジェットのデータ照会時間を変更する時、他のウィジェットのチャート時間も一緒に変更するにはオプションをオンにします。

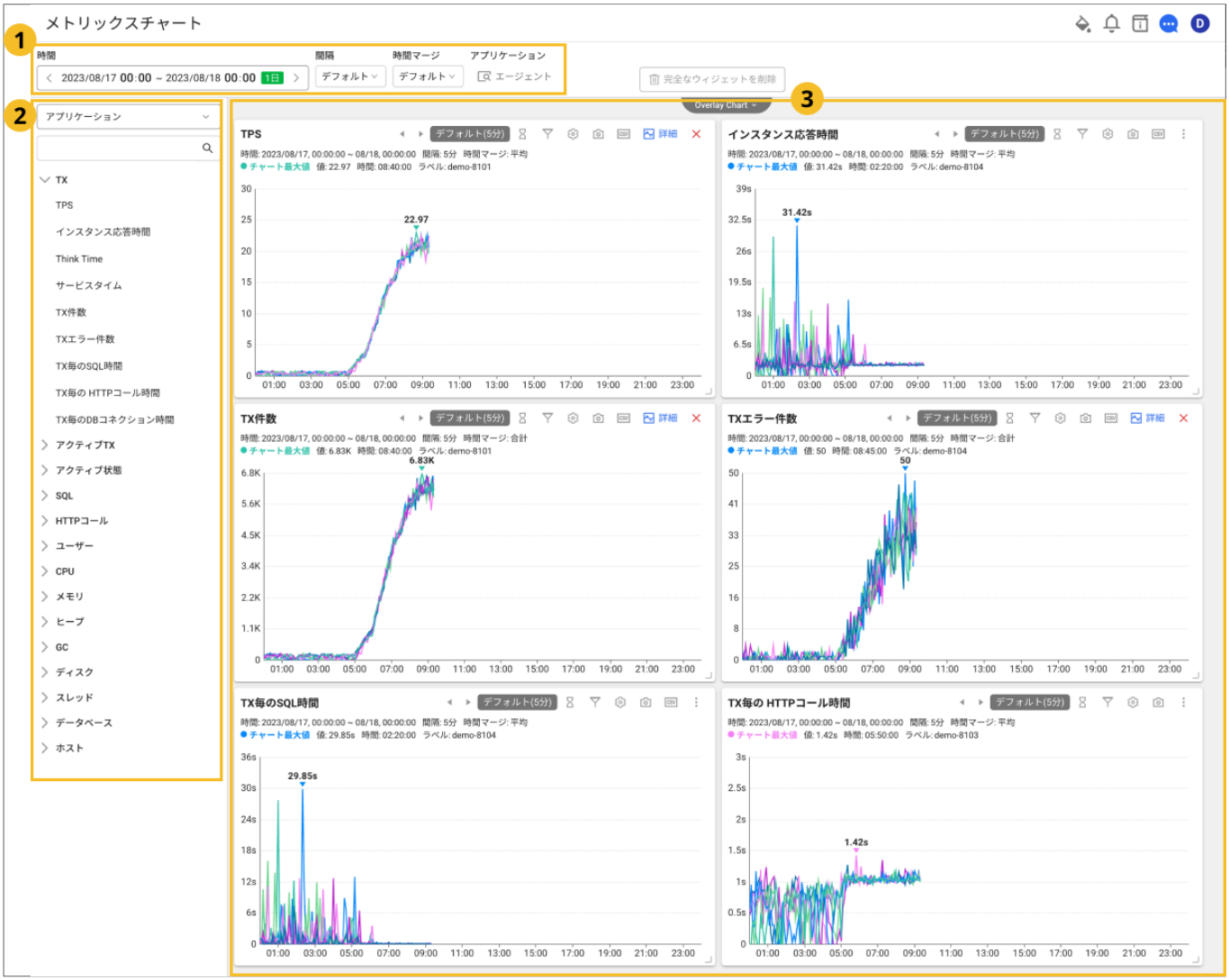
チュートリアルを利用する

画面の右上の**チュートリアル**ボタンを選択すると、Service 2.8.0 アップデート以降の変更を確認できます。

メトリクスチャート

サイトマップ > 分析 > メトリクスチャート *Old*

メトリクスチャートメニューでは、モニタリング対象から収集されたメトリクスデータを次のようにチャートで照会できます。時間と指標の指定が必須です。



上部オプション

1領域の**メトリクスチャート**の上部オプションからチャートの時間範囲とモニタリング対象エージェントを指定できます。

- **時間**：時間の範囲でX軸の開始と終了を指定できます。

  > **時間**オプションの利用方法の詳細については。

- **間隔**：時間間隔でX軸データの間隔を指定できます。
- **時間マージ**：データマージ方式の1つを選んで、指定した時間間隔内のデータをマージできます。

  例）**平均**は、1時間のデータ平均値を意味します。

- **エージェント**：照会するエージェントを指定できます。指定しない場合は、すべてを対象に照会します。

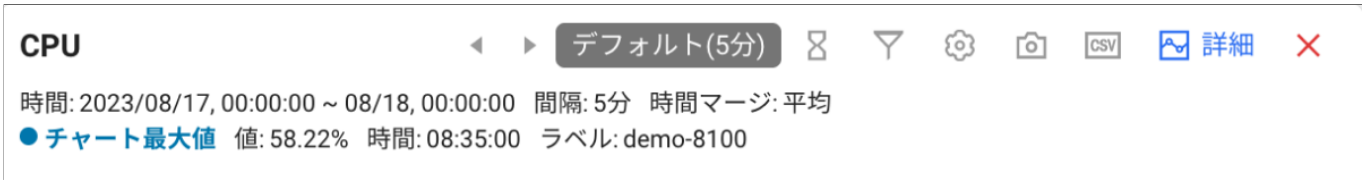
指標一覧

2領域は、オプションを検索する指標のリストです。まず、**カテゴリ**を選択します。 **カテゴリ**を選択すると、そのカテゴリに含まれる指標が表示されます。目的の指標を選択してください。**カテゴリ**と指標を選択すると、1領域の上のメニューで指定した時間範囲のデータから3領域でチャートウィジェットを照会できます。

> **データ結合**
>
> データのマージは、**オブジェクトのマージ**と**時間マージ**方法を提供します。
>
> - **時間マージ**は、ソースデータからフィールド値が同じデータを一定の間隔でマージします。
> - **オブジェクトのマージ**は、異なるフィールド値を持つデータの中からタグが一致する場合、そのデータをマージします。

チャートウィジェット

3領域のチャートウィジェットの左上で指標名を確認できます。チャートウィジェットの右上で次のオプションを確認できます。



- タイムシフト：◀ ▶左矢印または右矢印ボタンを使用して、選択した時間範囲だけ-1または、+1ずつ移動することができます。

  例えば、時間範囲が2月13日00:00~2月14日00:00の場合、◀左矢印を選択すると、2月12日00:00~2月13日00:00までのデータが表示できます。

- 間隔/時間のマージ：1トップメニューで指定した間隔と時間のマージを修正できます。
- モニタリング対象：アイコンを選択して、モニタリング対象を指定できます。選択しない場合は、すべて表示されます。
- 時間比較：アイコンを選択すると、同じ指標の以前の時間帯の推移を比較できます。
- スナップショット：アイコンを選択すると、ウィジェットのオプションを除くチャートのスナップショットを取得できます。
- CSV：CSVアイコンを選択して、チャートを描画したデータをCSVファイルとしてダウンロードできます。
- **詳細**：アイコンを選択して詳細を表示します。モニタリング対象が多い場合、モニタリング対象の指標推移を特定して確認できるため便利です。

> ⓘ メトリクスチャートウィジェットの上部にオプションが表示されない場合は、⋮アイコンを選択します。

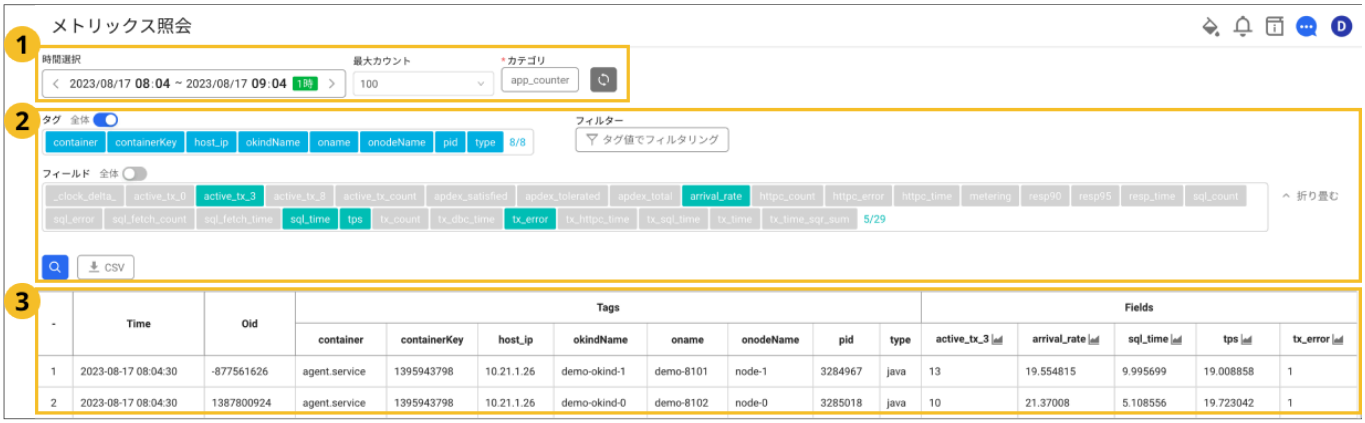
ウィジェットを削除する

画面に配置したすべてのウィジェットを削除するには、**全体ウィジェットの削除**ボタンを選択します。

メトリクス照会

ホーム画面 > プロジェクト選択 > **分析** > **メトリクス照会**

メトリクス照会メニューからタグベースで特定のメトリクスを照会できます。



1 時間とカテゴリの選択

1 領域でメトリクスが収集された**時間選択**と**最大カウント**および**カテゴリ**を指定できます。 **時間選択**と**カテゴリ**は、必ず指定してください。

- **時間選択**：メトリクスが収集された時間を指定して照会できます。 既定値は `1時間` です。 基本オプションとして提供する照会時間以外に、ユーザーが**時間選択**タブを選択して日付と時間を指定できます。

  > ⓘ **時間選択**オプションの利用方法の詳細については。

- **最大カウント**：3 一覧から参照するメトリクスの最大数を指定できます。 `10` 、 `50` 、 `100` 、 `200` 、 `300` 、 `1000` 、 `2000` 、 `3000` 個まで設定できます。
- **カテゴリ**：関連指標の分類単位です。 **カテゴリ**タブを選択し、**カテゴリ**を指定できます。
- **カテゴリ、タグ、フィールドオプションの更新**：**カテゴリ**、**タグ**および**フィールド**オプションを再読み込みできます。

2 タグとフィールドの選択

2 領域で**タグ**と**フィールド**を選択します。 ユーザーが個別に指定しない場合、デフォルト設定が完全な選択になります。

- **タグ**：収集された対象を区分できる固有情報データです。
- **フィールド**：モニタリング対象から収集された指標です。
- **フィルター**：**タグ値でフィルタリング**ボタンを選択し、タグ値を設定してフィルタリングできます。

  例） `oname` の値を `demo-8101` に設定すると、フィルタリングしたデータを表示することができます。

- **検索**：条件を設定して**検索**アイコンを選択すると、3 領域でそのメトリクスの原本データを検索できます。
- CSV：メトリクスの原本データをCSVファイルとしてダウンロードできます。

3 メトリクステーブル

収集されるメトリクスを事前に特定できないため、収集されるすべてのメトリクスの原本データを確認することが重要です。 上記の条件領域で目的の条件を設定した後3 領域で、そのメトリクスの原本データをテーブル形式で検索できます。 ユーザーが**タグ**と**フィールド**の各条件を指定することによって、テーブルのカラムが変更されます。

> - メトリクスの参照時、**時間選択**と**カテゴリ**を指定する必要があります。
> - メトリクス参照時、**タグ**と**フィールド**の指定はオプションです。

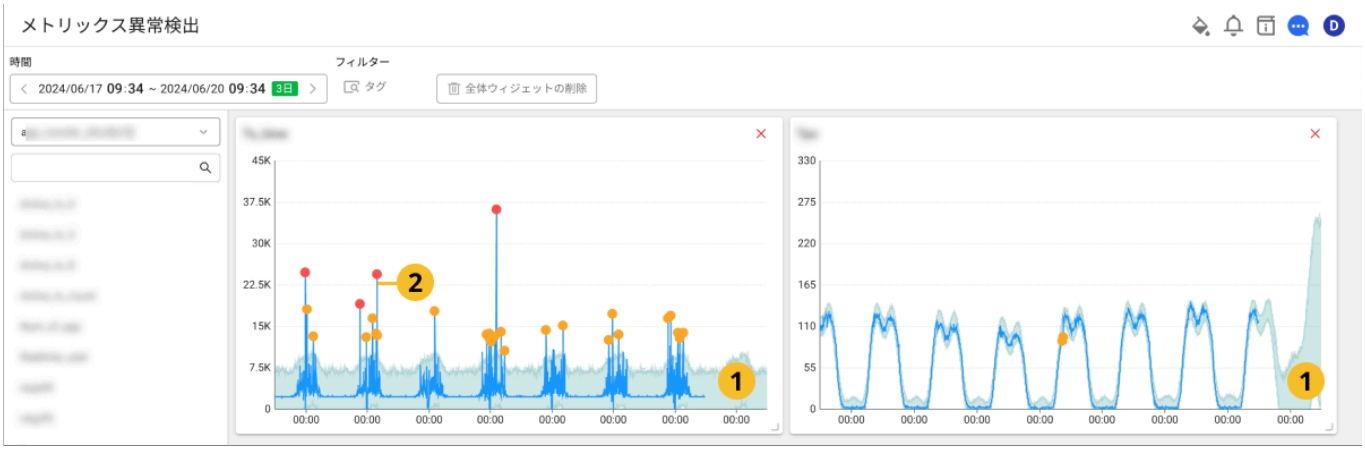
メトリクス異常検知

ホーム画面 > プロジェクト選択 > **サイトマップ** > **メトリクス異常検出**

さまざまなメトリクスのパターンをAIが学習した予想パターンと比較できます。予想パターンを外れた異常探知をグラフチャートで確認できます。過去のデータに基づいて繰り返されるパターンを確認し、今後の指標値の予測に活用できます。

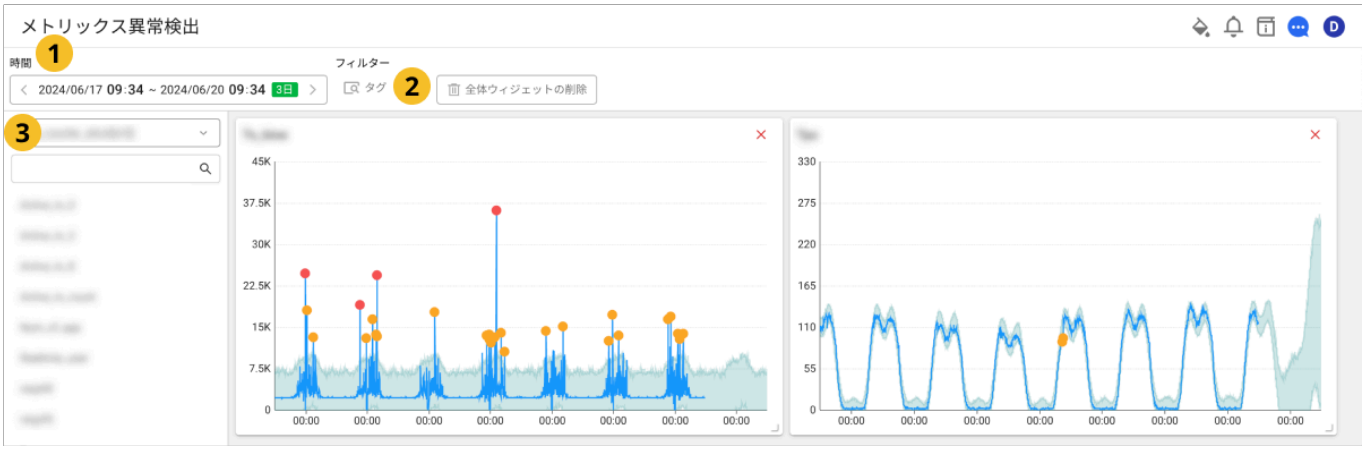
- 画面に配置したウィジェットは、別のメニューに移動すると保存されずに初期化されます。
- パターン分析と異常値検出機能を除くと**分析** > **メトリクスチャート**メニューに似ています。
- **異常値**(Anomaly Detection)アラート通知機能の技術的根拠は、**メトリクス異常値**です。**異常値**アラート通知設定の詳細については、次の文書を参照してください。

ウィジェットを確認する



- 1 明るい色のグラフ領域は、AIによって分析した予想されるパターンです。
- 2 青色グラフは、プロジェクトのメトリクスの推移です。
- AIによって分析された予想パターンから外れるとオレンジ色と、赤色のステップでグラフにドットが表示されます。予想されるパターンの範囲から大幅に外れた値を赤色として表示します。

ウィジェットを配置する



1. 1 **時間**で、目的の時間間隔を設定してください。最大1か月間隔まで設定できます。
2. 2 **フィルター**でメトリクスの範囲を選択します。
3. 以下の一覧でモニタリングするメトリクスを選択します。

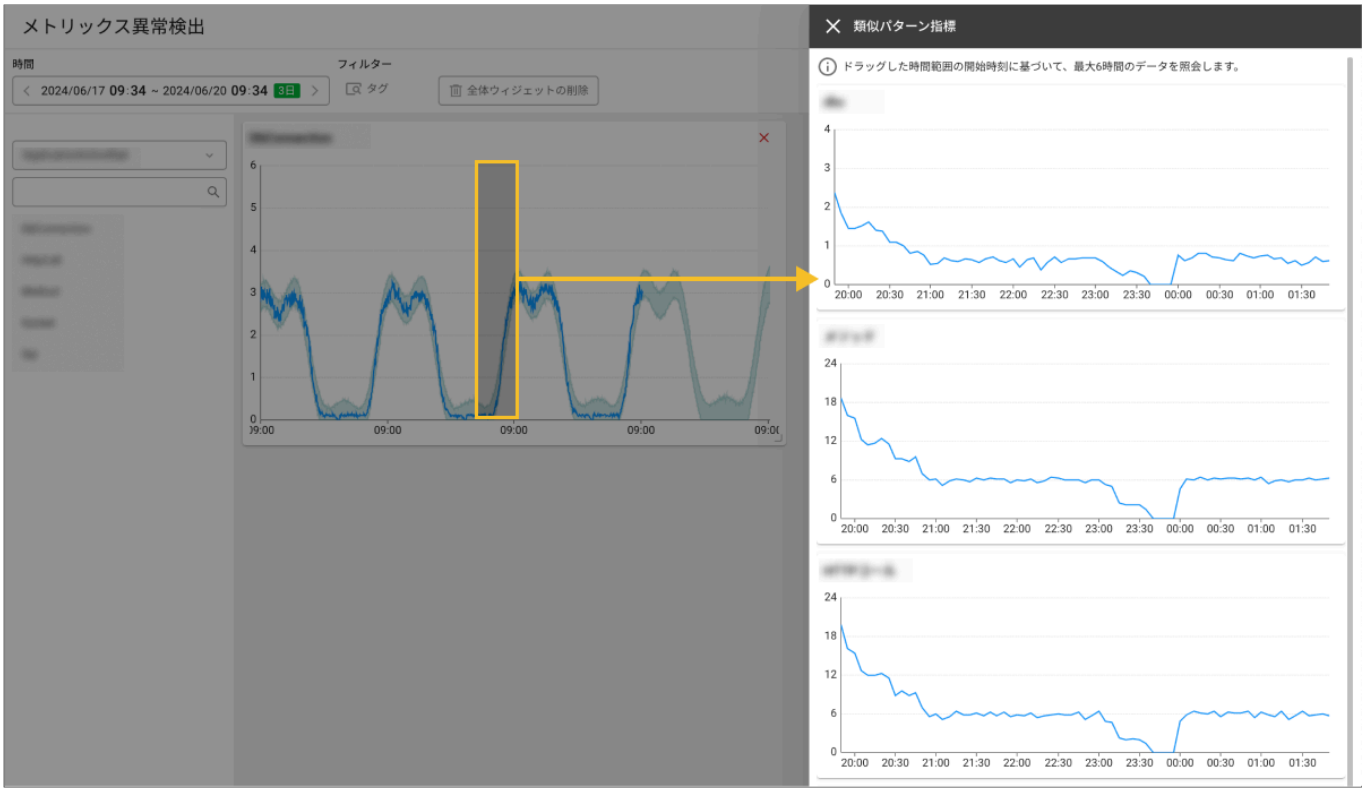
選択したメトリクスを画面の右側に配置します。

- モニタリング対象を選択してメトリクスを絞り込んで確認するには、2 **タグ**を選択します。画面右側にタグ選択リストが表示されます。目的の項目を選択し、ウィジェットを追加します。
- 画面に配置したウィジェットは、**時間**または**タグ**値を変更してもチャートに反映されません。
- 画面に配置したウィジェットを削除するには、ウィジェット右上の✕ボタンを選択します。
- ウィジェットの左上を選択した状態でドラッグすると、ウィジェットの位置を変更できます。
- ウィジェットの右下を選択した状態でドラッグすると、ウィジェットのサイズを変更できます。

ウィジェットを削除する

画面に配置したすべてのウィジェットを削除するには、**全体ウィジェットの削除**ボタンを選択します。

関連メトリクスを確認する



ウィジェットでチャートの一部の領域をドラッグします。画面の右側に**類似パターン指標**画面が表示されます。ドラッグした時間範囲の開始時間に基づいて、最大6時間の関連メトリクスを照会できます。

トランザクショントラッキング概要

トランザクションとは？

ユーザーがブラウザからのリクエストを処理するためのサーバーサイドのLUW(Logical Unit of Work)です。個々のウェブサービス(URL) のリクエストに対する処理過程がトランザクションです。ウェブアプリケーションでは、トランザクションは、ウェブサービス(URL)に対して、HTTP Requestを受信し、Responseを返すプロセスです。

アプリケーションの性能は、これらのトランザクションの性能であると言えます。トランザクションの性能には、トランザクションの開始から終了までの時間、レスポンス時間およびリソースの使用量、トランザクション呼び出し元の属性などの情報が含まれます。

基本的には、トランザクションのレスポンス分布とトランザクション統計を使用してトランザクションの性能を分析できます。

トランザクション名

トランザクションの名前はURLです。 ただし、Getパラメータ(Query String)は除外されます。

ブラウザ要請

```
http://www.whatap.io/hr/apply.do?name='kim'
```

トランザクション名

```
/hr/apply.do
```

> WhaTapでは、**ウェブサービス名**と**トランザクション名**を組み合わせて使用します。特定のURLとそのリクエストを処理するためのサービスモジュールとして表示します。トランザクションリクエストの1つの処理を参照するために、両方の名前を同じ方法でURLと呼ぶことができます。

トランザクション名の正規化

MSAベースのシステム開発に伴い、URL + ? の引数パラメータ形式よりもURLパスにパラメータを入れる方式をよく使います。

```
http://www.whatap.io/hr/kim/apply.do
```

このようにパスパラメータをそのままトランザクション名として使用すると、統計的な観点から性能分析が困難になるため、これを正規化する必要があります。WhaTapは、正規化のためのオプションと機能を提供します。

whtap.conf

```
trace_normalize_urls=/hr/{name}/apply.do
```

上記のように設定した場合、トランザクション名は、/hr/kim/apply.do → _/hello/:name/apply.do_に置き換えられ収集されます。対象URLの設定はそのままにして、この機能をoffにする場合は、次のオプションを指定することができます。 既定値は `true` です。

whatap.conf

```
trace_normalize_enabled=false
```

トランザクション照会

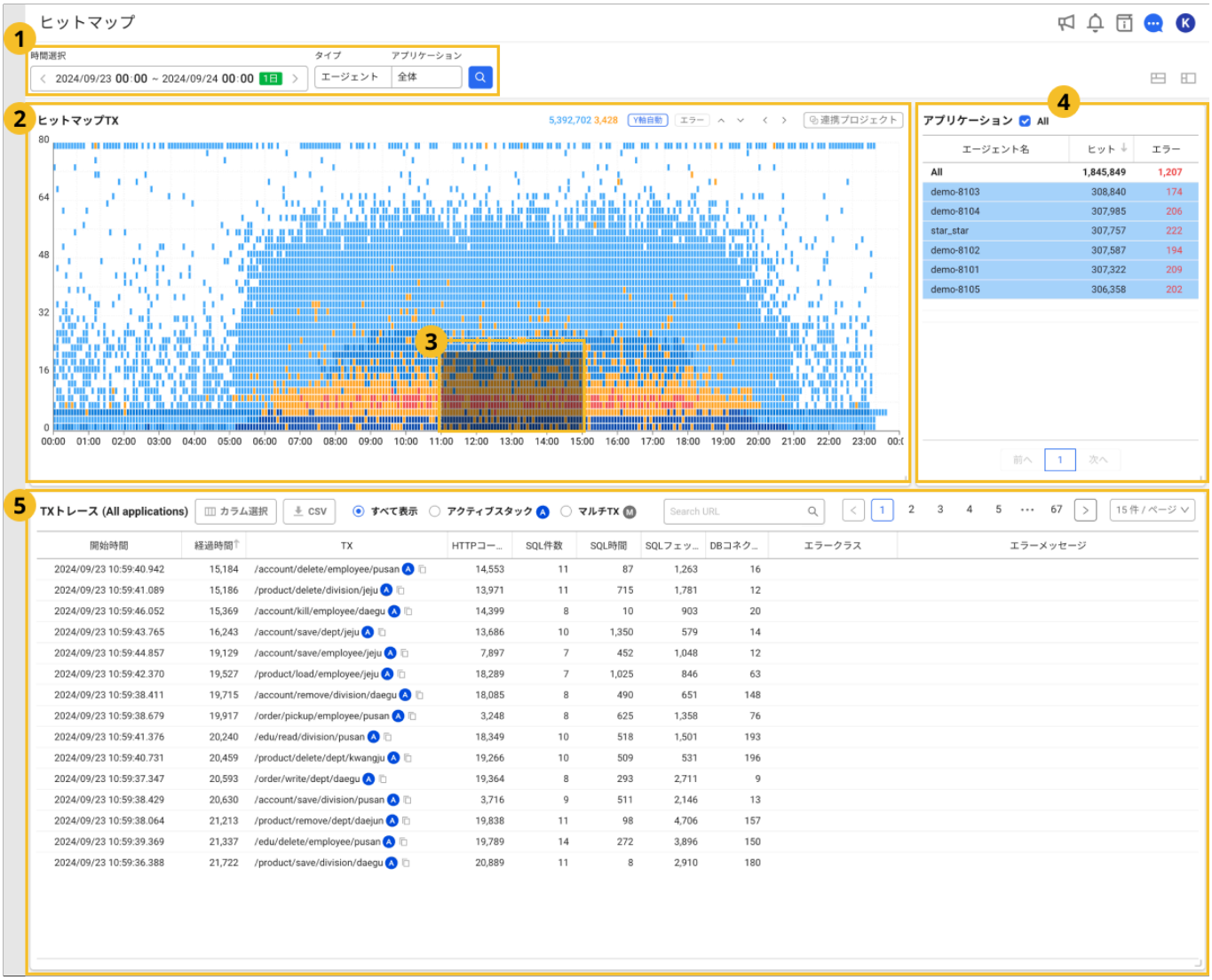
ヒットマップ

ホーム画面 > プロジェクト選択 > **分析** > **ヒットマップ**

WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**分析** > **ヒットマップ**メニューを選択します。

> **ヒットマップ**は、**ダッシュボード** > **アプリケーション**の**ヒットマップ**ウィジェットからもアクセスできます。

詳細分析



1 エージェント領域

現在のプロジェクトと関連づけられたエージェントを選択し、フィルタリングできます。🔍ボタンを選択すると、2 領域で**ヒットマップTX**チャートが表示します。

2 ヒットマップTXチャート

遅延の問題があるトランザクションが含まれている場合**ヒットマップTX**チャートに、オレンジや赤で表示されます。通常のトランザクションが含まれている場合、青で表示されます。

3 選択

リアルタイムヒットマップチャートをドラッグすると、4 **アプリケーション**一覧とトランザクション情報を確認できる5 **TXトレース** 一覧が表示されます。

4 アプリケーション一覧

チャートで選択された領域に該当するTXの数とエラーの数を確認できます。一覧からアプリケーションを選択すると、該当アプリケーションの5 **TXトレース**一覧に詳細が表示されます。

5 TXトレース一覧

TXトレース一覧には詳細情報が含まれています。一覧からトランザクションを選択すると、**トランザクション情報**画面が表示されます。**トランザクション情報**画面から該当トランザクションのトレース分析を確認できます。詳細については、次の文書を参照してください。

ヒットマップパターンを理解する

ヒットマップは、トランザクションの終了時間をX軸、レスポンス時間をY軸とした分布チャートです。通常のWebアプリケーションは、数秒以下の集中分布を表示します。

hitmap-pattern

ヒットマップラインの分析

- **縦線(LOCK現象) パターン**

  トランザクション処理中に一時的なロック(Notonly DB Lock)が発生すると処理が待機されます。ロックが解除されると、処理待機中のトランザクションは同じ時間帯で同時に終了します。これにより、次のように垂直線が作成されます。

  hitmap-lock

垂直パターンでロックを検知することは非常に強力な概念です。特にマイクロサービスアーキテクチャでは、バックエンドシステムで発生するロックについても同様に検知できます。

hitmap-front-api-db

Frontアプリケーションのレスポンスパターンの縦線は、バックエンドシステムが使用するDBでロックが発生した場合も検知されます。

- **横線(タイムアウト) パターン**

  10秒のタイムアウト条件でリソースが不十分な場合、トランザクションは10秒間待機後にタイムアウトエラーが発生します。この時点で、ヒットマップの約10秒後に、次のような横線が作成されます。

  hitmap-timeout

  タイムアウト後に再実行するロジックがある場合は、図のような横線が10秒単位で繰り返されます。以下は、実際の**トラブル状況**のヒットマップです。

  hitmap-error

  (1) セクションでレスポンス時間が増加し、(2) セクションの赤い線は典型的な**横線パターン**です。(1) セクションの負荷によりConnectionPoolが使い果たされ、(2) セクションはConnectionPoolの不足で2回目の**タイムアウト**トラブルが発生した状態です。

パターン分析活用

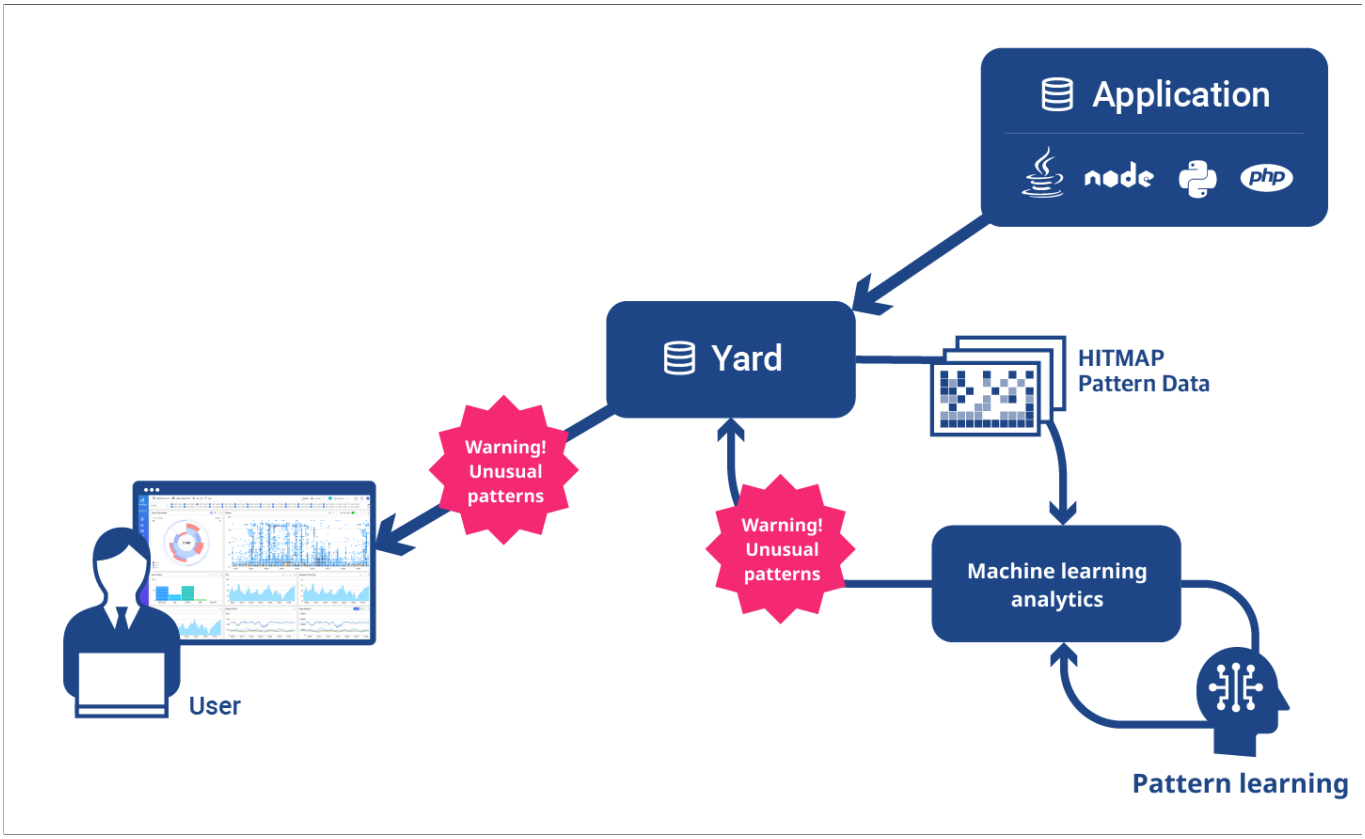
トランザクションレスポンスの分布に線が表示されている場合は、ボトルネックがあることを意味します。一時的なロッキングの場合は縦線が作成され、ボトルネックがタイムアウトになると横線が作成されます。

hitmap-pattern-analysis

問題を分析する際に特定のラインに含まれるトランザクションのみを選択的に分析して、問題をすばやく発見できます。

機械学習ベースの応答パターン分析

機械学習技術によりヒットマップパターンを分析し、異常を自動検知してアラートを発報する機能です。



異常パターンの例

毎月数百TBのパフォーマンスデータから異常パターンを学習し、学習した異常パターンと同様のパターンが発生するとアラートを発報します。

- **縦線パターン**

  hitmap-abnormal1

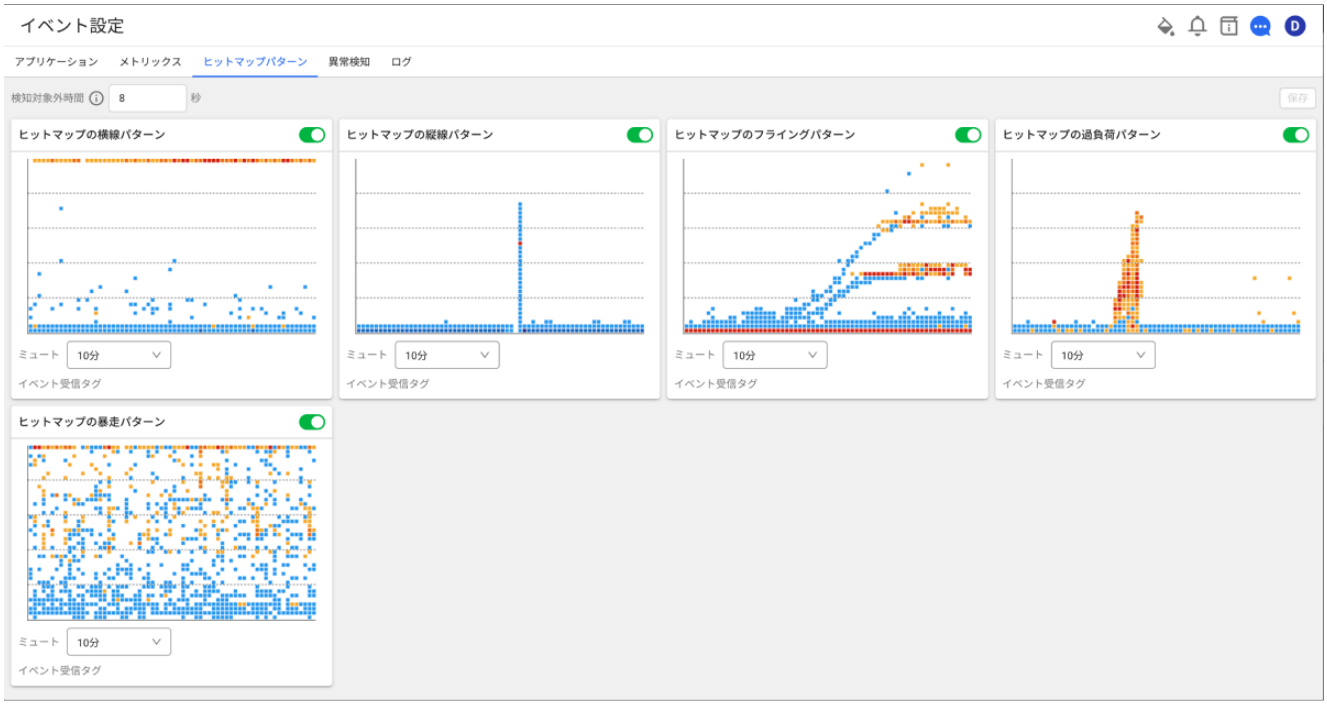
- **横線パターン**

  hitmap-abnormal2

- **複合パターン**

  hitmap-abnormal3

- **ヒットマップ通知**

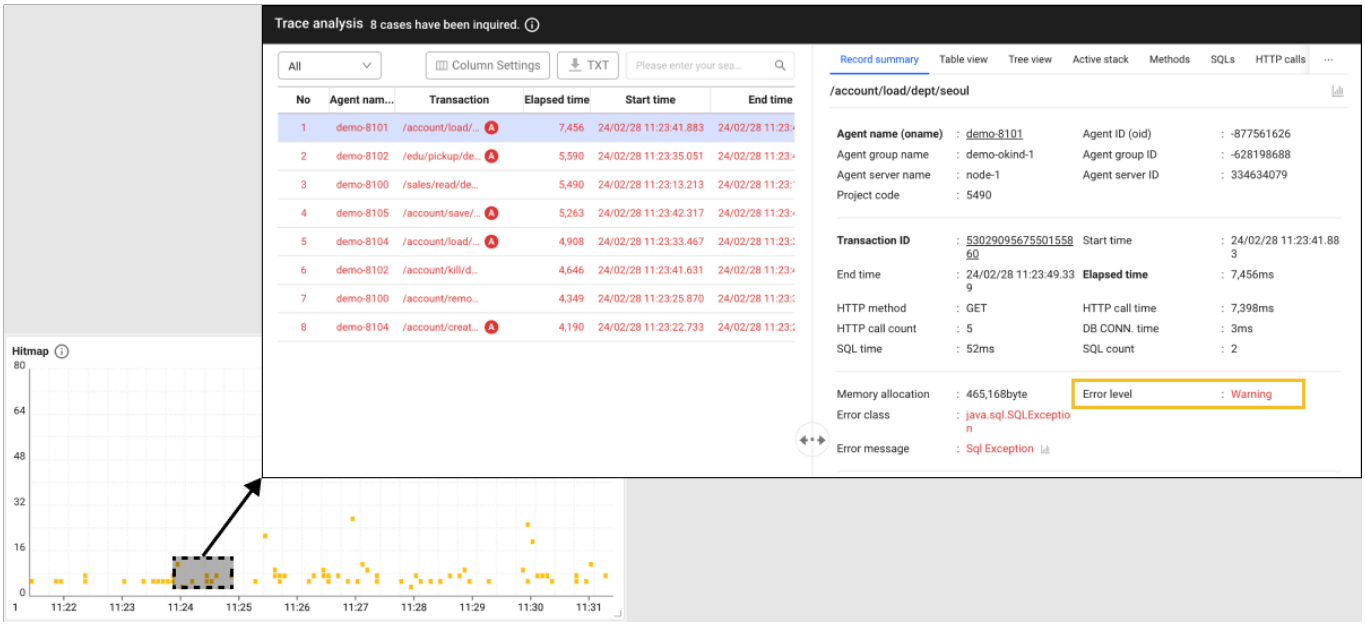


> **ヒットマップTX**チャートを分析する方法については、次の文書を参照してください。

Javaエージェントの例外処理(WARNING) 基準

Java言語を使用するアプリケーション環境で発生するエラーをJavaエージェントが例外処理する基準について案内します。次の内容は、Javaエージェントv2.2.27 基準で作成されました。

サービス画面でのエラー表示



ヒットマップウィジェット、**トレース分析**画面から確認できる**エラーレベル**は、殆どWARNINGレベルに該当します。

Spring ExceptionHandler通過時のエラー処理

Springフレームワークでは、`org.springframework.web.servlet.DispatcherServlet.processHandlerException` メソッドを通過する場合、エラーとして処理します。ただ、開発者がTry-Catch文を使用する場合は、`ExceptionHandler` を通過しませんので、WhaTapではエラー処理を行いません。

Status Code 400以上のエラー処理

HTTPレスポンスコードが400以上の場合は、例外クラス(Exception Class)が発生しなくてもエラーとして処理します。

JDBC driver呼び出し時のエラー処理

JDBC接続呼び出しやSQL実行呼び出し時に発生するエラーはエラーとして処理されます。

HTTP callエラー処理

HTTP呼び出し中に発生するエラーは、`SLOW_HTTPC`、`httpc error`などの形で表示されます。

WhaTapで定義したエラー処理

WhaTapモニタリングサービスでは、特定の条件に従ってエラーとして処理します。

- **TOO_MANY_RECORDS**：fetch件数10,000件以上

  関連するエージェントオプションについては、次の文書を参照してください。
  `profile_error_jdbc_fetch_max=10000`

- **SLOW_SQL**：SQL時間30,000ms以上

  関連するエージェントオプションについては、次の文書を参照してください。
  `profile_error_sql_time_max=30000`

- **SLOW_HTTPC**：HTTP call経過時間が10,000ms以上

  関連するエージェントオプションについては、次の文書を参照してください。
  `profile_error_httpc_time_max=10000`

- **httpc error**：HTTP call status code 400以上の場合

  関連するエージェントオプションについては、次の文書を参照してください。`httpc_status_error_enable=true`

- **status error**：トランザクションstatus code 400以上の場合

  関連するエージェントオプションについては、次の文書を参照してください。
  `transaction_status_error_enable=true`

トランザクションエラー段階のINFO処理

エージェント設定により、一部のエラーを正常な状態で表示または無視できます。**ヒットマップ**ウィジェットで表示されるレベルは、INFO(青)です。

- `biz_exceptions`：レコードエラー表示

  whatap.conf

```
biz_exceptions=io.home.test.baseapp.base.exception.custom.CustomParameterizedException
```

- `ignore_exceptions`：レコードエラー未表示

  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
  ```

> ⓘ エージェント設定は、whatap.confファイルを編集するか**管理** > **エージェント設定**メニューで確認できます。詳しい内容は、次の文書を参照してください。

メソッド例外処理

`hook_method_patterns`、`hook_service_patterns`オプションで設定したメソッドが実行中にExceptionを発生させる場合、当該例外はエラーとして処理されます。

> Try-Catch構文を使用してメソッド内部でExceptionを処理した場合、Javaエージェントは当該Exceptionを収集対象から除外し、エラーとして処理しません。

トランザクションプロファイル

トランザクショントレーシングとは？

トランザクション性能とは、トランザクションの開始と終了までの要約指標や属性を意味します。一方、トランザクショントレースは、トランザクションが実行されているステップのトレースです。 トランザクションが遅いかエラーがある場合は、その原因を追跡するために実行履歴をステップごとに追跡する必要があります。 これを**トランザクショントレーシング**と言います。

トランザクションのパフォーマンス追跡のために収集するステップの種類は次のとおりです。

- **DB接続ステップ** START-END

  RDBへの接続のパフォーマンスが含まれます。 ステップ情報には、名前、レスポンス時間、エラーが含まれます。

- **SQLステップ** START-END

  JDBC SQLのパフォーマンスが含まれます。 ステップ情報には、接続情報、SQL文、エラーが含まれています。

- **HTTP Callステップ** START-END

  外部HTTPサービス呼び出しのパフォーマンスが含まれます。 ステップ情報には、url、host、port、レスポンス時間、エラーが含まれます。

- **Messageステップ** ADD

  トレースを収集する過程で、非構造化のすべての間隔の履歴を収集する際にメッセージステップを使用します。

- **SOCKETステップ** ADD

  Socketオープンを表現するステップです。

- **METHODステップ** START-END

  メソッド応答時間を追跡します。

- **ACTIVE STACK ステップ** START-END

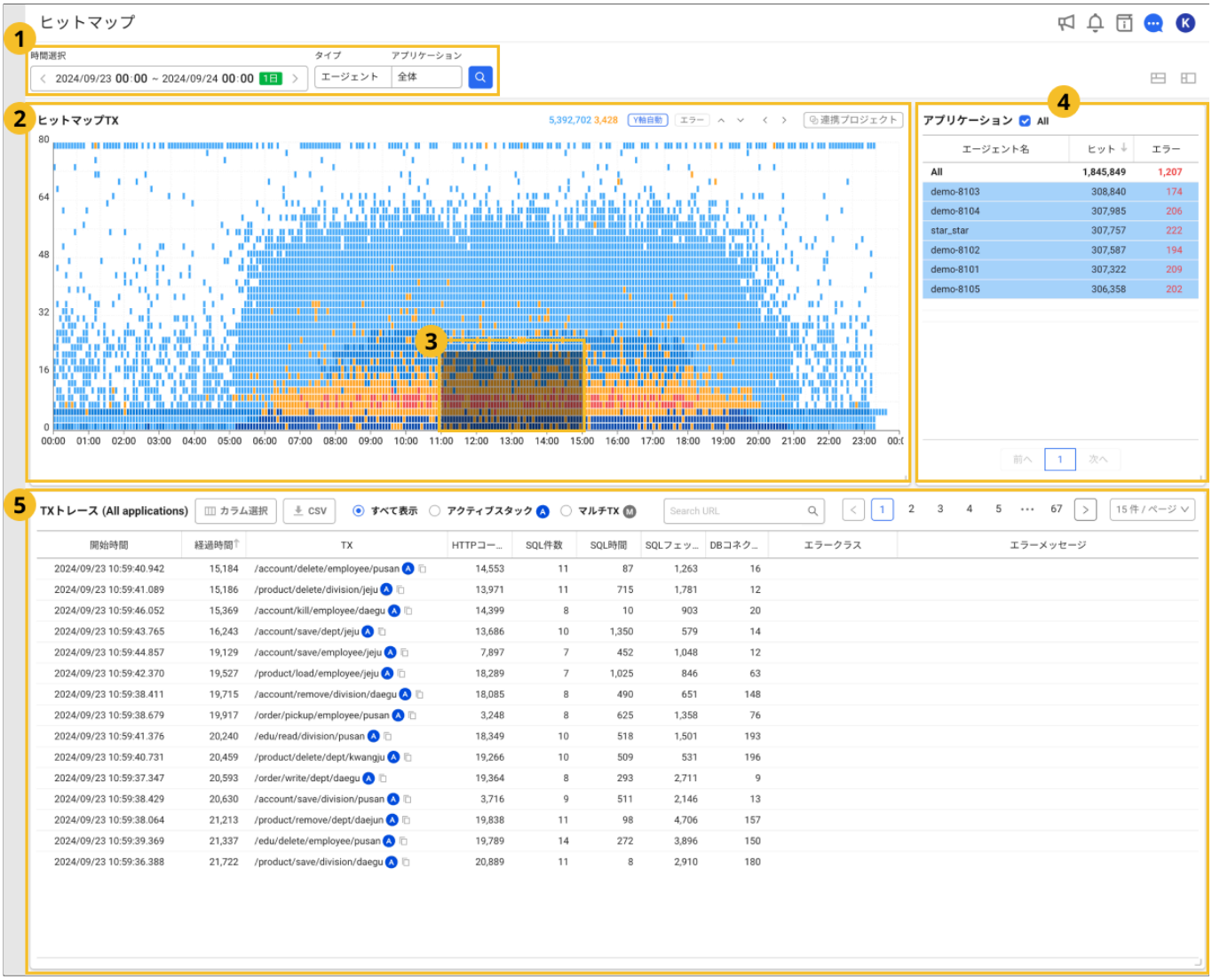
  アクティブスタックに関する情報を含みます。 別のスレッドが作成し、トレースに追加する方法で収集します。

連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、連携したプロジェクトの性能メトリクスを照会できます。 連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

WhaTapモニタリングサービスでは、**ステップ**(**Step**)は**スパン**(**Span**)と同じ意味で使用されます。

詳細分析

ホーム画面 > プロジェクト選択 > **分析** > **ヒットマップ**

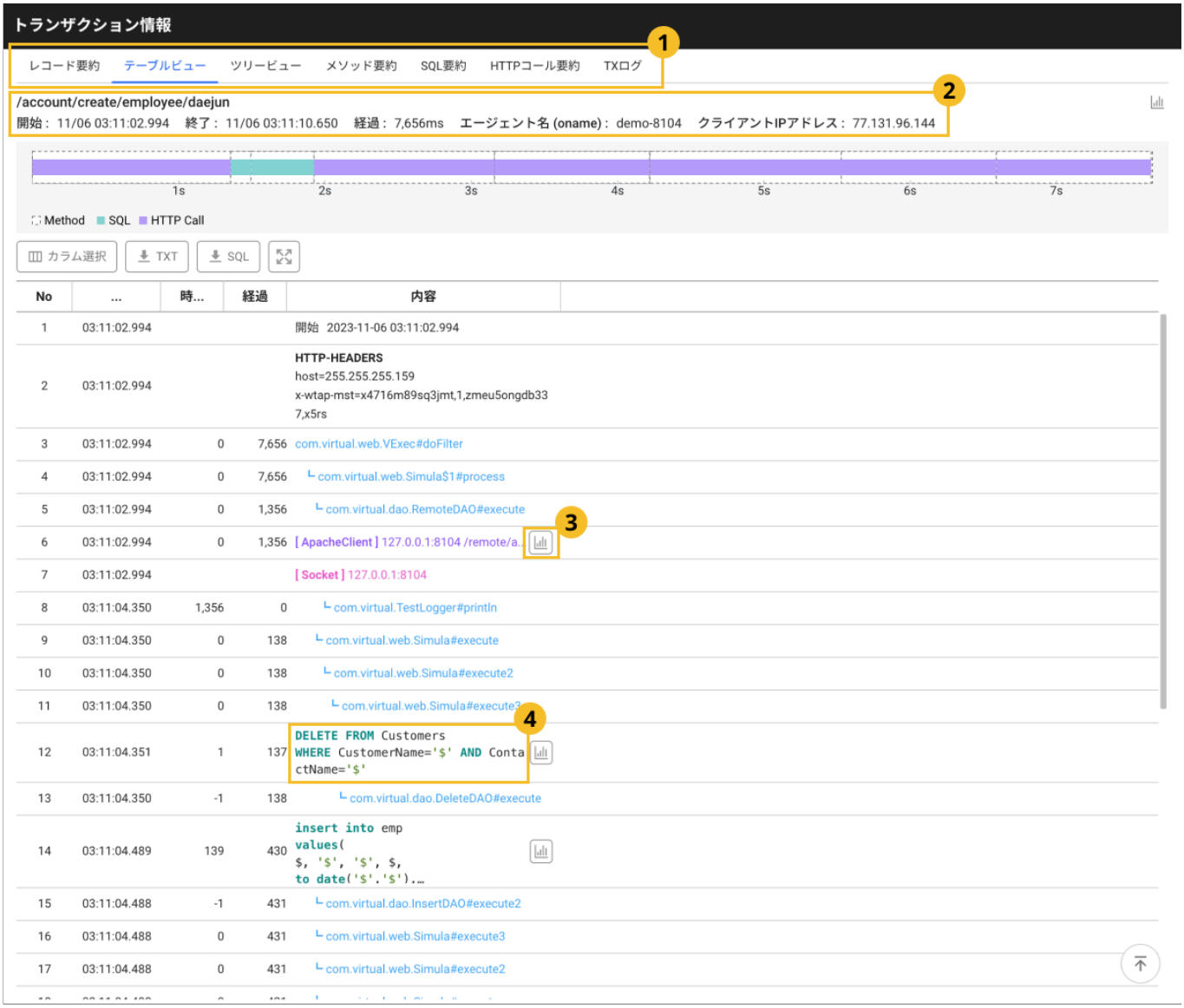


1. 1 **時間選択**で、ヒットマップを照会する期間を選択します。
2. 1 **タイプ**、**アプリケーション**から照会する対象を選択します。
3. 検索ボタンを選択します。
4. 2 **ヒットマップTX**から照会する領域を3のようにドラッグします。
5. 5 **TXトレース**に一覧が表示されたら、分析するトランザクションを選択します。

トランザクションのパフォーマンス分析のためのクライアント情報などの属性、トランザクションの処理性能、各区間別の詳細な実行履歴などを確認できる**トランザクション情報**画面が表示されます。

- **ヒットマップ**メニューの機能については、次の文書を参照してください。
- アプリケーションダッシュボードのヒットマップウィジェットの詳細については、次の文書を参照してください。

トランザクション情報画面の案内



- 1領域では、ステップ情報をさまざまな形式で確認できます。 希望する形式のタブを選択します。

- 2領域では、トランザクションに関する基本情報を確認できます。
- 3：該当URLまたは各実行区間の統計データ画面が表示されます。 統計データ画面のグラフチャートで、希望する時間をクリックすると**統計**または**TX検索**メニューに移動します。 選択した時間に基づいて統計データを照会することができます。

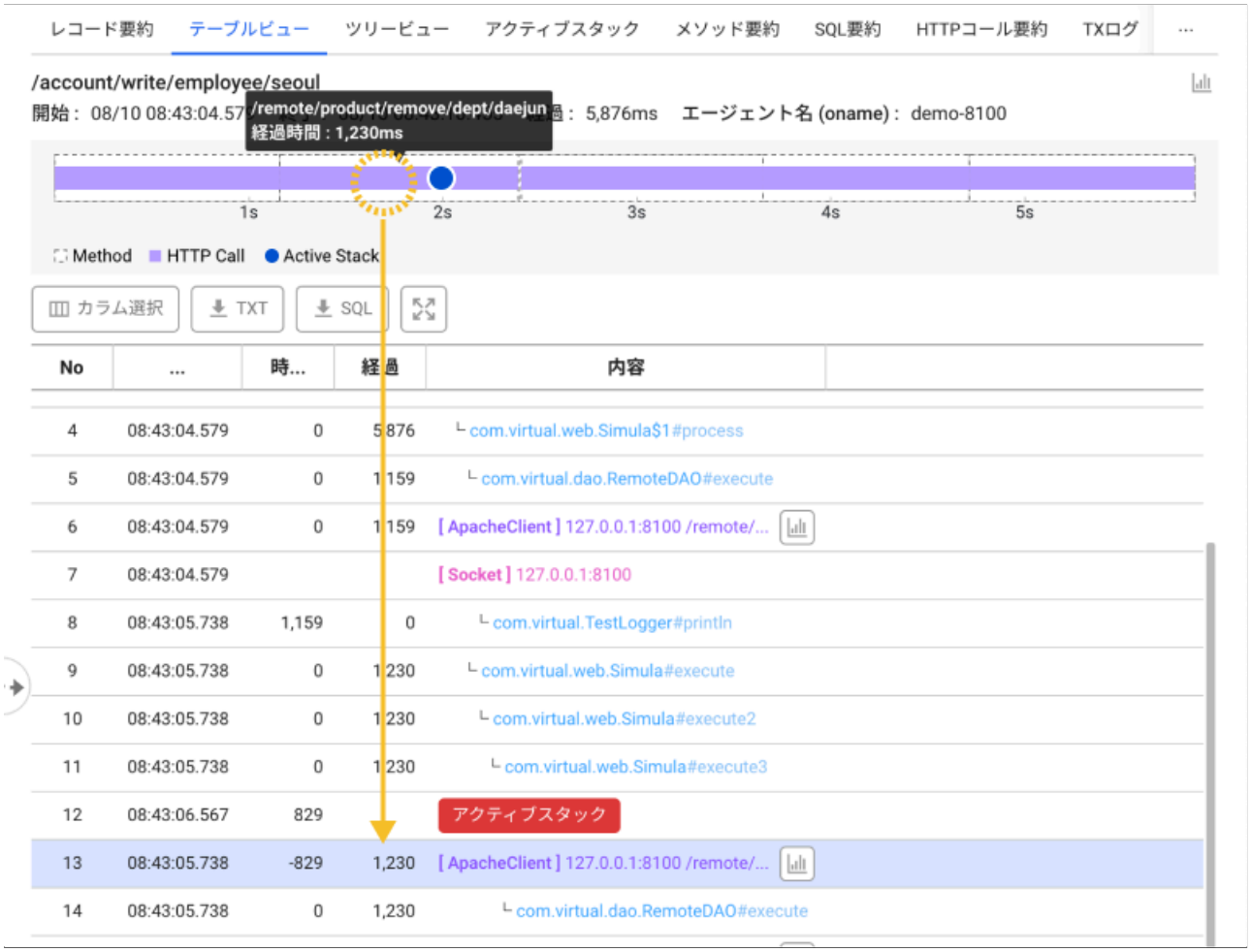
  > ◦ **統計**メニューの詳細については、次の文書を参照してください。
  > ◦ **TX検索**メニューの詳細は次の文書を参照してください。

- 4SQLステップを選択すると、SQL変数とHTTPクエリを照会することができます。 詳細については、次の文書を参照してください。

テーブルビュー

テーブルビュータブでは、トランザクションの実行過程を時間順に確認できます。

- トランザクションを実行区間別に分類し、ダイアグラムで確認できます。 全経過時間のうち、各区間別に所要された時間、最も長くかかった区間を素早く把握できます。
- ダイアグラムの各区間を選択すると、そのステップが位置するテーブルの一覧に移動します。



- 青い丸(●)が位置する領域は、アクティブスタックが収集された時点です。 青い丸を選択すると、**アクティブスタック**ボタンがあるテーブル一覧に移動できます。 **アクティブスタック**ボタンをクリックすると、その区間で行われたステップ情報を確認できます。

> ⓘ ◦ トランザクション一覧からⒶアイコンが表示された項目で確認できます。
> ◦ アクティブスタックの詳細については、次の文書を参照してください。

- **カラム選択**：テーブル一覧にカラムでメモリ累積情報およびCPU累積情報を追加または非表示できます。 カラム情報は次のとおりです。
  - **No**：ステップの実行順序
  - **時間**：各ステップの開始時刻

  - **時間差**：直前ステップの開始時刻から現在のステップに移るまでの待ち時間、外部要因により遅延される場合、経過時間と異なる場合があります。
  - **経過**：各メソッドの開始から終了までの所要時間
  - **内容**：該当ステップの詳細な実行内容
- **TXT**：トランザクションの基本情報および区間ごとの実行情報をTXTファイルでダウンロードできます。
- **SQL**：トランザクションの基本情報およびSQL実行情報をTXTファイルでダウンロードできます。
  - **置換値を含むダウンロード**: バインド変数値を元の値に置き換え、SQL統計データをダウンロードします。 セキュリティキー(`paramkey`)が設定されている場合は、セキュリティキー入力画面が表示されます。 セキュリティキーを入力しないとダウンロードできません。 セキュリティキーに関するエージェント設定の詳細については、次の文書を参照します。
  - **置換値を含まないダウンロード**: バインド変数値を置き換えない状態でSQL統計データをダウンロードします。
- :トランザクションの要約情報とダイアグラムを非表示にし、テーブル一覧のみを確認できます。 ボタンをクリックすると、隠された情報を再表示します。 テーブル一覧が長い場合は、この機能を利用すると便利です。
- SQLステップを選択すると、パラメータを照会できる**SQL**画面が表示されます。 詳細については、次の文書を参照してください。

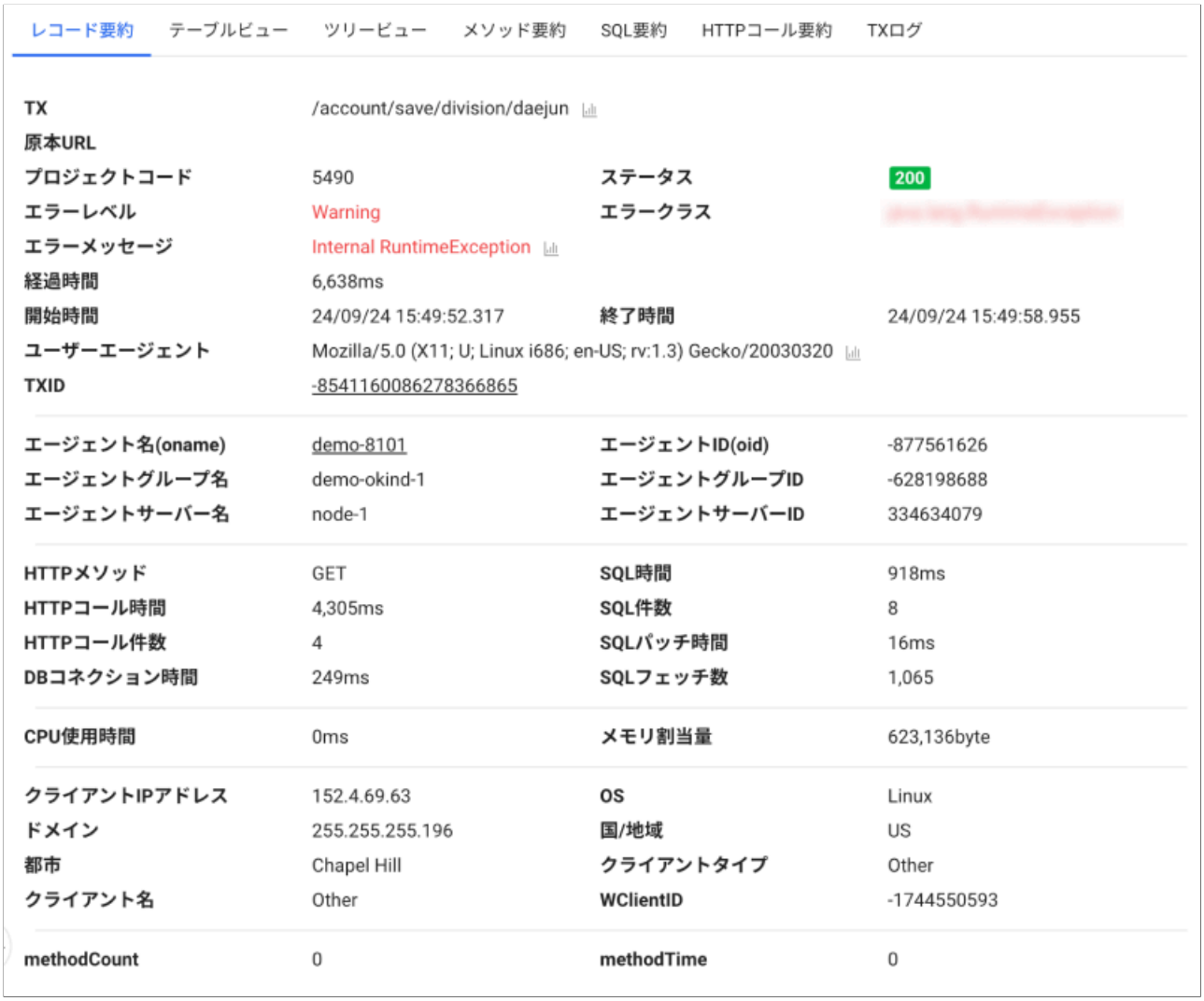
連携プロジェクト管理メニューでデータベースプロジェクトを追加すると、**DB Connectionステップ**で連携したプロジェクトの性能メトリクスを照会できます。



連携プロジェクトの追加と分析についての詳細は、次の文書を参照してください。

レコード要約

トランザクションの基本情報を確認するには、**レコード要約**タブを選択します。 トランザクションの基本情報およびそのトランザクションが実行されたエージェント関連情報、メモリ割り当て使用量、クライアント関連情報を確認できます。



分類	プロパティ	説明
トランザクション	TX	アプリケーションで実行されたAPIまたはURL呼び出し、そのトランザクションのパスです。 正規表現式を使用して一定の形式に変換します。

分類	プロパティ	説明
	原本URL	実際に呼び出されたURLパスです。 変数を含まない具体的なリクエストパスです。 元のURLがある場合は元のURL表示、元のURLがない場合は表示しません。
	プロジェクトコード	WhaTapモニタリングサービスに登録したプロジェクトの識別コードです。
	ステータス	そのトランザクションのHTTP応答ステータスのコードです。
	エラーレベル	そのトランザクションで発生したエラー レベルです。 WarningまたはCriticalと表示されます。
	エラークラス	そのトランザクションで発生したエラー関連クラスです。
	エラーメッセージ	そのトランザクションで発生したエラーメッセージです。
	経過時間	トランザクションの実行時間です。
	開始時間	トランザクションの開始時刻です。
	終了時間	トランザクションの終了時刻です。
	ユーザーエージェント	クライアント関連情報の抽出に利用するブラウザ情報です。
	リファラー	クライアントがブラウザで以前に利用したページアドレスまたは流入パスです。
	TXID	該当トランザクションの固有識別子(ID)です。
	マルチTXID	マルチトランザクションの固有識別子(ID)です。 他のアプリケーションのトランザクションと並列に実行された場合に表示されます。
エージェント	エージェント名(oname)	エージェント名です。
	エージェント	エージェントの固有識別子(ID)です。

分類	プロパティ	説明
	ID(oid)	
	エージェントグループ名	`okind`オプションで設定されたエージェントグループ名です。
	エージェントグループID	エージェントグループの固有識別子(ID)です。
	エージェントサーバー名	`ondoe`オプションで設定したエージェントサーバー名です。
	エージェントサーバーID	エージェント サーバーの固有識別子(ID)です。
ステップ (Step)	HTTPメソッド	HTTPメソッドです。 (GET、POST、PUT、HEADなど)
	HTTPコール時間	外部Http Call時間です。
	HTTPコール件数	外部HTTP Call件数です。
	DBコネクション時間	データベースに接続するまでにかかる時間です。
	SQL時間	SQLの実行時間です。
	SQL件数	SQLの実行件数です。
	SQLパッチ時間	レコードの照会にかかった時間です。 ネストが発生するか、他のロジックが含まれる場合があります。
	SQLフェッチ数	SQLでデータを照会したレコードの件数です。

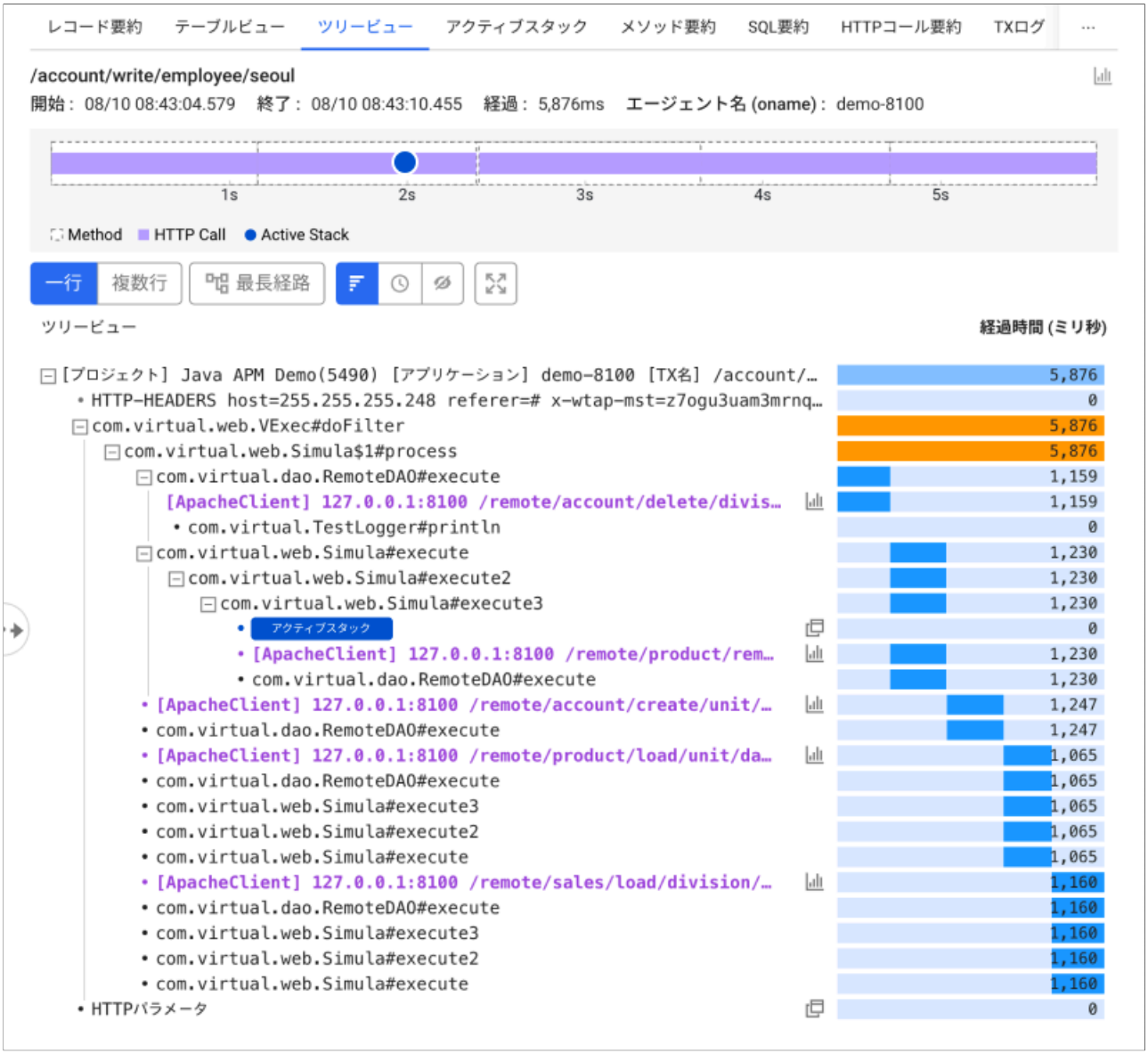
分類	プロパティ	説明
リソース (Resource)	CPU使用時間	トランザクションの処理に使用されたCPU時間です。
	メモリ割当量	そのトランザクションで使用されたメモリ量です。
コンテナ	Pod名	該当トランザクションを処理したKubernetes Podの名前です。
	コンテナID	Pod内で実行されている特定のコンテナの固有識別子です。
クライアント	クライアントIPアドレス	クライアントのIPアドレスです。
	OS	ブラウザが実行されるOS環境です。
	ドメイン	クライアントが接続したIPアドレスに指定されたインターネットアドレスです。
	国/地域	クライアントの国の情報です。
	都市	クライアントの国内の都市情報です。
	クライアントタイプ	クライアントが利用したブラウザの種類です。
	クライアント名	クライアントが利用したデバイス名です。
	WClientID	クライアントの固有識別子です。
Method	methodCount	トランザクションで呼び出されたメソッドの数です。
	methodTime	メソッドの実行にかかった時間です。

- アプリケーションの種類や設定、ステップの種類によって収集する情報は異なる場合があります。 **コンテナ**項目は、アプリケーションがコンテナ環境で実行されている場合にのみ表示されます。
- エラー関連項目は、そのトランザクションでエラーが発生した場合に表示されます。

- トランザクション属性の一部は、統計機能を提供します。 ボタンを選択すると、時系列チャートを通じて照会時間中に発生したトランザクション状況を把握できる画面が表示されます。 チャート内の棒グラフを選択すると、**分析** > **TX検索**メニューに移動して詳細照会できます。
- WhaTapは、クライアントに関する情報をデフォルトで保存します。 ユーザーデータの収集に関するエージェント設定の詳細については、次の文書を参照してください。

ツリービュー

トランザクションの実行過程をツリー形式で確認するには、**ツリービュー**タブを選択します。 各トランザクションとそれに属するトレースの詳細、トレースの開始時刻と所要時間、呼び出し関係を確認できます。 ダイアグラムの各区間を選択すると、そのステップが位置するツリービューに移動します。



- **一行**：各区間別の実行情報に表示されたテキストを一行で表示し、ツリー形式をシンプルに整理できます。
- **複数行**：各区間別の実行情報に表示されたテキストを改行してすべて表示します。
- **最長経路**：最長の経路で移動できます。
- **タイムバーの表示**：経過時間を棒チャートで表示します。
- **時間を表示する**：各区間ごとのタイムスタンプ、ギャップ、経過時間をテキスト形式で表示します。
  - 8秒以上：超過遅延状態で赤色で表現します。

- 3秒以上8秒未満：遅延状態でオレンジで表現します。
- 3秒未満：正常な状態で青色で表示します。

- **時間を表示しない**：時間情報を非表示にします。
- ：トランザクションの要約情報とダイアグラムを非表示にし、ツリービューのみを確認できます。 ボタンをクリックすると、隠された情報を再表示します。 ツリービュー一覧が長い場合は、この機能を利用すると便利です。
- ：SQL変数とHTTPクエリを照会できる画面が表示されます。 詳細については、次の文書を参照してください。

> 開始および所要時間の場合、トランザクション呼び出し環境によって生じる時差を上位トランザクションのトレースやマッピングで修正および表示することで、実際に収集された時間データとの差異が生じる可能性があります。

アクティブスタック



アクティブスタックタブを選択すると、該当トランザクションから収集したアクティブスタックの一覧を確認できます。

- 左側の**アクティブスタック**一覧から収集時点（No、時間）を選択すると、右側に該当時点のスタック情報を表示し

ます。

- **スタック**：アクティブスタックの詳細情報をTXTファイルでダウンロードできます。

> - トランザクション一覧でAアイコンが表示された項目からアクティブスタックを確認できます。
> - アクティブスタックの詳細については、次の文書を参照してください。

AIアクティブスタック分析

Beta

アクティブスタック分析ボタンをクリックすると、収集したアクティブスタックをAIが自動的に分析します。進行中のトランザクションのスタックダンプを基にパフォーマンスの問題を特定し、改善方法を案内します。

分析結果は次の4つのセクションで構成されます。

- **全体要約**：分析したスタック数と全体の実行フローを要約します。
- **共通パターン**：繰り返し現れる実行パターンを表示します。パターン名、発生回数、説明を含みます。
- **発見されたイシュー**：特定したパフォーマンスの問題を深刻度（**High**、**Medium**、**Low**）とともに表示し、判断の根拠となるスタック情報も提供します。

深刻度	基準
High	デッドロック、完全ブロッキング、同一ブロッキング4回以上繰り返し
Medium	I/O待機、2〜3回のロック競合、遅い外部呼び出し
Low	単一発生の待機、軽微な最適化の機会

- **推奨事項**：優先順位に基づいて具体的な改善方法を案内します。

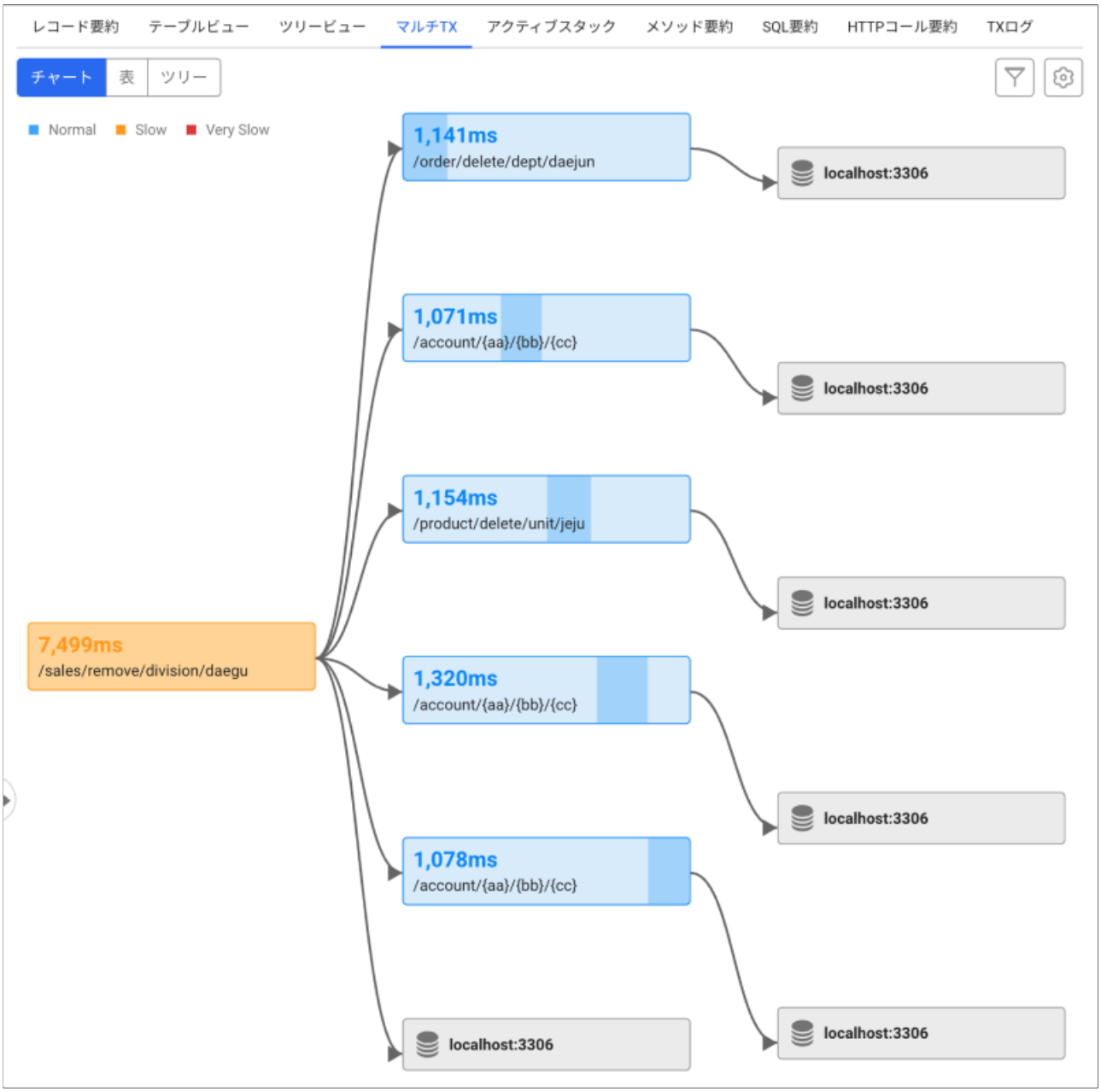
各分析項目には収集したスタック番号（ No.N ）が表示されます。スタック番号をクリックすると、該当スタックの原本を確認できます。

> - AIが生成した分析結果のため、実際の状況と異なる場合があります。参考資料としてご活用ください。
> - プロジェクト**メンバー**権限以上が必要です。

マルチトランザクション

マルチトランザクションとは、他のエージェントやプロジェクトに関連付けられたトランザクションを意味します。**マルチTX**タブでは、WhaTapモニタリングサービスに登録したアプリケーション間の呼び出し関係を確認できます。

> マルチトランザクションを追跡するには、**管理** > **エージェント設定**メニューで `mtrace_enabled` オプションを `true` に設定します。 エージェント設定の詳細については、次の文書を参照してください。



- **チャート**：各トランザクションの呼び出し関係をフローチャート形式で提供します。 トランザクションノードを選択すると、そのトランザクションノードのトレース分析情報を確認できます。 **チャート**モードではマウスを利用して位置を移動したり、スクロールして拡大または縮小できます。
  - ⚙**チャートビュー設定**：チャートに表示するアイテムを表示または非表示します。
- **表**：テーブル形式で、マルチトランザクション内の各トランザクションごとの情報を確認できます。 カラム選択

アイコンを選択して、テーブルヘッダーカラムを編集できます。 各トランザクション項目を選択すると、トレース分析情報を確認できます。

- **ツリー**：ツリー形式でトランザクション間の呼び出し関係を把握できます。 関連した付加機能は、**ツリービュー**タブの機能と同じです。
- **プロジェクト選択**：チャートに表示されるプロジェクトを選択または解除できます。

> ⓘ
> - トランザクション一覧からⓂアイコンがある項目で確認できます。
> - マルチトランザクションの詳細については、次の文書を参照してください。

メソッド要約



No	ク...	メソ...	パラメータ/戻...	数	合計 (ms)	最大 (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

メソッド情報のみを確認するには、**メソッド要約**タブを選択しますて。 エージェントに追跡が設定されたメソッド名と所要時間を表示します。 不要なロジックが繰り返し実行される場合、低い件数に比べて実行時間が長くかかる場合を把握し、メソッドロジック改善のための分析情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。

- **メソッド**: メソッドに関する詳細情報をTXTファイルにダウンロードできます。

> メソッド(method)と関連するエージェント設定オプションの詳細については、次の文書を参照してください。 ご利用中の製品によって、メソッド追跡サポートの可否は異なる場合があります。

SQL要約



No	データベ...	SQL	数	合計 (ms)	最大 (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文についての情報を確認するには、**SQL要約**タブを選択します。 不要に繰り返し実行される場合、低い件数に比べて実行時間が長くかかる場合を把握し、クエリ性能改善のための分析情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **SQL**: SQLに関する詳細情報をTXTファイルにダウンロードできます。

HTTPコール要約



No	リモートサー...	URL	数	合計 (ms)	最大 (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呼び出しのリクエスト件数、合計時間、平均時間などを確認するには、**HTTPコール要約**タブを選択します。 不要な外部への呼び出しが繰り返して実行される場合や、低い件数だが比較的に実行時間が長くかかる場合などを把握するなど、トランザクションの遅延要因が外部か内部かを分析する情報として活用できます。

- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **HTTP**: HTTP Callに関する詳細情報をTXTファイルにダウンロードできます。

TXログ

トランザクションに関するログ情報を確認するには、**TXログ**タブを選択します。



- : **TX統計**情報を確認できます。 チャートのグラフを選択すると**分析** > **TX検索**メニューに移動します。 選択した時間帯のトランザクション一覧を確認できます。
- **ログ検索機**: ログ検索に移動して、目的のログデータを簡単に照会できます。
- **キーワード検索**：検索バーに検索するテキストを入力し、エンターキーを入力するか、ボタンを選択します。 入力したテキストと一致するキーワードをハイライトで表示します。
- **キーワードハイライト**：キーワードと色を設定すると、自動的にログ一覧でキーワードと一致するテキストをハイライトします。 詳細については、次の文書を参照してください。
- **カラム設定** : ログリストテーブルに表示されるカラムの追加や順序変更ができます。 詳細については、次の文書を参照してください。
- **ログ表示詳細設定**: ログリストに表示される内容を設定できます。 詳細については次の文書を参照してください。

> - **TXログ**タブを有効にするには、ログに関連するエージェント設定オプションを適用する必要があります。 ログ設定の詳細については、次の文書を参照してください。
> - **TXログ**タブは**ログ読取り**権限を持つメンバーのみアクセスできます。 メンバー権限の詳細については、次の文書を参照してください。

キーワードハイライトの設定

1. ハイライトで表示するキーワードを追加するにはボタンを選択します。



2. ご希望のキーワードを入力します。
3. 色アイコンを選択し、ご希望の色を選択します。
4. エンターを入力します。 追加したキーワードが一覧に作成されます。

キーワードハイライト削除

ハイライト一覧から利用しない項目を削除するには、—ボタンを選択します。

テーブルカラムの表示または非表示

1. テーブルの右上の⊞ボタンを選択します。
2. **カラム設定**ウィンドウが表示されたら、左側のリストで目的の項目を選択・解除します。
   右のリストから×ボタンを選択してカラムを非表示にすることもできます。
3. すべての設定を完了した後**確認**ボタンを選択します。 設定した内容がテーブルに適用されます。

テーブルカラムの表示順番の変更

1. テーブルの右上の⊞ボタンを選択します。
2. **カラム設定**ウィンドウが表示されたら、右側のリストで目的の項目をドラッグして順序を変更します。
3. すべての設定を完了した後**確認**ボタンを選択します。 設定した内容がテーブルに適用されます。

テーブルコンテンツの表示設定

1. テーブル右上の⚙ボタンを選択します。
2. **ログ表示詳細設定**ウィンドウが表示されたら、目的のオプションを設定します。
   - **content**または**タグ**オプションを選択して、ログにコンテンツの表示有無を選択します。
   - **タグ管理**: タグは、追加した順にログに表示されます。 タグの順序はドラッグで変更できます。 非活性化されているタグはログに表示されません。
3. すべての設定を完了して×ボタンを選択します。 設定した内容がテーブルに適用されます。

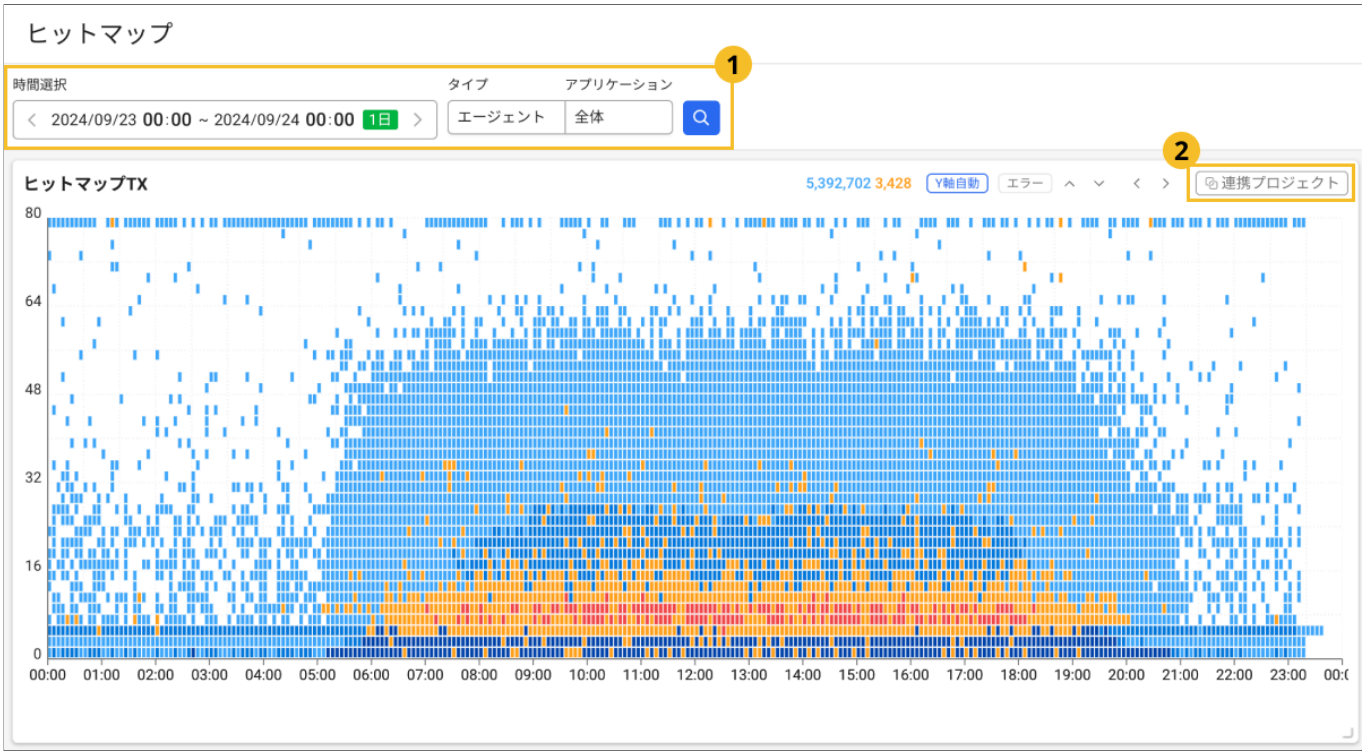
> ⓘ **ログ表示詳細設定**の詳細については次の文書を参照してください。

追加機能

連携プロジェクトメトリクスの確認

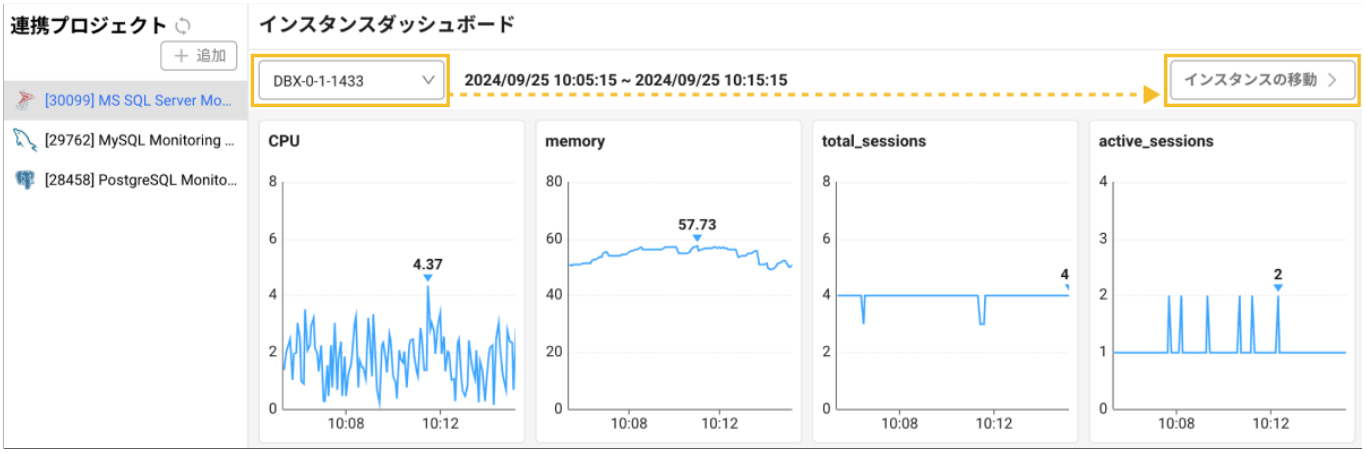
データベースモニタリング(DPM) プロジェクトとアプリケーションのモニタリング(APM) プロジェクトを生成した場合、2つのプロジェクトを連携してDPMで収集したモニタリングデータをAPMプロジェクトで確認できます。 **ヒットマップTX**セクションの右上にある**連携プロジェクト**機能で、トランザクションの照会中に連携したプロジェクトの性能メトリクスを一緒に確認できます。

連携プロジェクト機能は、APMプロジェクトでトランザクションを分析する際に、遅いSQLがある場合、類似な複数のDBを使用する環境から、正確に問題が発生したDBを把握しにくい状況に役立ちます。



1. トランザクションを照会する時間と**アプリケーション**を設定した後 ボタンを選択します。
2. **ヒットマップTX**セクションで右上にある**連携プロジェクト**ボタンを選択します。
3. 連携プロジェクトのメトリクスを確認できる新しい画面が表示されます。

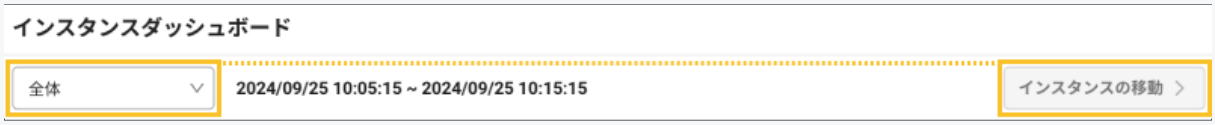
左の一覧で連携プロジェクトを選択し、**インスタンスダッシュボード**セクションのメトリクスを確認します。



選択したプロジェクトのダッシュボードメニューに移動するには、インスタンスを選択し、右上の**インスタンスの移動**

ボタンをクリックします。 **ダッシュボード** > **インスタンスモニタリング**メニューに移動します。

- **ヒットマップTX**セクションの**連携プロジェクト**ボタンは、連携されたプロジェクトがある場合にのみ表示されます。 連携プロジェクトの追加についての詳細は、次の文書を参照してください。
- 左の**連携プロジェクト**一覧の上にある**追加**ボタンを選択するとプロジェクトを追加できます。
- **インスタンスの移動**ボタンは、**全体**を選択した状態では無効になります。



- データベース商品の種類によって**分析** > **カウント推移**メニューに移動できます。

HTTPパラメータ照会

テーブルビュー タブでそのトランザクションのHTTPパラメータを照会することができます。

1. ページの下にスクロールして、トランザクションの実行の最後のステップに移動してください。
2. **HTTPパラメータ**項目を選択します。
3. **HTTP-PARAMETERS**画面が表示されると、**パスワード** ボタンを選択します。
4. 設定したParam Keyを入力します。隠されたパラメータを確認できます。

- HTTPパラメータに関するエージェント設定は、次の文書を参考にしてください。
- **パスワード**：復号化されたパラメータの値を確認できます。 パスワードは `WHATAP_HOME /security.conf (paramkey.txt)`ファイル内の6桁の文字列です。 別の文字列に変更できます。 SQL変数の照会、HTTPクエリの照会、Threadの中止に必要です。 詳細については、次の文書を参照してください。

SQLパラメータの照会

テーブルビュータブでSQLステップを選択するか、**ツリービュー**タブで  ボタンを選択します。 パラメータを照会できる**SQL**画面が表示されます。



- **パスワード**：復号化されたパラメータの値を確認できます。 パスワードは* `WHATAP_HOME` /security.conf*ファイル内の6桁の文字列(`paramkey`)です。 別の文字列に変更できます。

  > **パスワード**は、SQL変数の照会、HTTPクエリの照会、Threadの中止に必要です。 詳細については、次の文書を参照してください。

- **SQLフォーマット**：SQL文をインデントまたはフォーマット化して可読性を高めることができます。
- ：SQL文をクリップボードにコピーできます。
- **統計**: **統計** > **SQL**メニューに移動します。

SQL変数とHTTPクエリを照会するには、次のオプションをエージェント設定に追加します。

- SQLパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。
- HTTPパラメータ情報の記録に関するエージェント設定は、次の文書を参照してください。

whatap.conf

```
# SQLパラメータ照会オプション：オプションが適用されるとSQLパラメータを暗号化して収集します。
profile_sql_param_enabled=true

# HTTPパラメータ照会オプション：オプションが適用されると、HTTPクエリパラメータを暗号化して収集します。
profile_http_parameter_enabled=true
```

トレース一覧カラムを設定する

TXトレースセクションのカラム設定により、照会された結果の詳細項目を追加で確認できます。

1. **TXトレース** エリアで **カラム選択** ボタンをクリックします。
2. 表示したいカラム項目を選択または解除します。設定内容はテーブルに即時反映されます。
   - **すべて選択**: すべてのカラム項目を選択します。
   - **すべて解除**: すべてのカラム項目を解除します。
   - **初期化(既定値)**: カラム項目をデフォルトに設定します。
3. 設定が完了したら、右上の × アイコンをクリックします。

トレース一覧をフィルタリングする

TXトレースセクションで照会された結果を次のオプションでフィルタリングできます。



- **すべて表示**: 照会された結果からすべてのトランザクションが確認できます。
- **アクティブスタック**: 照会された結果から、アクティブスタックがあるトランザクションのみを確認できます。
- **マルチTX**: 照会された結果から、マルチトランザクションのトランザクションのみを確認できます。
- **Search URL**: 入力した文字列と一致する項目を含むトランザクションのみが表示されます。 文字列を入力した後、キーボードの `Enter` キーを押します。
  - **トランザクション**
  - **クライアントIPアドレス**
  - **ドメイン**
  - **エラークラス**
  - **エラーメッセージ**

照会一覧をダウンロードする

TXトレースセクションで照会された結果をCSV形式のファイルにダウンロードできます。 ダウンロードしたCSVファイ

ルの名前は、profile {project_code} YYYYMMDD .csv形式です。

1. トランザクションを照会した後に CSV ボタンを選択します。
2. **最大CSV行数**オプションに希望する値を入力します。
3. **ダウンロード**ボタンを選択します。

アクティブTX

進行中のトランザクションをアクティブTXと呼びます。**アクティブTXで定期的にダンプされるスタックはアクティブスタック**と呼ばれます。

> WhaTapエージェントは、10秒(オプションで調整可能)ごとにアクティブTXに対して、アクティブスタックをダンプし、サーバーに送信します。 `active_stack_second=10`

コンパクトなアクティブスタックの収集

アクティブスタックはスレッドダンプを定期的に実行するため、誤った実装をするとエージェントのオーバーヘッドが増加する可能性があります。WhaTapはエージェント負荷を最小限に抑えながらアクティブスタックを収集するために、様々なオプションがあります。

> **インスタンス性能管理** > **スレッドリスト/ダンプ**メニューのスレッドリストで**WhaTap-ActiveStackDump**スレッドの**CPU時間**を確認し、オーバーヘッドを判断できます。

アクティブスタックの例

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(ServletInvocableHandlerMethod.java:114)
org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerMethod(RequestMappingHandlerAdapter.java:827)
```

最適化されたデータ収集

- トランザクションを実行しているスレッドに対して、スタックをダンプします。
- アクティブスタックダンプの時間間隔を調整できます。

  whatap.conf

  ```
  active_stack_second=10
  ```

- アクティブスタックの最大行に制限されます。トップラインから基本の50行を収集します。

  whatap.conf

  ```
  trace_active_callstack_depth=50
  ```

- アクティブスタックの各行をハッシュ処理して収集します。テキストは１度だけ収集します。
- 一度に収集される最大のアクティブスタックの数も制限されます。

  whatap.conf

  ```
  active_stack_count=100
  ```

Background Threadのアクティブスタック

基本的に**アクティブスタック**は、トランザクションが実行されているスレッドのスタックを意味します。ただし、一部のバックグラウンドスレッドのスタックも分析が必要な場合があります。この場合、オプションを通じてバックグラウンドスレッドに対する**アクティブスタック**が取得できます。この機能はJavaエージェント1.6.2バージョンからサポートします。

- `async_stack_enabled` の値を `true` に設定すると活性化されます。

  whatap.conf

  ```
  async_stack_enabled=false
  ```

- スタックダンプの間隔は、フォアグラウンドアクティブスタックの設定に準じます。

  whatap.conf

  ```
  active_stack_second=10
  ```

- 対象スレッド名を指定する場合は、 `*` を使用して文字列パターンを指定します。

  whatap.conf

  ```
  async_thread_match=http*,abc*
  ```

  スレッド名でスタックダンプの対象を識別します。コンマ(,)を区切り文字として使用してmatchを複数指定できます。

- スタックの上位メソッドが `async_thread_parking` に登録されたクラスまたはメソッドの場合、スレッドがパーキング状態にあると判断してダンプを生成しません。

  whatap.conf

  ```
  async_thread_parking_class=sun.misc.Unsafe
  async_thread_parking_method=park
  ```

参考資料

- アクティブTX
- トラブルを最も早く特定するアクティブTX

マルチトレース

マルチトランザクションは、別のエージェントやプロジェクトに関連付けられたトランザクションを意味します。WhaTapプロジェクトに登録されたアプリケーションサービス間の呼び出しを追跡するのが、**マルチトランザクション追跡**です。

> Javaエージェントは、3つのHTTPヘッダーキー値(`x-wtap-po` 、 `x-wtap-mst` 、 `x-wtap-sp1`)でマルチトランザクションを追跡します。ゲートウェイを通過するHTTPトランザクションの連携追跡ができない場合は、HTTPヘッダー条件を確認してください。

> **マルチトランザクションの活性化**
>
> マルチトランザクションを追跡するには**管理** > **エージェント設定**メニューから `mtrace_enabled` オプションを `true` に設定します。エージェント設定の詳細については次の文書を参照してください。

マルチトランザクション追跡機能の利用方法

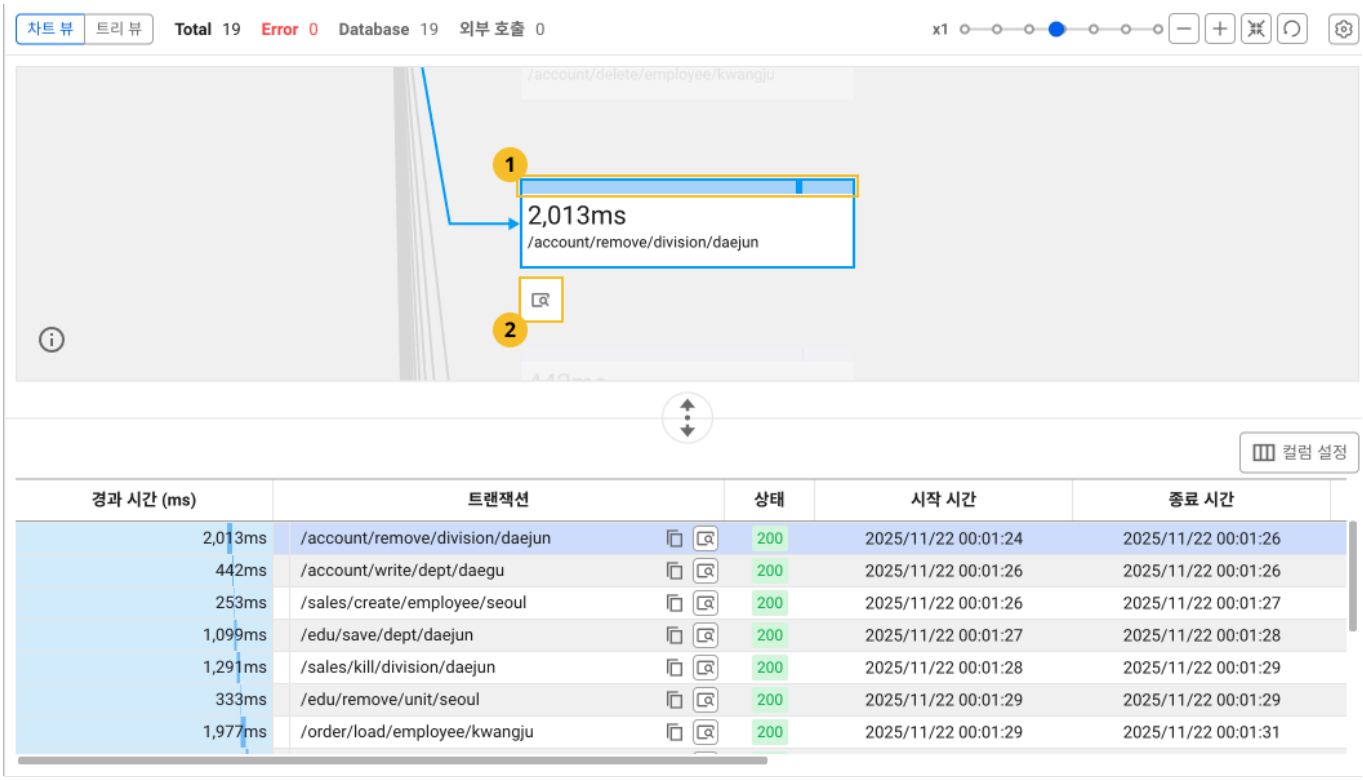
1. **分析** > **マルチTX追跡** メニューに移動します。
2. **トランザクション情報** 画面で確認した **MTID** の値を、**MTID/CUSTID照会** 項目に入力します。
3. 照会する日付とプロジェクトを選択します。
4. 画面下部の **適用** ボタンをクリックします。
    - 以下の場合、**適用** ボタンは無効化されます：
        - 前回と同じフィルター値（MTID/CUSTID、日付、選択プロジェクト）が設定されている場合
        - フィルター値（MTID/CUSTID、日付、選択プロジェクト）が一つも入力されていない場合
5. 右側の **チャート** タブには、各トランザクション間の呼び出し関係を把握できるダイアグラムとテーブルが表示されます。

    > ダイアグラムエリアとテーブルエリアは上下に分割されており、比率を自由に調整できます。

- 上向き矢印をクリックするとテーブルが全体領域を占めます。
  ダイアグラムが全体表示だった場合は 50/50 にリセットされます。
- 下向き矢印をクリックするとチャートエリアが全体領域を占めます。
  テーブルが全体表示だった場合は 50/50 にリセットされます。

チャート

チャート は、各トランザクション間の呼び出し関係を迅速かつ明確に確認できる機能です。
同じ **マルチTXID** を持つトランザクションサービスの個別実行時間を確認できます。
各トランザクションノード上部に表示される処理時間（1 **タイムバー**）を通じて、トランザクション間の呼び出し関係を把握できます。



チャートはマウスで自由に移動・拡大・縮小できます。

- チャート領域をドラッグすると表示位置を移動できます。
- スクロール操作:
  - 縦スクロール：上下移動
  - 横スクロール：左右移動
- `Ctrl` + 縦スクロールで拡大・縮小できます。
  （タッチパッドのピンチ操作にも対応）

トランザクションを選択すると、関連するノードのみが強調表示され、関連しないノードは薄く表示されます。
選択したトランザクションの下部に **詳細表示** ボタン（2）が有効化され、クリックすると **トランザクション情報** ウィンドウが開きます。
ここでトランザクションの詳細情報を確認できます。
トランザクション情報 を利用した詳細分析については、こちらのドキュメント を参照してください。

- **拡大**: ズームイン
- **縮小**: ズームアウト
- **画面に合わせる**: 全ノードを画面に収めて表示
- **ズーム初期化**: 表示位置とズーム倍率を初期状態に戻す
- **チャートビュー設定**: 表示項目のオン/オフ
  - **詳細**：アプリ名、プロジェクト情報、IP、ステータスコードの表示/非表示
  - **データベース** / **外部呼出し**：DB接続や外部HTTP呼び出しをノードとして表示

表

表 タブでは、マルチトランザクションに含まれる各トランザクション情報をテーブル形式で確認できます。

トランザクションを選択すると、**チャート** 図内の対応する **ノード** が強調表示され、関連ノードもハイライトされます。
トランザクション列の詳細ボタンをクリックすると **トランザクション情報** ウィンドウが開き、トレースの詳細情報を確認できます。
詳しくは こちらのドキュメント を参照してください。

- **カラム設定** — テーブル列の編集

ツリー

ツリー ボタンを選択すると、各トランザクションとその配下のトレースに関する詳細情報を確認できます。
全体トランザクションの実行時間内で、各サブトランザクションおよびトレースの開始時間・経過時間を可視化し、呼び出し関係をツリー形式で表示します。

- **プロジェクト選択**：チャートに表示するプロジェクトを選択または解除します。
- **一行**：各区間の実行情報を1行で表示し、ツリーを簡潔にします。
- **複数行**：各区間の実行情報を複数行で表示します（改行あり）。
- **最長経路**：最も長いパスへ移動します。
- **タイムバーの表示**：経過時間を棒グラフとして表示します。
- **時間を表示する**：各セグメントのタイムスタンプ、ギャップ、経過時間をテキスト形式で表示します。
  - 8秒以上：赤色で 過剰遅延 として表示
  - 3〜8秒：オレンジ色で 遅延 として表示
  - 3秒未満：青色で 正常 として表示

> 開始時間および経過時間は、上位トランザクションとのマッピングによる補正処理が行われるため、実際に収集されたデータと差異が生じる場合があります。

- **時間を表示しない**：時間情報を非表示にします。

> - 各トレースの または を選択すると、**HTTPコール統計**、**アクティブスタック**、**SQL** などの詳細情報を確認できます。
> - SQL ステップの をクリックすると、そのステップの SQL 文をコピーできます。

エンドポイント設定

トランザクションエンドポイントは、トランザクションの開始メソッドです。HTTPトランザクションの場合は、`HttpServlet.service()` または `Filter.doFilter()` がトランザクションの開始点であり、**トランザクションエンドポイント**と呼びます。

NON HTTP追跡

トランザクションエンドポイントとして指定されたメソッドが開始されて終了するまでのパフォーマンスをトランザクションパフォーマンスと呼びます。Non HTTPトランザクションを追跡するには、**エンドポイント**を指定する必要があります。

> **エンドポイント検索**
>
> 1. トランザクションが呼び出されると推定されるすべてのメソッドのトレース追跡を設定してください。
> 2. 再起動後、トランザクションを発生させてモニタリングしてください。
> 3. `back stack` オプションをオンにして、入力メソッドを正しく理解してください。

対象選択

1. **メソッドトレース設定**

   メソッドトレースを設定します。トランザクションで呼び出されるクラスを指定してください。DBを使用するプログラムでは、**JDBCドライバー**が便利です。

   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. **トランザクション開始オプションの設定**

   メソッドが呼び出されたら、トランザクションを開始します。トランザクション開始オプションとトランザクション開始時にスタックをダンプするオプションをオンにします。

   whatap.conf

```
trace_auto_transaction_enabled=true
trace_auto_transaction_backstack_enabled=true
```

3. **再起動後のトレース分析**

   再起動してください。あとからサービスを呼び出すと、トランザクションが追跡されていることがわかります。

   トランザクションを検索すると、`jdbc.*` で始まるすべてのクラスのメソッドがこのトランザクションで表示されることがわかります。トランザクショントレースを検索すると、**TRANSACTION BACKSTACK**というメッセージステップが表示されます。

   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. **スタックの内容を確認**

   スタックの内容を確認することで、どのメソッドから開始するかを推定できます。

   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)
```

   上記の3つのメソッドのいずれかをトランザクションの開始点として判断できます。この状況では、逆コンパイルを実行して適切な**トランザクションエンドポイント**を決定する必要があります。

ロジックを見ると、`SimulaNonHtp.run`内で`while()`が動作しながら`SimulaNonHtp.process()`を呼び出し、`SimulaNonHtp.execute()`が実行されます。`process()`が適当であることがわかります。この部分はソースを見て判断する必要があります。

> **エンドポイント**の最も**重要な基準は終了機能です。**通常の状態では、遅延することなく直ぐに終了する必要があり、パフォーマンスの判断ができます。

トランザクションのエンドポイント指定

1. トランザクションの開始点を次のように設定します。

   whatap.conf

   ```
   hook_service_patterns=com.virtual.web.SimulaNonHttp.process
   ```

2. アプリケーションを再起動します。

`process()`メソッドは、新しいトランザクションの**エンドポイント**になります。

> `hook_service_patterns`オプションではワイルドカード文字を使用できません。2つ以上の値を設定するにはコンマ(,)を区切り文字として利用します。

トランザクション名の定義

一般的に、メソッド名でトランザクションを区別できます。

whatap.conf

```
service_name_mode=[full,class,method,string,arg]
service_name_index=0
```

- **service_name_mode**

`full` , `class` , `method` , `string` , `arg` の5つのオプションを指定できます。

- `full` : Full Class名をサービス名として使用します。
- `class` : Class名をサービス名として使用します。
- `method` ：Method名をサービス名として使用します。
- `string` ：文字列の最初のパラメータをサービス名として使用します。
- `arg` ：パラメータで `service_name_index` オプションに指定されたインデックスにパラメータをサービス名として使用します。

プラグイン使用

> プラグインは、十分に理解している場合にのみ使用してください。

_WHATAP_HOME_経路の下位にpluginフォルダを作成し、pluginへ移動します。 `vi` コマンドでAppServiceStart.xファイルを作成します。

プラグインの使用例

`println("test");` と入力して保存すると、画面に、"test"という文字列が出力されます。確認後、パラメータから情報を抽出してください。

次の例では `HashMap` この引数で渡され、この `HashMap` には'url'というキーが含まれています。

```
Object url =((java.util.HashMap)$point.getArgs()[0]).get("url");
$ctx.service((String)url);
//println("url="+url);
```

このようにプラグインを作成すると、トランザクション名が変更できます。

スタック収集

アプリケーション・モニタリング・エージェントは、線形収集方式と環状収集方式を提供します。

線形収集

個々のトランザクショントレースは、無期限にステップを収集することができません。トランザクションごとに長さが制限されているバッファーにトレースステップを保存します。各トランザクションは、最大で収集できるステップの数がオプションとして指定されます。

- **profile_step_max_count**

  既定値 `1024`

  最大で収集可能なステップの数です。

- **profile_step_normal_count**

  既定値 `800`

  通常、何の制限もなく収集されるステップの数です。

- **profile_step_heavy_count**

  既定値 `1000`

  normal countを超えた場合、ステップは応答時間が遅いステップとアクティブスタックのステップのみを収集します。

- **profile_step_heavy_time**

  既定値 `100`

  heavy countで収集されるステップの標準時間は `profile_step_heavy_time` です。

  step_buffer

  収集されるトレースのステップ数がheavy countを超える場合は、アクティブスタックのみが収集されます。 この場合でも、最大ステップ数は `profile_step_max_count` を超えません。

環状収集

線状収集は、トレースの最初の部分を収集し、バッファサイズを超えた部分の内容を破棄します。一方、環状収集は、最初のステップを破棄する方式です。

```
circular_profile_enabled=true
```

circular_profile

バッファサイズを `profile_step_max_count` に設定します。

```
profile_step_max_count=1024
```

アプリケーショントポロジー

アプリケーショントポロジー は、トランザクションのフローに関連するデータベースおよび外部サービス呼び出しを視覚的に表示します。
ユーザーリクエストからデータベース呼び出し、外部サービス連携までの全経路を一目で把握でき、性能ボトルネックを簡単に発見できます。

- ユーザーリクエスト → DB/外部サービスまでの経路を可視化
- URL パターンによるフィルタリング検索
- トランザクション性能状態（遅延/エラーなど）の確認
- ボトルネック特定および詳細情報の確認

トポロジー表示

時間範囲とアプリケーションを選択した後、以下のオプションで検索範囲を設定します。
検索するたびに新しいタブで表示され、タブにマウスを乗せると検索条件のツールチップが表示されます。
ツールチップ内のポップアウトボタンをクリックすると、該当条件のトポロジーが別ウィンドウで開きます。

トランザクション URL 検索

特定の URL パターンに一致するトランザクションを呼び出したドメインのみを表示します。
ワイルドカード（ `*` ）が使用できます。

使用例

- `/api/*` : `/api/` で始まるすべての URL
- `*/users/*` : `users` を含むすべての URL
- `/api/v1/users` : 完全一致

> **スラッシュ(`/`) とワイルドカード(`*`) の組み合わせに注意してください。**
>
> - `/api/*` : `/api` は一致しない（例: 🔵 `/api/users` , ❌ `/api` ）
> - `/api*` : `/api` 自体も一致（例: 🔵 `/api` , `/api/users` , `/apitest` ）

パターン制限

- **必ずワイルドカード以外の文字が 1 文字以上必要**
  - ❌ 不可: `***`, `**`
  - 🔵 可: `a*`, `*a`, `*a*`, `/api*`
- 空文字列や null は一致しない
- `**` は `*` と同じ扱い
- `*` を 3 個以上含む複雑なパターンは完全にはサポートされない
- 正規表現は非対応
- 大文字・小文字は区別される
- `*` 文字そのものをエスケープしてマッチさせることはできない

トレース件数制限

取得する最大トランザクショントレース件数です。

- デフォルト: 10,000 件
- 最大: 1,000,000 件

> Limit を上げても、フィルタ条件に一致するすべてのトランザクションが取得されるとは限りません。
> 上限に到達すると警告が表示されます。

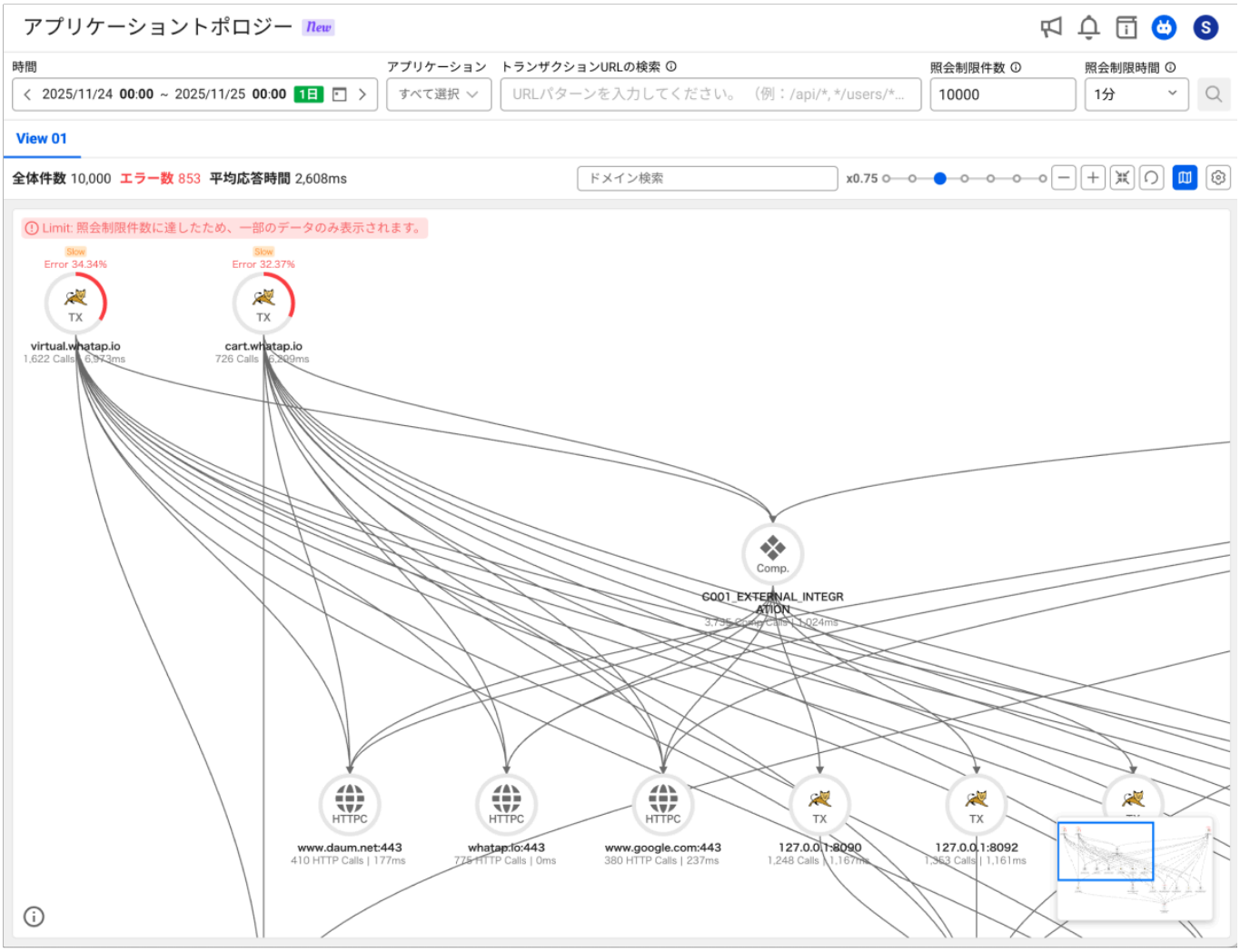
時間制限

時間制限に達すると、その時点までのデータのみ表示されます。

- 範囲: 30 秒 〜 5 分
- デフォルト: 1 分

画面構成と可視化

画面には円形の **ノード** と、それらを結ぶ **接続線** によってトランザクションフローが表示されます。



ノード構造

ノードはトランザクションの実行フローに基づき、上から下へ配置されます。

ノードタイプ

User → TX → DB/HTTPC の順に配置。

ノードタイプ	説明
User	トランザクションを呼び出したユーザー数
TX (Transaction)	アプリケーションの HTTP トランザクション
DB (Da	

統計

ホーム画面 > プロジェクト選択 > **統計**

WhaTapモニタリングサービスのメイン画面でプロジェクトを選択し**統計**の下位メニューを選択します。WhaTapエージェントは、トランザクション情報やSQLなどの主要なトレース情報について統計情報を収集します。5分ごとに一覧を作成してサーバーに送信します。

基本画面ガイド

時間選択および**フィルター**, **並替え順**, **アプリケーション**項目を設定した後、 ボタンを選択します。



- 各**統計**タブは、**合計**、**最大**、**平均時間**および**件数**による並べ替えオプションを提供し、任意の視点から上位リストを表示できます。
- **カラム設定**ボタンて既定値の他に照会するカラムを追加または除外できます。
- ボタンをタップすると、照会結果とともに照会リストの上に CSV ボタンが表示されます。ボタンを選択すると、照会リストをCSV形式のファイルでダウンロードできます。

> 統計照会の一覧が表示された後に照会条件を変更して、 ボタンを選択せずに CSV ボタンをクリックすると、変更された条件でCSVファイルをダウンロードできます。

- 統計リストから**詳細**アイコンを選択すると、件数と平均時間チャートが確認できるし、エージェント間の件数およびエラー件数の比較、応答時間の比較チャートを追加で提供します。そのチャートで照会する時点を選択すると**TX検索**メニューに移動します。
- 統計リストで T ボタンを選択すると **トップスタックURL**メニューに移動します。

- 統計リストで U ボタンを選択すると**ユニークスタックURL**メニューに移動します。

> ⓘ
> - 最大31日間のデータを照会できます。ただし、**クライアントIP**メニューでは最大7日間のデータを照会できます。

TX統計

ホーム画面 > プロジェクト選択 > **統計** > **トランザクション**

トランザクション統計を収集します。5分ごとに最大5,000個のURLごとの実行統計を収集してサーバーに送信します。異なるURLの数が5分間で5,000個を超えた場合は無視されます。

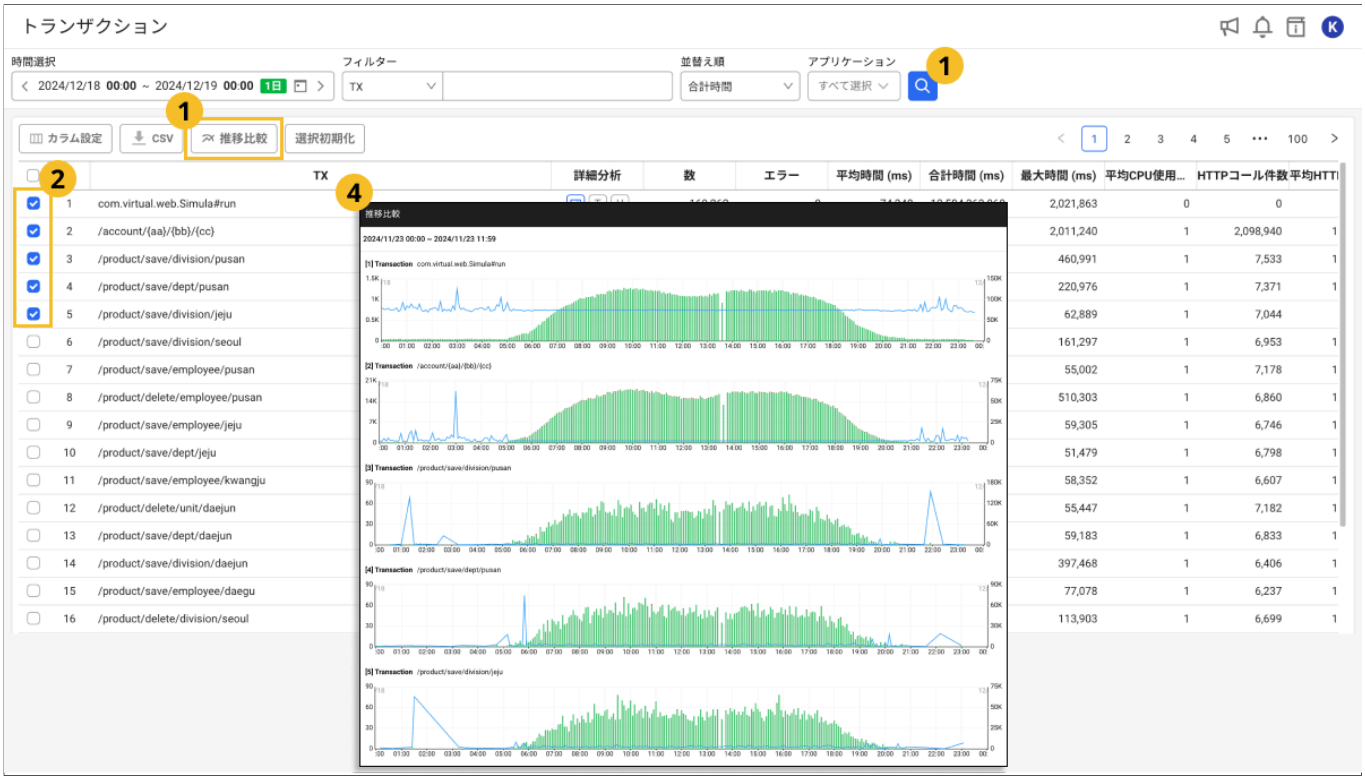
カラム	説明
TX	トランザクションアドレス
数	実行件数
エラー	実行されたトランザクションのエラー発生件数
平均時間 (ms)	応答時間の合計の平均
合計時間 (ms)	応答時間の合計
最大時間 (ms)	最大応答時間
平均CPU使用時間	CPU使用時間の平均
HTTPコール件数	HTTPコール件数
平均HTTP呼出し時間 (ms)	HTTP呼び出しの合計時間/トランザクションの発生件数
HTTP呼出し当りの平均時間 (ms)	HTTP呼び出しの合計時間/トランザクションで発生したHTTP呼び出しの合計件数
SQL件数	SQL実行件数

カラム	説明
平均SQL時間 (ms)	SQLの合計時間/トランザクションの発生件数
SQL当りの平均時間 (ms)	SQLの合計時間/トランザクションで発生したSQLの合計件数
SQLフェッチ	SQLフェッチ件数
平均SQLフェッチ時間 (ms)	SQLパッチの合計時間/トランザクションの発生件数
SQLパッチ当りの平均時間 (ms)	SQLパッチの合計時間/トランザクションで発生したSQLパッチの合計件数

> ⓘ **平均メモリ割当量**を収集するためには、エージェント設定(whatap.conf)に `trace_malloc_enabled=true` を追加する必要があります。該当オプションの既定値は `false` です。

推移比較機能を利用する

推移比較機能を利用して、多数のトランザクションの推移を一目で比較できます。この機能は、確認するトランザクション数が多い場合、主要な問題を迅速に識別し、トラフィックの問題を解決するのに役立ちます。



1. 照会したい時間とモニタリング対象を選択した後 ボタンを選択します。
2. トランザクション一覧から比較したいトランザクションを1つ以上選択します。
3. **推移比較**ボタンを選択します。
4. 選択したトランザクションの推移グラフが一覧表示される**推移比較**ウィンドウが表示されます。

推移比較ウィンドウで推移グラフを確認して、トランザクション間のトラフィックパターンを比較します。 特定の時間帯、サービスパスのトラフィック分布を比較できます。

- 最大100個のトランザクションを選択して照会できます。
- 選択したトランザクションが多い場合、ロード時間が増加する可能性があります。
- 正確な分析のために**フィルター**オプションを適切に活用します。

トランザクションドメイン統計

ホーム画面 > プロジェクト選択 > **統計** > **ドメイン**

WhaTapエージェントは、ドメインごとのトランザクション統計を収集できます。1台のサーバーに複数のドメインを個別にサービスを提供するシステムでは、ドメインごとの分析が必要になる場合があります。次のオプションを設定してください。

whatap.conf

```
stat_domain_enabled=true
```

収集されるデータは、ドメイン別のURLの処理状況を把握するために使用できます。

カラム	説明
ドメイン	サービスドメイン
TX	トランザクションアドレス
数	実行件数
エラー	実行されたトランザクションのエラー発生件数
平均時間(ミリ秒)	応答時間の合計の平均
時間合計(ミリ秒)	応答時間の合計

TX Caller統計

ホーム画面 > プロジェクト選択 > **統計** > **Caller**

マルチサーバーがrestコールに接続されると**Caller**と**Callee**間の関連付けられた統計を収集できます。このデータを収集するには、次のオプションを設定してください。

whatap.conf

```
mtrace_rate=100
mtrace_spec=v1
stat_mtrace_enabled=true
```

msa-system

上記のアーキテクチャでは、**Caller & Callee**の統計情報は、api1、api2のみ表示することができます。ユーザーのブラウザから呼び出されるシステムでは、**Caller**統計を表示できません。

ただし、**Caller**側のサーバーからデータを送信する必要があるため、すべてのサーバーに適切な設定を行う必要があります。

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
```

収集される統計データは次のとおりです。**Callee**側で参照する必要があります。

カラム	説明
Callerプロジェクトコード	Callerに対応するプロジェクトコード
Callerエージェントの種類	Callerに対応するエージェントが属するグループの単位
Caller URL	Callerトランザクションのトランザクションアドレス
TX	Calleeトランザクションアドレス
数	実行件数
エラー	実行されたトランザクションのエラー発生件数

カラム	説明
平均時間(ミリ秒)	応答時間の合計の平均

> ⓘ **Caller**メニューで一度にダウンロードできるCSVライン数は最大10,000行です。

IPトランザクション統計

ホーム画面 > プロジェクト選択 > **統計** > **IP Transaction**

特定のIPアドレスごとにアプリケーションから呼び出したURLと、呼び出し件数を照会できる機能を提供します。この機能は、異常に多くの要求が発生するIPアドレスを確認し、そのIPアドレスから呼び出されたURLを分析して、潜在的なセキュリティ脅威や性能問題を事前に識別するのに役立ちます。ユーザーはこの情報に基づいて、不要または悪意のあるリクエストをブロックできます。

フィルターオプションを通じてIPアドレスまたはトランザクションアドレスをフィルタリングし、条件に合うデータを素早く照会できます。特定のIPやトランザクションアドレスのパターンを分析する時に役立ちます。

IPトランザクション統計を収集するには、次のオプション設定が必要です。既定値は `false` です。

whatap.conf

```
stat_ipurl_enabled=true
```

カラム	説明
IP	リクエストを発生させたクライアントのIPアドレス
TX	呼び出されたトランザクションのアドレス、特定経路の変数は `{}` で表示
数	当該IPアドレスで発生したリクエストの合計件数
エラー	リクエスト中に発生したエラー数
平均時間(ミリ秒)	当該リクエストの平均応答時間(ミリ秒、ms)

- **異常なトラフィック検出**: 異常に呼び出し件数の多いIPを探知し、そのIPで発生したトランザクションアドレスのパターンを分析し、潜在的な攻撃を遮断できます。例えば、特定のIPから `/account/{aa}/{bb}/{cc}` URLに対して過度な要請が発生した場合、そのIPをブロックしたり要請を制限する措置ができます。
- **応答時間分析**: 平均応答時間が異常に高いIPを確認し、ネットワークまたはサーバー側の性能問題を把握して改善できます。
- **エラー発生分析**: 特定のIPでエラーが多数発生した場合、そのIPの要請に問題があるか、またはシステムに特定の問題があるかを診断できます。

トランザクションReferer統計

ホーム画面 > プロジェクト選択 > **統計** > **リファラー**

WhaTapエージェントは、Refererごとにトランザクション統計を収集できます。次のオプションを設定してください。既定値は `false` です。

whatap.conf

```
stat_referer_enabled=true
```

収集されたデータは、Refer情報の統計を使用して確認できます。

カラム	説明
リファラー	Refererアドレス
TX	トランザクションアドレス
数	実行件数
エラー	実行されたトランザクションのエラー発生件数
平均時間(ミリ秒)	応答時間の合計の平均
時間合計(ミリ秒)	応答時間の合計

エラー統計

ホーム画面 > プロジェクト選択 > **統計** > **エラー**

5分間に発生したサービスエラーに関する統計です。異なるエラーとトランザクション名をキーとして発生件数を収集します。エラー発生時点のスタックを提供します。5分あたり最大1,000個の異なるエラーを統計化します。

カラム	説明
クラス	エラークラス
TX	トランザクションアドレス
メッセージ	エラーメッセージ
数	エラー発生件数
エージェント	エージェント名
エージェントノード	`whatap.onode`に分類されたエージェントグループのユニット
エージェント種類	`whatap.okind`に分類されたエージェントグループのユニット
最初作成時刻	照会期間中にトランザクションのエラーが最初に生成された統計時間(5分または1時間)

SQL実行統計

ホーム画面 > プロジェクト選択 > **統計** > **SQL**

5分間のSQL性能統計を収集します。5分間に異なるSQL文が最大5,000まで許可されます。Javaプロセスで制限を超えるSQLを検出すると、そのSQLは統計データから破棄されます。

カラム	説明
データベース	データベース種類
SQL	SQLクエリ文
TX	SQLを呼び出したURLのいずれかを任意に収集した情報
CRUD	Create, Read, Update, Delete
全体件数	クエリ実行件数
エラー件数	実行されたクエリのエラー発生件数
平均時間 (ms)	クエリ実行時間の合計の平均
合計時間 (ms)	クエリ実行時間の合計
標準偏差	クエリ実行時間の標準偏差

> ⓘ TXは、5分間該当SQLを呼び出したURLから1つ(最後の呼び出しURL)を分析のため収集します。

HTTP Call性能統計

ホーム画面 > プロジェクト選択 > **統計** > **リモートHTTP呼出**

HTTPおよびHTTPSプロトコルを使用して、５分間のアプリケーション外部呼び出し統計を収集します。5分間、異なるHttp Call文章は最大5,000まで許可します。１つのアプリケーションプロセスで制限を超えた外部Http Callが発生する場合、統計データは捨てられます。

カラム	説明
HTTP呼び出しURL	HTTP呼び出しURLハッシュ(hash)
TX	HTTP APIをリクエストしたURLのいずれかを任意に収集した情報

カラム	説明
ホスト	HostまたはIPアドレス
ポート	TCPポート
合計	すべての実行件数
エラー	エラー発生件数
平均時間(ミリ秒)	応答時間の合計の平均
時間合計(ミリ秒)	応答時間の合計
最小時間(ミリ秒)	最小応答時間
最大時間(ミリ秒)	最大応答時間
標準偏差(ミリ秒)	HTTPコール時間の標準偏差

クライアントIPの地域別呼び出し件数

ホーム画面 > プロジェクト選択 > **統計** > **クライアントIP**

サーバーとクライアントの物理的距離は、データ送信時間に比例します。クライアントIPのリージョン別の呼び出し件数を統計的に収集します。5分当たりに収集できる異なるIP数は、インスタンスあたり最大70,000個です。

カラム	説明
クライアントIPアドレス	クライアントのIPアドレス
国/地域	ユーザーの接続リージョン
都市	ユーザー接続都市
数	全体呼び出し件数

- クライアントIPに関するデータは、最大7日間のデータを照会できます。
- WhaTapは、クライアントに関する情報を保存します。ユーザーデータの収集に関するエージェント設定の詳細については、次の文書を参考にしてください。

リージョン別の稼働率チャート

稼働率チャートは、ユーザーの接続場所を国と都市別で収集され表示します。

- 国のエリアを選択すると、その国の都市と接続されているユーザーのIPなどを確認できる詳細なチャートに移動します。
- 中心円を選択すると、既存の国のエリア選択チャートに戻ります。

ユーザエージェント別の呼び出し件数

ホーム画面 > プロジェクト選択 > **統計** > **ユーザーエージェント**

ユーザーエージェント文字列のHashごとに呼び出し件数を収集します。5分ごとに収集できる異なるUser Agent Hashはインスタンス当たり最大500個です。

カラム	説明
ユーザーエージェント	ブラウザのユーザーエージェント情報
OS	ブラウザが実行されるOS環境
WEBブラウサ	ブラウザの種類
数	収集された全体件数

クライアントブラウザの統計

ホーム画面 > プロジェクト選択 > **統計** > **クライアントブラウザ**

ユーザーエージェントの文字列を分析して、エンドユーザーが使用するブラウザの種類、OS基準で集計した統計情報を提供します。収集した統計情報に基づいて作成されたパイチャートを通じて占有率を確認できます。

カラム	説明
ユーザーエージェント	ブラウザのユーザーエージェント情報
OS	ブラウザが実行されるOS環境
WEBブラウサ	ブラウザの種類
数	収集された全体件数

Javaレポート

ホーム画面 > プロジェクト選択 > **レポート**

レポートは、個々のプロジェクトのサービスの使用状況と障害の発生履歴を報告する文書です。サービスモニタリング担当者は、レポートを通じて関連部署の担当者とステータスを共有します。

モニタリングデータ分析は、サービスの改善方向を定める指標となるため、極めて重要です。ただし、複数のダッシュボードからデータを収集して文書化するには煩雑な場合があります。

WhaTapの**レポート**メニューは、**レポート作成業務の自動化**を支援します。毎週報告書を作成すること、決まった時間に報告書を共有すること、様々な書式を管理することなど、すべて**レポート**メニューからできます。

> **レポート**のダウンロード、印刷やメール送信の予約を希望する場合は、そのボタンをクリックします。
> HTML形式でダウンロードできます。

レポートと統合レポートの違い

- **レポート**は、1つのプロジェクトに関するレポートを作成するメニューです。したがって、プロジェクトを選択した後、サイドメニューの**レポート**メニューから進入できます。
- **統合レポート**は、複数のプロジェクトに関するレポートを作成するメニューです。プロジェクトを選択せずにホーム画面のサイドメニューから進入できます。**統合レポート**の詳細については、次の文書で確認できます。

レポートの種類

基本レポートは、大手企業、公共機関やITサービス企業などで実際に使用しているフォームです。 ご要望のフォームを作成したい場合は、support@whatap.ioまでご連絡ください。 要望されたレポートは、**レポート**一覧に追加されます。

レポートフォーム追加

- リクエストフォームに一般的な場合、**レポート**一覧に追加して提供します。

- リクエスト時にHTMLおよびCSSフォーマットに付属されている場合は、簡単に追加できます。

リクエストフォームの例

レポートフォームの追加をリクエストする場合は、例に示すように、目的のHTMLおよびCSSフォームを添付することをお勧めします。詳細については、次のサンプルファイルを参照してください。

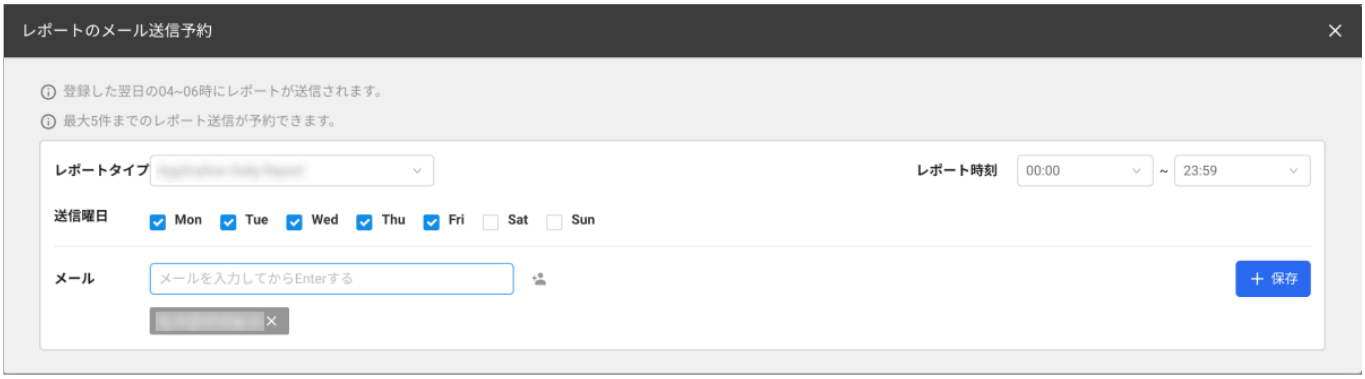
- サンプルファイル：ダウンロード後ZIP形式を解凍してHTML形式のサンプルファイルを確認してください。ファイル内のコメントにカスタマイズ用のCSS領域とHTML領域のガイダンスが記述されています。

制約事項

- WhaTapモニタリングで現在提供しているレポートテンプレートの各項目を追加または修正、削除が必要な場合に提供できます。
- 現在提供しているレポートテンプレートにないデータや分析結果を追加するには、開発者によるレビューが必要です。
- データの追加および加工は、WhaTapモニタリングで収集および保存されたデータ(メトリクス)に限定します。他のツールや外部データは追加できません。
- 特定の顧客企業のニーズを満たす汎用的でないレポートフォームは提供できません。
- 詳細については、support@whatap.ioまたは営業担当者にお問い合わせください。

メール送信予約

定期的にレポートをメールで受け取るには**レポートのメール送信予約**を選択します。出勤直後に行っていた様々なサービス点検手順をメール確認に置き換えることができます。


レポートのメール送信予約
登録した翌日の04~06時にレポートが送信されます。
最大5件までのレポート送信が予約できます。
レポートタイプ
レポート時刻
00:00
~
23:59
送信曜日
Mon
Tue
Wed
Thu
Fri
Sat
Sun
メール
メールを入力してからEnterする
＋ 保存

APMレポート

ホーム画面 > プロジェクト選択 > **レポート**

画面の上に**レポート**で ボタンをクリックすると、日刊、週刊、月刊に分類された各種分析レポートを確認できます。希望するレポート様式を選択し、照会したい日付と時間、フィルターオプションを設定してください。 ボタンを選択すると、選択したレポートを確認できます。

> 週次および月次レポートを作成する際は、データの照会範囲によって多少時間がかかる場合があります。

日次レポート

一日間のアプリケーション性能の変化が確認できるレポートを提供します。 日次レポートで提供するレポートのフォームは次の通りです。

- **日次レポート：アプリケーション**
- **日次レポート：アプリケーションピーク性能比較**
- **日次レポート：アプリケーションユーザー統計**
- **日次レポート：メトリクス推移(平均/最大)**
- **日次レポート：アプリケーションユーザー定義**

日次レポート：アプリケーション

一日中のアプリケーション性能に関する主要な指標を総合的に分析できるレポートです。

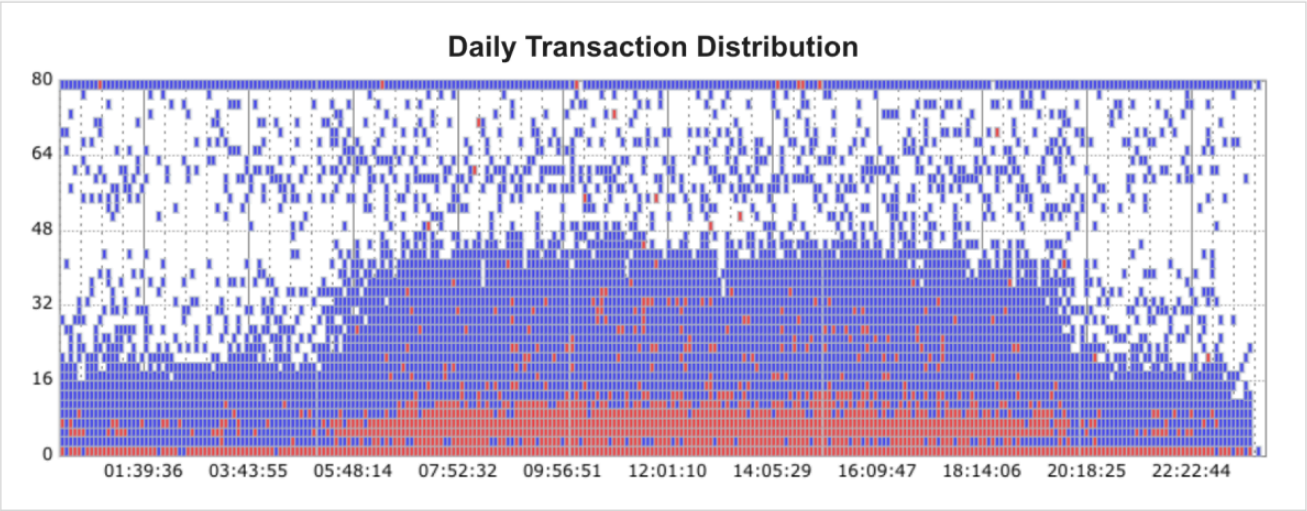
> 主要用語や指標の詳細は、次の文書を参照してください。

- **概要**: アプリケーションの主な性能指標を確認できます。
- **ピークタイム性能要約**: 一日の処理量(TPS)が最も高かった時間とその時間の指標値を確認できます。
- **指標別平均/最大値**: 一日の主要指標別の平均値と発生時間基準の最大値を確認できます。

- **性能推移チャート**: システムの主要指標を視覚的に表現した時系列チャートです。 時間の流れによって主要指標の数値を確認できます。 システムの安定性と性能を評価する上で重要な指標です。

  > ◦ **週次レポート：アプリケーション**は日別変化量を追加提供します。
  > ◦ チャートの右上にあるCSVボタンを選択すると、データをCSV形式のファイルでダウンロードできます。

- **日別TX分布図**: 1日の時間帯別のトランザクション分布とエラー発生状況を一目で把握できます。 グラフのx軸は各トランザクションの終了時間、y軸は応答時間(秒)を意味します。 正常トランザクションの場合は青で表示され、エラーが発生した場合は赤で表示されます。



- **アプリケーション別の要約**: 監視対象エージェントごとのステータスを確認できます。 **アクティブカウント数**は、進行中のトランザクション数です。
- **TX TOP 10**: **TX数**、**エラー件数**、**平均応答時間**基準で上位10件のトランザクションを確認できます。
- **Topスタック TOP 10**: Stack Trace上のステップに基づいて、ステップ間の呼び出し頻度を割合で計算した情報をまとめて確認できます。

  > **分析** > **スタック** > **トップスタック**メニューでスタックの詳細を照会できます。

- **平均SQL応答時間 TOP 10**: 使用されたSQLクエリ文に対する平均レスポンスタイムを基準にソートした統計情報です。

- **増加%**: 照会前日までの3日間の平均データと照会当日のデータを比較して増加した割合を計算した数値です。

  例） 4月25日00時00分 ~ 23時59分 データ検索 → 4月22日00時00分 ~ 4月24日23時59分 データと比較
  数値が10%以上の場合赤で表示され、5%~10%の間の場合オレンジで表示されます。 5%未満の増加率は表示しません。

- **統計** > **SQL**メニューで各クエリの発生時間と性能を詳細に照会できます。

日次レポート：アプリケーションピーク性能比較

モニタリング対象エージェントごとにアプリケーションの最大性能指標を日刊(赤)、前日(青), そして先週(緑) のデータと比較して提供します。 各指標は、特定の時間帯の最大値とそれに対応する時間の指標値を一緒に表示します。 これにより、各アプリケーションの性能の変動を一目で把握できます。

主要用語や指標の詳細は、次の文書を参照してください。

日次レポート：アプリケーションユーザー統計

10秒、5分、30分単位の同時接続ユーザーと1時間の訪問者数を時系列チャートで提供するレポートです。 このレポートは、使用パターンを理解し、ユーザートラフィックを効果的に管理するための重要なデータとして活用できます。

- **10s 同時接続ユーザー数**: リアルタイムのトラフィックの変化を細かくモニタリングすることで、ユーザー数が急増する時間帯を確認できます。 また、ユーザーが集中する時間帯にコンテンツやサービスを提供する戦略を策定できます。
- **5m / 30m 同時接続ユーザー数**: 短期・中期的なユーザーパターンを分析して、サーバーリソースとネットワーク帯域幅を効率的に配分できます。 ピーク時間帯を予測して、その時間帯により多くのリソースを割り当てる計画を策定できます。
- **1h ユーザー数**: 時間ごとにユーザーの訪問パターンを分析し、マーケティング戦略および運営計画に活用できます。

日次レポート：メトリクス推移(平均/最大)

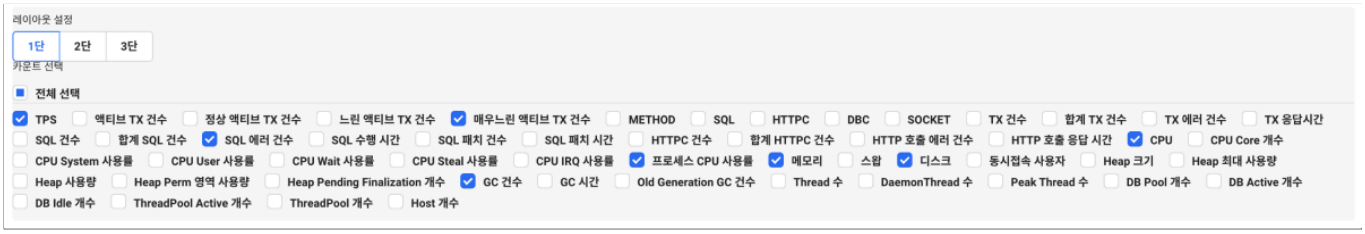
アプリケーションの様々な性能指標を時系列チャートで提供するレポートです。 各指標の平均および最大値を視覚的に表示します。 これにより、システムの性能推移をモニタリングし、性能の最適化と問題解決に必要なデータとして活用できます。

チャートのタイトルは、**Category[Field]**形式で構成されています。 アプリケーションの性能指標の詳細は、次の文書を参照してください。

- **Avg.**: 青線で表示され、各時間帯の平均値です。
- **Max.**: 赤線で表示され、各時間帯の最大値です。

日次レポート：アプリケーションユーザー定義

ユーザーが希望する指標を選択してレポートを作成できます。 選択した指標に基づいて時系列チャートを提供するレポートを作成できます。



1. 画面の上の**レポート**で**日次レポート：アプリケーションユーザー定義**を選択します。
2. **レイアウト設定**で希望する段数を設定します。
3. 照会する指標を選択します。 全体を選択するには、**すべて選択**をクリックします。
4. ボタンを選択します。

レポートが作成されます。

週次レポート：アプリケーション

アプリケーションの1週間の主要指標を総合的に分析できるレポートです。

> ⓘ 主要用語や指標の詳細は、次の文書を参照してください。

- **概要**: 1週間の総訪問者とトランザクション数を提供します。
- **稼働率**: アプリケーションが与えられた時間の間に動作した時間をパーセントで表したものです。 たとえば、稼働率が100%であれば、そのアプリケーションが照会期間中に一度も中断されずに動作し続けたことを意味します。 一方、稼働率が低いと、そのアプリケーションが照会期間に中断されたり、異常に動作した時間があったことを示します。
- **性能推移チャート**: システムの主要指標を視覚的に表現した時系列チャートです。 時間の流れによって主要指標の数値を確認できます。 システムの安定性と性能を評価する上で重要な指標です。

  > ⓘ
  > - 日ごとの変化量と日ごとの平均および最大値を提供します。
  > - チャートの右上にあるCSVボタンを選択すると、データをCSV形式のファイルでダウンロードできます。

- **TX TOP 10 比較**: **TX数**、**エラー件数**、**平均応答時間**基準で上位10件のトランザクションを確認できます。 各トランザクションの1週間前と照会週の実行件数、増減の件を提供します。
- **Topスタック TOP 10 比較**: スタックトレース上位10個をに対し、1週間前と照会週間のパーセントと件数を比較したリストです。 最も呼び出されるスタック項目を識別できます。

  > ⓘ **分析** > **スタック** > **トップスタック**メニューでスタックの詳細を照会できます。

- **平均SQL応答時間 TOP 10 比較** : SQL平均レスポンスタイムを基準に上位10個のSQL文を比較したリストです。 各SQL文に対して、1週間前と照会注意の実行件数、平均レスポンスタイム、増減時間を提供します。
- **プロジェクトエラー現況(エラーメッセージを含む)**: 一週間のプロジェクトのエラー統計をエラークラスとエラーメッセージ基準で提供する統計情報です。

月次レポート：アプリケーション

一か月間のアプリケーション性能に関する主要な指標を総合的に分析できるレポートです。

> ⓘ 主要用語や指標の詳細は、次の文書を参照してください。

- **概要**: アプリケーションの主な性能指標を確認できます。
- **アプリケーション別の要約**: エージェント別**IPアドレス**、**OS情報**、**稼働率**を確認できます。
- **性能推移チャート**: システムの主要指標を視覚的に表現した時系列チャートです。 時間の流れによって主要指標の数値を確認できます。 システムの安定性と性能を評価する上で重要な指標です。

  > ⓘ チャートの右上にあるCSVボタンを選択すると、データをCSV形式のファイルでダウンロードできます。

- **TX TOP 10 比較**: **TX数**、**エラー件数**、**平均応答時間**基準で上位10件のトランザクションを確認できます。 各トランザクションの1か月前と照会当月の実行件数、増減件を提供します。
- **Topスタック TOP 10 比較**: スタックトレース上位10個に対し、1か月前と照会当月のパーセントと件数を比較したリストです。 最も呼び出されるスタック項目を識別できます。

  > ⓘ **分析** > **スタック** > **トップスタック**メニューでスタックの詳細を照会できます。

- **平均SQL応答時間 TOP 10 比較**: SQL平均レスポンスタイムを基準に上位10個のSQL文を比較したリストです。 各SQL文に対して、1か月前と照会当月の実行件数、平均レスポンスタイム、増減時間を提供します。

主要用語及び指標案内

各レポートに表示された主要指標に対する説明です。

項目	説明
日間ユーザー数(DAU)	日別アクティブユーザー数で、1日アプリケーションを使用したユーザー数です。

項目	説明
(**DAU**, Daily Active User) / **日間ユーザー数**	
週間ユーザー数	1週間のアプリケーションを使用した接続者の合計です。
月間ユーザー数 (**MAU**, Monthly Active User)	1か月間アプリケーションを1回以上使用したユニークユーザーの合計です。
同時接続ユーザー数	重複を削除した同時接続ユーザーの数です。
ユーザー数 / リアルタイムユーザー数	特定の時間帯にアプリケーションに接続したユーザーの数です。
TX件数	リクエストされたトランザクションの合計です。
日間TX数 / 日別TX数	1日で処理されたトランザクションの合計です。
週間TX件数	1週間リクエストされたトランザクションの数です。
日別TX分布図	1日の時間帯別トランザクション分布とエラー発生状況を一目で把握できる分布図チャートです。
エラー比率(%)	全体のトランザクション数のエラー率です。
イベント件数	メトリクスイベントのうちCriticalレベルのイベントの数です。
エラー件数	トランザクションの処理中に発生したエラーが原因で失敗したトランザクションの数です。
アプリケーション数	照会期間にモニタリングしたアプリケーションの数(エージェントの数)です。
全体コア	照会期間に使用されたCPUコアの数です。
APDEX*100 (Application Performance Index)	使用期間中のアプリケーションに対する顧客パフォーマンスの満足度です。

項目	説明
TPS (**TPS**, Transaction Per Second)	1秒間に処理されたトランザクション数です。
平均応答時間(ミリ秒) / 応答時間	トランザクションへの平均レスポンスタイムです。
CPUコア	使用されたCPUコアの数です。
時間毎ユーザー数	照会期間の1時間あたりの訪問者数です。
サーバーCPU使用率	サーバーのCPU使用率です。
サーバーメモリ使用率	サーバーのメモリ使用率です。
アクティブカウント数 / アクティブTX	進行中のトランザクション数です。
CPU	CPU使用率です。
ヒープ(最小値)	最小のヒープメモリ使用量です。
稼働率	アプリケーションが与えられた時間の間に正常的に動作した時間をパーセントで表したものです。

- **Java CPU使用率**: JavaアプリケーションのCPU使用率です。
- **Javaヒープメモリ使用量**: JVMのヒープメモリ領域で割り当てられたメモリ量です。

統合レポート

ホーム画面 > **統合レポート**

ユーザーは、複数のプロジェクトを統合したレポートを作成および共有できます。レポートの種類と含めるプロジェクトを選択してレポートを生成することができ、生成一覧で作業の進行状況を確認できます。特定のプロジェクトでエラーが発生すると、エラーの内容を確認できます。作成が完了したレポートは、すぐに照会するか、ダウンロードおよび印刷機能で共有できます。



> **統合レポート**メニューは**Application**、**Server**、**Kubernetes**、**Database**商品のプロジェクトに関するレポートのみ作成できます。

主な機能の案内

統合レポートメニューで提供する機能は次のとおりです。

- **マルチプロジェクトサポート**

  複数のプロジェクトを選択して一度にレポートを生成できるので便利です。プロジェクトで利用できる**レポート**メニューは、そのプロジェクトに関するレポートのみ作成できます。

- **タスクごとのレポート作成**

  レポート生成機能はタスクごとにプロジェクトを選択し、テンプレートとして保存できます。通常タスクごとにプロジェクトを分割して使用するマイクロサービスアーキテクチャ(Micro Service Architecture、MSA)の環境で有効な機能です。

- **大容量データの最適化**

  レポートの生成時間が大幅に短縮されます。また、一部のプロジェクトレポートの生成中にエラーが発生した場合でも、レポートの結果を表示できます。

- **レポート作成中に同時作業が可能**

  レポートを作成すると同時に、他の業務を進めます。プロジェクトで利用できる**レポート**メニューは、レポート作成タスクを完了するまで待つ必要があります。

  統合レポートメニューからレポートの生成を開始した後、別のページに移動できます。また、レポートの結果一覧から進捗状況と生成されたレポートを照会できます。

- **レポート作業の共有**

  同じプロジェクト権限を持つユーザーが作成したレポートを表示できます。関連するレポートがすでに作成されているかを確認できるため、重複の作成を回避できます。

レポートと統合レポートの違い

プロジェクトで利用できる**レポート**メニューでは、そのプロジェクトに関するレポートを作成できます。初期画面でプロジェクトを選択し、サイドメニューの**レポート**または**統計/レポート**メニューでアクセスできます。

統合レポートメニューでは、複数のプロジェクトに関するレポートを作成できます。プロジェクトを選択することなく、初期画面でアクセスできます。

レポート作成

レポートの種類を選択

レポートタイプ一覧で希望するレポートタイプを選択すると**レポート生成**ボタンが有効になります。レポート名の前の**アイコン**は、レポート作成に対応するプロジェクトプラットフォームを意味します。

- **Application**プロジェクトプラットフォームを示します。
- **Database**プロジェクトプラットフォームを示します。
- **Kubernetes**プロジェクトのプラットフォームを示します。
- **Server**プロジェクトプラットフォームを示します。



詳細設定を行う

レポート生成画面が表示されたら、各項目を設定します。

- **題名**: 同じ種類のレポートが複数ある場合、名前を指定するとレポートの結果を区別するときに役に立ちます。
- **時間**: レポートで使用するデータの期間を選択します。期間はレポ

ートの種類(日、週、月)によって異なります。

- **プロジェクト**：レポートを作成するプロジェクトを選択します。グループおよび個別のプロジェクト単位で選択できます。**検索**フィールドを使用して、目的のプロジェクトを検索できます。
- **レポートフォーマット保存**：現在入力したレポートの設定をテンプレートとして保存できます。今後、新しい報告書を作成する際**タイプ**一覧で読み込めます。

設定を完了したら、**レポート生成**ボタンを選択します。



選択したレポートの様式に従って照会期間を設定する方法は次のとおりです。

- **日刊報告書**: 開始と終了の日付、時間を選択できます。
- **週間報告書**: 開始日のみを選択すると、終了日を計算して自動的に選択します。

- **月間報告書**: 開始日のみを選択すると、終了日を計算して自動的に選択します。
  - 開始日が1日であれば、その月の月末までに自動的に選択されます。
  - 開始日が1日でなければ、来月の前日までに自動選択されます。

例1、開始日が**1月1日**であれば、**1月1日00時00分00秒**から**1月31日23時59分59秒**を自動選択します。

例2、開始日が**1月10日**であれば、**1月10日00時00分00秒**から**2月9日23時59分59秒**を自動選択します。

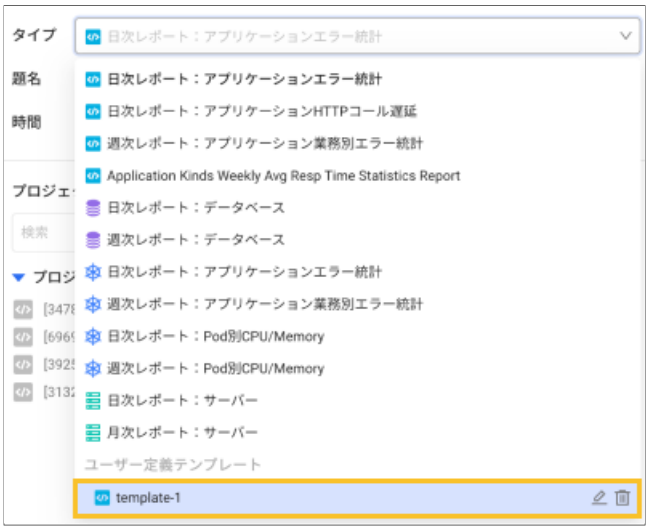
- **時間**オプションの利用方法の詳細については、次の文書を参照してください。
- 保存したレポート様式を読み込み方法の詳細は次の文書を参照してください。

レポートフォームの読み込み

新しいレポートを作成するとき**レポートフォーマット保存**機能を使用して保存した様式は、その後のレポート作成時に**タイプ**項目で確認できます。

業務単位でプロジェクトを頻繁に使用するユーザーは、アクティブに活用できます。

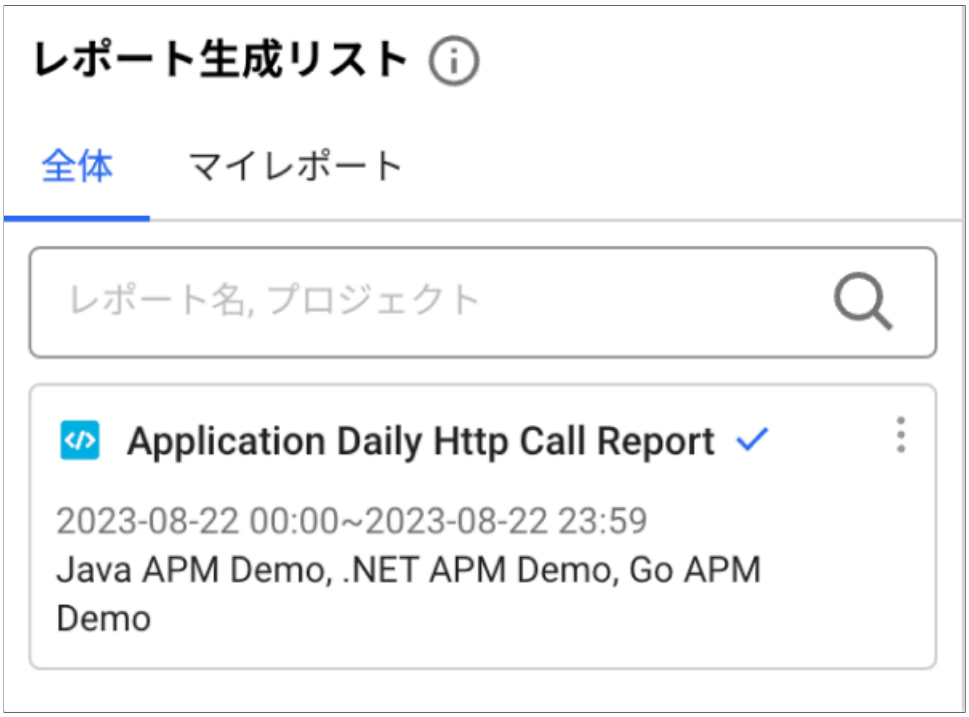
- ：保存したレポートフォームを修正できます。
- ：保存したレポートフォームを削除できます。



レポート一覧を確認する

レポート生成リストで以前に作成したレポートと現在作業中のレポートを確認できます。

- **全体**タブでは、ユーザー本人が作成したレポートの他にも、同じ権限を持つユーザーが作成したレポートを一緒に確認できます。
- **マイレポート**タブでは、ユーザー本人が作成したレポートのみ確認できます。
- レポート項目の右上に ⋮ ボタンを選択すると、そのレポートをコピーまたは削除できる機能を利用できます。



- エラーなく作成されたレポートは✓アイコンで表示され、エラーが含まれている場合は⚠アイコンで表示されます。⚠アイコンを選択すると、どのプロジェクトでエラーが発生したかを確認できます。
- 作成したレポートは最大100個まで7日間保管できます。

表示と共有

- **画面照会**

  レポート生成リスト一覧で作成したレポートを選択すると、レポートの詳細を確認できます。

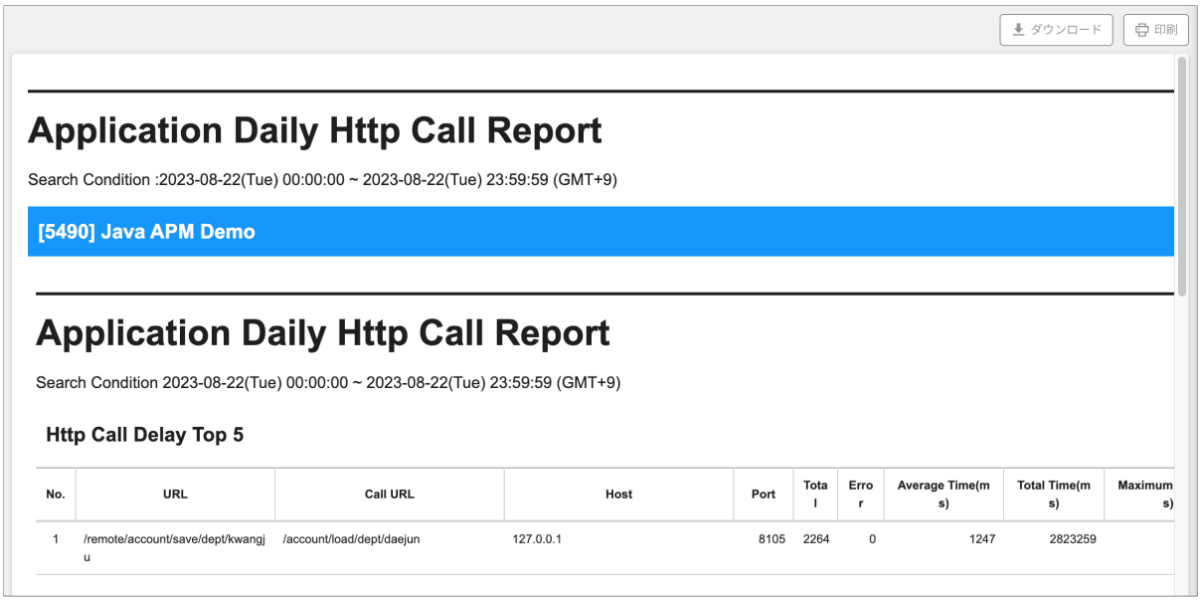
- **ダウンロード**

  現在照会中のレポートをhtml形式のファイルでダウンロードするには**ダウンロード**ボタンを選択します。モニタリングの管理

権限のない人にレポートを共有できます。

- **印刷およびPDF保存**

  レポートをプリンターで印刷するか、PDF形式のファイルとして保存するには**印刷**ボタンを選択します。



Javaアラート概要

監視対象のシステムまたはアプリケーションで問題が発生した際に、あらかじめ設定した条件に基づいてユーザーへ通知を送信する機能です。通知は、メール、SMS、メッセンジャー、アプリのプッシュ通知など、さまざまなチャネルを通じて配信されます。

たとえば、CPU 使用率、メモリ使用量、ディスク I/O、ネットワークトラフィック、トランザクション数、エラー発生、レスポンスタイムの遅延などの主要なパフォーマンス指標にアラート基準を設定しておくと、その基準を超過した場合や異常な兆候が検出された際に通知が送信されます。

WhaTap の AI ベースのアラート機能を使用すると、複雑な条件を手動で設定する必要はありません。AI がアプリケーションの正常な動作パターンを学習し、異常パターンを自動的に検知して通知します。また、タグカウントアラート機能を活用することで、特定のタグを基準に集計された値に条件を設定でき、複雑な状況でも高精度な検知が可能です。

主な機能

- **イベント設定**: 閾値ベースのアラート条件および通知チャネルの設定
- **通知設定**: ユーザーごとの通知チャネルおよび時間帯設定
- **イベント履歴**: アラート発生履歴の確認

権限

イベントを追加または設定するには、**アラート設定権限** が必要です。

- **閲覧**: メトリクスイベント設定を確認するには、**閲覧・分析権限** が必要です。
- **追加／編集／削除**: 設定を変更するには、**アラート設定権限** が必要です。
- **通知受信**: 通知を受信するには、**通知受信権限** が必要です。

> **参考資料**
>
> 詳細については、メンバー権限 を参照してください。

イベント設定

ホーム画面 > プロジェクト選択 > **アラート通知** > **イベント設定**

イベント設定 では、監視対象のシステムやアプリケーションで発生する可能性のある異常兆候を基準に、イベントを設定・管理できます。設定した条件が満たされると、メール、SMS、メッセンジャー、アプリのプッシュ通知など、希望するチャネルで通知を受信できます。監視の目的に応じて、イベントを柔軟に構成できます。

メトリクスイベント

Javaメトリクスに基づくイベント条件を設定しアラートを発生させる方法を案内します。

複合メトリクスイベント

複数のメトリクスを組み合わせてイベント条件を設定する複合メトリクスイベント機能を案内します。

ヒットマップパターン

ヒットマップに基づくパターン分析で異常状況を検知するイベント設定を案内します。

異常検知イベント

Javaメトリクスの異常パターンを自動検知しアラートを発生させるイベント設定を案内します。

リアルタイムログ

Javaログの内容に基づいてアラートを発生させるログイベントを設定します。

複合ログ

Javaログの内容を基準に警告通知を発生させる複合ログイベントを設定します。

トランザクションイベント

Javaトランザクション条件に基づいてアラートを発生させるイベント設定を案内します。

ホーム画面 > プロジェクト選択 > **アラート通知** > **イベント設定**

イベント設定 では、監視対象のシステムやアプリケーションで発生する可能性のある異常兆候を基準に、イベントを設定・管理する機能を提供します。設定した条件を満たすと、メール、SMS、メッセンジャー、アプリプッシュなど、希望するチャネルで通知を受信できます。監視目的に応じてイベント構成を設計できます。

メトリクスイベント

Javaメトリクスに基づくイベント条件を設定しアラートを発生させる方法を案内します。

複合メトリクスイベント

複数のメトリクスを組み合わせてイベント条件を設定する複合メトリクスイベント機能を案内します。

ヒットマップパターン

ヒットマップに基づくパターン分析で異常状況を検知するイベント設定を案内します。

異常検知イベント

Javaメトリクスの異常パターンを自動検知しアラートを発生させるイベント設定を案内します。

リアルタイムログ

Javaログの内容に基づいてアラートを発生させるログイベントを設定します。

複合ログ

Javaログの内容を基準に警告通知を発生させる複合ログイベントを設定します。

トランザクションイベント

Javaトランザクション条件に基づいてアラートを発生させるイベント設定を案内します。

参考

- イベント設定時は、不要なイベント発生を防ぐために適切なしきい値を設定してください。
- 通知設定では、問題解消時に通知される回復通知もあわせて検討すると、運用対応に役立ちます。

共通機能

すべてのイベント設定は **JSON または Excel 形式** でダウンロードできます。同じ種類のプロダクト設定を共有することで、繰り返し発生するイベント設定作業を削減できます。

- **JSON 一括編集**: すべてのイベント条件を JSON 形式で一括編集できます。
- **JSON 一括ダウンロード**: 現在登録されているイベント条件を JSON ファイルとして保存できます。
- **Excel 一括ダウンロード**: イベント一覧を Excel ファイルとしてダウンロードし、外部レポートやバックアップに活用できます。

JSON 一括編集

1. 画面上部の [JSON 一括編集] ボタンをクリックします。
2. **JSON 一括ダウンロード** 機能でダウンロードした JSON ファイルを読み込むか、直接編集します。
3. [上書き] ボタンをクリックして保存します。
4. 確認メッセージが表示されたら、[上書き] ボタンをクリックします。

上書き後は、以前のイベント設定に戻すことはできません。

- `productType` または `platform` プロパティの値を任意に変更すると、他のプロジェクトでイベントが正常に動作しなくなる可能性があります。
- JSON 編集画面には検証機能が組み込まれています。形式や値が正しくない場合、編集画面下部にエラーメッセージが表示され、[上書き] ボタンは無効になります。



JSON データ構造

```
{
 "productType": "{productType}",
 "platform": "{platform}",
 "events": {
  "HITMAP": {
   ...
  },
  "LOG_REALTIME": [
   {...}
  ],
  "METRICS": [
   {...}
  ],
  "COMPOSITE_LOG": [
   {...}
  ],
  "BASIC": [
   {...}
  ],
  "COMPOSITE_METRICS": [
   {...}
  ],
  "ANOMALY": [
   {...}
  ]
 }
}
```

JSON フィールド	説明
productType	プロダクトタイプ（例: APM、Infra など）
platform	プラットフォーム情報（例: java、node など）
events	イベントデータ
ㄴ HITMAP	ヒットマップパターン

JSON フィールド	説明
ㄴ LOG_REALTIME	リアルタイムログイベント
ㄴ METRICS	メトリクス
ㄴ COMPOSITE_LOG	複合ログイベント
ㄴ BASIC	基本イベント
ㄴ COMPOSITE_METRICS	複合メトリクス
ㄴ ANOMALY	異常値

JSON 一括ダウンロード

画面上部の [JSON 一括ダウンロード] ボタンをクリックします。

- JSON ファイル名は integrated-event-`YYYYMMDD`.json 形式です。
- JSON データの構造や `events` 属性配下のサブ属性は、プロダクト（ `platform` ）またはプロダクトタイプ（ `productType` ）によって異なる場合があります。

JSON データ内の `productType` または `platform` プロパティの値を任意に変更すると、他のプロジェクトでイベントが正常に動作しなくなる可能性があります。

Excel 一括ダウンロード

画面上部の [Excel 一括ダウンロード] ボタンをクリックします。

メトリクスイベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **メトリクス** タブ

プロジェクトに設定されているすべてのメトリクスベースのイベント一覧を確認できます。設定済みのメトリクスイベントを確認、修正、削除したり、新しいイベントを追加したりできます。

基本オプション

- **イベントレベル**: イベントの深刻度を示し、正常（Info）、警告（Warning）、危険（Critical）の3段階に分類されます。
- **エージェント重複アラート抑止**: 同じ名前のエージェントが同時に2つ以上動作している場合、選択した時間（30秒、1分、3分、5分、10分）の間はイベントが発生しません。
- **検索**: イベント名を基準に検索します。
- **JSON ダウンロード／アップロード**: JSON エディタでイベントルールを直接編集するか、ファイルとしてダウンロードして編集後に再アップロードできます。
- **+ イベント追加**: 新しいメトリクスイベントを追加します。
- **カラムサイズ調整**: 各カラムの境界をドラッグして幅を調整できます。

表 | メトリクスイベント一覧構成

項目	説明
No.	イベントの番号
	イベントの有効化状態 – 有効: 条件を満たすとイベントが発生 – 無効: イベントは動作せず、条件も評価されません
	イベント設定の編集および削除
イベント名	ユーザーが指定したイベント名 – 最大255文字

項目	説明
ルール	イベント発生条件
対象	イベントが適用される監視対象
発生回数	イベント発生の基準となる期間と回数 – 選択した時間内に設定回数分条件を満たすとイベントが発生
一時停止	アラート発生後、同一イベントの発生を一定時間停止 – 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 – 回復通知が有効な場合、RECOVERED（正常）通知後、選択した時間は同一イベントが発生しても通知されません
回復通知	Critical および Warning レベルのイベントが解消された際に、RECOVERED（正常）通知を受信するかどうか – 有効: 進行中から正常状態に遷移すると RECOVERED 通知を送信し、イベント履歴で進行中／解消状態を区別して確認可能 – 無効: しきい値を超えるたびに単発イベントが発生
イベント受信	該当イベント通知を受信するユーザーまたはグループ（受信タグ）

イベント設定

メトリクスイベントは、監視対象で発生する事象を条件に基づいて定義し、効率的に通知を受信できるよう設定するための機能です。

イベント追加

メトリクスイベントを追加するには、**イベント設定** > **メトリクス** タブに移動し、右上の [+ イベント追加] ボタンをクリックします。

メトリクスイベント追加手順
Step 1. テンプレート選択: イベントの基本構成を選択します。
Step 2. イベント条件定義: どの条件でイベントを発生させるかを設定します。
Step 3. イベント対象選択: イベントが適用される監視対象を選択します。
Step 4. 基本情報および通知設定: イベント名、メッセージ、受信対象を設定します。

テンプレート選択

1. イベントの構成方法に応じて、画面左側の 3 つのテンプレートのいずれかを選択します。
   - **新規作成**: カテゴリやメトリクスを一から自由に構成できます。複雑な条件や詳細な監視ルールが必要な上級ユーザーに適しています。
   - **クイック設定**: あらかじめ定義されたカテゴリ、メトリクス、基本しきい値が提供され、しきい値のみを調整して素早く適用できるため、初心者に適しています。
   - **高度な設定**: あらかじめ定義されたカテゴリやメトリクスを基に、条件を追加したり自由に変更したりできます。

> ⓘ メトリクスイベントでは、さまざまな設定方法をサポートしています。利用する監視プラットフォームによって、提供されるテンプレートは異なる場合があります。

イベント条件定義

イベント条件定義 のステップでは、イベント発生の判断に使用するメトリクス設定、発生回数、一時停止、回復通知、イベント動作時間を設定します。

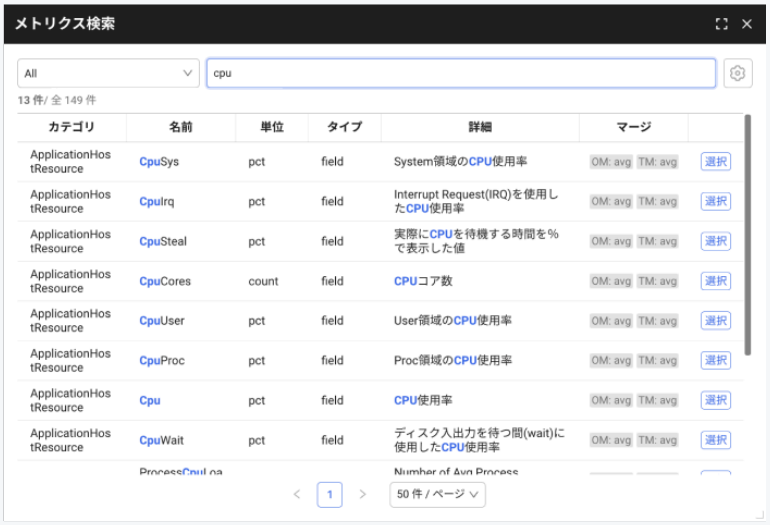
指標設定

1. 検索欄にカテゴリを入力するか、カテゴリー覧からカテゴリ（指標）を選択します。
   指標設定を進めるには、カテゴリの選択が必要です。

- カテゴリ一覧では、カテゴリ名、データ収集間隔、キーを確認できます。直近 3 時間にプロジェクトで収集されたメトリクスデータが表示されます。

2. 一覧にない指標を使用する場合は、**直接入力** を選択してカテゴリを手動で入力します。

> **カテゴリや指標が不明な場合は、メトリクス照会を活用してください。**
>
> 初めて指標を設定する場合、メトリクス照会を利用すると効率的に指標を見つけられます。
>
> - メトリクス照会で指標を検索すると、関連するカテゴリや詳細情報を確認できます。
> - 検索結果から必要なメトリクスの [選択] ボタンをクリックすると、選択した値がイベント設定画面に自動的に反映されます。



3. イベントレベル（Critical、Warning、Info）を選択します。1 つの指標に対して複数のレベルを設定できます。

> **イベントレベルの動作**
>
> 複数の条件を満たしても、最も優先度の高いレベル（Critical → Warning → Info）のみがイベントとして発生します。詳細は レベル動作ガイド を参照してください。
>
> - イベントレベルは、危険（Critical）、警告（Warning）、正常（Info）に分類されます。
> - 同一指標に対して段階的なしきい値でレベルを設定することを推奨します。

4. 選択したイベントレベルのフィールド、演算子、値を入力します。

   a. **+ 追加** をクリックすると条件を追加できます。

   - `&&` : **すべての条件を同時に満たした場合** にイベントが発生
   - `||` : **いずれか 1 つ以上の条件を満たした場合** にイベントが発生

   b. — をクリックすると、その条件が削除されます。

   > ⓘ 条件式の書き方や対応する演算子については、条件設定ガイド を参照してください。

発生回数

条件が何回満たされた場合にイベントを発生させるかを設定します。継続的な問題のみを検知したい場合に有効です。

1. 発生頻度を選択します。
   - **連続**: 設定した回数分、イベントが連続して発生した場合にイベントを発生
   - **直近**: 選択した時間内に、設定した回数分イベントが発生した場合にイベントを発生
     - 選択可能時間: `5秒` , `10秒` , `15秒` , `30秒` , `1分` , `2分` , `3分` , `4分` , `5分` , `10分` , `30分` , `1時間`
2. 発生回数を入力します。
   - **Interval** の値は、選択したカテゴリのデータ収集間隔を示します。（Interval: 10s）

一時停止

過剰なイベント発生を防ぐため、イベント動作を一時停止する時間を選択してください。イベント発生後、設定した時間が経過するまで同一イベントは発生しません。

- 選択可能時間: `使用しない` , `5分` , `10分` , `15分` , `20分` , `30分` , `1時間` , `2時間` , `3時間` , `6時間` , `12時間` , `1日`

> ⓘ **解消通知が有効な場合**
>
> 正常状態の通知（**RECOVERED**）を受信した時点から、設定した時間が経過するまで、同一イベントが再発しても通知は送信されません。

解消通知

解消通知機能を有効にすると、イベント発生後の状態変化（進行中 → 解消）を追跡できます。**イベント履歴** メニューでは、進行中のイベントとして表示されます。

- 解消通知は Critical および Warning レベルにのみ適用されます。Info レベルは常に単発イベントとして動作します。
- Critical または Warning レベルのイベントが解消されると、正常（RECOVERED）状態の通知が送信されます。
- トグルボタンをクリックして **有効化** または **無効化** してください。
  - **有効**: イベントは状態ベースで動作します。しきい値を超えると進行中となり、値がしきい値未満に回復すると解消状態に遷移します。
  - **無効**: しきい値を超えるたびに単発イベントが発生します。

イベント動作時間

イベントが特定の時間帯（勤務時間／非勤務時間／点検時間など）にのみ動作するよう、タグを設定してください。タグを設定しない場合、イベントが有効な状態では 24 時間常時動作します。

1. **+ 追加** をクリックします。
2. イベント動作時間で **+ 新しいタグを作成** をクリックします。
3. **イベント動作時間タグ作成** で **タグ名、曜日、時間、色** を選択し、[適用] ボタンをクリックします。
4. 作成したタグはタグ一覧で確認でき、チェックボックスを選択すると適用されます。
5. タグを修正または削除するには、 をクリックし、**イベント動作時間タグ修正** で修正または削除してください。

> タグを削除すると、そのタグが適用されているすべてのユーザーからタグが削除されます。ただし、イベントルールで使用中のタグは削除できません。

シミュレーション

シミュレーション機能では、設定したイベント条件を過去データに基づいて事前にテストできます。想定されるイベント件数が多すぎる、または少なすぎる場合は、条件を調整して最適なしきい値を見つけることができます。

- **ルール検証**: 新しく作成したイベントルールを過去データに適用した場合の結果を確認できます。
- **しきい値チューニング**: イベントが頻発する、またはまったく発生しない場合に、適切なレベルへしきい値を調整します。

シミュレーション開始

1. イベント条件の設定が完了したら、[シミュレーション] ボタンをクリックします。
2. 想定イベント件数と可視化された結果を確認します。

シミュレーション結果の理解

1. **時系列データ**

   指標の時間帯別変化と各レベルのしきい線がグラフで表示され、イベント発生タイミングを直感的に確認できます。

2. **レベル別発生状況**

   画面右側に Critical、Warning、Info レベルごとのイベント発生回数が表示され、全体の分布や頻度を把握できます。

3. **対象別詳細情報**

   グラフ上の特定ポイントにマウスを合わせると、その時点の正確な指標値と監視対象が表示されます。複数の対象がある場合は色で区別され、対象ごとにパターンが異なる場合はフィルタリング機能で個別分析が可能です。

4. **ピーク区間の特定**

   指標が急激に上昇する時間帯を確認し、問題が頻発する時間帯やパターンを把握できます。

> **シミュレーション活用のヒント**
>
> - **複数時間帯でのシミュレーション**
>
>   平日／週末、ピーク／非ピーク時間帯をそれぞれシミュレーションし、時間帯ごとの最適なしきい値を設定してください。

- **実際の障害期間を含める**

  過去に障害が発生した期間を含めてシミュレーションを実行し、現在のルールが障害を正しく検知できるか確認しましょう。

- **対象フィルタリングの追加**

  監視対象が多い場合、結果が複雑になります。フィルタリング機能を使って特定の対象のみを選択し、明確な結果を確認してください。

- **しきい値を段階的に調整**

  一度に大きく変更するのではなく、10 パーセント単位で段階的に調整しながら最適値を探すことをおすすめします。

イベント対象選択

イベント対象を指定しない場合、プロジェクト内のすべての対象に対してイベントが発生し、多数のイベントが生成される可能性があります。そのため、イベントを適用する対象を明確に設定することを推奨します。

イベント対象を変更すると、イベント件数が変わる場合があります。**シミュレーション** ボタンをクリックして再実行し、想定イベント件数を確認してください。

対象選択

イベントを適用する対象は、**選択入力** または **直接入力** の方法で指定できます。

- 選択入力 **直接入力**

1. タグ、演算子、値を選択または入力してください。
   a. タグが見つけにくい場合は、**メトリクス照会** を使ってタグを検索できます。
2. **+ 追加** をクリックして、条件（ `&&` , `||` ）でイベント対象を追加してください。

監視対象のタグ、演算子、値を直接入力してください。

- 直接入力を選択すると、入力欄の下に使用可能なタグ一覧（**タグ選択入力**）が表示されます。

```
ex. endsWith(okindName, 'example_name') && container == 'prod.billing'
ex. ${4xxErrorType} == '401'
```

> 入力方法を変更すると、作成中の内容は初期化されます。

基本情報および通知設定

イベント名、メッセージ、受信対象 を設定します。設定した内容はイベント履歴や実際の通知メッセージにそのまま表示されるため、識別しやすく分かりやすい名称にすることが重要です。

イベント有効化

設定した条件を満たした際にイベントを発生させるかどうかを設定します。

- **有効**: 設定した条件を満たすとイベントが発生します。
- **無効**: 条件を判定しません。

イベント名

イベントのタイトルとして使用するイベント名を入力してください。設定したイベント名は **イベント一覧** および **イベント履歴** メニューに表示され、この名前で検索することができます。

メッセージ

イベントメッセージを入力してください。入力したメッセージは、イベント履歴および通知に使用されます。 をクリックすると、過去に作成したメッセージ履歴を確認できます。メッセージには変数を使用して、イベント発生時点の実際の値を含めることができます。

- **変数の使用ルール**
  - 変数形式: `${指標名}`
  - 設定した **カテゴリと同一カテゴリ内** の指標のみを変数として使用できます。
  - 複数の変数を組み合わせて、より詳細なメッセージを作成できます。
  - 使用可能な変数一覧は **メトリクス照会** メニューで確認できます。

- **変数の使用例**

	イベント設定	イベント発生時
タイトル	Disk 使用量増加	Disk 使用量増加
メッセージ	`Disk = ${disk}%`	`Disk = 89.9%`

イベント受信

イベント通知を受信するメンバーを指定します。

- **全体受信** **タグ選択受信**

プロジェクトで通知設定が有効になっているすべての対象（メンバー、チャンネル）に通知が送信されます。

指定したタグを持つプロジェクトメンバーに通知が送信されます。

1. **タグ選択受信** を選択します。
2. **受信タグ** 項目で [+ タグ追加] または [+] ボタンをクリックします。
3. **イベント受信タグ** 画面の下にある **+ 新しいタグを作成** をクリックします。
4. **タグ作成** 画面でタグ名を入力し、色を選択した後、[タグ作成] ボタンをクリックしてタグを作成します。
5. 作成したタグは、タグ一覧の をクリックして修正または削除できます。
6. **イベント受信タグ** 画面の **タグ一覧** から希望するタグを選択すると適用されます。

> **受信タグ**
>
> 受信タグは、通知を受け取るユーザーやチャンネルを **グループ単位で管理するための機能** です。イベントに受信タグを設定すると、そのタグが付与されたユーザーにのみ通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

イベント修正／削除

1. **警告アラート** > **イベント設定** の **メトリクス** タブに移動します。

2. イベント一覧で、修正または削除したいイベントの をクリックします。
3. **イベントルール修正** 画面でオプションを修正し、[保存] ボタンをクリックします。

   a. 選択したイベントを削除する場合は、**イベントルール修正** 画面右上の [削除] ボタンをクリックします。

イベント共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules-`YYYYMMDD`.json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
   - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

> 同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。

2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

> ⓘ 修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。
>
> JSON error

JSON データ構造


```
{
  "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"
      ]
  }
}
```

表 | JSON データ構造

JSON フィールド	型	説明	備考
version	Integer	イベントルールのバージョン	変更制限
eventId	String	イベントルールの一意識別子	
enabled	boolean	イベントの有効／無効	
stateful	boolean	状態ベースイベントかどうか	
title	String	ユーザーが設定したイベントルール名	
message	String	ユーザーが設定したイベントルールメッセージ	
category	String	データカテゴリ	
alertLabel	List<String>	イベント状態管理のため、カテゴリごとに指定される基本識別子（プライマリキー）値	変更制限
repeatCount	Integer	繰り返し回数	
repeatDuration	Long	繰り返し期間	
silent	Integer	一時停止時間	
receiver	List<String>	受信タグのキー値一覧	

JSON フィールド	型	説明	備考
timeTag	List<String>	動作時間タグのキー値一覧	
selectString	String	対象選択	
conditions	List<Object>	レベル別の発生条件一覧	
ㄴ level	byte	レベル値	Critical: 30、Warning: 20、Info: 10
ㄴ enabled	boolean	レベルの有効／無効	
ㄴ rule	String	レベル発生条件	
createTime	Long	イベントルールの初回作成時刻	
lastModifiedTime	Long	イベントルールの最終修正時刻	
lastModifiedUser	String	イベントルールを最後に修正したアカウント	
basic	boolean	クイック設定で作成されたイベントかどうか	変更制限
metaId	String	クイック設定テンプレートの一意識別子	変更制限: この値はクイック設定で作成されたイベントにのみ存在します（basic=true）。
selectCondition	Object	クイック設定の対象選択値	この値はクイック設定で作成されたイベントにのみ存在します（basic=true）。

「変更制限」と表示されたフィールドはイベント動作に影響を与える可能性があるため、値を変更しないことを推奨します。

付録

発生条件・対象選択ガイド

メトリクスアラートにおけるイベントの発生条件とイベント対象の選択は、同一の構文を使用します。ただし、イベント発生条件ではフィールド（Field）のキーを変数として使用し、イベント対象の選択ではタグ（Tag）のキーを変数として使用します。

レベル動作ガイド

イベントレベルは、危険（Critical）、警告（Warning）、正常（Info）の3段階に分類されます。

1. **優先順位に基づくイベント発生**

   複数のレベル条件を同時に満たした場合、最も優先度の高いイベントのみが発生します。

   例

   ```
   設定:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   現在の状態: CPU 95%
   → 結果: Critical イベントのみ発生（Warning は抑制される）
   ```

2. **レベル上昇**

   進行中状態を持つイベントにおいて、低いレベルのイベントが進行中の状態で高いレベルの条件を満たすと、両方のレベルが進行中状態になります。

   例

   ```
   設定:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   シナリオ:
   ```

```
1) CPU 80% → Warning 発生（進行中）
2) CPU 95% → Warning（進行中）、Critical（進行中）
```

3. **レベル下降**

   進行中状態を持つイベントで、高いレベルから低いレベルへ遷移した場合、高いレベルは解消され、低いレベルが維持されます。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%

シナリオ:
1) CPU 97% → Critical 発生
2) CPU 85% → Critical 解消、Warning 維持
3) CPU 65% → Warning 解消
```

4. **動作フロー**

```
- Warning: CPU > 70%
- Critical: CPU > 90%

メトリクス値: 60%  →  75%  →  92%  →  85%  →  60%
状態:     正常     Warning    Critical    Warning    正常
                （発生）      （発生）      （維持）      （解消）
                          Warning
                            （維持）
```

> ⓘ 進行中状態を持つイベントは、条件が満たされている間は継続して有効となり、条件が解除されるまで維持されます。

条件設定ガイド

メトリクス名が数字で始まる場合、または特殊文字を含む場合は、`${メトリクス名}` の形式で記述する必要があります。

- 括弧 `()`, `[]`
- 演算子 `+`, `-`, `*`, `/`, `%`
- 区切り文字 `:`, `@`, `#`, `,`, (空白)
- その他の特殊文字 `!`, `^`, `&`, `|`, `~`, `` ` ``, `=`

`${}` 使用例:

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` なしで使用可能: 英字、アンダースコア（`_`）、ドット（`.`）のみを含む場合

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. **比較演算子**

   数値比較

```
より大きい:     cpu > 80
以上:          cpu >= 80
より小さい:     memory < 1000
以下:          memory <= 1000
等しい:        status == 200
等しくない:     error != 0
```

文字列比較

```
status == 'OK'
region == "us-east-1"
```

2. 算術演算子

- 基本的な算術演算

```
加算:     cpu + 10 >= 90
減算:     memory - 100 >= 500
乗算:     cpu * 2 >= 100
除算:     disk / 1024 >= 100
剰余:     value % 10 == 0
```

- 括弧による優先順位の制御

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 負の値の表現

```
cpu > -100
```

3. 論理演算子

- AND 演算（&&）

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 演算（||）

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 複合論理演算

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. パターンマッチング演算子

- LIKE 演算子

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 演算子

```
oname not like 'test-*'
url not like '*debug*'
```

5. 組み込み関数

- Null チェック関数

```
isNull(value)          # null かどうかを確認
isNotNull(value)       # null でないかを確認
nvl(value, 0)          # null の場合に既定値を返す
isEmpty(str)           # 空文字列かどうかを確認
isNotEmpty(str)        # 空文字列でないかを確認
```

- 集計関数

```
sum(cpu, memory, disk)          # 合計
avg(cpu, memory)                # 平均
max(cpu, memory, disk)          # 最大値
min(cpu, memory, disk)          # 最小値
count(value1, value2, value3)   # 件数
```

- 数学関数

```
round(cpu, 2)         # 四捨五入（小数点以下2桁）
```

- 文字列関数

```
length(str)           # 文字列の長さ
startsWith(str, 'prefix')  # 接頭辞の確認
endsWith(str, 'suffix')    # 接尾辞の確認
indexOf(str, 'search')     # 文字列の位置
substring(str, 0, 10)      # 部分文字列
trim(str)             # 空白の削除
replace(str, 'old', 'new') # 文字列の置換
hasStr(str, 'search')      # 文字列の包含判定
```

- 条件関数
  - **if**: 条件式が `true` の場合は `true値` を返し、 `false` の場合は `false値` を返す条件関数

    形式

```
if(条件, true値, false値)
```

    例

```
if(value >= 90, 'High', 'Medium') == 'High'
```

  - **decode**: 値と条件を比較し、一致した条件の結果を返します。一致しない場合は既定値を返します

    形式

```
decode(値, 条件1, 結果1, 条件2, 結果2, ..., 条件N, 結果N, 既定値)
```

    例

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

  - **in**: 値が指定した候補値のいずれかと一致した場合は `true` 、それ以外は `false` を返します

形式

```
in(値, 候補値1, 候補値2, ..., 候補値N)
```

例

```
in(status, 200, 201, 204) == false
```

記述時の注意事項

- 推奨事項
  - 簡単な変数名は `${}` を使わずに記述（例: cpu, memory）
  - 特殊文字を含む場合は必ず `${}` を使用
  - 複雑な演算は括弧で明確に区切る
  - 関数名は正確に入力（大文字と小文字を区別）
- 避けるべき事項
  - 不完全な式の記述（例: `cpu >`, `memory &&` ）
  - 括弧の不一致（例: `(cpu > 80 && cpu > 70)` ）
  - 演算子の連続使用(例: `cpu >> 80`, `cpu >< 80`)

複合メトリクスイベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **複合指標** タブ

複雑なルールを利用してイベントを作成し、警告通知を送信できます。

基本オプション

- **検索**: イベント名またはイベント受信タグを選択して、目的のイベントを検索します
- **JSON ダウンロード／アップロード**: JSON エディタでイベントルールを直接編集するか、ファイルとしてダウンロードして編集後に再アップロードできます
- **通知メッセージ設定**: 受信するメッセージの詳細項目を選択し、項目ごとに内容を編集します
- **+ イベント追加**: 新しい複合メトリクスイベントを追加します

> **権限**
>
> **複合指標** イベントを設定するには **アラート設定** 権限が必要です。この権限は **ホーム > プロジェクト選択 > 管理 > 統合メンバー管理** で、対象メンバーの をクリックして付与できます。

表 | 複合メトリクスイベント一覧の構成

項目	説明
イベントタイトル	ユーザーが指定したイベント名 - 最大 255 バイトまで入力可能
条件	イベント発生条件
イベント状態が解消された場合の追加通知	Critical および Warning レベルのイベントが解消された際に、RECOVERED（正常）状態の通知を受信するかどうか – 有効: 進行中 → 正常状態に切り替わると RECOVERED 通知を送信し、イベント履歴で進行中／解消状態を区別して確認できます – 無効: しきい値を超えるたびに単発のイベントが発生します

項目	説明
イベント受信タグ	該当イベントの通知を受信するユーザーまたはグループ
	イベントの修正および削除
	イベントの有効／無効状態 – 有効: 条件を満たすとイベントが発生 – 無効: イベントは動作せず、条件も評価されません

イベント設定

複合指標 イベントは、メトリクスデータのクエリ言語である **MXQL** を基にイベント条件を作成します。**チャートから作成** 機能では、**MXQL** の自動補完を支援するコンボボックス機能を提供します。

イベント追加

1. **アラート通知** > **イベント設定** から **複合指標** タブをクリックします。
2. 画面右側の [+ イベント追加] ボタンをクリックします。
3. **複合指標** 画面で **チャートから作成** をクリックします。

   a. 提供されているテンプレートを利用できます。
4. **イベント設定** 画面で、先にイベントデータを取得し、その後イベント条件を設定します。

イベントデータ照会

イベントデータを照会してチャートを構成します。**イベント設定** 画面右側の **イベントデータ照会** で、**ウィジェット** または **テキスト** オプションを選択してイベントを設定します。

- ウィジェット
- テキスト

時系列チャートを構成するオプションを使用して、イベント設定に利用する **MXQL** を自動補完できます。

1. データを照会する日付を選択します。
2. データを照会する **カテゴリ**（対象）を選択します。

3. **フィルター** 項目の **+ フィルター追加** をクリックしてイベント条件を作成します。

   a. フィルター画面で、演算式、タグ、フィルター値を選択します。
   b. **+ 追加** をクリックして条件を追加できます。
   c. 条件を削除する場合は、条件の横にある – をクリックします。

4. **グループ化** 項目で、グループ化されたメトリクスデータを選択します。複数選択が可能です。
5. **タイムユニット** 項目で、`秒`、`分`、`時間` 単位のいずれかを選択し、データを分割する時間基準を設定します。
6. **フィールド** 項目で、イベント発生条件に使用するフィールドを選択します。複数選択が可能です。

> テキストモードで入力した MXQL は、ウィジェットモードと互換性がありません。

MXQL を直接編集できるエディタが表示されます。

1. **MXQL** を直接修正し、**照会** ボタンをクリックします。
2. 日付を選択すると、該当内容をチャートまたはグラフで確認できます。

通知

警告通知設定の基本情報を入力します。

- **イベント有効化**: トグルボタンを有効にすると、イベントが動作します。
- **レベル**: イベントレベル（Critical、Warning、Info）を選択します。
  - **イベント状態が解消された場合の追加通知**: トグルを有効にすると、イベントが解消され、正常（RECOVERED）状態の通知が送信されます。
- **タイトル**: イベントルールのタイトルを入力します。
- **メッセージ**: イベント発生時に受信する通知メッセージの内容を入力します。メッセージ内で変数を使用すると、イベント発生時点の実際の値を含めることができます。

  Message example

> **変数使用ルール**

- 変数形式: `${メトリクス名}`
- 設定した **カテゴリと同一カテゴリ** 内のメトリクスのみ変数として使用できます。
- 複数の変数を組み合わせて、詳細なメッセージを構成できます。
- 使用可能な変数一覧は **メトリクス照会** で確認できます。

イベント発行条件

- **データ照会範囲**: リアルタイムデータの照会範囲を選択します。
  - 選択可能な範囲: `30秒` , `1分` , `2分` , `5分` , `10分` , `30分` , `1時間` , `2時間` , `3時間`
- **条件**: イベント発行条件を入力します。イベントデータ照会に含まれるフィールドのみ使用できます。

付加情報

- **インターバル**: 選択した時間間隔（ `1分` , `5分` , `10分` , `30分` , `1時間` , `12時間` ）ごとにアラート条件を確認します。
- **サイレンス**: イベント発生後、選択した時間（ `unset` , `1分` , `5分` , `10分` , `30分` , `1時間` , `12時間` ）の間、同一イベントは発生しません。
- **イベント動作時間**: 勤務時間、非勤務時間、点検時間など、特定の時間帯にイベントが動作するようタグを設定します。タグを設定しない場合、イベントは有効な状態で 24 時間常時動作します。

  1. **+ 追加** をクリックします。
  2. **イベント動作時間** で **+ 新しいタグ作成** をクリックします。
  3. **イベント動作時間タグ作成** で **タグ名、曜日、時間、色** を選択し、[適用] ボタンをクリックします。
  4. 作成したタグはタグ一覧で確認でき、チェックボックスを選択すると適用されます。
  5. 修正または削除するタグの をクリックし、**イベント動作時間タグ修正** で編集または削除できます。

  タグを削除すると、そのタグが適用されているすべてのユーザーから削除されます。ただし、イベントルールで使用中のタグは削除できません。

- **イベント受信タグ**: 受信タグを設定すると、該当タグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

1. **+ タグ追加** または + をクリックします。
2. **イベント受信タグ** 画面下部の **+ 新しいタグ作成** をクリックします。
3. **タグ作成** 画面でタグ名を入力し、色を選択した後、[タグ作成] ボタンをクリックします。
4. 作成したタグは、タグ一覧の をクリックして修正または削除できます。
5. **イベント受信タグ** 画面の **タグ一覧** から適用したいタグを選択します。

> イベント設定時にイベント受信タグを選択すると、該当タグを持つプロジェクトメンバーおよび 3rd-party プラグインに通知を送信できます。**アラート通知** > **通知設定** で、それぞれにタグを割り当てることができます。

イベントルールテスト

設定した条件を過去データに基づいてテストし、どの程度のイベントが発生するかを予測できます。

1. イベントルールをテストする日付を選択し、[確認] ボタンをクリックします。
2. [実行] ボタンをクリックして、イベント発生件数を確認します。

イベント修正／削除

1. **アラート通知** > **通知設定** の **複合指標** タブに移動します。
2. 複合メトリクス一覧で、修正または削除したいイベントの をクリックします。
3. **通知設定** 画面でオプションを修正し、[保存] ボタンをクリックします。

   a. 選択したイベントを削除する場合は、**通知設定** 画面右上の [削除] ボタンをクリックします。

イベント共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules- `YYYYMMDD` .json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
   - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。
2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。

JSON error

通知メッセージ設定

受信するメッセージの詳細項目を選択し、項目ごとに内容を編集できます。

付録

MXQL 文法ガイド

複合指標 は MXQL クエリを基に動作します。詳細な文法については、MXQL 文法ガイド を参照してください。

ヒットマップパターンイベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **ヒットマップパターン** タブ

ヒットマップパターン は、レスポンスタイム分布や呼び出し密度など、アプリケーション実行区間の時間帯別パターン変化を基に異常を自動検知する機能です。

複雑な条件やしきい値を手動で設定しなくても、システムがリアルタイムで収集したヒットマップを分析し、通常とは異なるパターンを検出すると自動的に警告通知を発生させます。ヒットマップパターンアラートは、次のような状況で特に有効です。

- 特定のトランザクションが、通常よりも特定の時間帯に過度に集中する場合
- トラフィック分布やレスポンスパターンが通常の状態から変化した場合
- 条件設定だけでは捉えにくい、パターンベースの異常値が必要な場合

イベント設定

検知するイベントパターンのオプションを設定します。

有効化

トグルをオンにすると、該当パターンが検知された際に警告通知が送信されます。

- **有効:** 条件を満たすとイベントが発生します。
- **無効:** イベントは発生せず、条件も評価されません。

サイレンス

イベント発生後、選択した時間の間はイベントが発生せず、**イベント履歴**にも記録されません。

イベント受信タグ

受信タグを設定すると、そのタグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

1. **+ タグ追加** または + をクリックします。
2. **イベント受信タグ** 画面下部の **+ 新しいタグ作成** をクリックします。
3. **タグ作成** 画面でタグ名を入力し、色を選択した後、[タグ作成] ボタンをクリックします。
4. 作成したタグは、タグ一覧の をクリックして修正または削除します。
5. **イベント受信タグ** 画面の **タグ一覧** から適用したいタグを選択します。

イベント動作時間

勤務時間、非勤務時間、点検時間など、特定の時間帯にイベントが動作するようタグを設定します。タグを設定しない場合、イベントは有効な状態で 24 時間常時動作します。

1. **+ 追加** をクリックします。
2. **イベント動作時間** で **+ 新しいタグ作成** をクリックします。
3. **イベント動作時間タグ作成** で **タグ名、曜日、時間、色** を選択し、[適用] ボタンをクリックします。
4. 作成したタグはタグ一覧で確認でき、チェックボックスを選択すると適用されます。
5. 修正または削除するタグの をクリックし、**イベント動作時間タグ修正** で編集または削除できます。

> タグを削除すると、そのタグが適用されているすべてのユーザーから削除されます。ただし、イベントルールで使用中のタグは削除できません。

ヒットマップパターン種別

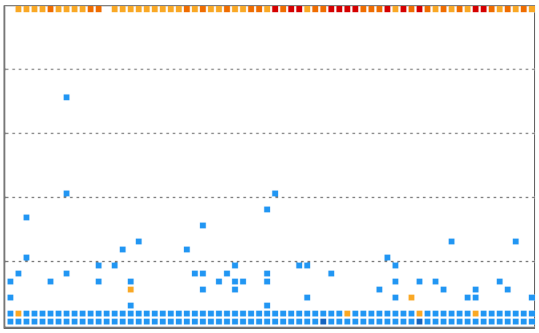
各パターンのトグルをオンにすると、該当パターンが検知された際にイベントが発生します。

ヒットマップ横ラインパターン

複数のトランザクションが、ほぼ同じ応答時間帯で一斉に終了するパターンです。特定の応答時間帯に集中が発生しており、サービス遅延の増加や外部リソースの応答不安定を示します。

- 外部 API、DB、HTTPS 呼び出しでのタイムアウトまたは遅延発生
- 特定リソースを使用しようとしたリクエストが、一定時間待機した後に一括処理される場合

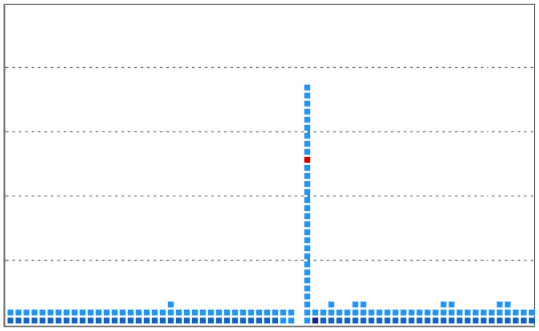
- 共通の I/O またはネットワーク区間で繰り返し遅延が発生している場合



ヒットマップ縦ラインパターン

トランザクションの開始時点は異なるものの、同じタイミングで同時に終了するパターンです。複数のリクエストが特定の瞬間にまとめて処理され、共有リソースのボトルネックを示します。

- DB ロックやファイルロックなど、共有リソースでの待機時間増加
- キューやスレッドプールにおける同時処理遅延
- GC ポーズなどのシステム停止により、一時的にリクエストが止まり、その後同時に再開される状況

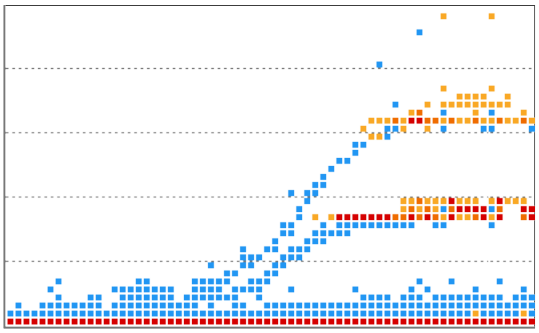


ヒットマップ縦ラインパターンは、ロック競合やボトルネックリソースの状態確認が必要であることを示すサインです。

ヒットマップフライングパターン

特定のリソースやログなどの共通リソース不足により、応答時間が一定間隔で波のように上下するパターンです。

- GC、コネクションプール、ファイルハンドルなどの共通リソース不足
- ログフラッシュやストレージ I/O など、周期的な遅延要因の存在
- バッチ処理やバックグラウンドジョブによる周期的なリソース占有

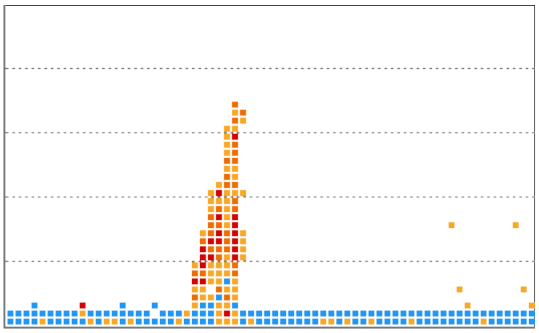


ヒットマップフライングパターンは、周期的なボトルネックやリソース不足の可能性が高いことを示します。

ヒットマップ過負荷パターン

特定の応答時間帯にトランザクションが異常に集中するパターンです。一時的な負荷増大や障害の前兆により、リクエストが正常に分散されず、特定の区間に偏る場合に発生します。

- 一時的な障害や遅延によりトランザクションがキューに溜まり、一括処理されるケース
- バックエンドサービスの応答遅延
- 特定の時間帯におけるトラフィック集中による処理余力不足

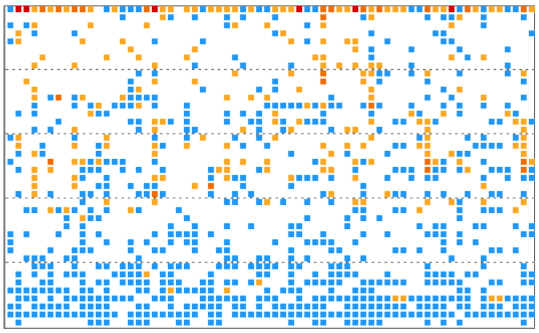


過負荷パターンは、一時的なサービス品質低下や負荷分散失敗を示します。

ヒットマップ暴走パターン

過剰なリクエストや負荷によって、全体的な応答時間が大きく増加するパターンです。サービス全体でパフォーマンス低下が発生している状態を示します。

- ユーザートラフィックや自動化トラフィックの急増
- CPU、メモリ、GC などのシステムリソース不足
- バックエンドのボトルネックによる全体的な応答品質の悪化



暴走パターンは、即時の点検が必要な状態です。

異常検知イベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **異常検知** タブ

異常検知 は、メトリクスデータから異常な上昇または下降パターンを検出し、通常とは異なる動きが検知された場合にイベントを発生させる機能です。

基本オプション

- **検索**: イベント名またはイベント受信タグを選択して、目的のイベントを検索します。
- **+ イベントルール追加**: 新しい異常値イベントを追加します。

表 | 異常値イベント一覧の構成

項目	説明
イベントメッセージ	イベント発生時に送信されるメッセージ
カテゴリ	メトリクスデータを分類する単位で、メトリクスイベント設定の基準となるカテゴリ
フィールド	異常を検知する指標（メトリクスフィールド）
フィルター	異常値の対象となるデータ範囲を制限する条件
上昇／下降パターン	上昇または下降パターンの感度
イベント受信タグ	該当イベントの通知を受信するユーザーまたはグループ
	イベント設定の修正
	イベントの削除
	イベントの有効／無効状態 - 有効: 条件を満たすとイベントが発生 - 無効: イベントは動作せず、条件も評価されません

イベント設定

メトリクスデータを基に、値の動きが通常とは異なり上昇または下降した場合に、警告通知を送信するよう設定できます。

イベントルール追加

1. 画面右側の [+ イベントルール追加] ボタンをクリックします。
2. **イベント基本情報**, **メトリクス**, **異常値**, **付加情報** を順に設定します。

イベント基本情報

1. イベントレベル（Critical, Warning, Info）を選択します。
   - イベントの重要度および通知レベルが決定されます。
2. イベント発生時に受信する通知メッセージを入力します。
   - メッセージ内で変数を使用すると、イベント発生時点の実際の値を含めることができます。

> **使用可能な変数**
> - `$pcode` : プロジェクトコード
> - `$category` : カテゴリ
> - `$field` : フィールド
> - `$value` : 発生値
> - `$oname` : イベント対象

メトリクス

1. **カテゴリ** を選択します。
   - 異常を検知するメトリクスデータの基準となるカテゴリです。

2. イベント発生条件に使用する **フィールド** を選択します。複数選択が可能です。
3. イベント対象を絞り込むためのフィルターを選択します。複数選択が可能です。

異常値

1. **上昇パターン** と **下降パターン** の感度を選択します。
2. 各パターンのトグルボタンをクリックして、有効または無効を切り替えます。
   - 上昇パターンのみを検知する場合: 上昇パターン **ON**, 下降パターン **OFF**
   - 下降パターンのみを検知する場合: 下降パターン **ON**, 上昇パターン **OFF**

付加情報

イベント動作時間

勤務時間、非勤務時間、点検時間など、特定の時間帯にイベントが動作するようタグを設定します。タグを設定しない場合、イベントは有効な状態で 24 時間常時動作します。

1. **+ 追加** をクリックします。
2. **イベント動作時間** で **+ 新しいタグ作成** をクリックします。
3. **イベント動作時間タグ作成** で **タグ名、曜日、時間、色** を選択し、[適用] ボタンをクリックします。
4. 作成したタグはタグ一覧で確認でき、チェックボックスを選択すると適用されます。
5. 修正または削除するタグの をクリックし、**イベント動作時間タグ修正** で編集または削除できます。

> タグを削除すると、そのタグが適用されているすべてのユーザーから削除されます。ただし、イベントルールで使用中のタグは削除できません。

イベント受信タグ

イベント受信タグを設定すると、該当タグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

1. **+ タグ追加** または **+** をクリックします。
2. **イベント受信タグ** 画面下部の **+ 新しいタグ作成** をクリックします。

3. **タグ作成** 画面でタグ名を入力し、色を選択した後、[タグ作成] ボタンをクリックします。
4. 作成したタグは、タグ一覧の をクリックして修正または削除できます。
5. **イベント受信タグ** 画面の **タグ一覧** から適用したいタグを選択します。

> イベント設定時にイベント受信タグを選択すると、該当タグを持つプロジェクトメンバーおよび 3rd-party プラグインに通知を送信できます。**アラート通知** > **通知設定** で、それぞれにタグを割り当てることができます。

イベント修正

1. **アラート通知** > **イベント設定** の **異常検知** タブに移動します。
2. **異常値** 一覧で、修正するイベントの右側にある をクリックします。
3. **異常値** 画面でオプションを修正し、[保存] ボタンをクリックします。

イベント削除

1. **アラート通知** > **イベント設定** の **異常検知** タブに移動します。
2. **異常値** 一覧で、削除するイベントの右側にある をクリックします。
3. 削除確認画面で [削除] ボタンをクリックします。

リアルタイムログイベント

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **リアルタイムログ** タブ

収集したログデータを条件に合わせてフィルタリングして警告通知を設定できます。ログイベントを使用するには、ログモニタリングを有効化する必要があります。**ログモニタリングの有効化**については、各製品のログドキュメントを参照してください。

- **リアルタイムログイベント**: リアルタイムで収集したログで、設定したイベント条件が満たされるとイベントが発生します。

基本オプション

- **検索**: イベント名またはイベント受信タグを選択して目的のイベントを検索
- **JSONダウンロード/アップロード**: イベントルールはJSONエディタで直接編集するか、ファイルにダウンロードして修正後に再アップロード可能
- **通知メッセージ設定**: 受信するメッセージの詳細項目を選択し、項目ごとに内容を編集
- **+ イベント追加**: 新しいログイベントを追加

表 | リアルタイムログイベント一覧構成

項目	説明
イベント名	通知タイトルとして使用されるイベント名 - 最大255バイトまで入力可能
カテゴリ	ログ分類名（ログフォルダ名）
検索キー	ログデータで検索したい項目 - 例: HTTPレスポンスステータスコードの値にアクセスする場合、検索キー `status`
検索値	検索キーに対応する実際のデータ。ログに入力した文字列が含まれる場合に警告通知を送信 - 例: 検索キー `status`、検索値 `200` を設定した場合、HTTPステータスコード200を含むログデータ収集時に警告通知が発生

項目	説明
イベント発生一時停止	イベント通知発生後、同じイベントの発生を一定時間停止 - 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 - 解消通知が有効な場合、正常（RECOVERED）状態の通知受信後、選択した時間中は同じイベントが発生しても通知されない
イベント対象フィルタリング	イベントが適用される対象の条件 - 対象条件を設定しない場合、収集されるすべてのデータに対して通知を判断 - 選択入力: タグ、演算子、値を選択して条件を構成 - 直接入力: 条件を直接入力して対象を指定
イベント受信タグ	イベント通知を受け取るユーザーまたはグループ（受信タグ）
	イベントの編集と削除
	イベントの有効化状態 - 有効: 条件を満たすとイベント発生 - 無効: イベントが動作せず、条件をチェックしない

イベント設定

ログイベントは、リアルタイムで収集されるログ、または特定条件を満たすログが一定数蓄積された際に警告通知を設定できる機能です。

リアルタイムログイベントの追加

1. **アラート通知** > **イベント設定**の**リアルタイムログ**タブをクリックしてください。
2. 画面右側の [+ イベント追加] ボタンをクリックしてください。
3. **イベントルール追加**ウィンドウでログイベント条件を設定してください。
    - **イベント名**: イベントタイトルとして使用する名前を入力してください。

- **イベント有効化**: トグルボタンを有効にすると、条件を満たしたときにイベントが発生します。
- **レベル**: イベントレベル（Critical、Warning、Info）を選択してください。
- **メッセージ**: イベント発生時に受け取る通知メッセージを入力してください。メッセージに変数を使用して、イベント発生時点の実際の値を含めることができます。

  > **変数使用ルール**
  >
  > - 変数形式: `${指標名}`
  > - 設定した**カテゴリと同じカテゴリ**内の指標のみ変数として使用できます。
  > - 複数の変数を組み合わせて詳細なメッセージを構成できます。
  > - 使用可能な変数一覧は**メトリクス照会**で確認できます。

- **カテゴリ**: ログ分類名（ログフォルダ名）をリストから選択するか、直接入力してください。カテゴリを選択してから検索キーと検索値を入力できます。
- **検索キー**: ログデータで検索したい項目を入力してください。
  - 例: HTTPレスポンスステータスコードの値にアクセスする場合、**検索キー** `status`
- **検索値**: **検索キー**に対応する実際のデータを入力してください。ログに入力した文字列が含まれる場合に警告通知を送信します。
  - 例: **検索キー** `status` 、**検索値** `200` を設定した場合、HTTPステータスコード200を含むログデータ収集時に警告通知が発生
- **イベント対象フィルタリング**: イベントが適用される対象の条件を入力してください。入力値がない場合、収集されるすべてのデータに対して通知を判断します。
  - **選択入力**: タグ、演算子、値を選択して対象条件を設定してください。

    > 条件式の記述方法とサポートされる演算子については、条件設定ガイドを参照してください。

  - **直接入力**: 条件を直接入力してください。
- **イベント発生一時停止**: 通知を一時停止する時間を選択してください。イベント発生後、設定した時間中は同じイベントが発生しません。

    - 選択可能時間: `使用しない`、`1分`、`2分`、`3分`、`5分`、`10分`、`20分`、`30分`、`1時間`、`3時間`、`6時間`、`12時間`

- **イベント動作時間**: 特定の時間帯（業務/非業務/メンテナンス時間など）にイベントが動作するようにタグを設定してください。タグを設定しない場合、イベントが有効な状態で24時間常時動作します。

    1. **+ 追加**をクリックしてください。
    2. **イベント動作時間**で**+新しいタグを作成**をクリックしてください。
    3. **イベント動作時間タグの作成**で**タグ名、曜日、時間、色**を選択し、[適用] ボタンをクリックしてください。
    4. 作成したタグはタグリストで確認でき、チェックボックスを選択すると適用されます。
    5. 変更または削除するタグの アイコンをクリックして、**イベント動作時間タグの編集**でタグを編集または削除できます。

    > タグを削除すると、そのタグが適用されているすべてのユーザーのタグが削除されます。ただし、イベントルールで使用中のタグは削除できません。

- **イベント受信タグ**: イベントに受信タグを設定すると、そのタグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

    1. **+ タグ追加**または+をクリックしてください。
    2. **イベント受信タグ**ウィンドウ下部の**+ 新しいタグを作成**をクリックしてください。
    3. **タグ作成**ウィンドウでタグ名を入力し、色を選択してから、[タグ作成] ボタンをクリックしてタグを作成してください。
    4. 作成されたタグはタグリストの アイコンをクリックして編集または削除してください。
    5. **イベント受信タグ**ウィンドウの**タグリスト**から目的のタグを選択すると適用されます。

    > イベント設定時にイベント受信タグを選択して、そのタグを持つプロジェクトメンバーとサードパーティプラグインに通知を送信できます。**アラート通知 > 通知設定**でプロジェクトメンバーとサードパーティプラグインにそれぞれタグを指定できます。

4. **イベントルール追加**ウィンドウで設定が完了したら、[保存] ボタンをクリックしてください。

イベントの編集/削除

1. **アラート通知** > **イベント設定**の**リアルタイムログ**タブに移動してください。
2. ログイベントリストで編集または削除するイベントの アイコンをクリックしてください。
3. **イベント設定**画面でオプションを変更し、[保存] ボタンをクリックしてください。

   a. 選択したイベントを削除するには、**イベント設定**画面の右上の [削除] ボタンをクリックしてください。

イベントの共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules-`YYYYMMDD`.json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
   - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

> 同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。
2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

> 修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。
>
> JSON error

付録

発生条件・対象選択ガイド

メトリクスアラートにおけるイベントの発生条件とイベント対象の選択は、同一の構文を使用します。ただし、イベント発生条件ではフィールド（Field）のキーを変数として使用し、イベント対象の選択ではタグ（Tag）のキーを変数として使用します。

レベル動作ガイド

イベントレベルは、危険（Critical）、警告（Warning）、正常（Info）の3段階に分類されます。

1. **優先順位に基づくイベント発生**

   複数のレベル条件を同時に満たした場合、最も優先度の高いイベントのみが発生します。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%
```

```
現在の状態: CPU 95%
→ 結果: Critical イベントのみ発生（Warning は抑制される）
```

2. **レベル上昇**

   進行中状態を持つイベントにおいて、低いレベルのイベントが進行中の状態で高いレベルの条件を満たすと、両方のレベルが進行中状態になります。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%

シナリオ:
1) CPU 80% → Warning 発生（進行中）
2) CPU 95% → Warning（進行中）、Critical（進行中）
```

3. **レベル下降**

   進行中状態を持つイベントで、高いレベルから低いレベルへ遷移した場合、高いレベルは解消され、低いレベルが維持されます。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%

シナリオ:
1) CPU 97% → Critical 発生
2) CPU 85% → Critical 解消、Warning 維持
3) CPU 65% → Warning 解消
```

4. **動作フロー**

```
- Warning: CPU > 70%
- Critical: CPU > 90%
```

```
メトリクス値: 60%  →  75%  →  92%  →  85%  →  60%
状態:     正常    Warning   Critical   Warning   正常
                （発生）    （発生）    （維持）    （解消）
                        Warning
                          （維持）
```

進行中状態を持つイベントは、条件が満たされている間は継続して有効となり、条件が解除されるまで維持されます。

条件設定ガイド

メトリクス名が数字で始まる場合、または特殊文字を含む場合は、`${メトリクス名}` の形式で記述する必要があります。

- 括弧 `()`, `[]`
- 演算子 `+`, `-`, `*`, `/`, `%`
- 区切り文字 `:`, `@`, `#`, `,`, (空白)
- その他の特殊文字 `!`, `^`, `&`, `|`, `~`, `` ` ``, `=`

`${}` **使用例:**

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` **なしで使用可能:** 英字、アンダースコア（`_`）、ドット（`.`）のみを含む場合

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. 比較演算子

数値比較

```
より大きい:      cpu > 80
以上:           cpu >= 80
より小さい:      memory < 1000
以下:           memory <= 1000
等しい:          status == 200
等しくない:      error != 0
```

文字列比較

```
status == 'OK'
region == "us-east-1"
```

2. 算術演算子

- 基本的な算術演算

```
加算:      cpu + 10 >= 90
減算:      memory - 100 >= 500
乗算:      cpu * 2 >= 100
除算:      disk / 1024 >= 100
剰余:      value % 10 == 0
```

- 括弧による優先順位の制御

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 負の値の表現

```
cpu > -100
```

3. 論理演算子

- AND 演算（&&）

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 演算（||）

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 複合論理演算

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. パターンマッチング演算子

- LIKE 演算子

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 演算子

```
oname not like 'test-*'
url not like '*debug*'
```

5. 組み込み関数

- Null チェック関数

```
isNull(value)          # null かどうかを確認
isNotNull(value)       # null でないかを確認
nvl(value, 0)          # null の場合に既定値を返す
isEmpty(str)           # 空文字列かどうかを確認
isNotEmpty(str)        # 空文字列でないかを確認
```

- 集計関数

```
sum(cpu, memory, disk)          # 合計
avg(cpu, memory)                # 平均
max(cpu, memory, disk)          # 最大値
min(cpu, memory, disk)          # 最小値
count(value1, value2, value3)     # 件数
```

- 数学関数

```
round(cpu, 2)         # 四捨五入（小数点以下2桁）
```

- 文字列関数

```
length(str)            # 文字列の長さ
startsWith(str, 'prefix')  # 接頭辞の確認
endsWith(str, 'suffix')    # 接尾辞の確認
indexOf(str, 'search')    # 文字列の位置
substring(str, 0, 10)     # 部分文字列
trim(str)              # 空白の削除
replace(str, 'old', 'new') # 文字列の置換
hasStr(str, 'search')     # 文字列の包含判定
```

- 条件関数
  - **if**: 条件式が `true` の場合は `true値` を返し、 `false` の場合は `false値` を返す条件関数

    形式

    ```
    if(条件, true値, false値)
    ```

    例

    ```
    if(value >= 90, 'High', 'Medium') == 'High'
    ```

  - **decode**: 値と条件を比較し、一致した条件の結果を返します。一致しない場合は既定値を返します

    形式

```
decode(値, 条件1, 結果1, 条件2, 結果2, ..., 条件N, 結果N, 既定値)
```

例

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

- **in**: 値が指定した候補値のいずれかと一致した場合は `true`、それ以外は `false` を返します

形式

```
in(値, 候補値1, 候補値2, ..., 候補値N)
```

例

```
in(status, 200, 201, 204) == false
```

記述時の注意事項

- 推奨事項
  - 簡単な変数名は `${}` を使わずに記述（例: cpu, memory）
  - 特殊文字を含む場合は必ず `${}` を使用
  - 複雑な演算は括弧で明確に区切る
  - 関数名は正確に入力（大文字と小文字を区別）
- 避けるべき事項
  - 不完全な式の記述（例: `cpu >`, `memory &&` ）
  - 括弧の不一致（例: `(cpu > 80 && cpu > 70)` ）
  - 演算子の連続使用(例: `cpu >> 80`, `cpu >< 80`)

複合ログイベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **複合ログ** タブ

収集したログデータを条件に合わせてフィルタリングして警告通知を設定できます。ログイベントを使用するには、ログモニタリングを有効化する必要があります。**ログモニタリングの有効化**については、各製品のログドキュメントを参照してください。

- **複合ログイベント**: 設定した**データ照会範囲**内のログを基準に、ログ検索条件を満たすログ数が閾値以上の場合にイベントが発生します。

> **権限**
>
> 複合メトリクスイベントを設定するには**通知設定**権限が必要です。 通知設定権限は、**ホーム画面 > プロジェクト選択 > 管理 > 統合メンバー管理**で、権限を付与するメンバーの アイコンをクリックして設定できます。

基本オプション

- **検索**: イベント名またはイベント受信タグを選択して目的のイベントを検索
- **JSONダウンロード/アップロード**: イベントルールはJSONエディタで直接編集するか、ファイルにダウンロードして修正後に再アップロード可能
- **通知メッセージ設定**: 受信するメッセージの詳細項目を選択し、項目ごとに内容を編集
- **+ イベント追加**: 新しいログイベントを追加

表 | 複合ログイベント一覧構成

項目	説明
イベント名	通知タイトルとして使用されるイベント名
テンプレート	複合ログテンプレート

項目	説明
カテゴリ	ログ分類名（ログフォルダ名）
ルール	イベントが発生する条件
イベント状態が解消されたら追加通知	CriticalとWarningレベルのイベントが解消されたとき、正常（RECOVERED）状態の通知を受信するかどうか - 有効: 進行中 → 正常状態に移行したときRECOVERED通知を送信し、イベント履歴で進行中/解消状態を区別して確認可能 - 無効: 閾値を超えるたびに単発のイベントが発生
イベント発生一時停止	イベント通知発生後、同じイベントの発生を一定時間停止 - 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 - 解消通知が有効な場合、正常（RECOVERED）状態の通知受信後、選択した時間中は同じイベントが発生しても通知されない
インターバル	ログデータを照会する時間範囲
イベント受信タグ	イベント通知を受け取るユーザーまたはグループ（受信タグ）
	イベントの編集と削除
	イベントの有効化状態 - 有効: 条件を満たすとイベント発生 - 無効: イベントが動作せず、条件をチェックしない

イベント設定

設定した**データ照会範囲**内のログを基準に、ログ検索条件を満たすログ数が閾値以上の場合に警告通知を設定できる機能です。

複合ログイベントの追加

1. **アラート通知** > **イベント設定**の**複合ログ** タブをクリックしてください。
2. 画面右側の [+ イベント追加] ボタンをクリックしてください。
3. **イベントルール追加**ウィンドウでログイベント条件を設定してください。詳細については、基本設定、詳細設定、追加設定を参照してください。

   複合ログイベントの追加
4. **イベントルール追加**ウィンドウで設定が完了したら、[保存] ボタンをクリックしてください。

基本設定

- **イベント名**: イベントタイトルとして使用する名前を入力してください。
- **イベント有効化**: トグルボタンを有効にすると、条件を満たしたときにイベントが発生します。
- **レベル**: イベントレベル（Critical、Warning、Info）を選択してください。

詳細設定

テンプレート（プリセット）、メッセージ、カテゴリ、**フィルター**、イベント発行条件を設定します。

テンプレート（プリセット）

定義済みのプリセットを選択すると、**フィルター、クエリタイプ、イベント発行条件**が自動的に設定されます。プリセットを選択後、必要に応じて変更できます。

複合ログテンプレートの種類

プリセット	クエリタイプ	デフォルトルール
使用しない	none	`rows > 10`
2xxステータスコード件数	count	`count > 10`

プリセット	クエリタイプ	デフォルトルール
3xxステータスコード件数	count	`count > 10`
4xxステータスコード件数	count	`count > 10`
5xxステータスコード件数	count	`count > 10`
正常ステータスコード(2xx, 3xx)件数	count_v2	`include_minus_exclude_count > 10`
エラーステータスコード(4xx, 5xx)件数	count_v2	`include_minus_exclude_count > 10`
エラー受信件数	count	`count > 10`
例外受信件数	count	`count > 10`
毎秒ログ発生件数	rps	`rps > 100`
フィールド集計（Aggregation）	aggr	`avg > 1000`
比率（Rate）	rate	`rate > 10`
ユニーク値数（Cardinality）	cardinality	`cardinality > 100`
パーセンタイル（Percentile）	percentile	`p99 > 3000`

メッセージ

イベント発生時に受け取る通知メッセージを入力してください。メッセージに変数を使用して動的なメッセージを構成できます。

- **例示:** メッセージ `${host}` でエラー `${count}` 件発生 → 通知メッセージ web-01でエラー15件発生

表 | メッセージ変数

変数	説明	使用条件
${count}	条件に一致するログ件数	常に使用可能
${groupField}	グループ化フィールド名	グループ化通知のみ
${groupValue}	グループ化フィールド値	グループ化通知のみ
${フィールド名}	グループ化フィールド値（フィールド名で直接参照）	グループ化通知のみ

> ${count} 変数はグループ化の有無に関わらず、すべての通知で使用できます。 ${groupField} 、 ${groupValue} 、 ${フィールド名} 変数はグループ化フィールドが設定されている場合のみ置換されます。

カテゴリ

ログ分類名（ログフォルダ名）をリストから選択するか、直接入力してください。カテゴリを選択してから検索キーと検索値を入力できます。

フィルター

Luceneクエリ構文でログ検索条件を入力します。2種類の入力モードをサポートします。

- **フィルターモード**: フィールドと値を選択してクエリを構成します。
- **コードモード**: Luceneクエリを直接入力します。

表 | Luceneクエリ例

クエリ	説明
level:ERROR	levelフィールドがERRORのログ
status_nxx:status_4xx OR status_nxx:status_5xx	4xxまたは5xxステータスコード
appender:accessLog	accessLog appenderのログ
response_time.n:*	response_time.nフィールドが存在するログ

クエリ	説明
`host:web-01 AND level:ERROR`	web-01ホストのエラーログ

イベント発行条件

クエリタイプ別に設定します。選択したテンプレート（プリセット）またはクエリタイプに応じて、追加設定項目が表示されます。

表 | イベント発行条件キーの意味

キー	意味
`rows` / `count`	フィルター条件に一致するログ件数
`rps`	毎秒ログ発生件数（件数 / インターバル秒）
`avg`	対象フィールド値の平均
`sum`	対象フィールド値の合計
`min`	対象フィールド値の最小値
`max`	対象フィールド値の最大値
`rate`	分子件数 / 分母件数 × 100（%）
`cardinality`	対象フィールドのユニーク値数
`p99` / `p95` / `p90` / `p75` / `p50`	対象フィールドのパーセンタイル値
`include_minus_exclude_count`	含むフィルター件数 - 除外フィルター件数

- **件数（count / none）**: 設定したフィルター条件に一致するログの件数をカウントします。

  表 | 件数設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
イベント発行条件	`件数` または `条件に一致するログ` `{演算子}` `{閾値}`（例: `count > 10`）

- **毎秒ログ発生件数（rps）**: 設定したフィルター条件に一致するログの毎秒発生件数（Requests Per Second）を計算します。インターバル時間中の総件数をインターバル秒数で割って算出します。

表 | 毎秒ログ発生件数設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
イベント発行条件	`RPS` `{演算子}` `{閾値}`（例: `rps > 0.5`）

- **フィールド集計（aggr）**: 指定した数値フィールドの集計値（平均、合計、最小値、最大値）を計算します。イベント発行条件で集計方式をドロップダウンから選択できます。

表 | フィールド集計設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
対象フィールド	集計対象の数値フィールド（例: `response_time.n`）
イベント発行条件	`平均` / `合計` / `最小値` / `最大値` `{演算子}` `{閾値}`（例: `avg > 1000`）

- **比率（rate）**: 分子フィルターと分母フィルターの件数比率（%）を計算します。全ログに対する特定条件の比率を監視する際に有用です。

表 | 比率設定項目

設定項目	説明
フィルター（分子）	比率計算の分子となるログ条件（例: `level:ERROR`）
分母フィルター	比率計算の分母となるログ条件（例: `level:*`）
イベント発行条件	`比率（%）` `{演算子}` `{閾値}`（例: `rate > 10`）

  - **例示**: フィルター `level:ERROR`、分母フィルター `level:*`、ルール `rate > 10` → 全ログ中のエラー比率が10%を超えると通知が発生

- **ユニーク値数（cardinality）**: 指定したフィールドのユニーク値数を計算します。異常に多くの種類の値が発生する場合を検出する際に有用です。

表 | ユニーク値数設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
対象フィールド	ユニーク値を計算するフィールド（例: `host`）
イベント発行条件	`ユニーク値数` `{演算子}` `{閾値}`（例: `cardinality > 100`）

- **パーセンタイル（percentile）**: 指定した数値フィールドのパーセンタイル値を計算します。レスポンスタイムのテールレイテンシを監視する際に有用です。パーセンタイルの選択を変更すると、イベント発行条件のキーが自動的に更新されます。

表 | パーセンタイル設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
対象フィールド	パーセンタイルを計算する数値フィールド（例: `response_time.n`）
パーセンタイル	P99、P95、P90、P75、P50から選択

設定項目	説明
イベント発行条件	`P99` / `P95` /... `{演算子}` `{閾値}`（例: `p99 > 3000`）

表 | パーセンタイルの意味

パーセンタイル	意味
P50	中央値。全データの50%がこの値以下
P75	全データの75%がこの値以下
P90	全データの90%がこの値以下
P95	全データの95%がこの値以下
P99	全データの99%がこの値以下

- **含む/除外件数（count_v2）**: 含むフィルターの件数から除外フィルターの件数を引いた純件数を計算します。

表 | 含む/除外件数設定項目

設定項目	説明
フィルター（含む）	含むログ条件（例: `status_nxx:status_4xx OR status_nxx:status_5xx`）
フィルター除外	除外するログ条件（例: `status_nxx:status_4xx`）
分母フィルター	比率計算用の分母（任意）
イベント発行条件	`含む-除外件数` `{演算子}` `{閾値}`（例: `include_minus_exclude_count > 10`）

  - **例示**: フィルター `status_nxx:status_4xx OR status_nxx:status_5xx`、フィルター除外 `status_nxx:status_4xx`
    → 4xx+5xxから4xxを除いた純5xx件数が10を超えると通知が発生

グループ化フィールド（Group By）

グループ化フィールドを設定すると、そのフィールドの値ごとに独立して通知条件が判定されます。ドロップダウンからグループ化するフィールドを選択します。ドロップダウンをクリックすると、現在のカテゴリで使用可能なフィールド一覧がオートコンプリートで表示されます。

- **例示**: フィルター `level:ERROR`、グループ化フィールド `host`、ルール `rows > 3`

表 | グループ化設定の比較

項目	グループ化なし	グループ化あり
判定単位	全ログを1つに集計	グループ値ごとに独立して集計
通知発生	1件	条件を満たすグループ数分
通知タイトル	`イベントタイトル`	`[グループ値] イベントタイトル`
Stateful	全体基準での状態遷移	グループごとに独立した状態遷移

  - `[web-01] エラー通知` — web-01でエラー8件 → 通知発生
  - `[web-02] エラー通知` — web-02でエラー5件 → 通知発生
  - `[api-01] エラー通知` — api-01でエラー2件 → 条件未達、通知なし

> ⓘ **イベント状態が解消されたら追加通知**を有効にした場合、グループごとに独立してABNORMAL → RECOVERED状態が遷移します。例えば `web-01` のエラーが解消されると `[web-01] RECOVERED` 通知が発生し、他のホストの状態には影響しません。

追加設定

- **インターバル**: 選択した時間（`unset`、`1分`、`5分`、`10分`、`30分`、`1時間`、`12時間`）間隔で通知条件を確認します。
- **無音**: イベント発生後、選択した時間（`unset`、`1分`、`5分`、`10分`、`30分`、`1時間`、`12時間`）中は同じイベントが発生しません。

- **イベント動作時間**: 特定の時間帯（業務/非業務/メンテナンス時間など）にイベントが動作するようにタグを設定してください。タグを設定しない場合、イベントが有効な状態で24時間常時動作します。

    1. **+ 追加**をクリックしてください。
    2. **イベント動作時間**で**+新しいタグを作成**をクリックしてください。
    3. **イベント動作時間タグの作成**で**タグ名、曜日、時間、色**を選択し、[適用] ボタンをクリックしてください。
    4. 作成したタグはタグリストで確認でき、チェックボックスを選択すると適用されます。
    5. 変更または削除するタグの✎アイコンをクリックして、**イベント動作時間タグの編集**でタグを編集または削除できます。

    > タグを削除すると、そのタグが適用されているすべてのユーザーのタグが削除されます。ただし、イベントルールで使用中のタグは削除できません。

- **イベント受信タグ**: イベントに受信タグを設定すると、そのタグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

    1. **+ タグ追加**または+をクリックしてください。
    2. **イベント受信タグ**ウィンドウ下部の**+ 新しいタグを作成**をクリックしてください。
    3. **タグ作成**ウィンドウでタグ名を入力し、色を選択してから、[タグ作成] ボタンをクリックしてタグを作成してください。
    4. 作成されたタグはタグリストの✎アイコンをクリックして編集または削除してください。
    5. **イベント受信タグ**ウィンドウの**タグリスト**から目的のタグを選択すると適用されます。

    > イベント設定時にイベント受信タグを選択して、そのタグを持つプロジェクトメンバーとサードパーティプラグインに通知を送信できます。**アラート通知 > 通知設定**でプロジェクトメンバーとサードパーティプラグインにそれぞれタグを指定できます。

シミュレーション

通知を保存する前に、過去のデータでシミュレーションして通知発生予測件数を確認できます。

1. 通知設定を完了後、下部の時間範囲を選択してください。
2. [シミュレーション] ボタンをクリックしてください。
3. チャートに時間帯別データと通知発生回数が表示されます。

> ⓘ シミュレーション時間範囲を調整して、さまざまな時間帯の通知発生パターンを確認できます。直近1〜2分のデータはインデックスビルド遅延により照会されない場合があります。

4. **イベントルール追加**ウィンドウで設定が完了したら、[保存] ボタンをクリックしてください。

ログエクスプローラーから通知を作成

ログエクスプローラーから直接通知を作成できます。

1. **ログ** > **ログ検索**メニューに移動してください。
2. カテゴリと検索クエリを設定してください。
3. 画面右上の [イベント追加] ボタンをクリックしてください。
4. Drawerで通知設定フォームが開き、現在のカテゴリと検索クエリが自動的に入力されます。
5. 追加設定を完了し、[保存] ボタンをクリックしてください。
6. 保存完了後、[移動] ボタンをクリックすると通知一覧に移動します。

イベントの編集/削除

1. **アラート通知** > **イベント設定**の**複合ログ**タブに移動してください。
2. ログイベントリストで編集または削除するイベントの アイコンをクリックしてください。
3. **イベント設定**画面でオプションを変更し、[保存] ボタンをクリックしてください。

   a. 選択したイベントを削除するには、**イベント設定**画面の右上の [削除] ボタンをクリックしてください。

イベントの共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules-`YYYYMMDD`.json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
    - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

> 同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。
2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。

JSON error

トランザクションイベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **TX** タブ

トランザクションイベントは、特定の URL、エラー、ステータスコードなどを基準にトランザクションを詳細に監視する機能です。設定した条件を満たすトランザクションが検出されるとイベントが発生します。トランザクションデータは 1 分単位で分析され、条件を満たすトランザクション数がしきい値を超えた場合に通知が送信されます。

最新エージェントバージョン

トランザクションイベントを使用するには、最低対応バージョン以上のエージェントが必要です。

言語	最低対応バージョン
Java	2.2.50 以上
PHP	未定（最新バージョンの使用を推奨、別途お問い合わせください）
Node.js	0.5.21 以上
Python	1.7.8 以上
.NET	2.3.5 以上
Go	未定（最新バージョンの使用を推奨、別途お問い合わせください）

基本オプション

- **イベントレベル**: イベントの重大度を示し、Info、Warning、Critical の 3 段階に分類されます。
- **検索**: イベント名またはイベント受信タグでイベントを検索します。
- **+ イベント追加**: 新しいトランザクションイベントを追加します。
- **カラムサイズ調整**: 各カラムの境界をドラッグして幅を調整できます。

表 | トランザクションイベント一覧の構成

項目	説明
No.	イベントの番号
	イベントの有効／無効 - 有効: 条件を満たすとイベントが発生 - 無効: イベントは動作せず、条件を判定しません
	イベント設定の修正および削除
イベント名	ユーザーが指定したイベント名 - 最大 255 バイト
条件タイプ	イベント発生条件の種類（経過時間、エラー発生、パターンのみ）
集計基準	トランザクション数を集計する基準
対象パターン	イベントが適用される包含／除外パターン（URL、ステータスコードなど）
発生回数	イベント発生の基準となる回数 - 集計周期: 1分固定（毎正時00秒〜59秒基準、期間選択不可） - この1分間に条件を満たしたトランザクションが設定回数以上発生するとイベントが発生
一時停止	アラート発生後、同一イベントの発生を一定時間抑制 - 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 - 解消通知が有効な場合、RECOVERED 通知を受信後、選択した時間内は同一イベントが発生しても通知されません
イベント受信	当該イベント通知を受信するユーザーまたはグループ（受信タグ）

イベント設定

トランザクションイベントは、**イベント条件、対象パターン、基本情報および受信設定** の各ステップで構成されます。
イベント設定により、特定のトランザクションパターンを検知し、通知を受け取ることができます。

イベント追加

トランザクションイベントを追加するには、**イベント設定** > **TX** タブに移動し、画面右上の [+ イベント追加] ボタンをクリックしてください。

トランザクションイベント追加手順
Step 1. イベント条件定義: イベント発生の条件を定義します。
Step 2. イベント対象選択: イベントが適用される監視対象を選択します。
Step 3. 基本情報および受信設定: イベント名、メッセージ、受信対象を設定します。

イベント条件定義

トランザクションイベントの発生条件を設定します。

条件タイプ

イベントを発生させる基準を選択してください。

- **経過時間**: 指定した時間（ms）以上かかったトランザクション
  - **経過時間しきい値**: 条件タイプが経過時間の場合、ミリ秒（ms）単位で入力します。設定した時間（ms）以上かかったトランザクションが発生するとイベントが発生します。
- **エラー発生**: エラーが発生したトランザクション
- **パターンのみ**: 特定のパターン（ヒットマップパターンなど）に該当するトランザクション

集計基準

アラート条件をどの単位で集計するかを選択してください。

- 集計基準: エージェント別、各エージェントの URL 別、各エージェントのエラークラス別、各エージェントの業務別、URL 別、エラークラス別、業務別、全体。

発生回数（必須）

条件を満たしたイベントが何回発生したときに通知を送信するかを設定します。

例: 発生回数 **1回** は1分以内に条件を満たしたトランザクションが1件でもあれば即時通知、発生回数 **3回** は同じ1分間に条件を満たしたトランザクションが3件以上累積した場合に通知されます。

一時停止

過剰なイベント通知を防ぐため、通知を一時停止する時間を選択してください。

- イベント発生後、設定した時間が経過するまで同一イベントは再度発生しません。
- 選択可能時間: `1分` , `5分` , `10分` , `15分` , `30分` , `1時間` , `2時間` , `3時間` , `6時間` , `12時間` , `1日`

イベント動作時間

イベントが動作する時間帯を設定します。**非勤務時間や点検時間帯の通知発生を制限**できます。タグを設定しない場合、イベントが有効な状態では 24 時間常時動作します。

1. **+ 追加** をクリックします。
2. イベント動作時間で **+ 新しいタグを作成** をクリックします。
3. **イベント動作時間タグ作成** で **タグ名、曜日、時間、色** を選択し、[適用] ボタンをクリックします。
4. 作成したタグはタグ一覧で確認でき、チェックボックスを選択すると適用されます。
5. タグを修正または削除するには、 をクリックし、**イベント動作時間タグ修正** で修正または削除してください。

> タグを削除すると、そのタグが適用されているすべてのアラートから削除されます。ただし、イベント設定で使用中のタグは削除できません。

イベント対象選択

トランザクションをフィルタリングするためのパターンを設定します。

パターン設定方法

- `Enter` キーで絞り込んで複数のパターンを入力できます。
- パターンを設定しない場合、プロジェクト内のすべてのトランザクションが対象となります。
- 同一タイプのパターンは **OR** 条件で適用されます。
- 異なるタイプのパターンは **AND** 条件で適用されます。
- ワイルドカード（`*`）を使用できます。ワイルドカードの位置は最大 2 箇所まで指定できます。
- ワイルドカード（`*`）やスラッシュ（`/`）以外の文字を必ず含める必要があります。

> **スラッシュ（`/`）とワイルドカード（`*`）の組み合わせに注意してください**
>
> - `/api/*` ： `/api` 自体はマッチしません。 `/api/` の後にパスが必要です（例: ✅ /api/users、❌ /api）
> - `/api*` ： `/api` 自体もマッチします（例: ✅ /api、✅ /api/users、✅ /apitest）

一致パターン

指定した項目と一致するデータをフィルタリングします。一致パターンは適用後、青色タグ として表示されます。
各項目につき、最大 10 個までパターンを入力できます。

項目	説明
業務名	業務名パターン
URL	トランザクション URL パターン - 例: `/api/*`, `*/users/*`
エラークラス	エラークラス名パターン - 例: `*Exception`, `*Error`
エラーメッセージ	エラーメッセージ内容パターン

項目	説明
ステータスコード	HTTP ステータスコードパターン - 例: `4**`, `5**`

除外パターン

指定した項目と一致するデータをフィルタリング対象から除外します。除外パターンは適用後、赤色タグ として表示され、一致パターンと同じ形式を使用します。各項目につき、最大 10 個までパターンを入力できます。

パターン例

- **完全一致**

```
パターン: "hello"
"hello"   → 一致
"Hello"   → 不一致（大文字・小文字を区別）
```

- **前方一致（プレフィックス一致）**

```
パターン: "hello*"
"hello"     → 一致
"helloworld" → 一致
"hi"        → 不一致
```

- **後方一致（サフィックス一致）**

```
パターン: "*world"
"world"     → 一致
"helloworld" → 一致
"hello"     → 不一致
```

- **部分文字列一致**

```
パターン: "*ell*"
"hello"     → 一致（h'ell'o）
"excellent" → 一致（exc'ell'ent）
"welcome"   → 不一致（"ell" を含まない）
```

- **複合パターン**

```
パターン: "hello*world"
"hellobeautifulworld" → 一致

パターン: "hello*test*"
"hellomytestcase"    → 一致

パターン: "*test*world"
"mytestcaseworld"    → 一致
```

表 | 実際の使用例

パターン種別	例
URL パターン	- `/api/*`：`/api/` で始まるすべての URL - `/users/*`：`users` を含むすべての URL - `/api/v1/users`：完全一致する URL のみ
エラークラスパターン	- `Exception`：`Exception` で終わるすべてのエラー - `java.lang.*`：`java.lang` で始まるすべてのクラス - `Timeout*`：`Timeout` を含むすべてのエラー
ステータスコードパターン	- `4**`：400 番台のすべてのステータスコード - `404`：404 ステータスコードのみ - `5**`：500 番台のすべてのステータスコード

パターンの制限事項

- **必須文字列**：パターンにはワイルドカード（`*`）以外の文字を最低 1 文字以上含める必要があります。ワイルドカードのみではパターンを設定できません。
  - ❌ 不可：`**`, `*`
  - ✅ 可：`a*`, `a`, `a*`, `/api*`
- **空文字列**：空文字列や null 値はマッチしません。
- **連続ワイルドカード**：連続した `*` は 1 つのワイルドカードとして扱われます。
- **複雑なパターン**：ワイルドカードを 3 個以上含むパターンはサポートされていません。

- **正規表現非対応**：文字クラス（`[a-z]`）、量指定子（`+`, `?`）、グループ化（`()`）は使用できません。
- **大文字・小文字の区別**：すべてのパターンマッチングは大文字と小文字を区別します。
- **エスケープ不可**：文字をそのままリテラルとしてマッチさせる方法はありません。

基本情報および受信設定

イベントの基本情報と通知の受信方法を設定します。

イベント有効化

イベント発生時に通知を送信するかどうかを設定します。

- **有効**: 設定した条件を満たすとイベント通知が送信されます。
- **無効**: 設定した条件を満たしても通知は送信されません。

イベント名（必須）

通知タイトルとして使用されるイベント名を入力してください。

イベントレベル（必須）

イベントレベル（Info、Warning、Critical）を選択してください。

イベント受信

イベント通知を受信するメンバーを指定します。

- **全体受信** **タグ選択受信**

プロジェクトで通知設定が有効になっているすべての対象（メンバー、チャンネル）に通知が送信されます。

指定したタグを持つプロジェクトメンバーに通知が送信されます。

1. **タグ選択受信**を選択します。
2. **受信タグ**項目で [+ タグ追加] または [+] ボタンをクリックします。
3. **イベント受信タグ**ウィンドウ下部の **+ 新しいタグ作成** をクリックします。

4. **タグ作成**ウィンドウでタグ名を入力し、色を選択して [タグ作成] をクリックします。
5. 作成したタグは、タグ一覧の をクリックして編集または削除できます。
6. **イベント受信タグ**ウィンドウの **タグ一覧** から適用したいタグを選択します。

> **受信タグ**
>
> 受信タグは、通知を受け取るユーザーやチャンネルを **グループ単位で管理するための機能** です。
> イベントに受信タグを設定すると、そのタグを持つユーザーにのみ通知が送信されます。
> 受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

参考. イベント動作例

トランザクションイベントが実際に発生した例です。

> **設定条件**
>
> - 集計基準: エージェント別
> - 発生回数: 2回
> - 一時停止: 5分
> - URL パターン: `/api/posts/**` , `/delete/*` , `/*test*/`
> - エラークラスポターン: `*TimeoutException*` , `*NullPointerException`
> - ステータスコードパターン: `4**`
> - 除外エラークラス: `*IOException`
> - 除外ステータスコード: `5**`

発生した通知メッセージ例

イベントメッセージ内のトランザクション情報は、条件に合致した最後のトランザクションの値が表示されます。イベントメッセージには、設定した条件と実際に発生したトランザクションの情報の両方が含まれるため、どの条件によって通知が発生したのかを明確に確認できます。

```
(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
```

イベントメッセージ構成の説明

- **(ignored count: n)**: 一時停止時間中に条件を満たしたが、通知が送信されなかったトランザクションの件数です。一時停止時間中に無視された件数がない場合、この項目は表示されません。
- **集計情報**: 設定した集計方式に応じて表示内容が異なります。
  - **エージェント別**: `[Per Agent] limit: n, count: n`
  - **URL別**: `[Per Url] limit: n` の後に、各URLと発生件数を表示
  - **エラークラス別**: `[Per Error Class] limit: n` の後に、各エラークラスと発生件数を表示
  - **コンテキスト別**: `[Per Context] limit: n` の後に、各コンテキストと発生件数を表示
  - **エージェント別 URL**: `[Per Agent & Url] limit: n` の後に、各URLと発生件数を表示
  - **エージェント別 エラークラス**: `[Per Agent & Error Class] limit: n` の後に、各エラークラスと発生件数を表示
  - **エージェント別 コンテキスト**: `[Per Agent & Context] limit: n` の後に、各コンテキストと発生件数を表示
  - **全体**: `[Total] limit: n, count: n`
- **[Condition]**: イベント発生時点で設定されていた条件を表示します。設定されたパターンのみが表示されます。
  - **context patterns**: コンテキスト一致パターン
  - **url patterns**: URL 一致パターン
  - **error class patterns**: エラークラス一致パターン
  - **error message patterns**: エラーメッセージ一致パターン
  - **status patterns**: ステータスコード一致パターン

  - **exclude context patterns**: コンテキスト除外パターン
  - **exclude url patterns**: URL 除外パターン
  - **exclude error class patterns**: エラークラス除外パターン
  - **exclude error message patterns**: エラーメッセージ除外パターン
  - **exclude status patterns**: ステータスコード除外パターン
- **[Transaction]**: 条件に合致した最後のトランザクションの詳細情報です。
  - **url**: トランザクションURL
  - **txid**: トランザクション固有識別子
  - **status**: HTTP 応答ステータスコード（HTTP トランザクションでない場合は非表示）
  - **error class**: エラークラス（エラークラスが存在しない場合は非表示）
  - **error message**: エラーメッセージ（エラーメッセージが存在しない場合は非表示）

> **参考**
>
> - トランザクションイベントは1分単位で集計・処理されます。
> - 条件タイプが「パターンのみ」の場合、少なくとも1つ以上の一致または除外パターンを設定する必要があります。
> - パターンマッチングは大文字・小文字を区別します。
> - 通知メッセージには、条件に合致した最後のトランザクションの詳細情報が含まれます。

アプリケーションイベント設定(クラシック)

アプリケーションの **サーバーリソース使用量（CPU、メモリ、ディスク）**、**進行中トランザクション数**、**トランザクションエラー**、**応答時間の増加** などの条件に基づいて、アラート通知を設定できます。

> イベント設定 *classic* は、従来方式のイベント設定を継続して利用できるよう分離されたメニューです。
> より多くの機能は イベント設定 で利用できます。

基本オプション

エージェント非アクティブアラート

設定した時間内にエージェントからデータを受信できなかった場合、イベントが発生します。トグルボタンで有効または無効を切り替えられます。

- **有効**: 一定期間データ未受信時にエージェント非アクティブイベントが発生
- **無効**: データ受信状況に関係なくイベントは発生しません

> 従来のエージェント非アクティブ確認間隔は **管理 > エージェント一覧** で設定できます。

状態解消時の追加通知

条件を満たしてイベントが発生した後、状態が正常に復旧した際に **解消（RECOVERED）通知** を送信するかを選択できます。トグルボタンで切り替えます。

- **有効**: イベントは状態ベースで動作し、しきい値超過で進行中、しきい値未満に復旧すると解消状態に切り替わります
- **無効**: しきい値を超えるたびに単発イベントが発生します

エージェント重複アラート

同一名のエージェントが同時に 2 つ以上起動すると、正常なモニタリングが困難になるためイベントが発生します。ただし、設定した時間内は重複エージェントイベントは再発しません。

- エージェント名の重複有無は **イベント設定** メニューで確認してください。
- 選択可能時間: 30秒 , 1分 , 3分 , 5分 , 10分

条件

項目ごとにイベント発生条件を設定できます。項目によって入力形式が異なります。

- **サーバー CPU / サーバー メモリ / サーバー ディスク**
- **アクティブTX / トランザクションエラー**
- **トランザクション応答時間**

条件（%）

Warning と Critical の 2 段階のしきい値を設定します。
しきい値超過時にアラートが発生し、Critical が優先されます。

- 左の値: 警告（Warning）しきい値
- 右の値: 危険（Critical）しきい値

条件（件数）

イベント発生の基準となる件数を入力します。指定した継続時間中に設定値を超えた状態が続くとイベントが発生します。

条件（件数）

条件を設定します。設定した応答時間を超えたトランザクションが指定件数以上発生するとイベント条件を満たします。

条件（応答時間）

基準となる応答時間を選択します。

- 選択可能時間: 1秒 , 2秒 , 3秒 , 5秒 , 10秒 , 15秒 , 30秒 , 1分 , 2分 , 3分 , 5分 , 10分 , 30分 , 1時間
- 例: 件数基準が **10件**、応答時間基準が **1秒**、継続時間が **5秒** の場合

→ 1秒を超えるトランザクションが 10 件以上発生し、その状態が 5 秒継続するとイベントが発生します。

継続

選択した時間中にイベント条件が持続する場合、イベントが発生します。

- 選択可能時間: `5秒` , `10秒` , `15秒` , `30秒` , `1分` , `2分` , `3分` , `5分` , `10分` , `30分` , `1時間`

サイレンス

イベント発生後、選択した時間中に同一条件のイベントは発生しません。適切なサイレンス時間の設定により、過剰なイベント発生を防止できます。

- 選択可能時間: `1分` , `2分` , `3分` , `5分` , `10分` , `30分` , `1時間` , `2時間` , `3時間` , `6時間` , `12時間` , `1日`

イベント有効化

トグルボタンでイベントの動作を制御できます。

- **有効:** 設定した条件を満たすとイベントが発生します
- **無効:** 条件を評価しません

イベント設定

次の項目にイベントを設定することで、システム状態をリアルタイムで監視できます。各項目を設定後、画面右上の **保存** ボタンをクリックしてください。

サーバー CPU / サーバー メモリ / サーバー ディスク

サーバーリソース使用量が設定したしきい値（%）を超えるとイベントが発生します。Warning と Critical で異なるしきい値を設定できます。

- 例: CPU 使用率を Warning: 80%、Critical: 90% に設定した場合、90% を超えると Critical イベントが発生します。

アクティブTX

現在処理中のトランザクション数を監視します。設定した基準件数を超え、その状態が **継続時間** 以上維持されるとイベントが発生します。大量の同時トランザクションを検知するのに有効です。

- 発生件数のデフォルト: 100 件

トランザクションエラー

正常に終了しなかったトランザクションをエラーとして判断します。エラー件数が設定した基準を超え、その状態が継続するとイベントが発生します。

- 収集間隔: 5 秒

トランザクション応答時間

各トランザクションの完了までにかかった時間（応答時間）を基準にイベントを設定します。期待される応答時間を基準に、サービス遅延の検知が可能です。

- 例: 応答時間基準を 10 秒、件数基準を 10 件に設定した場合
  「応答時間 10 秒を超えるトランザクションが 10 件以上発生し、その状態が 5 秒間継続するとイベントが発生」

通知設定

ホーム > プロジェクト選択 > **アラート通知** > **通知設定**

ユーザー別アラート受信設定

ユーザー別アラート受信設定 における他メンバーの情報は、**メンバー管理** 権限を持つメンバーのみが確認できます。それ以外のメンバーには、必須情報がマスキングされて表示されます。

- イベント通知の一括受信設定およびアクセス設定のためのモバイル端末管理は、**アカウント情報** で行うことができます。
- メンバー権限の詳細については、こちらのドキュメント を参照してください。

通知手段の設定

メール、**SMS**、**WhatsApp**、**モバイル** による通知を提供します。

ユーザー別アラート受信設定 セクションで、受信したい通知手段のチェックボックスを選択すると警告アラートを受信できます。通知手段のチェックボックスを解除すると、その手段ではアラートは送信されません。

プロジェクトの最高管理者を除き、すべてのユーザーは自分自身の受信設定のみ変更できます。

メール通知

メール通知は、会員登録時に入力したメールアドレス宛に送信されます。

SMS通知

SMSで通知を受信できます。

SMS通知機能は、有料プロジェクトでのみ利用できます。

1. 画面右上の **プロフィールアイコン** をクリックし、**アカウント管理** に移動します。
2. **アカウント情報** の **ユーザー電話番号** セクションで、[一般携帯電話] ボタンをクリックします。
3. 認証番号を受信する **電話番号** を入力します。
    - SMS通知を受信できる電話番号は、**韓国の携帯電話番号のみ**対応しています。
4. [保存] ボタンをクリックします。

> **登録した電話番号の変更および削除**
>
> - 電話番号を変更する場合は、[変更] ボタンをクリックしてください。SMS認証が必要です。
> - 電話番号を削除する場合は、[削除] ボタンをクリックしてください。削除後は、SMSによる通知を受信できなくなります。

WhatsApp通知

WhatsAppを通じて通知を受信できます。

> WhatsApp通知機能は、有料プロジェクトでのみ利用できます。

1. 画面右上の **プロフィールアイコン** をクリックし、**アカウント管理** に移動します。
2. **アカウント情報** の **ユーザー電話番号** セクションで、[WhatsApp] ボタンをクリックします。
3. 認証番号を受信する **電話番号** を入力します。
4. [認証番号送信] ボタンをクリックします。
5. WhatsAppアプリケーションに送信された6桁の認証番号を入力します。
    - 10分以内に認証番号を受信できなかった場合は、[認証番号再送信] ボタンをクリックしてください。
6. [認証] ボタンをクリックします。

モバイル通知

モバイル通知を受信するには、モバイル端末にWhaTapアプリケーションをインストールし、ログインする必要があります。

- WhaTapアプリケーションをダウンロード – **Android**
- WhaTapアプリケーションをダウンロード – **iOS**

繰り返し通知（エスカレーション）設定

アラート発生後も状況が解消されない場合、初回アラート発生時刻を基準に通知を繰り返す間隔を設定できます。この設定は **Critical** レベルのアラートにのみ適用されます。

例えば、初回アラート発生から即時（0分）、1時間後、1日後に通知を繰り返す場合は、**繰り返し通知（エスカレーション）** カラムに `0,1H,1D` と入力します。

1. **アラート通知** > **通知設定** に移動し、**ユーザー別アラート受信設定** セクションを開きます。
2. **繰り返し通知（エスカレーション）** 項目で設定します。
   - **M**: 分、**H**: 時間、**D**: 日。単位を省略した場合は分単位として扱われます。
   - 数値、または 数値＋単位（**M**, **H**, **D**）の形式で入力してください。入力形式が正しくない場合はメッセージが表示されます。
3. [保存] ボタンをクリックしてください。保存しない場合、設定は反映されません。

イベント受信タグ

イベント設定時に **イベント受信タグ** を選択すると、そのタグを持つプロジェクトメンバーや 3rd-party プラグインに通知を送信できます。イベント受信タグを設定しない場合、**すべてのプロジェクトメンバー** に警告アラートが送信されます。

イベント受信タグの追加・編集・削除

> イベントに **適用中のイベント受信タグ** は削除できません。

1. **アラート通知** > **通知設定** に移動します。
2. **ユーザー別アラート受信設定** セクションのユーザー一覧で、**+ タグ追加** または + をクリックします。
3. **イベント受信タグ** 画面で **+ 新しいタグ作成** をクリックします。
4. **タグ作成** 画面でタグ名を入力し、色を選択して [タグ作成] ボタンをクリックします。

5. 作成されたタグは、タグ一覧の をクリックして編集または削除できます。
6. **イベント受信タグ** 画面の **タグ一覧** から使用したいタグを選択すると適用されます。

イベント受信タグを解除する

1. **アラート通知** > **通知設定** に移動し、**ユーザー別アラート受信設定** セクションを開きます。
2. イベント受信タグを解除したいユーザーの **イベント受信タグ** 項目にある + をクリックします。
3. **イベント受信タグ** 画面で、解除したいタグの **X** をクリックすると、そのタグが解除されます。

受信タグ未設定のアラートを受信する

イベント設定時にイベント受信タグを選択すると、そのタグを持つプロジェクトメンバーや 3rd-party プラグインに通知を送信できます。

- イベント受信タグが設定されていないアラートを受信するには、**受信タグ未設定のアラートを受信** オプションを有効にしてください。
- イベント受信タグが設定されているアラートのみ受信したい場合は、このオプションを無効にしてください。

> すべての警告アラートを受信しない場合は、このオプションを無効にし、**イベント受信タグ** が一つも選択されていない状態にしてください。

イベント受信モードの選択

ユーザー別アラート受信設定セクションで、メンバーまたはサードパーティチャネルごとにイベント通知の受信範囲を次の3つのモードから選択できます。受信モードは、各メンバーまたはチャネル行の**イベント受信タグ**列で選択します。

Mode	Description
すべての通知を受信	イベント受信タグに関係なく、すべての警告通知を受信
選択したタグのみ通知を受信	選択したイベント受信タグを持つ警告通知のみ受信。タグを最低1つ以上選択
すべての通知を受信しない	いかなるイベント通知も受信しない。選択したタグは削除されずそのまま保存

- **選択したタグのみ通知を受信**モードでは、タグ選択領域で通知を受信するタグを1つ以上選択してください。
- イベント受信タグが指定されていない警告通知まで受信するには、タグ選択領域で**未分類**仮想タグを選択してください。**未分類**タグは通常のタグと同様に選択または解除でき、タグ自体を編集・削除することはできません。

> ⓘ **すべての通知を受信しない**モードでは、該当メンバーまたはチャネルがイベント通知をまったく受信しません。このとき選択したタグは削除されず保存されるため、再び**選択したタグのみ通知を受信**モードに切り替えると、以前に選択したタグ設定をそのまま使用できます。

通知受信言語の選択

ユーザーが受信するアラート通知の言語を選択できます。

1. **アラート通知** > **通知設定** に移動します。
2. **ユーザー別アラート受信設定** セクションのユーザー一覧で、言語を設定したいユーザーの一番左にある ▶ ボタンをクリックします。
3. **通知受信言語** を選択します。

曜日・時間別通知設定

ユーザーごとに、曜日および時間帯別の通知受信可否を設定できます。

1. **アラート通知** > **通知設定** に移動します。
2. **ユーザー別アラート受信設定** セクションのユーザー一覧で、曜日・時間別通知を設定したいユーザーの一番左にある ▶ ボタンをクリックします。
3. 警告アラートを受信したい曜日を選択するか、時間帯を入力します。
    - 通知手段ごとに設定できます。

サードパーティプラグイン

外部アプリケーションを通じて警告アラートを受信できます。

- **対応アプリケーション**: Slack、Telegram、Teams、Teams Workflows、Jandi、AlertNow、Webhook、Webhook

JSON、Opsgenie、PagerDuty、LINE、ilert

> WhaTap のサポート範囲に含まれない社内メッセンジャーは、標準の **Webhook** または **Webhook JSON** を使用して連携できます。

表 | サードパーティプラグイン一覧

項目	説明
プラグイン名	プラグインの名称
認証キー	認証キー
認証値	認証値
受信レベル	警告、全体、危険のいずれか
繰り返し通知（エスカレーション）	問題が解消されない場合、初回通知時刻から再通知する間隔 - Critical レベルのアラートにのみ適用
Smart メッセージ（AI）	Telegram プラグイン追加時に AI 通知機能を提供 - 対応言語：韓国語、英語、日本語
イベント受信タグ	プロジェクト内の特定メンバーまたはチームに対する通知受信設定
削除	サードパーティプラグインの削除

サードパーティプラグインの追加

1. **アラート通知** > **通知設定** に移動します。
2. **サードパーティプラグイン** セクションで **追加** ボタンをクリックします。
3. **サードパーティプラグイン追加** 画面で、使用するサービスのタブを選択します。
4. 選択したサービスの案内（サードパーティプラグイン通知の追加方法）に従って設定します。
5. **受信テスト** ボタンをクリックして設定を確認します。

6. すべて完了したら **追加** または **登録** ボタンをクリックします。

サードパーティプラグイン通知の追加方法

Slack

サードパーティプラグイン追加 画面の **Slack** タブで、[**Slackに登録**] ボタンをクリックすると、Slack のチームおよびチャンネルに **WhaTapNotiBot** を登録できます。

Telegram

1. **Telegram**で **@WhaTapApmNotiBot** を検索してください。

   Telegram

2. ボットを選択し、[開始] ボタンをクリックします。
3. チャット画面に `/license` コマンドと、通知を受信したいプロジェクトのアクセスキーを入力してください。

   Telegram

使用可能なコマンド一覧

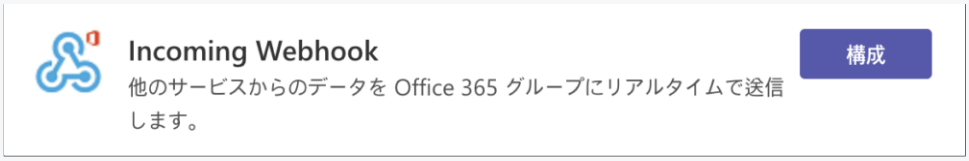
- `/help` : コマンド一覧を表示
- `/license` : 通知を受信するプロジェクトのアクセスキーを入力
- `/remove` : 通知を受信しないプロジェクトコードを入力
- `/list` : 現在通知を受信しているプロジェクトコードを表示

Teams

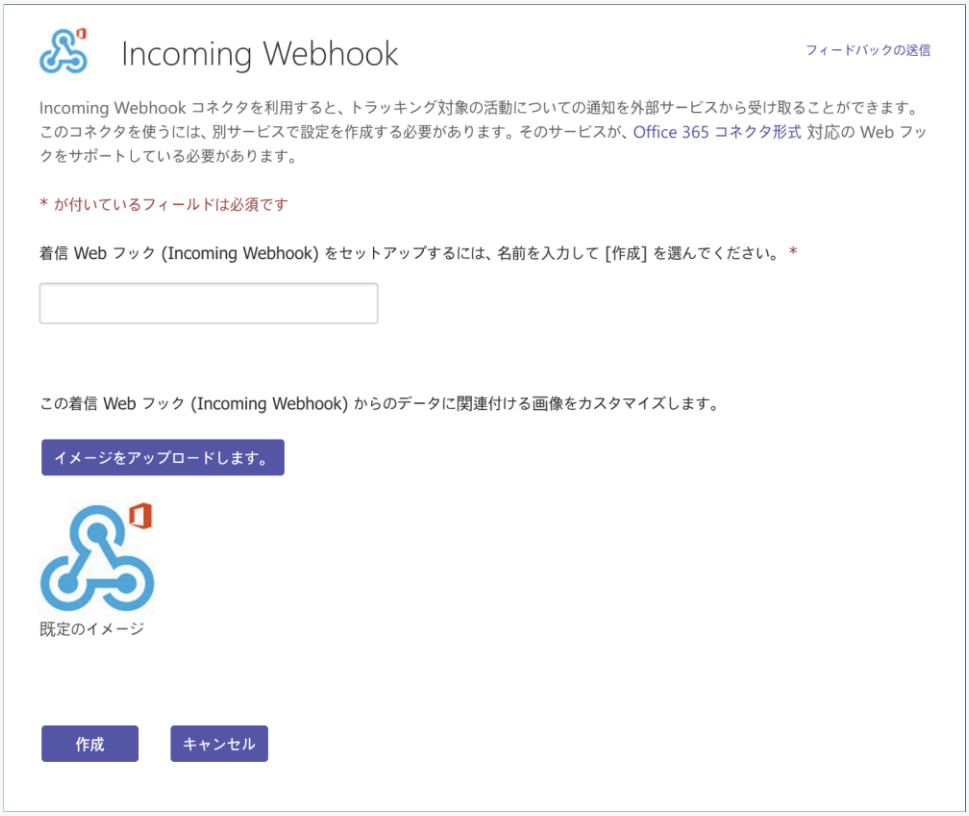
1. 通知を受信したいチャンネルの **コネクタ** に移動します。



2. **Webhook** を追加し、[構成] ボタンをクリックします。



3. Webhook 名を入力し、[作成] ボタンをクリックします。



4. 生成された **Webhook URL** をコピーします。

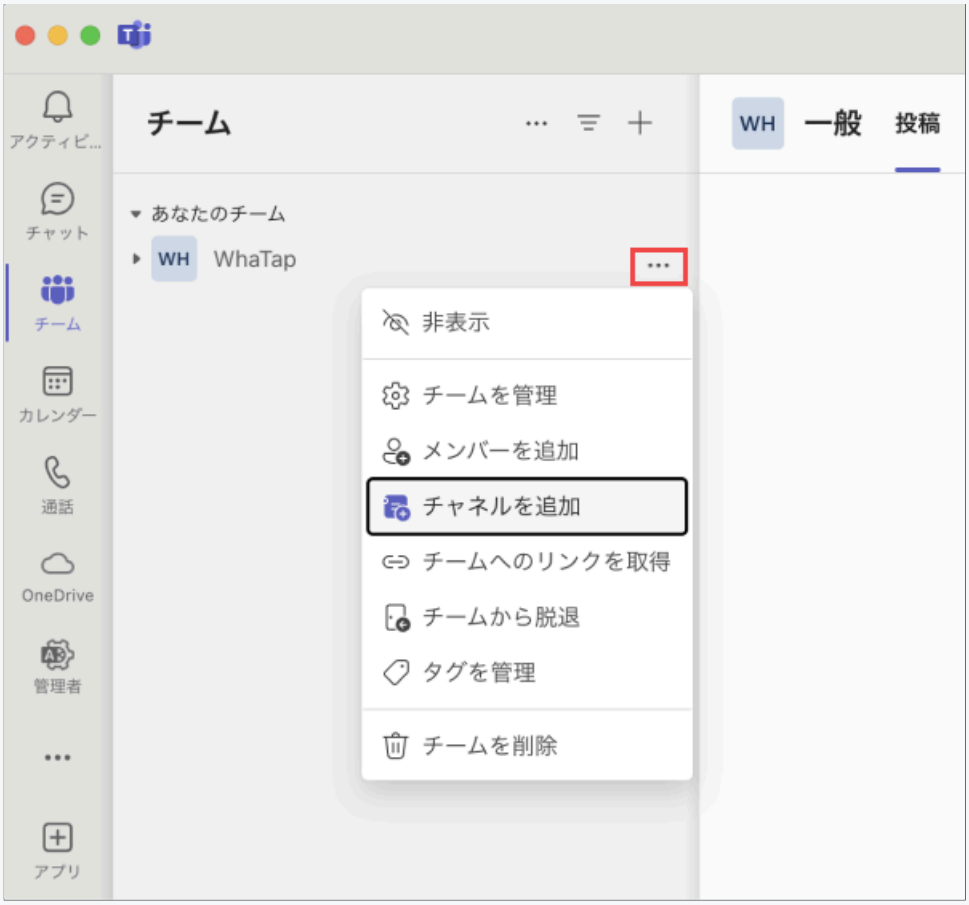


5. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Teams** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
6. [**登録**] ボタンをクリックします。

Teams Workflows

ワークフローを使用するには、**Microsoft Teams** にアクセス可能なアカウントと **ビジネスプランのサブスクリプション**が必要です。

1. **チーム** メニューの **その他の操作 (...)** > **チャネルを追加** をクリックします。

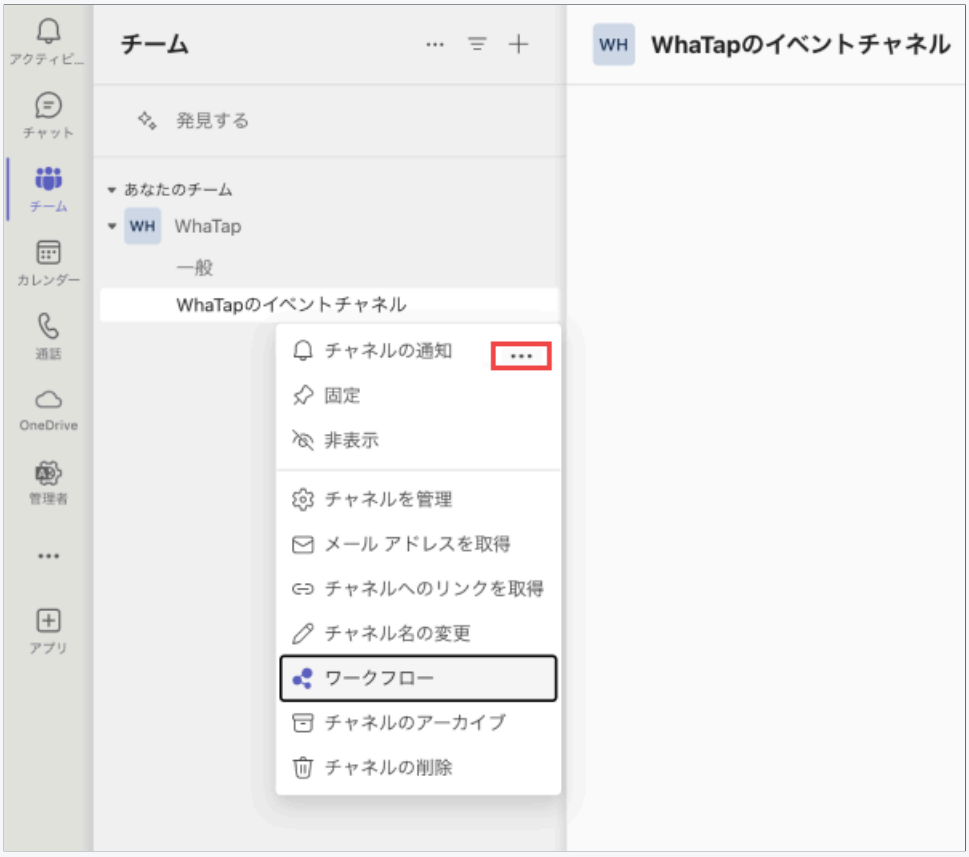


2. **チャネルの作成** 画面で **チャネル名** と **説明** を入力します。**チャネルの種類** は **標準** を選択し、[作成] ボタンをクリックします。

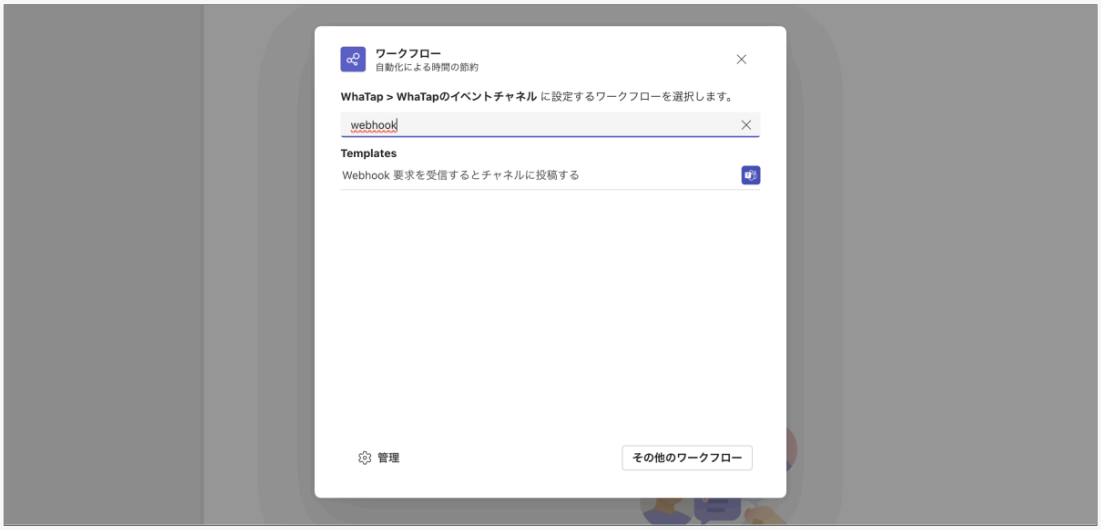


チャネルの種類を **共有** にするとワークフローは使用できません。**プライベート** を選択した場合、ワークフローは追加できますが通知は受信できません。

3. 作成したチャネルの **その他の操作 (...)** > **ワークフロー** をクリックします。



4. **ワークフロー** ダイアログの一覧から **Webhook 要求を受信したらチャネルに投稿** を選択し、[ワークフローを追加] ボタンをクリックします。



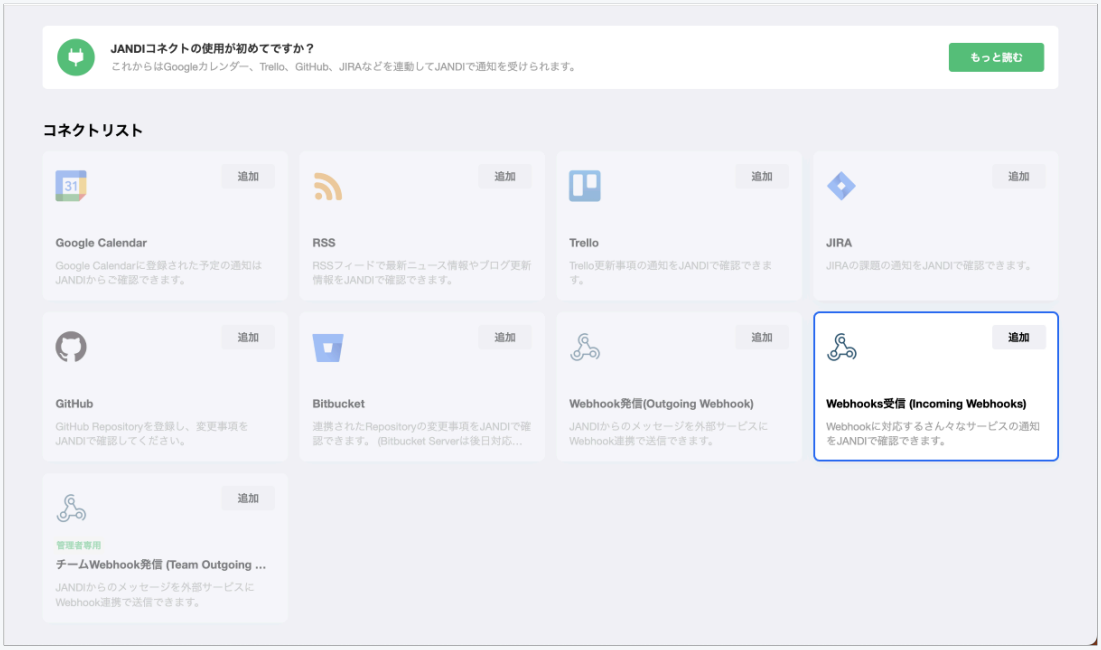
5. **名前** と **接続** の状態を確認し、[次へ] ボタンをクリックします。
6. **詳細** を確認後、[ワークフローを追加] ボタンをクリックします。
7. ワークフローが追加されました。生成された URL をコピーします。



8. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Teams Workflows** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
9. [**登録**] ボタンをクリックします。

Jandi

1. 通知を受信する **トピック** または **チャット** で、**Jandi Connect 設定** に移動します。
2. **コネクト一覧** から **Webhook 受信（Incoming Webhook）** を選択し、[連携する] ボタンをクリックします。



3. [Webhook 受信を追加] ボタンをクリックします。
4. **Webhook URL 登録** 項目に表示される Webhook URL をコピーします。



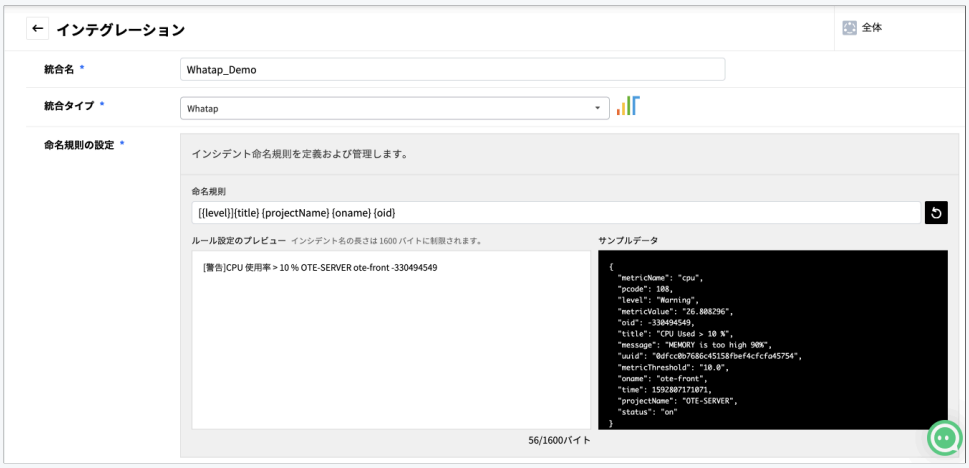
5. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Jandi** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
6. [**登録**] ボタンをクリックします。

AlertNow

1. **AlertNow** サービスにアクセスし、**設定** > **Integration** メニューに移動します。



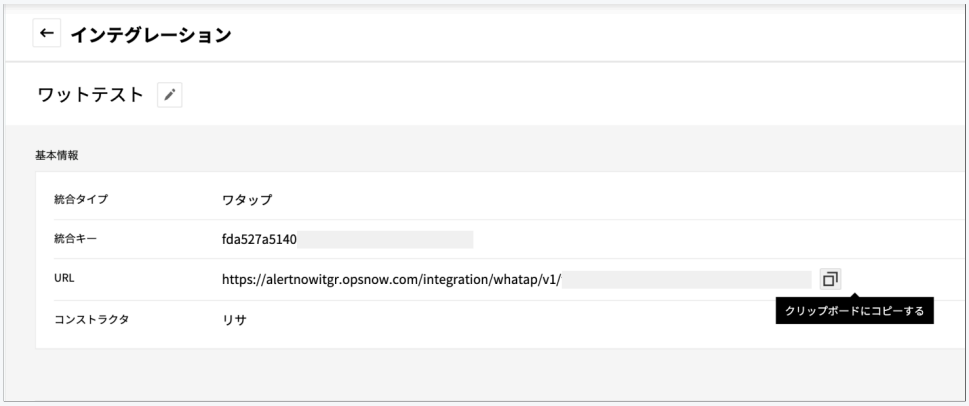
2. **Integration** 画面で [Integration 作成] ボタンをクリックします。
3. Integration タイプの中から **WhaTap** を選択します。
4. **Integration 作成** 画面で、**名前** および **命名規則** を設定します。



5. [確認] ボタンをクリックすると、Integration が作成されます。
6. 作成された Integration の一覧を確認し、該当する Integration を選択します。



7. 詳細画面で、URL 右側のコピーアイコンをクリックして URL をコピーします。



8. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**AlertNow** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
9. [**登録**] ボタンをクリックします。

Webhook

1. 通知を受信する Webhook を登録します。登録した URL に対して、通知の `title` と `text` が **POST** メソッドで送信されます。

```
{
  "title":"Alert Title",
  "text": "Sample warning message"
}
```

2. **Webhook Name** と **Webhook URL** の項目を入力します。
3. [**登録**] ボタンをクリックします。

Webhook JSON

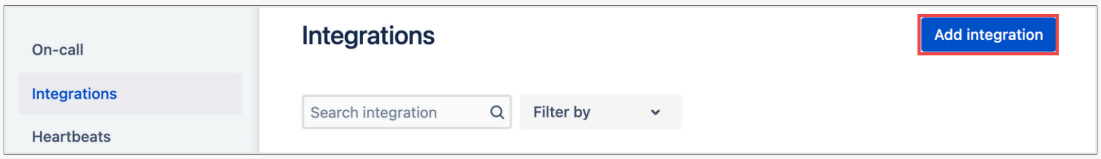
1. 通知を受信する Webhook を登録します。登録した URL に対して、通知データが **JSON 形式**で **POST** メソッドにより送信されます。JSON に含めることができる置換変数については、**使用可能な変数**を参照してください。
   - `${something}` は WhaTap が提供する置換値を表します。対応する全一覧と意味については、**使用可能な変数**を参照してください。
   - イベント状態（`status`）は、**イベント発生**（`on`）と **イベント解消**（`off`）に区別されます。
   - `level` は `None`、`Info`、`Warning`、`Critical` に区別されます。

```
{
  "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}"
}
```

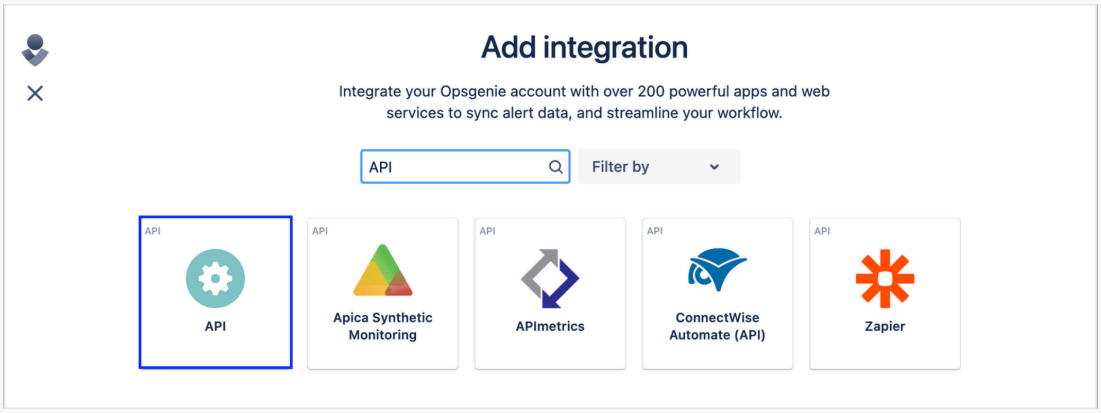
2. **Webhook Name** と **Webhook URL** の項目を入力してください。
3. ヘッダー値を追加するには、**ユーザーヘッダー値の追加** の **+ 追加** をクリックし、key と value を入力してください。
4. [**登録**] ボタンをクリックしてください。

Opsgenie

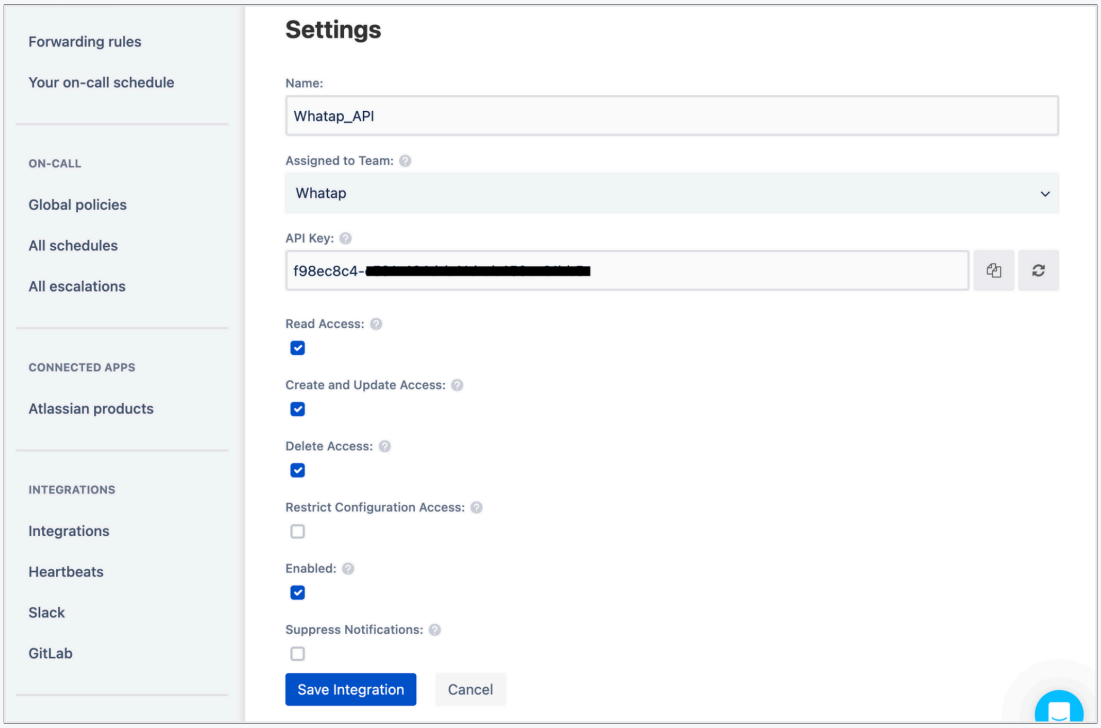
1. **Teams** メニューで、アラートを受信するチームを選択してください。
2. **Integration** メニューに移動し、[Add Integration] ボタンをクリックしてください。



3. インテグレーション一覧で **API** を検索し、選択してください。



4. API 名を設定し、チームを割り当て、各権限を設定した後、[Save Integration] ボタンをクリックしてください。



5. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Opsgenie** タブ下部の **Name** と **API Key** を入力してください。
6. [**登録**] ボタンをクリックしてください。

PagerDuty

1. PagerDuty アカウントを登録し、ログインしてください。
2. PagerDuty で **Services** > **Service Directory** メニューに移動し、[+ New Service] ボタンをクリックしてください。

   PagerDuty
3. サービス情報（**Name**）を入力し、[Next] ボタンをクリックしてください。

   PagerDuty
4. **Escalation Policy** を設定し、[Next] ボタンをクリックしてください。

PagerDuty

5. **Alert Grouping** と **Transient Alerts** を設定し、[Next] ボタンをクリックしてください。

   PagerDuty

6. **Integrations** で **Event API V2** を選択し、[Create Service] ボタンをクリックしてください。

   PagerDuty

7. 作成されたサービス画面で **Integration Name** と **Integration Key** の値をコピーしてください。

   PagerDuty

8. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Integration Name** と **Integration Key** を入力してください。
9. [登録] ボタンをクリックしてください。

> メッセージを受信できない、またはプラグインが登録されない場合は、**Integration Name** または **Integration Key** の値が正しいか確認してください。

LINE

1. LINE アプリをインストールしてください。
2. 以下の QR コードをスキャンして、**WhaTapLabs** 公式チャンネルを友だち追加してください。

   LINE 3rd パーティプラグイン追加

3. 次のコマンドを使用して、プロジェクトを LINE と連携し、通知を設定してください。

   コマンド

```
/help # 使用可能なコマンド一覧を表示
/list # 連携されているプロジェクト一覧を表示
/license [アクセスキー] # 新しいプロジェクト連携を追加
```

```
/remove [プロジェクトコード] # プロジェクト連携を解除
/language [ko|ja|en] # 通知言語を設定（韓国語／日本語／英語）
```

- プロジェクトのアクセスキーは **管理 > プロジェクト管理** で確認できます。
- LINE 内で `/language` コマンドを使用した場合、その設定は **該当する LINE チャットルーム内のみ** に適用されます。他の 3rd パーティプラグインやプロジェクトの通知言語は変更されません。

使用例

```
/license XXXXXXXXXX # プロジェクトを連携
/language ko # 言語を韓国語に設定
/list # 連携済みプロジェクトを確認
```

- **プロジェクトアクセスキー**: WhaTap プロジェクト内の **管理 > プロジェクト管理** で確認できます。
- **言語設定**: `/language` コマンドは LINE チャットルーム内でのみ有効で、他の通知チャネルの言語は変更されません。

ilert

1. ilert コンソールの **Alert sources** 画面右上にある [+ Create alert source] をクリックします。
2. **Integration type** ステップ下部の **Filter integrations** 検索欄に `WhaTap` を入力して検索します。
3. 検索結果から **WhaTap** タイルを選択し、[Next] ボタンをクリックします。
4. アラートソース名を入力し、チームを指定した後、[Next] ボタンをクリックします。
   - エスカレーションポリシーは、新規作成または既存のものを選択できます。
5. アラートのグループ化方式を選択し、[Continue setup] ボタンをクリックします。
   - 最初に選択した **Do not group alerts** は後から変更可能です。
6. 次の画面で **アラートテンプレートや優先度などの追加設定** を行うことができます。ここでは [

Finish setup] ボタンをクリックして設定を完了します。

7. 設定が完了すると、完了画面で **API キー** または **Webhook URL** が生成されます。
8. **Webhook JSON** タブで生成された **Webhook URL** をコピーし、下部の **Webhook URL** 入力欄に貼り付けます。
9. [登録] ボタンをクリックします。

大量アラート発生の防止

短時間に大量のアラートが発生した場合、指定した時間だけアラート通知を一時的に停止します。たとえば、5 分間に 20 件のイベントが発生すると、その後 5 分間は警告アラートが送信されません。設定した **停止時間** が経過すると、大量アラート防止機能は自動的に解除されます。

1. 次の画面に移動してください。**アラート通知** > **通知設定** > **大量アラート通知発生防止** セクション
2. トグルボタンをクリックして **有効化** または **無効化** します。
   - **有効**: 大量アラートを検知した場合、一時的に通知を停止
   - **無効**: アラート通知を再開
3. **検知時間**, **検知回数**, **停止時間** を設定してください。
   - **検知時間** 内に **検知回数** 以上のイベントが発生すると、**停止時間** の間、警告アラートが停止されます。

大量アラートによってブロックされた場合、ブロック案内ボックスが表示されます。大量アラート防止機能をすぐに解除するには、ボックス内の **大量アラート発生中止** ボタンをクリックしてください。

イベント通知フォーマット

共通アラートテンプレート

共通テンプレートは、ほとんどのケースで適用できます。さまざまな環境において、同一のフォーマットでアラートを受信できます。

- プロダクトタイプ: アプリケーション、データベース、Kubernetes
- イベントタイプ: アプリケーションアラート、データベースアラート、メトリクスアラート
- イベントチャネル: SMS、Mobile、3rd-party プラグイン、プラグイン

イベントタイトル

Event title format

```
[Level][Platform][ProjectName][ApplicationName][EventTitle]
```

Event title example

```
[Info][JAVA][アプリケーションプロジェクト][TC-0-1-8081][CRITICAL_HIGH_MEMORY]
```

ApplicationName が設定されていない場合、イベントタイトルから省略されます。

- Platform は以下のいずれかで表示されます。
  - 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

イベントメッセージ

イベントメッセージに含めることができる情報です。`optional` が `false` の場合、常にメッセージに含まれます。`optional` が `true` の場合、該当データが確認できるときのみ表示されます。

イベントメッセージ例

```
Project Name : アプリケーションプロジェクト
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
```

表 | イベントメッセージ構成要素

En	Ko	対応するアラートタイプ	説明
Project Name	プロジェクト名	全て	-
Project Code	プロジェクトコード	全て	-
Agent Name	エージェント名	全て（optional）	oname
Message	イベントメッセージ	全て	-
Alert Type	イベント種別	全て	下記の AlertType 表を参照
Event Time	イベント発生時間	全て	`2022-04-13 10:40:49 +0900` の `+0900` は GMT オフセット
Event Off Time	イベント解除時間	全て（optional）	`2022-04-13 10:40:49 +0900` の `+0900` は GMT オフセット
Metric Name	メトリクス名	全て（optional）	イベント条件判定に使用されるメトリクス名
Metric Value	メトリクス値	全て（optional）	メトリクス値が閾値を超えた場合に含まれる
Metric Threshold	メトリクス閾値	全て（optional）	メトリクス条件の閾値

En	Ko	対応するアラートタイプ	説明
Stateful	復旧イベント通知	全て（optional）	復旧通知が有効な場合 `true`、それ以外は `false`
Event Rule	イベント発生条件	メトリクスアラート	-
Target Filter	イベント対象選択	メトリクスアラート	特定対象から収集されたメトリクスのみ評価
Repeat Count	イベント繰り返し回数	メトリクスアラート	繰り返し時間内に指定回数条件を満たすと発生
Repeat Duration	イベント繰り返し時間	メトリクスアラート	繰り返し回数を評価する時間
Receiver	受信者	メトリクスアラート	-
Query	MXQL クエリ	複合メトリクスアラート	-
Rule	イベント発生条件	複合メトリクスアラート	-
Query Period	クエリ期間	複合メトリクスアラート	-
Query Interval	クエリ間隔	複合メトリクスアラート	-
Silent Time	サイレント時間	複合メトリクスアラート	-
Query	URL	Exception アラート	例外を発生させたリクエストの URL
TXID	トランザクション ID	Exception アラート	-

En	Ko	対応するアラートタイプ	説明
Class	エラークラス名	Exception アラート	-
Log Message	ログメッセージ	サーバー－ファイルログアラート	-
Log File	ログファイルパス	サーバー－ファイルログアラート	-
IP	IP	サーバーアラート全体	-
CPU	CPU	サーバーアラート全体	イベント発生時のスナップショット
CPU_load1	CPU_load1	サーバーアラート全体	イベント発生時のスナップショット
CPU_loadPerCore	CPU_loadPerCore	サーバーアラート全体	イベント発生時のスナップショット
Memory	メモリ	サーバーアラート全体	イベント発生時のスナップショット
Swap	スワップ	サーバーアラート全体	イベント発生時のスナップショット
Disk Name | Used Percent | Free Size | IO Percent	ディスクパフォーマンス	サーバーアラート全体	イベント発生時のスナップショット
Name | Bps | Pps	トラフィックパフォーマンス	サーバーアラート全体	イベント発生時のスナップショット
Message | Time | Name	処理履歴メッセージ	サーバーアラート全体	-

このセクションでは、該当するイベントで提供可能な情報を最大限に表示します。

- **AlertType** は以下のいずれかの値で表示されます。

AlertType	説明
APPLICATION_CPU	アプリケーション CPU アラート
APPLICATION_MEMORY	アプリケーション メモリアラート
APPLICATION_DISK	アプリケーション ディスクアラート
APPLICATION_ACTIVE_TRANSACTION	アプリケーション アクティブTXアラート
APPLICATION_ERROR_TRANSACTION	アプリケーション エラートランザクションアラート
APPLICATION_SLOW_TRANSACTION	アプリケーション トランザクション応答時間アラート
METRICS	メトリクスアラート
COMPOSITE_METRICS	複合メトリクスアラート
ANOMALY	異常値アラート
LOG_REALTIME	リアルタイムログアラート
COMPOSITE_LOG	複合ログアラート
SERVER_REBOOT	サーバー 再起動アラート
SERVER_NO_DATA	サーバー データ未受信アラート
SERVER_PORT	サーバー ポートアラート
SERVER_NETWORK_IOPS	サーバー ネットワーク IOPS アラート
SERVER_NETWORK_BPS	サーバー ネットワーク BPS アラート

AlertType	説明
SERVER_DISK_IO	サーバー ディスク I/O アラート
SERVER_DISK_QUOTA	サーバー ディスク使用量アラート
SERVER_DISK_INODE	サーバー inode アラート
SERVER_CPU	サーバー CPU アラート
SERVER_MEMORY	サーバー メモリアラート
SERVER_CPU_STEAL	サーバー CPU steal アラート
SERVER_MEMORY_SWAP	サーバー スワップメモリアラート
SERVER_LOG_FILE	サーバー ログファイルアラート
SERVER_WINDOW_EVENT	サーバー Windows イベントアラート
SERVER_OFF	サーバー アラート OFF 通知
SERVER_ACKNOWLEDGE	サーバー 処理確認通知
SERVER_PROCESS_COUNT	サーバー プロセス数アラート
SERVER_PROCESS_CPU	サーバー プロセス CPU アラート
SERVER_PROCESS_MEMORY	サーバー プロセス メモリアラート
SERVER_PROCESS_OFF	サーバー プロセスアラート OFF 通知
AGENT_ACTIVE	エージェント 有効化アラート
AGENT_INACTIVE	エージェント 無効化アラート

AlertType	説明
AGENT_REACTIVATED	エージェント 再有効化アラート
URL	URL アラート
TOO_MANY_EVENT	イベント多発アラート
CLOUD_WATCH	CloudWatch アラート
EXCEPTION	例外アラート

アプリケーション警告アラート

アプリケーションアラートでは、Event Title と Event Message の両方が提供されます。**イベント状態が解消されたら追加に通知する** 機能を使用している場合、イベント発生条件が解除されると Event Off Message が送信されます。

Event Type	Event Level	Event Title	Event Message	Event Off Message
サーバー CPU	Warning	HIGH_CPU	CPU is high. `${value}% (>= ${threshold}%)`	RECOVERED: CPU is high. less than `${threshold}%`
サーバー CPU	Critical	CRITICAL_HIGH_CPU	CPU is too high. `${value}% (>= ${threshold}%)`	RECOVERED: CPU is too high. less than `${threshold}%`
サーバー メモリ	Warning	HIGH_MEMORY	Memory is high. `${value}% (>= ${threshold}%)`	RECOVERED: Memory is high. less than `${threshold}%`
サーバー メモリ	Critical	CRITICAL_HIGH_MEMORY	Memory is too high `${value}% (>=`	RECOVERED: Memory is too high. less than

Event Type	Event Level	Event Title	Event Message	Event Off Message
			${threshold}%)	${threshold}%
サーバーディスク	Warning	HIGH_DISK	Disk is high ${value}% (>= ${threshold}%)	RECOVERED: Disk id high. less than ${threshold}%
サーバーディスク	Critical	CRITICAL_HIGH_DISK	Disk is too high ${value}% (>= ${threshold}%)	RECOVERED: Disk is too highf. less than ${threshold}%
正常トランザクション	Warning	HIGH_ACTIVE_TRANSACTION	Active Transaction Count is over ${value} (>= ${threshold})	RECOVERED: Active Transaction Count is less than ${threshold}
エラートランザクション	Warning	HIGH_ERROR_TRANSACTION	Error Transaction Count is over ${threshold} (${value})	RECOVERED: Error Transaction Count is less than ${threshold}
遅いトランザクション	Warning	TOO_MANY_SLOW_TX	Too many delayed transactions (${value}, above ${time} ms)	RECOVERED: Too many delayed transactions. less than ${threshold}

イベント履歴

ホーム > プロジェクト選択 > **アラート通知** > **イベント履歴**

イベント履歴 では、プロジェクトで発生したさまざまなイベントを確認し、詳細情報を分析できます。Elasticsearch DSL クエリをサポートしており、指定した条件に一致するイベントを迅速かつ正確に検索できます。

イベント履歴の確認

検索結果で **イベントタイトルをクリック**すると、そのイベントが発生した時間帯の詳細データを分析できる画面に移動します。これにより、イベント発生前後の状況を総合的に把握できます。

1. **アラート通知** > **イベント履歴** に移動してください。
2. **イベント履歴** 画面の **時間選択** で、イベント履歴を確認する期間を設定してください。
3. フィルター入力欄にクエリを入力し、[虫眼鏡] ボタンをクリックしてください。
    - クエリを入力しない場合、すべてのイベントが表示されます。

カラム設定

表示するカラムや非表示にするカラムを設定できます。

1. **イベント履歴** 画面の左上にある [カラム設定] ボタンをクリックしてください。
2. **カラム設定** 画面で、表示または非表示にするカラムを選択してください。
    - **表示項目** でカラムの順序を変更できます。
    - **カラム名検索** 欄で目的のカラムを検索できます。

カラム項目	説明
イベント名	イベントのタイトル
メッセージ	イベントのメッセージまたはスナップショット情報

カラム項目	説明
イベント状態	イベントの現在の状態
エージェント名	イベントが発生した対象エージェントの名前（Oname） - 特定のエージェントに紐づかないイベントの場合は空欄で表示されます
エージェントグループ	イベントが発生した対象エージェントの種類（OkindName） - 特定のエージェントに紐づかないイベントの場合は空欄で表示されます
エージェントノード	イベントが発生した対象エージェントのサーバー（OnodeName） - 特定のエージェントに紐づかないイベントの場合は空欄で表示されます
イベント発生時刻	イベントが発生した時刻
イベント解消時刻	イベントが解消された時刻 - 解消状態を持たないイベントの場合は `-` と表示されます

3. 設定が完了したら、[確認] ボタンをクリックしてカラム設定を反映してください。

CSV ダウンロード

検索結果を CSV ファイル形式でダウンロードできます。

1. **アラート通知** > **イベント履歴** で [CSV] ボタンをクリックしてください。
2. 現在のフィルター条件に一致するイベント一覧が CSV ファイルとして保存されます。

進行中のイベントのみ表示

現在進行中の状態のイベントのみを表示できます。

1. **アラート通知** > **イベント履歴** で [進行中のイベントのみ表示] ボタンをクリックしてください。
2. フィルタークエリに次の条件が自動的に追加され、検索が実行されます。

```
Stateful: true and Status : "ON"
```

イベント検索

- **正確な検索**: 必要なイベントだけを取得できるよう、条件を正確に指定
- **複合条件**: 複数のフィールドを同時に検索し、複雑な条件も一度に処理
- **柔軟なパターン**: ワイルドカードや部分一致など、さまざまな検索方式をサポート
- **高速パフォーマンス**: インデックス化されたデータを活用した高速検索

検索可能なフィールド

フィールド名および値は大文字と小文字を区別しません。

- **フィールド名タグ**: フィールドがサポートされるイベントタイプ
- **フィールドタイプ**: フィールドのデータ型
  - `#`: Number
  - `T`: String
  - : Boolean
  - : Date

イベント情報フィールド

フィールド名	タイプ	説明	例
Title	String	イベントタイトル	"Database Connection Error"
OffTitle	String	イベント解消タイトル	"RECOVERED: Database Connection Error"
Message	String	イベントメッセージ	"Connection timeout occurred"
OffMessage	String	イベント解消メッセージ	"RECOVERED: Connection timeout occurred"

フィールド名	タイプ	説明	例
Level	String	現在のイベントレベル	`Critical`, `Warning`, `Info`
OriginLevel	String	元のイベントレベル	`Critical`, `Warning`, `Info`
Status	String	イベント状態	`ON`, `OFF`, `CANCEL`, `ACKNOWLEDGE`, `MAINTENANCE`, `DISABLED`
ActCount	Number	発生回数	2
MetricName	String	指標名	"memory"
MetricValue	String	指標値	"85.5"
OffValue	String	解消時の値	"72.8"
MetricThreshold	String	閾値	"80"
alertType	String	イベント種別	"BASIC", "METRICS", "TRANSACTION" など
alertId	String	イベントルールの一意識別子	"zf3uojer0fv4v7"

イベントタイプ別対応フィールド

> ⓘ 特定のイベントタイプでのみサポートされるフィールドです。該当しないイベントタイプの場合、値が存在しないことがあります。

フィールド名	タイプ	イベント種別	説明	例
eventRule	String	基本、メト	イベント発生	"memory ≥ 80"

フィールド名	タイプ	イベント種別	説明	例
		リクス	ルール	
field	String	リアルタイムログ	ログ検索キー	"content"
keyword	String	リアルタイムログ	ログ検索値	"Error"
logCategory	String	リアルタイムログ	ログカテゴリ	"AppLog"
logContent	String	リアルタイムログ	ログ内容	"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."

対象フィールド

フィールド名	タイプ	説明	例
Oid	Number	エージェントの一意識別子	-98765432
Oname	String	エージェント名	"web-server-01"
Okind	Number	エージェント種別の一意識別子	867318026

フィールド名	タイプ	説明	例
OkindName	String	エージェント種別名	"web-server"
Onode	Number	エージェントノードの一意識別子	334634079
OnodeName	String	エージェントノード名	"production-node-1"

イベント固有値フィールド

フィールド名	タイプ	説明	例
Id	Number	イベントの一意識別子	5768121
UUID	String	イベント固有識別子	"550e8400-e29b-41d4-a716"

状態／フラグフィールド

フィールド名	タイプ	説明	例
Stateful	Boolean	状態ベースイベントかどうか	true, false
Disabled	Boolean	無効化されたイベントかどうか	true, false
Escalation	Boolean	エスカレーション有無	true, false
SystemEvent	Boolean	システムイベントかどうか	true, false

検索タイプ別クエリ構文

> ⓘ 検索の基本構文については、WhaTap ログ検索クエリ構文 ドキュメントを参照してください。

1. キーワード検索

「Database Connection」という文字列を含むイベント

```
title: "Database Connection"
```

2. 複数値検索

指定した複数の値のうち、いずれかを含むイベントを検索します。

```
title: Database Connection
```

現在のイベントレベルが Info または Warning のイベント

```
level: info warning
```

3. パターン検索

ワイルドカードを使用してパターン検索ができます。

名前パターンで特定のエージェントで発生したイベントを検索

```
oname: web-*-prod
```

タイトルが Connection で終わるイベントを検索

```
title: *Connection
```

4. 複合条件検索

複数の条件を組み合わせて、より正確に検索できます。

レベルが Critical で、タイトルに Database を含むイベント

```
level: critical and title: Database
```

特定のエージェントにおける Warning または Critical のイベント

```
oname: web-server-01 and level: Warning Critical
```

5. OR 条件検索

複数の条件のうち、**いずれか一つでも満たす**イベントを検索します。

タイトルまたはメッセージに特定のキーワードを含むイベント

```
title: Connection or message: Connection
```

6. 除外条件

特定の条件を除外して検索できます。

Info レベルを除いたすべてのイベント

```
not level: info
```

リアルタイムアラート

リアルタイムアラート 機能は、**イベント設定** で設定したイベントが発生した際に、ポップアップ形式で表示されます。イベント発生と同時に通知を受け取ることで、問題を迅速に把握し、対応することができます。また、すでに発生した通知や見逃した通知の履歴を確認することで、リアルタイム対応が難しかった状況を把握し、解決に役立てることができます。

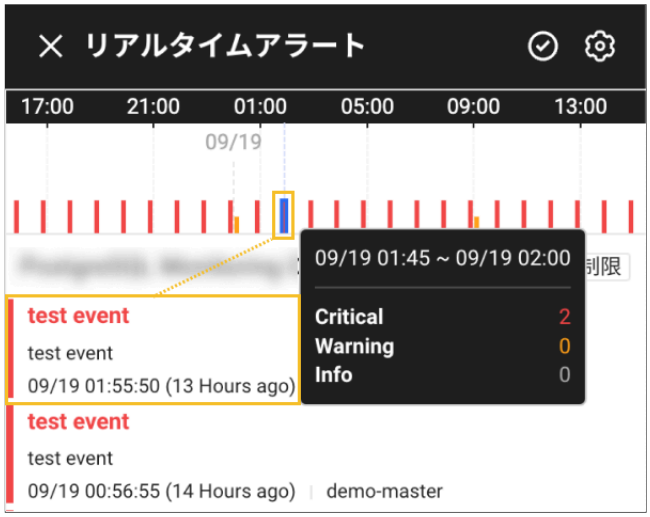
リアルタイムアラートを確認する

リアルタイムアラートは、画面右上にポップアップとして表示されます。通知が発生した時刻、タイトル、メッセージを確認できます。

- 個別の通知を閉じるには、メッセージ右上の × ボタンをクリックしてください。
- 画面に表示されているすべてのポップアップ通知を閉じるには、**すべての通知を閉じる** をクリックしてください。

通知履歴を確認する

1. プロジェクトを選択して表示される画面の最上部にある 🔔 をクリックしてください。
2. **リアルタイムアラート** ウィンドウが表示されます。
    - 現在のプロジェクトについて、直近24時間以内の通知を最大100件まで確認できます。
    - 個別のイベント項目を選択すると、関連するメニューに移動します。
    - **リアルタイムアラート** ウィンドウ内の時間別チャートで任意の時間帯を選択すると、該当するイベント項目に移動します。



- すべての通知履歴を確認するには、**全体の通知履歴** ボタンをクリックしてください。**アラート通知** > **イベント履歴** メニューに移動します。

> ⓘ **リアルタイムアラート** の一覧が空の場合、**全体の通知履歴** ボタンが表示されます。**受信タグ未設定の通知を受け取る** オプションを無効にしてください。

イベントを一括で解消する

リアルタイムアラート の一覧から、進行中のすべてのイベントを一括で解消できます。
イベント解消が完了するまでに、約30秒〜1分程度かかる場合があります。

- **イベント一括処理** 機能を使用すると、現在進行中のイベントに対する解消通知が大量に発生する可能性があります。
- **イベント一括処理** 機能を使用した後でも、イベントルールが有効であり、データが引き続きしきい値を超えている場合、イベントは再度発生することがあります。
- イベント解消通知は、**アラート通知** > **イベント設定** メニューでイベントを編集し、**イベントが解消されたときに追加通知** または **解消通知** オプションを無効にすることで停止できます。

使用方法:

1. プロジェクトを選択して表示される画面上部の をクリックします。
2. **リアルタイムアラート** ウィンドウが表示されたら、右上の をクリックします。
3. リアルタイムアラートリストの進行中イベントを一括で解消します。 というメッセージが表示されたら、**解消** ボタンをクリックします。

リアルタイムアラートを設定する

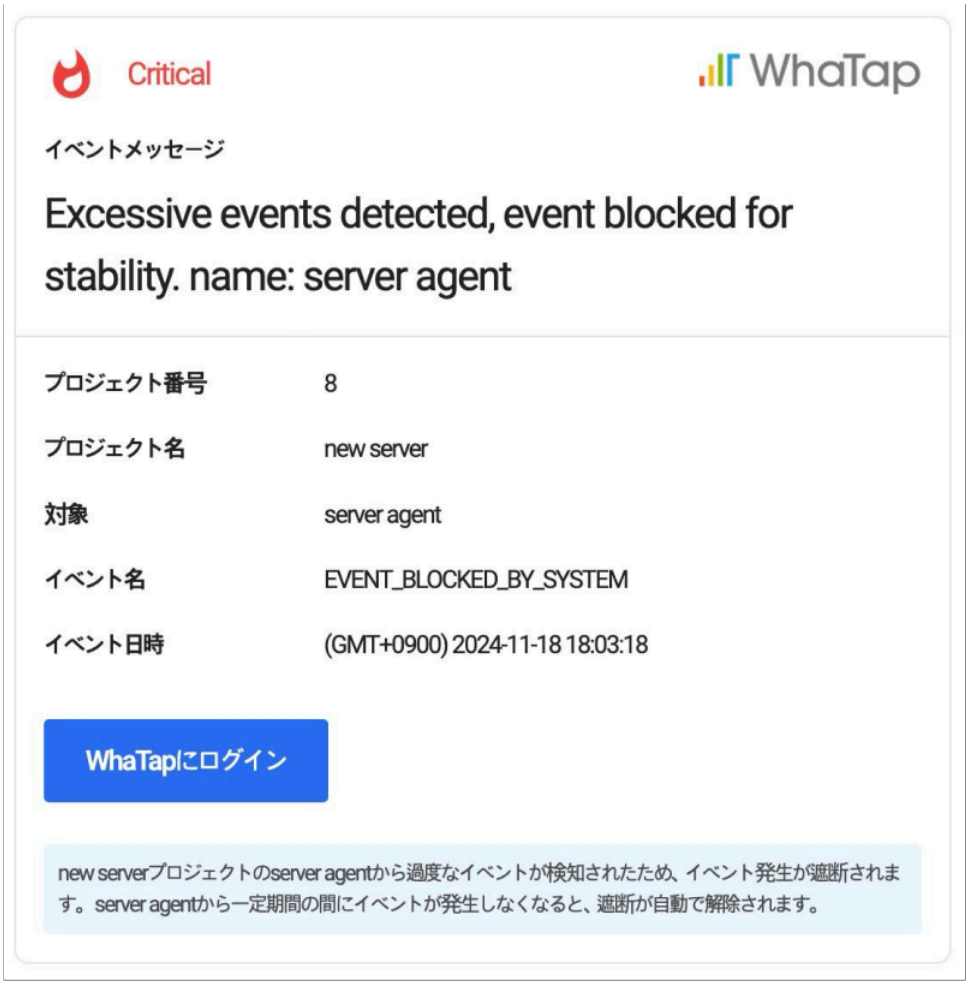
1. プロジェクトを選択して表示される画面上部の をクリックします。
2. **リアルタイムアラート** ウィンドウが表示されたら、右上の をクリックします。
3. リアルタイムアラート設定画面で各項目を設定します。
   - **ポップアップ通知**: **リアルタイムアラート** ウィンドウが閉じている状態でも、ポップアップ形式で通知を表示します。
   - **解消済みイベントを表示**: **リアルタイムアラート** 一覧に解消済みイベントも含めて表示します。
   - **通知音の試聴**: ブラウザで再生される通知音を事前に確認できます。Critical または Warning の横にある をクリックしてください。音が再生されない場合は、ブラウザの設定をご確認ください。

システムイベント

WhaTap モニタリングは、プラットフォーム全体で自動的に検知・生成される共通イベントを提供します。本ドキュメントでは、システムによって発生するイベントの種類、発生条件、および対応方法について説明します。

EVENT_BLOCKED_BY_SYSTEM

プロジェクト内で短時間に過剰な数のイベントが発生した場合、WhaTap システムがこれを検知し、イベントの発生を一時的にブロックします。イベントがブロックされると、システムは次のメッセージをユーザーに通知します。このイベントが発生した場合は、イベント設定内容を確認し、必要に応じて調整してください。



ブロック基準

イベントは次の基準に基づいてブロックされます。

1. **エージェント単位のブロック**

   1つのエージェントで **1分以内に10件** のイベントが発生した場合、そのエージェントから発生する以降のイベントはブロックされます。

2. **複数エージェントによる全体ブロック**

   条件 **1** によりイベントがブロックされたエージェントが **1分以内に3件以上** 発生した場合、プロジェクト全体のイベントがブロックされます。

3. **プロジェクト単位のブロック**

   1つのプロジェクトで **1分以内に100件** のイベントが発生した場合、プロジェクト全体のイベントがブロックされます。

ブロック解除方法

システムによってイベントがブロックされた場合、対象のエージェントまたはプロジェクトで **1分間** イベントが発生しない状態を維持する必要があります。

- **アラート通知 > イベント設定** メニューから設定されているすべてのイベントを無効化してください。**イベント設定を見直し、過剰なイベントが発生しないように閾値を調整する必要があります。**
- イベントを設定する前に、**シミュレーション** 機能を活用してイベント発生回数を事前に予測できます。

インスタンス性能分析

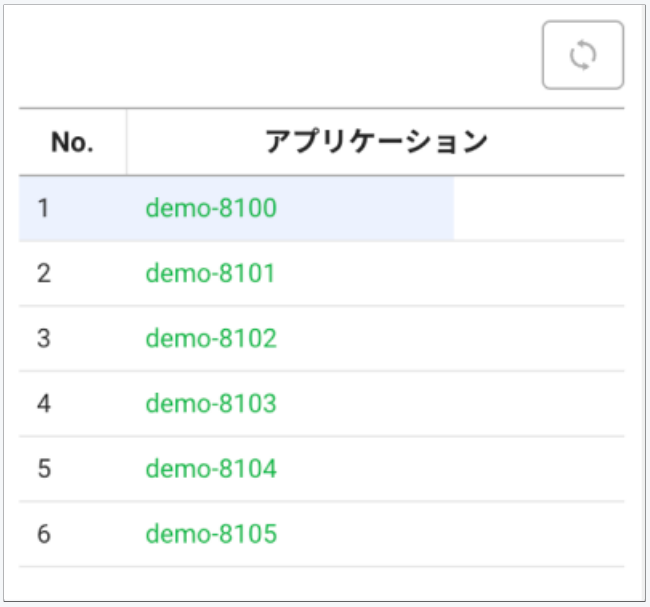
ホーム画面 > プロジェクト選択 > **インスタンス性能管理**

アプリケーションの環境を確認し、性能に関連する設定を確認できます。 エージェントおよびアプリケーションに関する詳細情報を確認できます。

- 右上のテキスト入力欄から希望する項目がフィルタリングできます。
- モニタリング対象サーバーに位置するエージェントおよびアプリケーションの変更が自動的に反映されない場合は、**リフレッシュ**ボタンを選択します。

エージェントリスト

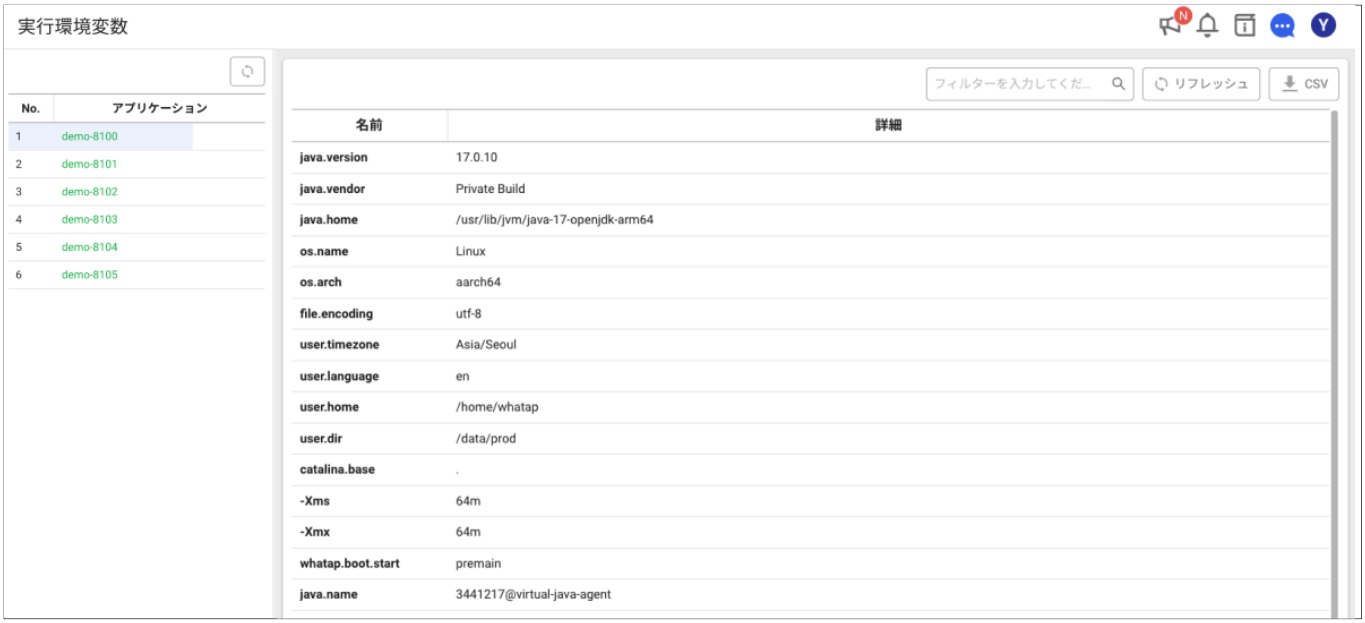
画面左側のアプリケーション一覧では、プロジェクトに割り当てられたエージェントの一覧を確認できます。 個々のエージェント項目を選択すると、右画面に、設定された環境変数、エージェント設定及びに性能に関する情報を照会できます。



エージェント一覧を更新するか、非活性化されたエージェントを再起動しても一覧に自動的に表示されない場合は、ボタンを選択します。

実行環境変数

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **実行環境変数**

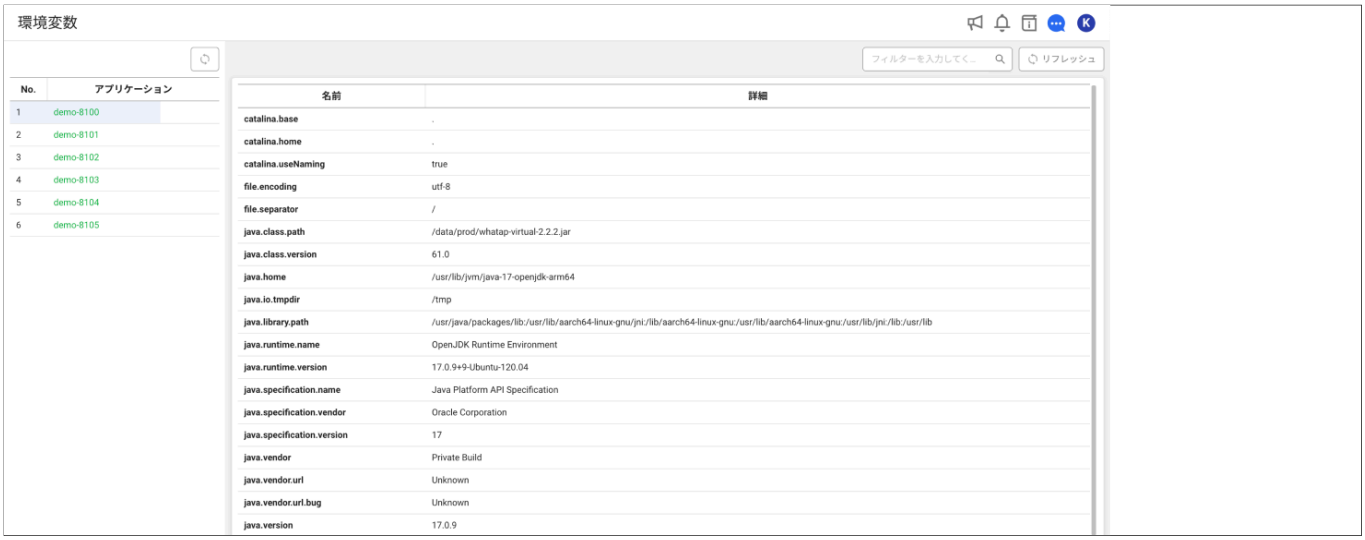


エージェントの実行に関連する環境変数を照会できます。 環境変数のうち、重要な情報またはアプリケーションの状態情報をWhaTapサーバーに保存し、エージェントがダウンしても照会が可能です。 エージェントバージョンやインストールパス、名前、IPアドレスなどを確認できます。 **CSV**ボタンをクリックすると、環境変数情報をダウンロードできます。 主な確認事項は次のとおりです。

- `(アプリケーション).version` : アプリケーションのバージョンを確認できます。
- `file.encoding` : `utf-8` といったファイルエンコーディング形式を確認できます。
- `os.name` : OSを確認できます。
- `user.language` : ユーザー言語を確認できます。
- `-Xms` , `-Xms` : ヒープメモリ設定に関連する環境変数を確認できます。
- `(アプリケーション).start` : アプリケーションの開始時間を確認できます。

環境変数

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **環境変数**

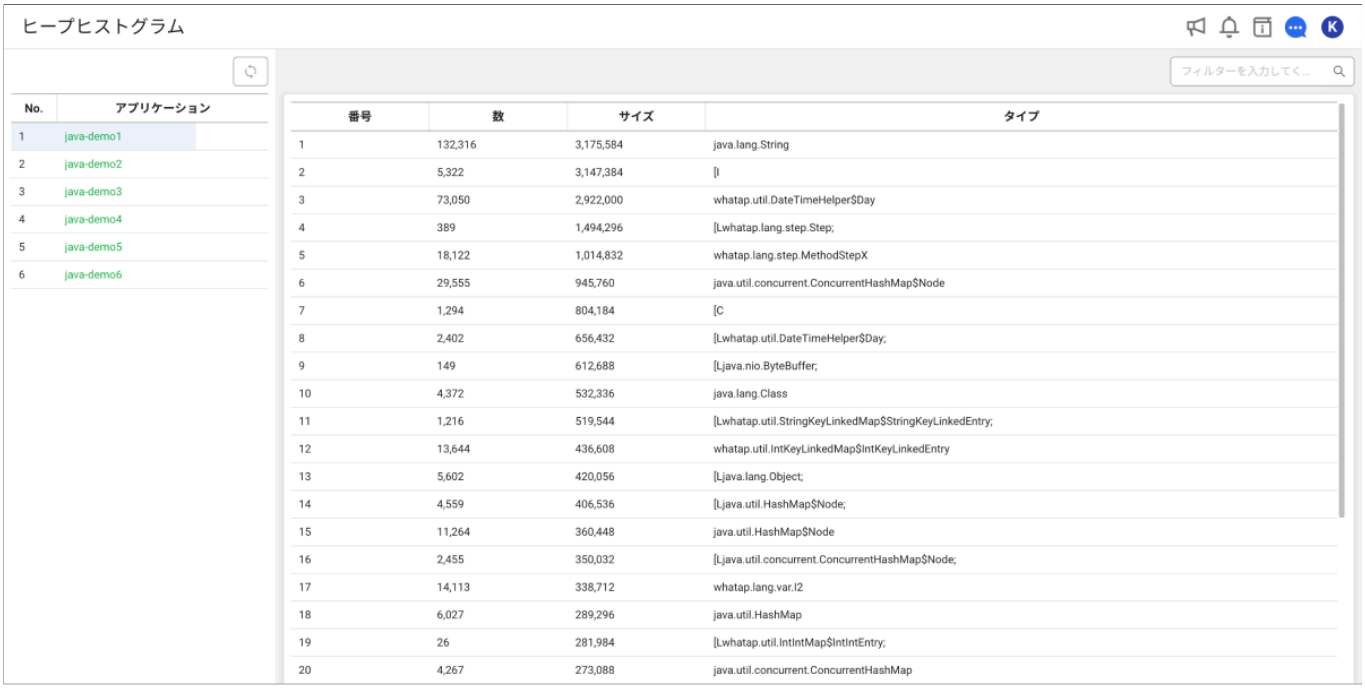


アプリケーションの実行に関連する環境変数情報を照会できます。 **環境変数**メニューは、照会時点の環境変数情報を収集して提供します。 `System.getProperty()` を使用して、パラメータとして渡されたキーの値を照会します。 主な確認事項は次のとおりです。

- `(アプリケーション).version` : アプリケーションのバージョンを確認できます。
- `file.encoding` : `utf-8` といったファイルエンコーディング形式を確認できます。
- アプリケーションステータス情報: 例えば `catalina.home` 環境変数が照会される場合、tomcat環境であることが把握できます。
- OS環境変数: 例えば `(env) LANG` 環境変数のように接尾辞 `(env)` 形式で提供されます。

ヒープヒストグラム

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **ヒープヒストグラム**



JVM(Java仮想マシン)のメモリにロードされているヒープ占有オブジェクトの状況(ヒープメモリ上のオブジェクト別サイズ)を照会することができます。 つまり、オブジェクトそのもののサイズを照会します。 **数**カラムを通じて主要オブジェクトの数を確認することが容易です。

JVMオプション

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

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

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

example

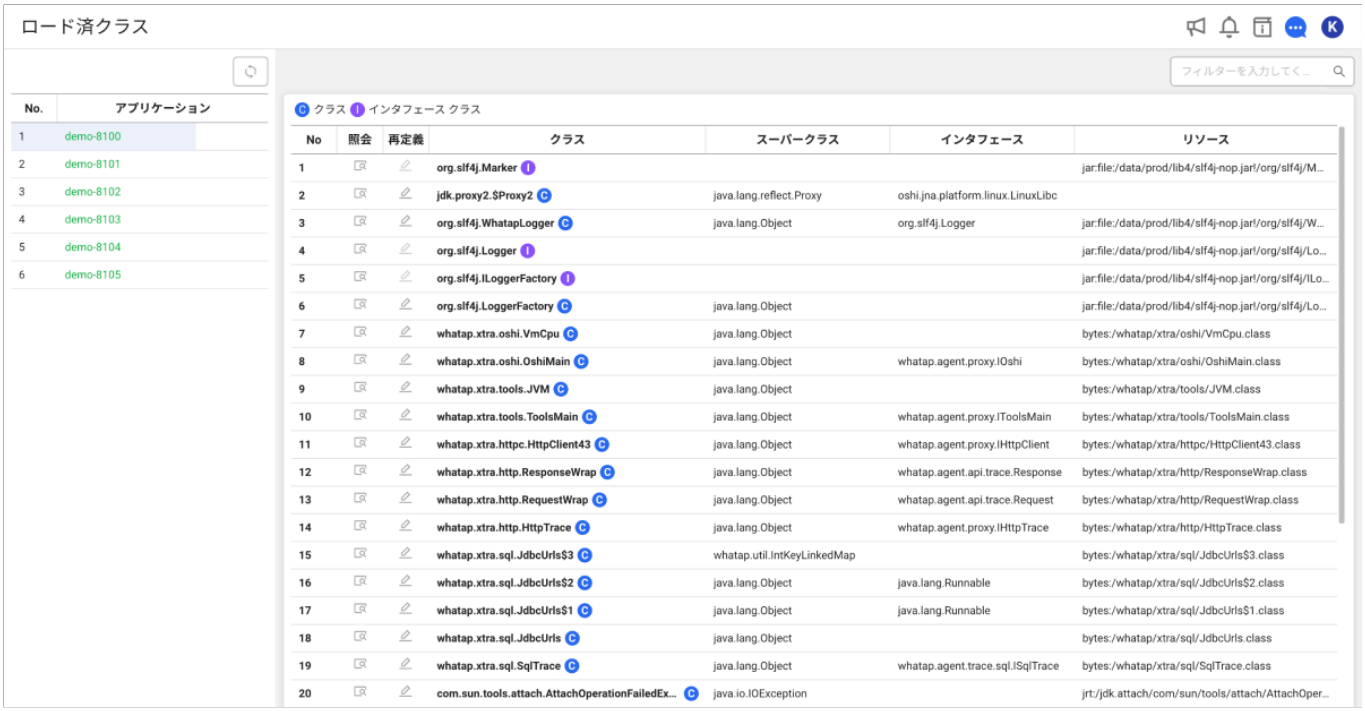
```
java -javaagent:{WHATAP_HOME}/whatap.agent-X.Y.Z.jar -Djdk.attach.allowAttachSelf=true --add-opens=java.base/
java.lang=ALL-UNNAMED --add-opens=jdk.attach/sun.tools.attach=ALL-UNNAMED -jar {application.jar}
```

- Java 5以前の旧バージョン、IBM Javaには対応していません。

> ⓘ GC前まで維持されるオブジェクトの数を照会するため、実際に使用されていなくても、該当一覧には表示されます。 正確な照会のために**システムGC**実行後にその一覧を確認することをお勧めします。 **システムGC**については、次の文書を参照してください。

ロード済クラス

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **ロード済クラス**



Javaアプリケーションにロードされたクラス情報を確認できます。 **ロード済クラス**メニューを通じてアプリケーションでロードされたクラスリストと関連詳細データを照会し、中断なくクラスの動作を変更できます。

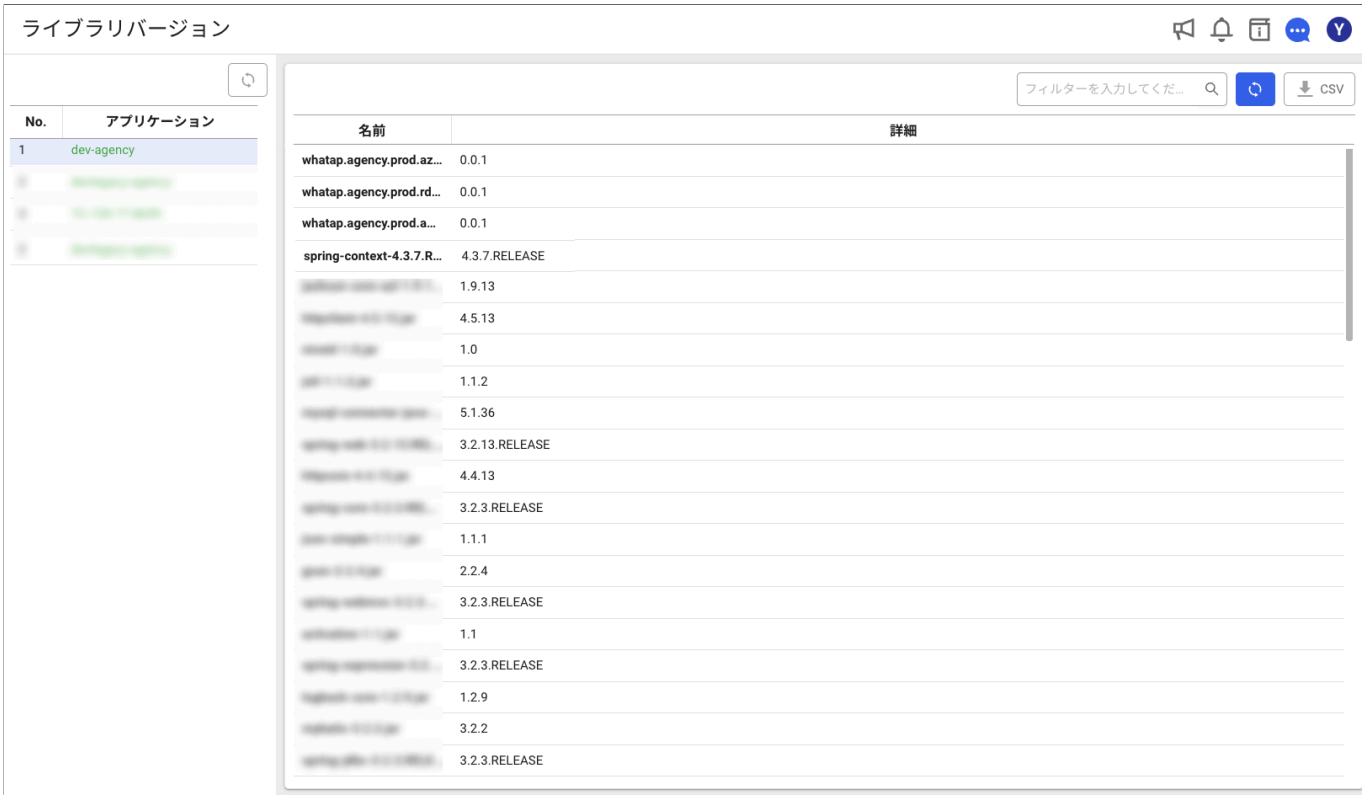
- **クラス詳細**: **照会**カラムでアイコンをクリックすると**クラス詳細**を確認できます。 **クラス詳細**は、クラス内のメソッド及び継承関係情報を提供します。
- **再定義**: **再定義**カラムでアイコンをクリックすると、そのクラスを再定義できます。 ランタイムに既にロードされたクラスを再びロードした後、バイトコード挿入(BCI)を通じてアプリケーションを中断することなく、該当クラスの動作変更が可能です。
- **リソース**: そのクラスが物理的にどのファイルからロードされたかを確認できます。 アプリケーションの複雑な多階層構造の特性上、クラスが正確にどのjarファイルからロードされたかを把握する**リソース**情報の確認が必要です。

> **再定義**機能と**リソース**情報確認活用の詳細は、次の文書を参照してください。

> **BCI(ByteCode Instrumentation)**
>
> Javaバイトコードは、プラットフォームに独立的な中間コードでJVMで実行されます。 **BCI**つまり、バイトコード挿入は、ソースファイルの修正なしでランタイムまたはビルドタイムにバイトコードを動的に変更して追加機能を実装する技術です。 詳細については、次の文書を参照してください。
>
> - Java SE Specifications
> - Java Package Summary

ライブラリバージョン

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **ライブラリバージョン**

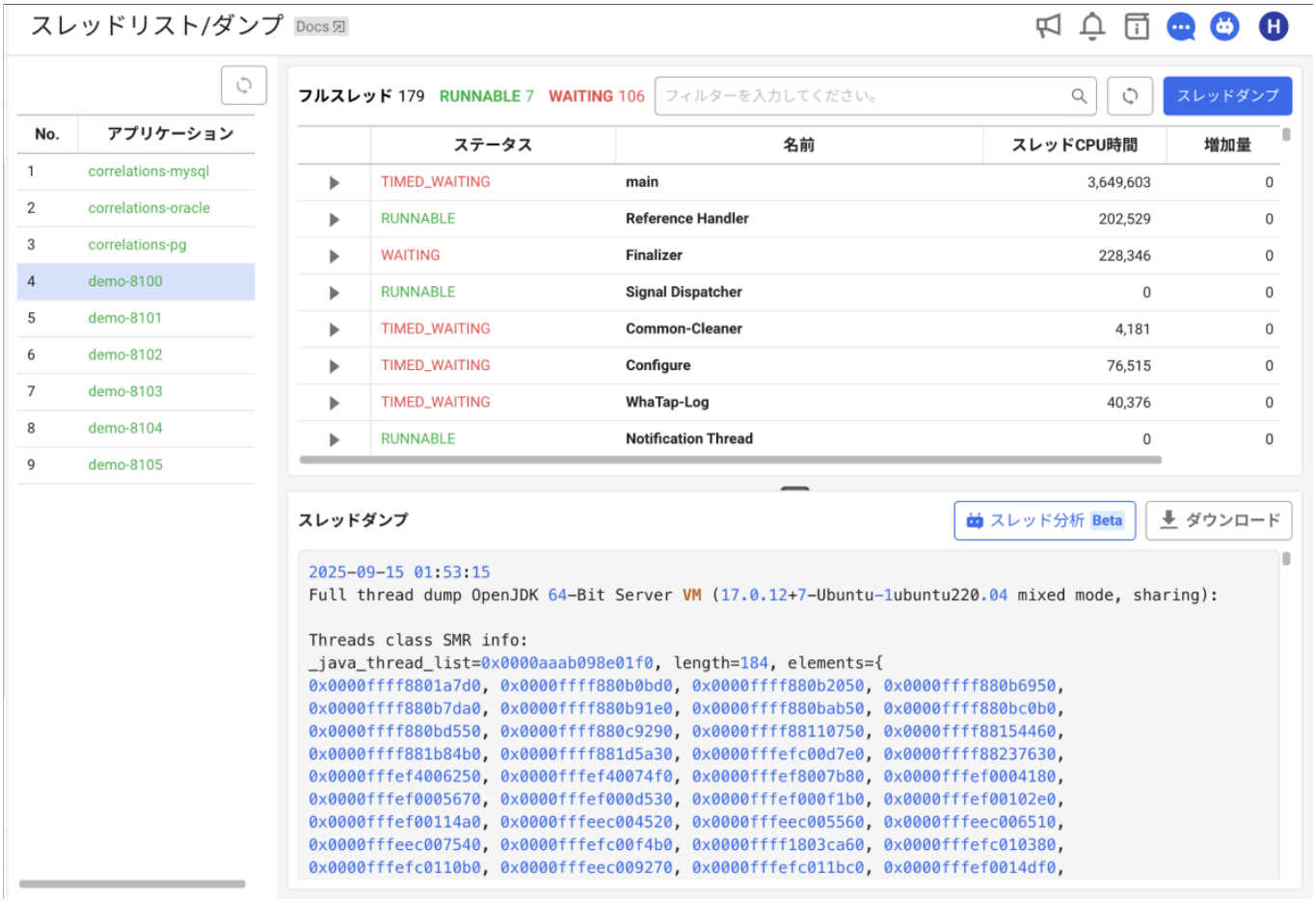


Javaアプリケーションに配布されたコンポーネントのリストを照会できます。 オープンソースライブラリを多く使用する環境では、jarファイルのバージョン情報を確認し、互換性の問題を迅速に識別して解決できます。 また、バージョン情報の確認は、セキュリティの脆弱性があるバージョンを見つけてパッチするなど、セキュリティ管理にも容易です。

- コンポーネント一覧情報: 当該jarファイルの**名前**及びバージョン情報を**詳細**で提供します。
- **CSV**: ダウンロードボタンをクリックして、バージョン情報をダウンロードできます。

スレッドリスト/ダンプ

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **スレッドリスト/ダンプ**



Javaプロセス内のすべてのスレッド状態のスナップショット情報を提供します。 **スレッドリスト/ダンプ**はCPU最適化作業において重要なメニューです。 プロセスではなくスレッド単位で状態を確認することで、CPUリソースをより細かく管理できます。 例えば、照会時点でCPUを最も多く使用しているスレッドを特定した後、該当スレッドのスタック情報を確認して性能問題を識別できます。

- スレッドステータス: **全体スレッド**数およびRUNNABLE、WAITING状態のスレッド数を一覧の上段にある状況要約で確認できます。
- **スレッドダンプ**: 上段右側の スレッドダンプ ボタンをクリックして、全体のスレッドダンプを作成できます。 スレッド全体のダンプを作成した後、スレッド一覧の下段でその情報を確認できます。
- 個別スレッドダンプ: 各スレッドごとに ▸ アイコンを選択すると、個々のスレッドダンプ情報を確認できます。 **スタック**及び**スタックトレース**で、スレッドの状態と実行パスに関する詳細及びスタックトレースを照会できます。 更新して確認できます。 詳細については次の文書を参照してください。
- **スレッドCPU時間**: 該当スレッドがCPUを占有した累積時間を意味します。 ミリ秒単位の時間を使用します。

- **増加量**: スレッドリストの上段で更新アイコンをクリックすると、既存のスレッドのCPU時間と現在のスレッドのCPU時間の差値が追加されます。

> **スレッドCPU時間**は、当該スレッドがCPUを占有した総累積時間であるため、照会時点で最も多くCPUを使用しているスレッドを特定するためには**増加量**値の確認が必要です。 詳細については、次の文書を参照してください。

> **CPU使用量急増タイプ**
>
> - ヒープメモリがいっぱいの場合: 頻繁なGCによりCPU使用量が急増
> - 実際のリクエストが多い場合: 大量のクライアントリクエスト処理によりCPU使用量が急増
> - 誤ったコードロジック: 非効率的なアルゴリズムや無限ループによりCPU使用量が急増

スレッドダンプの詳細

スレッドリスト/ダンプメニューで確認できる個々のスレッドダンプ情報について案内します。

JVMオプション

スレッドダンプ機能を各Javaバージョンごとに使用するためには、追加のJVMオプション設定が必要です。

Java 6~8バージョンでは、JVMオプションなしで標準対応しますが、一部のJavaバージョンによっては次のようにJVMオプションを適用する必要があります。

- Java 9 ~ Java 15バージョン

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

example

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

- Java 16バージョン以降

```
-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以前の旧バージョン、IBM Javaには対応していません。

スレッド状態

スレッドの状態は、`java.lang.Thread`クラス内に`State`という名前のEnumerated Types(列挙型)と宣言された項目です。

- `NEW`：スレッドは作成されましたが、まだ実行されていません。
- `RUNNABLE`：現在CPUを占有し、作業を行っている状態です。 OSのリソース分配により`WAITING`状態になる可能性があります。
- `WAITING`：`wait()`、`join()`、`park()`メソッドなどを利用して待機している状態です。
- `TIMED_WAITING`：`sleep()`、`wait()`、`join()`、`park()`メソッド等を用いて待機している状態、`WAITING`状態と異なる点は、与えられた時間に待機している状態です。 外部的な変化だけでなく、時間によっても待機状態が解除されることがあります。
- `BLOCKED`：使用するオブジェクトのロック(Lock)が解除されるまで待機している状態です。
- `TERMINATED`：実行済みの状態です。

スレッドダンプ情報

スレッド一覧で各スレッドの ▸ ボタンをクリックすると、個別スレッドダンプ情報を確認できます。



スタック

項目	プロパティ	説明
スレッドID	threadId	スレッドに割り当てられた固有ID
ロック所有者ID	lockOwnerId	スレッドがブロックされたオブジェクトのモニター(Monitor)のロックを維持するスレッドのID
待ち件数	waitedCount	スレッドがWAITINGまたはTIMED_WAITING状態になった総回数
ブロック数	blockedCount	スレッドがBLOCKED状態になった総回数
待ち時間	waitedTime	スレッドがWAITING状態を継続した経過時間(ミリ秒)、スレッドの競合モニタリングが非活性化された場合は-1を返す
スレッドCPU時間	threadCpuTime	スレッドのCPU時間合計(ナノ秒)をミリ秒で計算して表示、CPU時間測定を非活性化した場合は-1を返す
ステータス	threadstate	スレッド状態
ブロック時間	blockedTime	スレッドがBLOCKED状態になってからの経過時間(ミリ秒)、スレッドの競合モニタリングが非活性化された場合は-1を返す
スレッド名	threadName	スレッドの固有名
ロック名	lockName	スレッドの入力がブロックされたりObject.waitメソッドを介して通知を待つモニターロックを表示する文字列
ロック所有者名	lockOwnerName	スレッドがブロックされるオブジェクトのモニターロックを受け入れるスレッド名
スレッドユーザー時間	threadUserTime	スレッドがユーザーモードで実行したCPU時間(ナノ秒)をミリ秒で計算して表示

ⓘ スレッド情報の詳細については次のリンクを参考にしてください。

スタックトレース

例外が発生した時のスタック(Stack) 追跡を表示します。 スタックフレーム一覧には、コードが呼び出したメソッド情報が含まれています。

AI スレッド分析

スレッドダンプ詳細画面で **スレッド分析** ボタンをクリックします。

AI による分析結果はドロワー形式で表示されます。

分析結果項目

- サマリー
- 総スレッド数
- スレッド状態別の分布
- 問題点
- 推奨事項

スレッド分析の再試行

ドロワー上部の ボタンをクリックすると、同じスレッドダンプを再分析できます。

分析結果は WhaTap ページで設定された言語で表示されます。
一部の場合、英語で表示されることがあります。

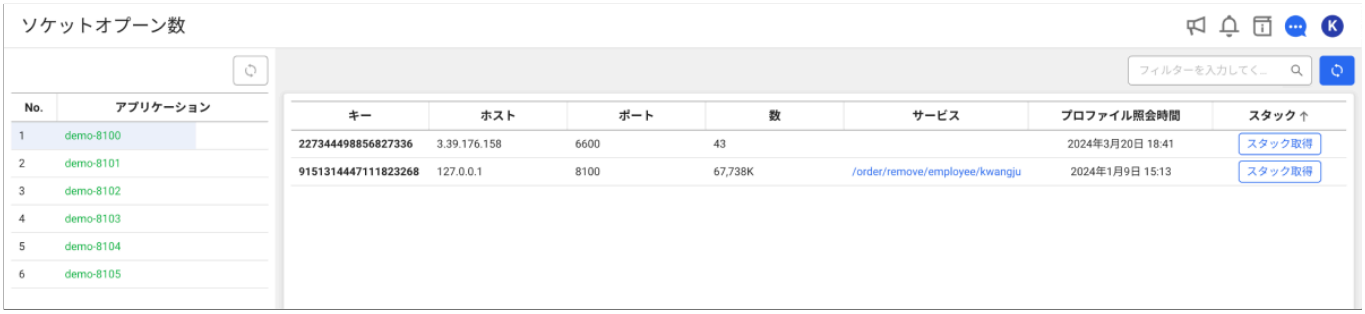
プロジェクトごとに1日最大20回まで分析でき、プロジェクトのタイムゾーンを基準に毎日00:00:00にリセットされます。

- 分析回数の消費基準
  - 新しいスレッドダンプを分析する場合、1 回分消費されます
    - アプリケーション選択状態が変更された場合
      - アプリケーションリストから別のアプリケーションを選択した場合
    - 時刻が変更された場合

    - 別のアプリケーションを選択した場合（選択状態と時刻が変更）
    - 同じアプリケーションを再度選択した場合（選択状態は維持されるが、時刻が変更されスレッドダンプが更新）
- ドロワー内で同じスレッドダンプを再分析しても 1 回分消費されます

ソケットオプーン数

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **ソケットオプーン数**



JavaアプリケーションがTCP機能を実行するためにオープンしたソケット(Socket)情報が確認できます。これにより、アプリケーションインスタンスがデータベースなど、どの外部システムに依存するか、ネットワーク依存性(Network Dependency)を照会できます。 また、誰とアウトバウンドコールを行っているかを追跡できます。 これらの情報は、TCP通信の効率性、安定性、及び性能最適化のために重要です。

- **数**: 外部システムへの接続試行回数を意味します。
- **スタック**: **スタック取得**を選択して、次の接続時に確保したスタックを照会・削除できます。

数基準にソートした後、**ホスト**及び**ポート**を確認して、どのようなサービスまたはシステムとの接続が多いかを確認できます。 以後、**スタック**を確保して詳細を照会してみてください。

> **データベースとコネクションプール**
>
> 例えば、3306**ポート**の**数**が増加時に、コネクションプールを使用しなかったり、あるいはコネクションプールを頻繁に切って作り直すなど、MySQLのコネクション管理が正しく行われていない状況である可能性があります。 該当3306**ポート**がどのようなコード経路で使用されているかを把握するため**スタック**を確保して追跡できます。

メソッド性能状態

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **メソッド性能状態**

Javaアプリケーションで実行されたメソッド(Method)の性能統計情報を提供します。

- **リフレッシュ**: データを再呼び出しします。
- **リセット**: 登録されたメソッド統計データを初期化します。

> **メソッド性能状態**メニューを使うには、**エージェント設定**メニューから次のカスタムオプションを追加します。 そして再起動または**ロード済クラス**メニューから**再定義**します。
>
> ```
> hook_method_stat_patterns=io.home.test.baseapp.app.post.mapper.PostMapperImpl.*
> ```

データソース状態

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **データソース状態**

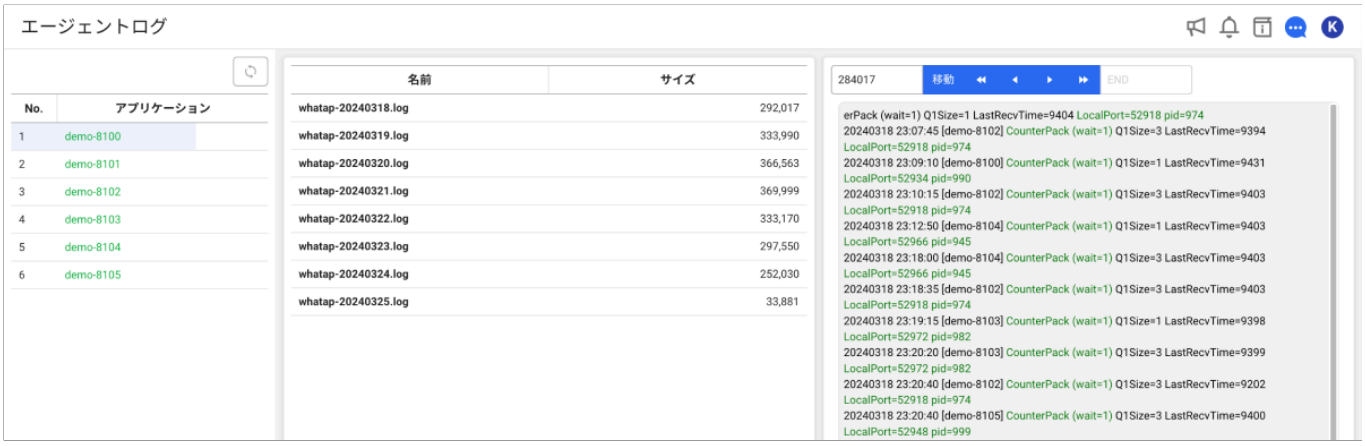


データソース(DataSource)の状態を確認できます。 アプリケーションが使用するデータベースコネクションプールの現在の状態と構成、接続設定情報、接続状態と設定の不一致などの情報を提供します。

> コネクションプールオブジェクトの生成と同時にモニタリングが開始されるため、そのオブジェクトが使用されていなくても、データソースの状態情報に表示される可能性があります。

エージェントログ

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **エージェントログ**



モニタリング対象サーバーに保存されているエージェントログを照会できます。 ログのファイル名は、whatap-`YYYYMMDD`.log形式です。 各ログを選択して、ログに含まれてあるエラーやイベントに関する情報が確認できます。

> ⓘ ログに関連するエージェント設定は、次の文書を参照してください。

システムGC

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **システムGC**



`System.gc()` を呼び出してJVM(Java仮想マシン)のヒープ領域で動的に割り当てたメモリのうち、不要になったメモリ

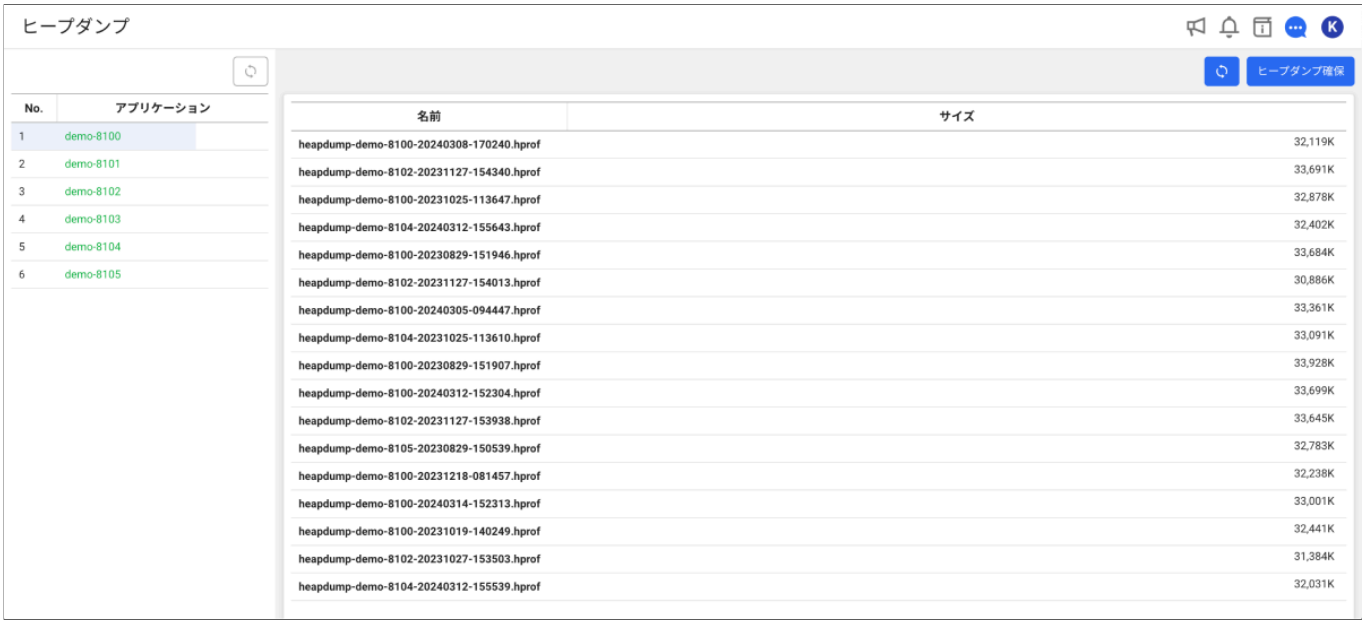
オブジェクト(garbage)を集めて削除できます。 **GC実行**ボタンを選択するとGCプロセスが実行され、実行前後のメモリ容量を確認できます。

> `-XX:+DisableExplicitGC` のように明示的なGCリクエストを非活性化するオプションが設定されている場合、**システムGC**が動作しません。

> **システムGCとメモリリーク**
>
> Heapメモリ使用量が持続的に上昇した場合、メモリリークの可能性を確認し、正確に診断するため**システムGC**が必要です。
>
> - **メモリリクエスト有無**: アプリケーションが継続的にメモリをリクエストしている状態であることを確認します。 メモリリクエストの増加が予想値以上であるか、またはメモリ解除が正確に行われているかを調べます。
> - **メモリ使用量の確認**: 実際にメモリが過剰に使用されているかを確認します。 GC実行により不要なオブジェクトを解除し、メモリ使用量の変化を調べます。
> - **GC周期**: メモリが十分に空いている状況でも、GCが実行されていない状態であるかを確認します。 GC周期または設定を調べます。

ヒープダンプ

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **ヒープダンプ**

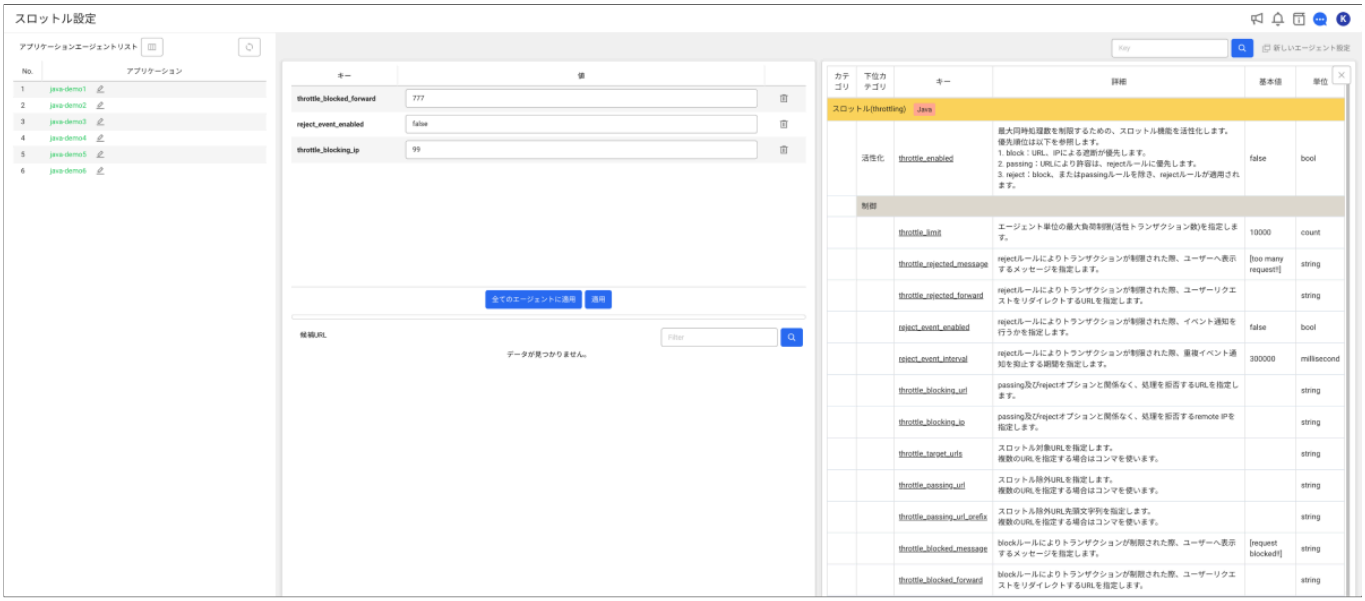


Javaアプリケーションの実行中にメモリ漏れなどの問題が発生すると、関連した問題を整理してダンプファイルを作成できます。 **ヒープダンプ確保**ボタンを選択します。 生成されたダンプファイルは、アプリケーションサーバーに保存されます。 エージェントをインストールしたパス下位のdumpディレクトリから確認できます。

> この機能は、Javaエージェント1.5.2バージョン以降に対応しています。 エージェント設定で `heapdump_enabled` オプションを `true` に設定してください。 **ヒープダンプ確保**ボタンを選択して、ヒープダンプファイルを作成する機能を活性化します。 毎回ヒープダンプを自動作成するわけではなく、そのオプションが性能に影響を与えることはありません。 ただし、ダンプの確保作業は性能に影響を与える可能性があります。
>
> **ヒープダンプの確保**作業は、**負荷**の大きい作業だと言えます。 負荷とは別に、ヒープダンプがすべて抽出されるまでアプリケーションのコードは実行中止(stop the world)になるため、ユーザーが感じる性能の差はさらに激しくなる可能性があります。

スロットル設定

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **スロットル設定**



Javaアプリケーションの負荷量制御に関連するエージェント設定を確認し、オプションを設定できます。 負荷量制御に関するエージェント設定の詳細については、次の文書を参照してください。

エージェント設定 *Old*

ホーム画面 > プロジェクト選択 > **インスタンス性能管理** > **エージェント設定** *Old*

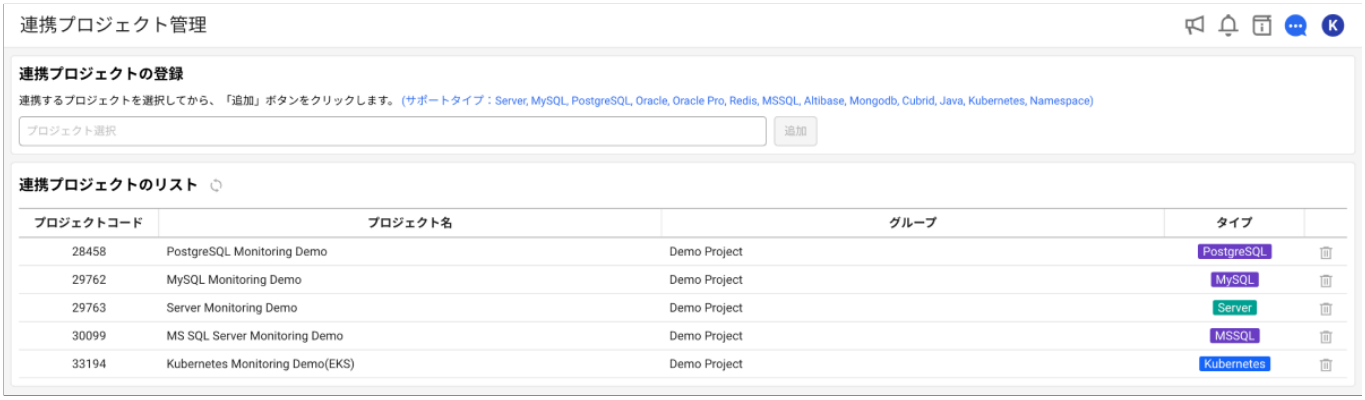
モニタリング対象サーバーに位置するwhatap.confファイルを直接修正することなく、エージェント設定オプションを追加、変更、または削除できます。 **設定ファイルのダウンロード**ボタンをクリックする時に選択したエージェントのwhatap.confファイルをダウンロードします。

> ⓘ エージェント設定機能は、使用性と機能を改善した**管理** > **エージェント設定**メニューの利用をお勧めします。 画面右上の**新しいエージェント設定**ボタンを選択します。 詳細については、次の文書を参照してください。

リンクされたプロジェクト

ホーム画面 > プロジェクト選択 > **管理** > **連携プロジェクト管理**

データベースモニタリング(DPM) プロジェクトとアプリケーションのモニタリング(APM) プロジェクトを生成した場合、2つのプロジェクトを連携してDPMで収集したモニタリングデータをAPMプロジェクトで確認できます。



連携プロジェクト管理機能を使用すると、製品または機器中心のモニタリングを超えて、複数のプロジェクト間のモニタリングデータを統合して確認できます。 これは、アプリケーション、データベースなど、 様々なシステムコンポーネントがどのように相互作用するかを分析するのに役立ちます。 特に、性能低下が発生した時に装備の問題なのか個別アプリケーションの問題なのか迅速に把握でき、問題解決時間を大幅に短縮できます。 **連携プロジェクト管理**これにより、ユーザーシステム単位で性能データを総合的に分析し、より可視的なインサイトを確保できます。

詳細については、次の動画を通じて確認できます。

> **開始前の参考事項**
>
> - **連携プロジェクト**機能をサポートするデータベースプラットフォームは次の通りです。
>
> PostgreSQL, Oracle, MySQL, SQL Server, CUBRID, Altibase, Redis, MongoDB
>
> - **連携DBセッション**機能の場合、Javaエージェント2.2.33バージョン以降が必要です。 この機能をサポートするデータベースプラットフォームは次の通りです。
>
> PostgreSQL, Oracle, Oracle Pro, MySQL

権限設定

OracleとOralce Proプラットフォームの場合、連携プロジェクトを追加するためには、次のような権限設定が必要です。

```
grant select on sys.v_$session to DB接続アカウント;
# ex) grant select on sys.v_$session to petapp;
```

連携プロジェクトを追加する

1. **管理** > **連携プロジェクト管理**メニューに移動します。
2. **連携プロジェクトの登録**セクションで**プロジェクト選択**入力欄を選択します。



3. 選択できるプロジェクトの一覧が表示されたら、連携するプロジェクトを選択します。 文字列を入力して、一致するプロジェクトを検索もできます。
4. プロジェクトを1つ以上選択した後に有効になった**追加**ボタンを選択します。

連携プロジェクトのリストに選択したプロジェクトが追加されます。

連携プロジェクトを追加するためには、プロジェクト**修正**権限が必要です。 **修正**権限がある場合**連携プロジェクト管理**メニューで**追加**ボタンが有効になります。

エージェントオプションを設定する

データベースプロジェクトとの連携分析のため、以下のようにエージェント設定ファイル(whatap.conf)にオプションを設定する必要があります。 または、**管理** > **エージェント設定**メニューで設定できます。

whatap.conf

```
dbc_sid_enabled=true
hook_db_native_enabled=true
debug_dbc_sid_enabled=true
```

- **dbc_sid_enabled** bool

  既定値 `false`

  SID発行のためのオプションです。 連携分析のために値を `true` に設定します。 **必須**オプションです。

- **hook_db_native_enabled** bool

  既定値 `false`

  トランザクションで使用するDBの種類を抽出するためのオプションです。 連携分析のために値を `true` に設定します。 **必須**オプションです。

- **debug_dbc_sid_enabled** bool

  既定値 `false`

  JavaエージェントログにSID関連情報をロギングするオプションで、選択オプションです。

> ⓘ **管理** > **エージェント設定**



連携プロジェクトのメトリクスを確認する

追加した連携プロジェクトに関連するメトリクスは、次のメニューパスで確認できます。

- **ダッシュボード > アプリケーション**
- **ダッシュボード > アクティブTX**
- **分析 > ヒットマップ**

- **アプリケーション**メニューについての詳細は、次の文書を参照してください。
- **アクティブTX**メニューについての詳細は、次の文書を参照してください。
- **ヒットマップ**メニューについての詳細は、次の文書を参照してください。

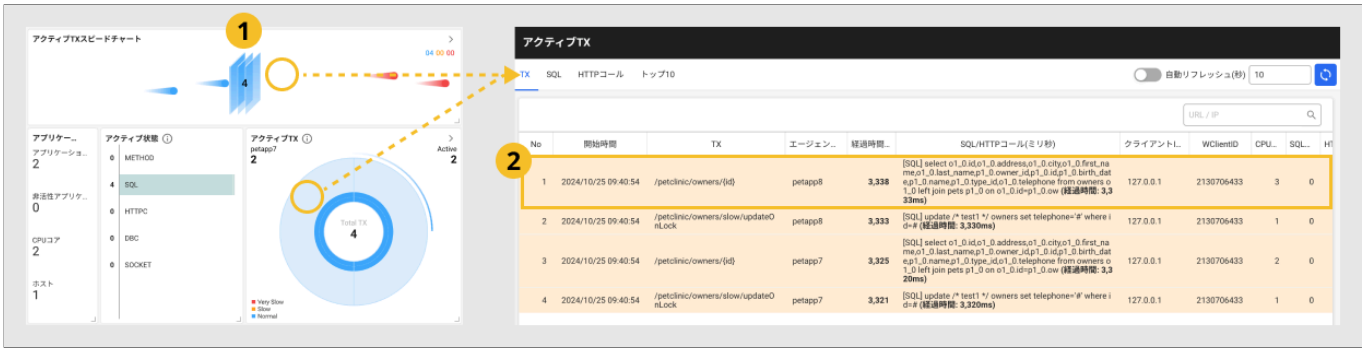
アプリケーション

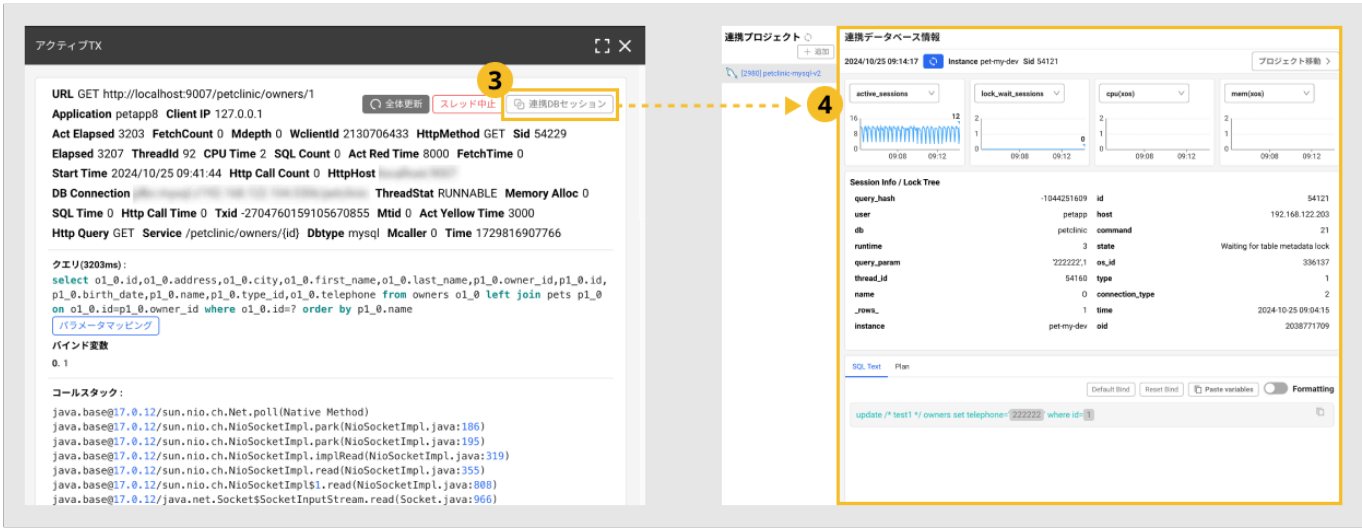
アプリケーションメニューでは、次のウィジェットで連携プロジェクトを確認できます。

- **アクティブTXスピードチャート**
- **アクティブTX**
- **ヒットマップ**

アクティブTXスピードチャート / アクティブTXウィジェット

現在進行中のトランザクションと連携されたデータベースのSQL性能を照会できます。 **ダッシュボード** > **アプリケーション**メニューに移動します。





1. ダッシュボードで**アクティブTXスピードチャート**または**アクティブTX**ウィジェットのチャート領域をクリックします。
2. **アクティブTX**画面が表示されたら、トランザクション項目を選択します。
3. トランザクションの詳細画面が表示されると、右上の**連携DBセッション**ボタンを選択します。
4. **連携データベース情報**画面が表示されたら、関連するメトリクスを確認します。

> ⓘ • 連携分析のために登録したデータベースプロジェクトのDBが現在選択したトランザクションのSQLを実

行した場合のみ、トランザクション詳細画面に**連携DBセッション**ボタンが表示されます。

- **連携データベース情報**画面についての詳細は、次の文書を参照してください。
- **Javaエージェント2.2.40**以前のバージョンでは**アクティブTX**ウィンドウで**ThreadStat**値が数字値形式で表示されます。 文字形式で表示するにはJavaエージェント2.2.40バージョン以降をインストールします。
- この機能をサポートするデータベースは、**PostgreSQL**、**Oracle**、**Oracle Pro**、**MySQL**プラットフォームです。

ヒットマップウィジェット

過去の時点でのトランザクションに連携されたデータベースの性能メトリクスを照会できます。 **ダッシュボード** > **アプリケーション**メニューに移動します。

DB Connectionステップで確認する

Correlated project

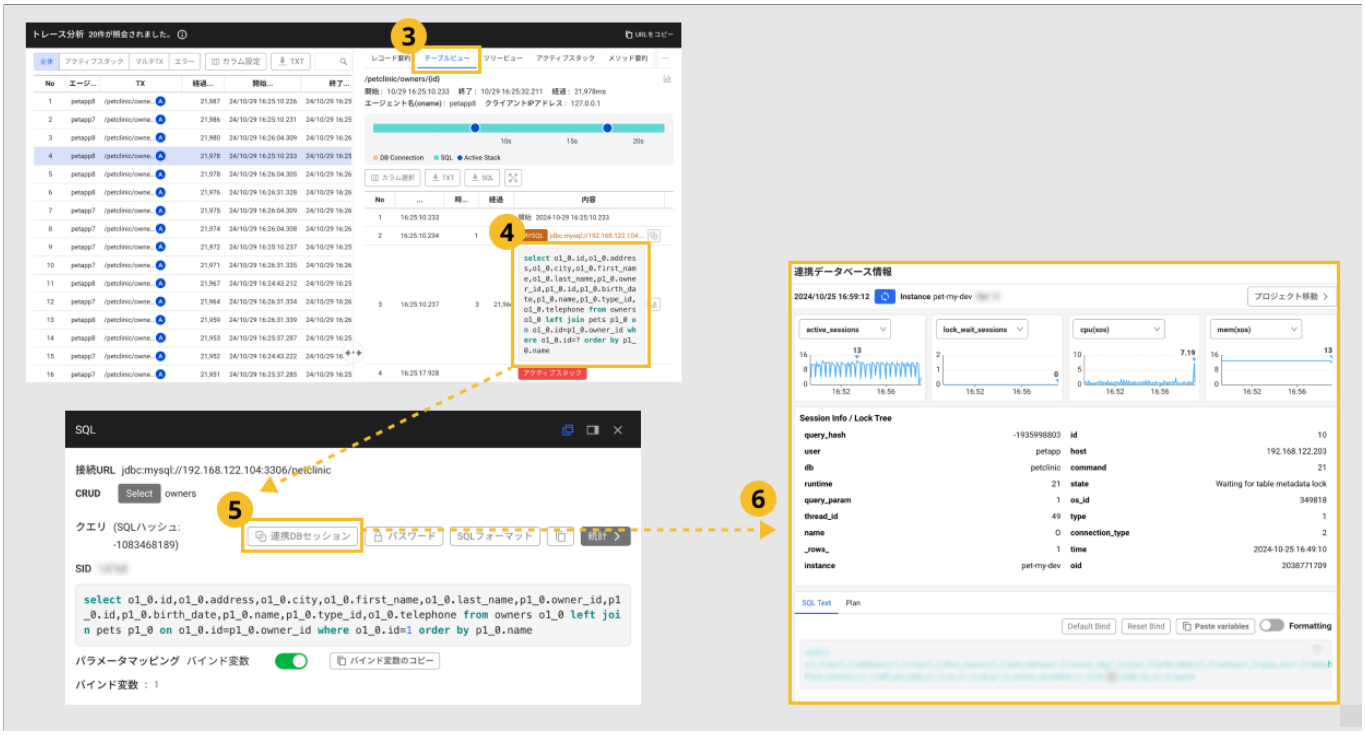


1. ダッシュボードで**ヒットマップ**ウィジェットのチャート領域をドラッグします。
2. **トレース分析**画面が表示されたら、照会したいトランザクション項目を選択します。
3. トランザクションの詳細情報が右側に表示されると、**テーブルビュー**タブを選択します。
4. DB Connectionステップでボタンを選択します。
5. 連携プロジェクトの**インスタンスダッシュボード**画面が表示されたら、関連するメトリクスを確認します。

- 連携分析のために登録したデータベースプロジェクトのDBが現在選択したトランザクションのSQLを実行した場合のみ、**テーブルビュー**タブにボタンが表示されます。
- **インスタンスダッシュボード**画面についての詳細は、次の文書を参照してください。
- **トレース分析**画面についての詳細は、次の文書を参照してください。
- この機能をサポートするデータベースは、**PostgreSQL**、**Oracle**、**Oracle Pro**、**MySQL**プラットフォームです。

SQLステップで確認する

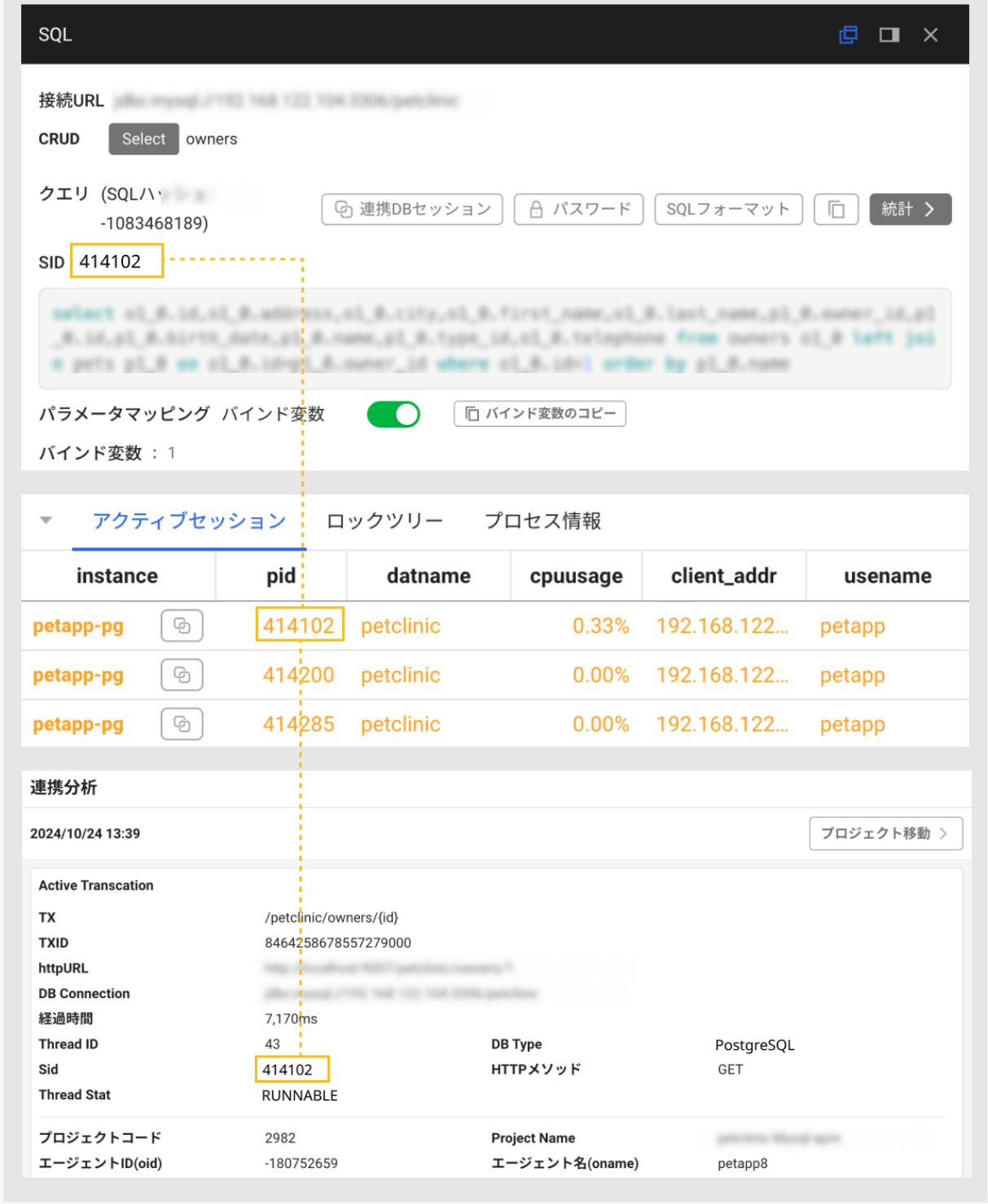
Correlated project



1. ダッシュボードで**ヒットマップ**ウィジェットのチャート領域をドラッグします。
2. **トレース分析**画面が表示されたら、照会したいトランザクション項目を選択します。
3. トランザクションの詳細情報が右側に表示されると、**テーブルビュー**タブを選択します。
4. SQLステップでSQL文を選択します。
5. **SQL**画面が表示されたら、**連携DBセッション**ボタンを選択します。

6. **連携データベース情報**画面が表示されたら、関連するメトリクスを確認します。

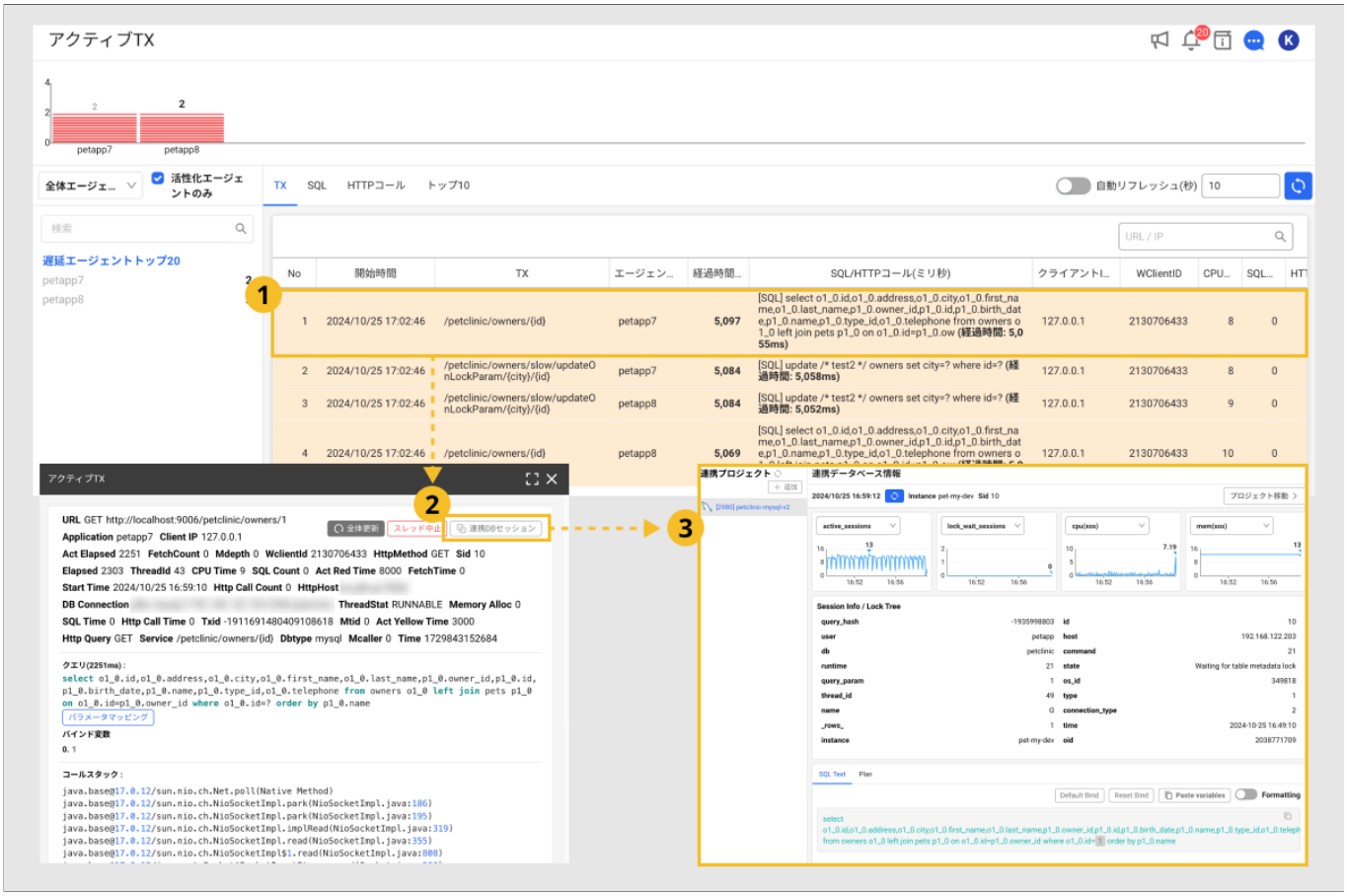
- 連携分析のために登録したデータベースプロジェクトのDBが現在選択したトランザクションのSQLを実行した場合のみ、**SQL**画面に**連携DBセッション**ボタンが表示されます。
- **バインド変数のコピー**機能を利用すると変数をコピーして**連携データベース情報**ウィンドウの**SQL Text**タブに変数を貼り付けることができます。 この機能は**パスワード**ボタンを選択してセキュリティキーを入力した後、バインド変数の値が表示されるときに利用できます。 詳細については、次の文書を参照してください。
- **連携データベース情報**画面についての詳細は、次の文書を参照してください。
- この機能をサポートするデータベースは、**PostgreSQL**、**Oracle**、**Oracle Pro**、**MySQL**プラットフォームです。
- **PostgreSQL**プラットホームで**アクティブセッション**テーブルの**PID**項目は、**SID**項目と同じです。


SQL
接続URL
CRUD
Select
owners
クエリ (SQLハッシュ
-1083468189)
連携DBセッション
パスワード
SQLフォーマット
統計
SID
414102
パラメータマッピング バインド変数
バインド変数のコピー
バインド変数 : 1
アクティブセッション
ロックツリー
プロセス情報
instance
pid
datname
cpuusage
client_addr
usename
petapp-pg
414102
petclinic
0.33%
192.168.122...
petapp
petapp-pg
414200
petclinic
0.00%
192.168.122...
petapp
petapp-pg
414285
petclinic
0.00%
192.168.122...
petapp
連携分析
2024/10/24 13:39
プロジェクト移動
Active Transcation
TX
/petclinic/owners/{id}
TXID
8464258678557279000
httpURL
DB Connection
経過時間
7,170ms
Thread ID
43
DB Type
PostgreSQL
Sid
414102
HTTPメソッド
GET
Thread Stat
RUNNABLE
プロジェクトコード
2982
Project Name
エージェントID(oid)
-180752659
エージェント名(oname)
petapp8

> **Javaエージェント2.2.40**以前のバージョンでは**アクティブTX**ウィンドウで**ThreadStat**値が数字値形式で表示されます。 文字形式で表示するにはJavaエージェント2.2.40バージョン以降をインストールします。

アクティブTX

現在進行中のトランザクションと連携されたデータベースのSQL性能を照会できます。 **ダッシュボード** > **アクティブTX**メニューに移動します。

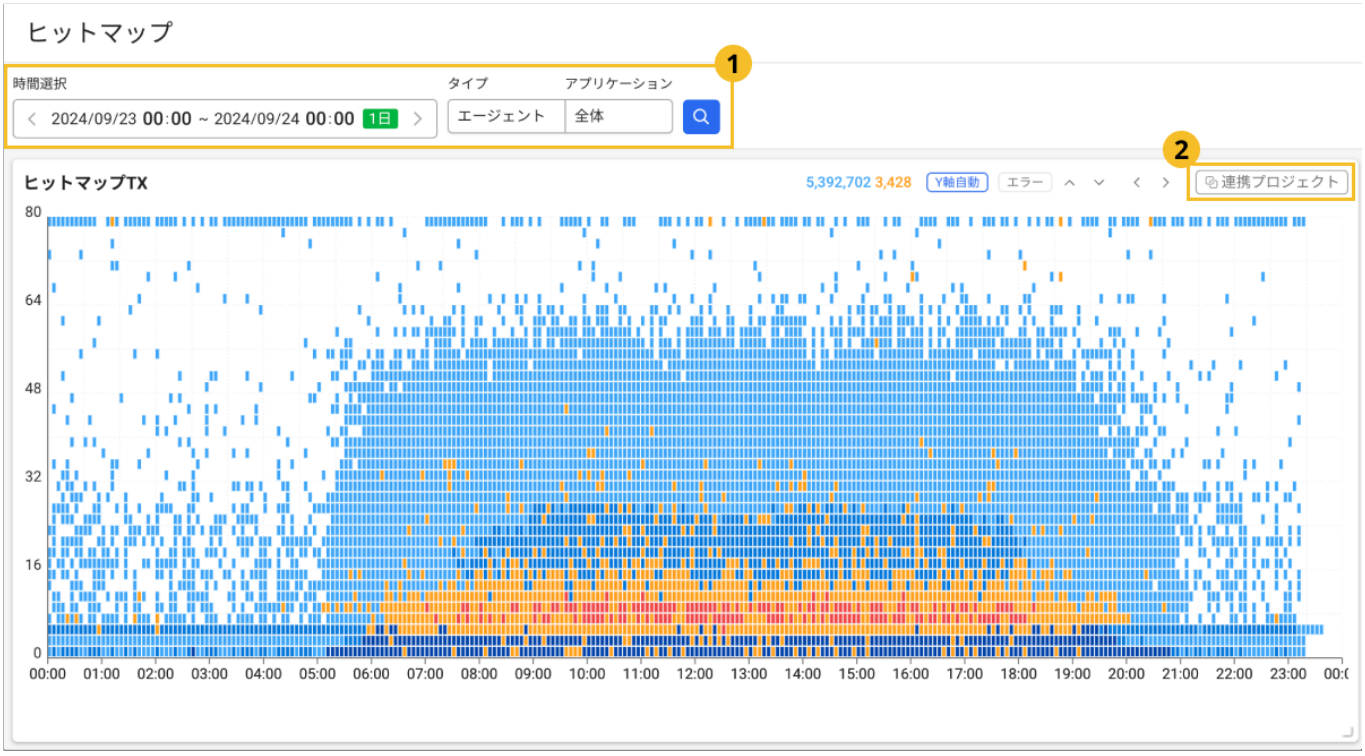


1. トランザクション一覧で照会したいトランザクション項目を選択します。
2. **アクティブTX**画面が表示されたら、右上の**連携DBセッション**ボタンを選択します。
3. **連携データベース情報**画面が表示されたら、関連するメトリクスを確認します。

- 連携分析のために登録したデータベースプロジェクトのDBが現在選択したトランザクションのSQLを実行した場合のみ、**アクティブTX**画面に**連携DBセッション**ボタンが表示されます。
- **連携データベース情報**画面についての詳細は、次の文書を参照してください。
- **Javaエージェント2.2.40**以前のバージョンでは**アクティブTX**ウィンドウで**ThreadStat**値が数字値形式で表示されます。 文字形式で表示するには**Javaエージェント2.2.40**バージョン以降をインストールします。
- この機能をサポートするデータベースは、**PostgreSQL**、**Oracle**、**Oracle Pro**、**MySQL**プラットフォームです。

ヒットマップ

トランザクションを照会する際に連携されたプロジェクトの性能メトリクスを一緒に確認できます。 この機能は、APMプロジェクトでトランザクションを分析する際に、SLOW SQLがあった場合、類似な複数のDBを使用する環境で、正確にどのDBで問題が発生したのかを確認しにくい状況に役立ちます。



1. トランザクションを照会する時間と**アプリケーション**を設定した後 ボタンを選択します。
2. **ヒットマップTX**セクションで右上にある**連携プロジェクト**ボタンを選択します。

3. 連携プロジェクトのメトリクスを確認できる新しい画面が表示されます。 左の一覧で連携プロジェクトを選択し、**インスタンスダッシュボード**セクションのメトリクスを確認します。

> ⓘ **インスタンスダッシュボード**画面についての詳細は、次の文書を参照してください。

バインド変数のコピー&ペースト

SQLステップで確認したバインド変数の値をコピーして**連携データベース情報**ウィンドウ**SQL Text**タブで貼り付けることができます。 **トレース分析**ウィンドウでSQLステップを確認します。



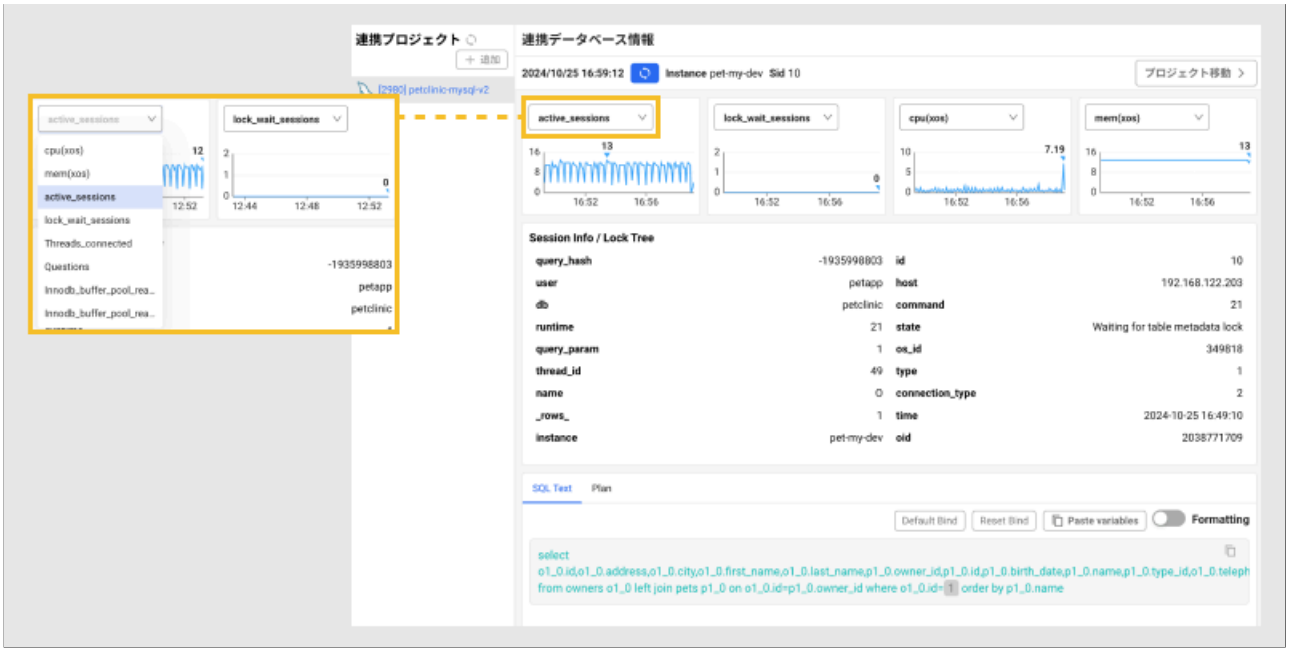
1. **SQL**ウィンドウで**パスワード**ボタンを選択し、セキュリティキーを入力します。
2. バインド変数を確認し**バインド変数のコピー**ボタンを選択します。
3. **連携DBセッション**ボタンを選択します。

4. **連携データベース情報**ウィンドウが表示されたら**SQL Text**タップで**変数の貼り付け**ボタンを選択します。

> ⓘ
> - **連携データベース情報**ウィンドウでSQLステップを確認する方法の詳細については、次の文書を参照してください。
> - セキュリティキーに関するエージェント設定の詳細については、次の文書を参照してください。

連携データベース情報

連携データベース情報では、アプリケーションで現在進行中のトランザクションと連携したデータベースのSQL性能を確認できます。



- 画面上部のチャートでは、データベースの主要なメトリクスのチャートを提供します。 選択ボックスを選択して他のメトリクスを照会できます。
- **Session Info / Lock Tree**セクションでは、データベースセッションに関する様々な詳細を提供します。

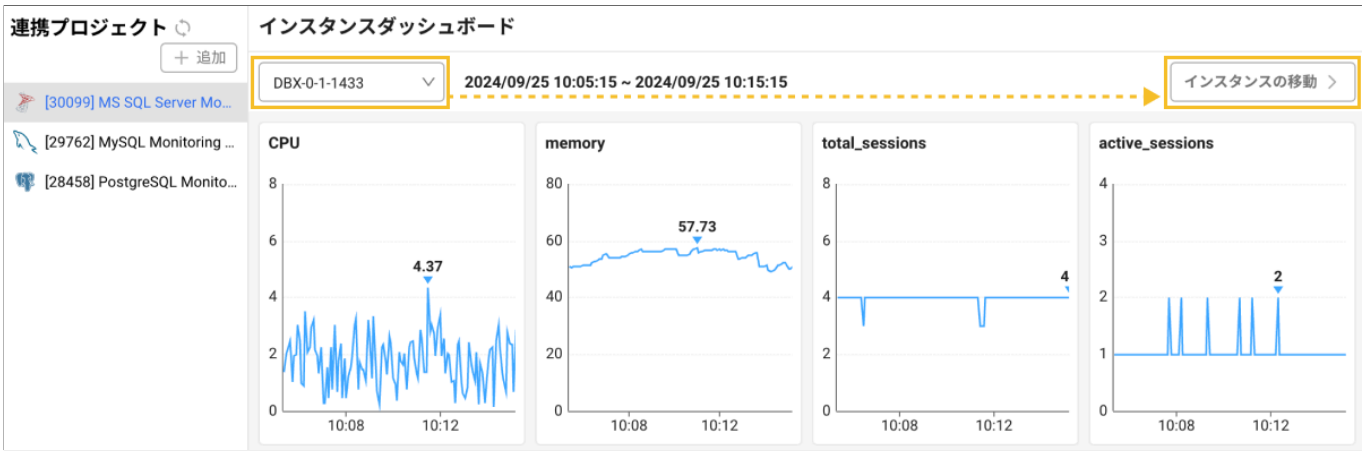
> ⓘ Session Info / Lock Tree

- **query_hash**: 実行中のSQLクエリの固有ハッシュ値です。
- **user**: データベースに接続されているユーザー名です。
- **db**: 割り当てられたデータベース名です。
- **runtime**: 当該セッションが実行された時間(ミリ秒単位)です。
- **query_param**: クエリで使用されたパラメータです。
- **thread_id**: このセッションに接続されたスレッドの固有識別子です。
- **name**: 当該セッションに関連付けられた名前です。
- **rows**: 実行されたクエリで処理された行(row)の数です。
- **state**: 当該セッションのステータスです。
- **id**: 現在のセッションの固有識別子です。
- **host**: データベースが実行中のサーバーのIPアドレスです。
- **command**: 現在実行中のコマンドまたはタスクを示します。
- **os_id**: オペレーティングシステムで使用されるそのセッションのIDです。
- **type**: 当該セッションのタイプです。
- **connection_type**: データベース接続のタイプです。
- **time**: セッションが開始された時間です。
- **oid**: 当該セッションが属するインスタンス(エージェント)の固有IDです。

- **SQL Text**タブでは、実行されたSQLクエリの実際の内容を確認できます。
  - **Default Bind**: クエリ文で変数として設定された部分を既定値としてバインディングできます。
  - **Reset Bind**：バインディングされた既定値を解除し、変数を確認できます。
  - **変数の貼り付け**: コピーしたバインド変数の値を貼り付けることができます。 詳細については、次の文書を参照してください。
  - **Formatting**: SQLクエリ文にインデントおよびフォーマットを適用して可読性を高めます。
- **Plan**: Plan情報を確認できます。 **DB Name**、**User Name**、**Password**を入力した上で  ボタンを選択します。

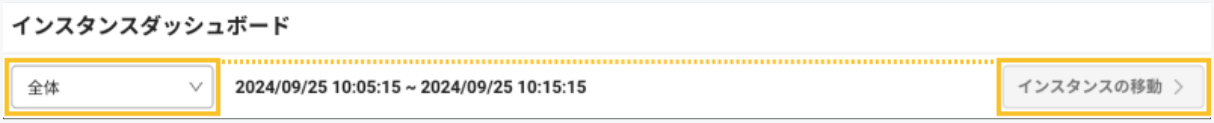
インスタンスダッシュボード

インスタンスダッシュボードでは、アプリケーションで実行されたトランザクションに連携されたデータベースの主要な性能メトリクスを確認できます。



選択したプロジェクトのダッシュボードメニューに移動するには、インスタンスを選択し、右上の**インスタンスの移動**ボタンをクリックします。 **ダッシュボード** > **インスタンスモニタリング**メニューに移動します。

- **ヒットマップTX**セクションの**連携プロジェクト**ボタンは、連携されたプロジェクトがある場合にのみ表示されます。 連携プロジェクトの追加についての詳細は次の文書を参照してください。
- 左の**連携プロジェクト**一覧の上にある**追加**ボタンを選択するとプロジェクトを追加できます。
- **インスタンスの移動**ボタンは、**全体**を選択した状態では無効になります。



- データベース商品の種類によって**分析** > **カウント推移**メニューに移動できます。

連携プロジェクトを削除する

1. **管理** > **連携プロジェクト管理**メニューに移動します。

2. **連携プロジェクトのリスト**セクションで連携を解除しようとするプロジェクトを削除するには、一覧の一番右にあるボタンを選択します。
3. 確認メッセージが表示されたら**削除**ボタンを選択します。

連携プロジェクトのリストセクションで削除したプロジェクトは除外されます。

Logモニタリング

ログは、アプリケーションやシステムで発生するイベントやメッセージを記録するファイルです。異常を特定してシステムの劣化を防いだり、発生したトラブルの原因を確認する場合は、ログを確認することが重要です。

最近のITサービス構築環境は、MSA（マイクロサービスアーキテクチャ）やKubernetesベースへと移行が進んでおり、管理対象が増加しています。一般的なログモニタリングでは、サーバーに接続して `tail` コマンドやエディタを使ってログを確認します。しかし、個別のサーバーに逐一接続するような単純な方法では対応が困難です。では、数百〜数千台のサーバーから発生するログを、どのようにして確認すればよいのでしょうか？

WhaTapログモニタリングサービスを使用すると、数多くのログを簡単に管理できます。

> Linuxの `tail` コマンドは、時間の経過とともにコンテンツを追加するログを確認するためによく使用されます。

主な機能

- **中央統合管理**

  WhaTapは、大量のログを中央で一元管理します。個別のサーバーにアクセスしなくても、中央からログを確認できるため、利便性に優れています。

- **すべてのログを収集**

  ログを選別して収集する場合、重要なデータが欠落する可能性があります。WhaTapはすべてのログを収集し、収集されたログは **ライブテール** でリアルタイムに確認できます。

- **可視性確保**

  WhaTapのさまざまなチャートからログを確認できます。これにより、エラーや問題に関する情報へのアクセス性が向上し、障害の早期検知が可能になります。

- **柔軟な容量管理**

  個別のサーバーにログを保存する場合、ログによってファイルシステムの容量が過剰に占有される問題が発生する可能性があります。WhaTapを利用してログを中央に集約すれば、各サーバーでログファイルを保持する必要はなく、中央に保存されたログデータの保持期間のみを管理すれば済みます。

- **複数の分析的視点**

  障害の状況を把握し予測するためには、さまざまな観点からログを分析できる必要があります。WhaTapでは、特定のタグを含むログ件数の推移を確認したり、フィルターを適用して分析したりすることができます。よく使うパターンはチャートとして保存し、いつでも呼び出して確認できます。

- **パターン通知**

  障害のパターンをアラートとして設定することで、問題を予防したり迅速に把握したりすることができます。WhaTapでは、個別のログに特定のキーワードが含まれている場合に通知を受け取る **リアルタイムログアラート** と、特定のタグを含むログ件数の推移を基準に通知を受け取る **複合ログアラート** を提供しています。

ログ分析

ライブテール

ライブテール メニューでは、リアルタイムで収集されたログを `tail` コマンドのように画面上で流れる形式で確認できます。

> ⓘ 詳しくは、ライブテール のドキュメントを参照してください。

ログトレンド

ログトレンド メニューでは、収集されたすべてのログ、または特定のタグを含むログの件数推移を確認できます。ログの発生件数は障害の発生および解消のタイミングと密接に関連しているため、ログ件数の推移を通じて障害の原因を分析し、対応のスピードを向上させることができます。

> ⓘ 詳しくは、ログトレンド のドキュメントを参照してください。

ログの検索

ログ検索 メニューでは、収集されたすべてのログ、または特定のタグを含むログを照会できます。特定の時間帯やサーバーで発生したログをタグを基準にして検索・確認することができます。隣接ログ機能を使用すれば、特定のError

やException発生の前後に記録されたログをあわせて確認し、原因分析に活用できます。

> ⓘ 詳しくは、ログ検索 のドキュメントを参照してください。

ログエクスプローラー

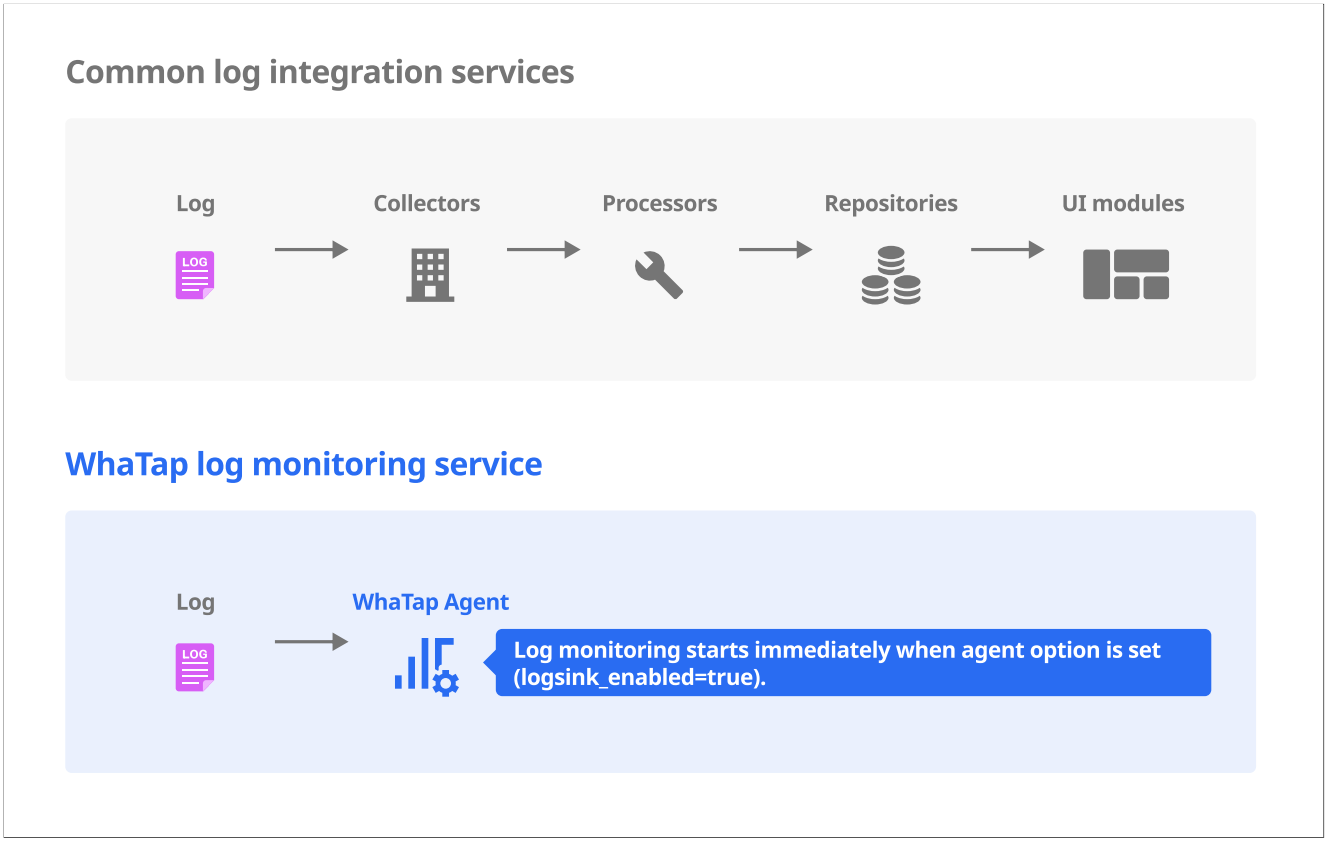
ログエクスプローラー メニューを使用すると、ログデータを迅速に検索、フィルタリング、分析できます。フィールド構造を把握し、クエリベースのフィルタリングによって必要なログを効率的に検索し、その結果をデータとして可視化することができます。ログエクスプローラー機能は、Luceneインデックスを基盤として動作します。

> ⓘ 詳しくは、ログエクスプローラー のドキュメントを参照してください。

ログモニタリングを適用する

WhaTapログモニタリングサービスを利用するための基本的な使用方法について説明します。WhaTapログモニタリングは、追加のエージェントを構成やログのパーサを適用する必要はありません。簡単な設定で迅速に起動できます。

ログモニタリングの動作原理



一般的なログ統合サービスは収集機、処理機、ストレージ、UIモジュールで構成されています。段階別設定と構成作業が必要なため、それぞれのモジュールを構築する過程が煩わしく、追加費用が発生します。

WhaTapログモニタリングは簡単に適用できます。既存のモニタリングエージェントがコレクターとして機能するため、エージェントオプションをオンにするだけでログモニタリングを開始できます。

- Java PHP Python Go Server Kubernetes

- 出力されたファイルからログを読まずに、Javaアプリケーションのログライブラリに渡されるログを直接収集します。
- ログを直接収集するため、ファイルI／Oを引き起こさず、システムに与える性能への影響が非常に低いです。
- トランザクショントレースとログの接続を追跡する特徴を確保し、トレースからログを確認できます。

> **Javaログのライブラリ**
>
> 典型的なJavaログライブラリはApache Log4j、Logbackです。

> Java Agent2.1.1バージョンから使用できます。

- 既存のエージェントにログ収集機能を追加しました。モニタリングエージェントがログファイルから追加で出力されたログを読み取り読み取り収集する方法を利用します。

> PHP Agent2.3.2バージョンから使用できます。

- 既存のエージェントにログ収集機能を追加しました。モニタリングエージェントがログファイルから追加で出力されたログを読み取り読み取り収集する方法を利用します。
- ログにトランザクションIDを出力すると、トランザクショントレースとログの接続追跡性を確保し、トレースからログを確認できます。

> Python Agent1.2.2バージョンから使用できます。

- 既存のエージェントにログ収集機能を追加しました。モニタリングエージェントがログファイルから追加で出力されたログを読み取り読み取り収集する方法を利用します。
- 既存のエージェントにログ収集機能を追加しました。モニタリングエージェントがログファイルから追加で出力されたログを読み取り読み取り収集する方法を利用します。

> Server Agent2.1.2バージョンから使用できます。

- Kubernetesコンテナにログを収集できます。

- Kubernetesコンテナ内のアプリケーションからログを収集できます。

> Kubernetes Agent1.1.35バージョンから使用できます。

ログモニタリングを適用する

使用するアプリケーションに応じた適用方法は以下のとおりです。ログモニタリングを適用する前に、対応バージョンを必ずご確認ください。

1. 対応するエージェントのバージョンを確認し、**アップデート**してください。
2. ログモニタリング**オプション**を設定してください。
3. ログモニタリングを**有効**にしてください。

Java

Javaアプリケーションでログを収集する方法について案内します。

PHP

PHPアプリケーションからログを収集する方法について説明します。

Node.js

Node.jsアプリケーションでログを収集する方法を案内します。

Python

Pythonアプリケーションからログを収集する方法について説明します。

Go

Goアプリケーションからログを収集する方法について説明します。

Server

サーバーアプリケーションからログを収集する方法について説明します。

Kubernetes

Kubernetesコンテナとコンテナ内部のアプリケーションのログを収集する方法について説明します。

Java

Javaアプリケーションでログを収集するための設定方法を案内します。

エージェントのアップデート

Javaエージェントのバージョン2.1.1以降で利用可能です。アップデート方法については、次のドキュメントを参照してください。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリング開始**タブの **1. エージェント設定の確認** の案内に従って、エージェント設定を確認してください。

エージェント設定

1. プロジェクトの **管理** > **エージェントの設定** メニューをクリックしてください。
2. **オプション作成** の **オプション選択** ドロップダウンリストから **直接入力** をクリックし、`logsink_enabled=true` オプションを入力して **適用** ボタンをクリックして追加してください。
   - キー入力欄に `logsink_enabled` を入力してください。
   - 値入力欄に `true` を入力してください。
3. ログモニタリングを適用するには、アプリケーションを再起動します。

主なオプション

- **hooklog_enabled** Boolean

既定値 false

Logライブラリをフックして、ログモニタリングを有効化します。

> アプリケーションを実行する前に、whatap.confオプションを有効にして、その後にログモニタリングのOn/Offを logsink_enabled 設定で動的に制御できます。アプリケーションを起動する前に logsink_enabled オプションを true に設定する場合、このオプションを個別に設定しなくてもログモニタリングが可能です。
>
> 今後ログモニタリングを活用する可能性がある場合は、事前にこのオプションを設定することをお勧めします。

- **hooklog_custom_methods**

  カスタムログを登録します。特定のログフレームワークの内容を収集します。サイトで個別に作成したログモジュールのログを追跡する際に利用してください。

  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

  既定値 false

  Logモニタリング機能をOn/Offします。

> アプリケーションが起動する前にwhatap.confに hooklog_enabled オプションが設定されている場合、このオプションを使用すると、ログモニタリングのOn/Offを動的に制御することができます。

- **logsink_trace_enabled** Boolean

既定値 false

Logにトランザクション IDを挿入して、トランザクショントレースのログタブの表示有無を指定します。

ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始する タブの **2. ログモニタリング機能の活性化** で、WhaTapログモニタリングを有効化または無効化できます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

> **ログモニタリングの有効化権限**
>
> エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

PHP

PHPアプリケーションでログを収集するための設定方法を案内します。

エージェントのアップデート

PHPエージェント2.3.2バージョンから可能です。アップデート方法は次の文書を参照してください。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリングを開始する** タブの **1. エージェント設定の確認** の案内に従って、エージェント設定を確認してください。

設定ファイルパスの確認

ログモニタリングするファイルの**設定ファイルパス**を確認してください。CLI環境または、ウェブで確認することができます。

CLI環境で確認する

```
$ php -i | grep 'Scan'
Scan this dir for additional .ini files => /etc/php8.0.d
```

> ⓘ パスが"(None)"の場合、設定ファイルのパスは*/usr/whatap/php*です。

ウェブで確認する

CLI環境とApache環境でPHPの設定が異なる場合は、Web上で `phpinfo()` 関数の結果を確認してください。

コマンド入力

設定ファイルパスを含むコマンドを入力すると、ログモニタリングが開始されます。

```
export LOGFILES=/some/path/file1,/some/other/file2
echo "whatap.logsink_enabled=true" | sudo tee -a [設定ファイルパス]/whatap.ini
echo "whatap.logsink.files=$LOGFILES" | sudo tee -a [設定ファイルパス]/whatap.ini
```

ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始する タブの **2. ログモニタリングの有効化** で、WhaTapのログモニタリングを有効または無効に設定できます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

> **ログモニタリングの有効化権限**
>
> エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

Node.js

Node.jsアプリケーションでログを収集するための設定方法を案内します。

エージェントのアップデート

Node.jsエージェント0.5.1バージョンから可能です。 アップデート方法は、次の文書を参照してください。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリング開始**タブの **1. エージェント設定の確認** の案内に従って、エージェント設定を確認してください。

エージェント設定

1. プロジェクトの **管理** > **エージェントの設定** メニューをクリックしてください。
2. **オプション作成** の **オプション選択** ドロップダウンリストから **直接入力** をクリックし、`logsink_enabled=true` オプションを入力して **適用** ボタンをクリックして追加してください。
   - キー入力欄に `logsink_enabled` を入力してください。
   - 値入力欄に `true` を入力してください。
3. ログモニタリングを適用するには、アプリケーションを再起動します。

主なオプション

- **logsink_enabled** Boolean

  既定値 `false`

ログモニタリングの活性化有無を設定します。

- **logsink_trace_txid_enabled** Boolean

  既定値 `true`

  ログにトランザクションIDを挿入して、トランザクション トレースのログタブの表示有無を指定します。

- **logsink_limit_content_enabled** Boolean

  既定値 `true`

  ログメッセージの最大サイズの制限有無を設定します。

- **logsink_limit_content_length** Number

  既定値 `10,000`

  ログメッセージの最大長さを設定します。 この値は、 `logsink_limit_content_enabled` オプションが `true` に設定された場合にのみ適用されます。

ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始するセクションの**ログモニタリングを有効にする**タブのトグルボタンで、WhaTapログモニタリングの有効または無効にできます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

> **ログモニタリングの有効化権限**
>
> エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

Python

Pythonアプリケーションでログを収集するための設定方法を案内します。

エージェントのアップデート

Pythonエージェント1.2.2バージョン以降で可能です。アップデート方法は次の文書を参照してください。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリングを開始する** タブの **1. エージェント設定の確認** の案内に従って、エージェント設定を確認してください。

ログ収集の活性化

設定ファイルパス(WHATAP_HOME)を含むコマンドを入力すると、ログ収集がすぐに開始されます。

```
export LOGFILES={ログファイル全体パス},...
echo "logsink_enabled=true" | sudo tee -a {設定ファイルパス}/whatap.conf
echo "logsink.files=$LOGFILES" | sudo tee -a {設定ファイルパス}/whatap.conf
```

ログとウェブトランザクションの連動

1.3.6 以降バージョン

1.3.6バージョンから次の方法でトランザクションとログの連動を設定できます。現在、WhaTapはPythonのlogging、loguruライブラリを対応しています。使用するPython Logライブラリに従ってwhatap.confを構成してください。

- loggingモジュール

  logging

```
echo "trace_logging_enabled=true" | sudo tee -a{設定ファイルパス}/whatap.conf
```

- loguruモジュール

  loguru

```
echo "trace_loguru_enabled=true" | sudo tee -a{設定ファイルパス}/whatap.conf
```

1.3.6以前バージョン

トランザクションごとに発生したログを個別に照会できるように、ログにトランザクションID `{txid}` を出力します。WhaTapモニタリングでは、PythonのLogRecordに `{txid}` を自動的に注入し、フォーマッタを設定することでログファイルに `{txid}` を出力できるようにします。

```
settings.py
...
LOGGING = {
...
  'formatters': {
    ...
  },
  'handlers': {
    ...
  },
  'loggers': {
    ...
    '{ロガー名}': {
      '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'][{ロガー名}]['handlers'].append('whatap')
except:
  pass
...
```

ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始する タブの **2. ログモニタリングの有効化** で、WhaTapのログモニタリングを有効または無効に設定できます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

> **ログモニタリングの有効化権限**
>
> エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

Go

Goアプリケーションでログを収集するための設定方法を案内します。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリングを開始する** タブの **1. エージェント設定の確認** の案内に従って、エージェント設定を確認してください。

ログ収集の活性化

次のコマンドを入力すると、すぐにログの収集が開始されます。

```
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
```

ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始する タブの **2. ログモニタリングの有効化** で、WhaTapのログモニタリングを有効または無効に設定できます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

ログモニタリングの有効化権限

エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

Server

サーバーアプリケーションでログを収集するための設定方法を案内します。

エージェントのアップデート

サーバーエージェントのバージョン2.1.2以降で利用可能です。アップデート方法は 次の文書 を参照してください。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリングを始める**の**1. エージェント設定の確認**にある**OS選択**タブで、**Linux（shell）または Windows（PowerShell、管理者権限）**を選択してください。
3. **基本インストール**または**カテゴリ付きインストール**を参考にして、エージェント設定を確認してください。

> ⓘ 次の設定は、エージェントを再起動する必要があります。

Linux Shell

- **基本インストール**

  Linux Shell

```
export LOGFILES=/some/path/file1,/some/other/file2
echo "logsink.files=$LOGFILES" | sudo tee -a /usr/whatap/infra/conf/whatap.conf
```

- **カテゴリ付きインストール**

Linux Shell

```
cd /usr/whatap/infra
sudo mkdir extension

cat <<EOL | sudo tee extension/logsink.conf > /dev/null
[[inputs.logsink]]
 category = "serverlog"
 ## ログ発生量統計の別途データとしての送信有無
 stats_enabled = true
 ## 対数発生量の統計カテゴリ
 stats_category = "logsink_stats"
 ## ログファイルパス(path)にアスタリスク(*)が含まれれ、除外するログ非対象ファイル名を設定
 excludeNames = [ ".gz",".zip" ]
 [[inputs.logsink.file]]
  ## ログファイルの指定時、日付パターン(strftime.org)指定可能
  path = "/some/path/%Y-%m-%d/.log"
  disabled = false
  encoding = "utf-8"

 [[inputs.logsink.file]]
  path = "/some/other/log"
  disabled = false
  encoding = "utf-8"

 [[inputs.logsink.file]]
  ## 行単位ログで該当キーワードを検索する時、以前のログにマージ
  nowrap_keywords = ["Caused by:", "Test"]

 [[inputs.logsink.file]]
 ## プロジェクトコード及びプロジェクトライセンスの入力時、ログを同じリージョンの希望するプロジェクトに転送可能
 # pcode = プロジェクトコード
 # license = "プロジェクトライセンス"

 [[inputs.logsink.file]]
 # whatap_host_ip = "収集サーバーIP1/収集サーバーIP2"
 # whatap_host_port = 6600

 [[inputs.logsink.file]]
```

```
 path = "/tmp/mixed.log"
 disabled = false
 encoding = "utf-8"
 ## [{"A":1,"B":2},{"C":3,"D":4},...] 形態をそれぞれのログに分離できる{"A":1,"B":2} {"C":3,"D":4}
 multiline_pattern = "simple_json_array"
EOL

sudo service whatap-infra restart
```

Windows Powershell

- **基本インストール**

  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"
```

- **カテゴリ付きインストール**

  Windows Powershell

```
# 管理者権限が必要
New-Item -type "Directory" -Path "C:\Program Files\WhatapInfra\extension"

$contentToAdd = @"
[[inputs.logsink]]
category = "serverlog"
## ログ発生量統計の別途データとしての送信有無
stats_enabled = true
## 対数発生量の統計カテゴリ
stats_category = "logsink_stats"
## ログファイルパス(path)にアスタリスク(*)が含まれ、除外するログの非対象ファイル名を設定
excludeNames = [ ".gz",".zip" ]
[[inputs.logsink.file]]
## ログ ファイルの指定時、日付パターン(strftime.org)指定可能
path = "c:\\whatap\\logs%Y-%m-%d_.log"
disabled = false
```

```
encoding = "utf-8"

[[inputs.logsink.file]]
## 行単位ログで該当キーワードを検索する時、以前のログにマージ
nowrap_keywords = ["Caused by:", "Test"]

[[inputs.logsink.file]]
## プロジェクトコード及びプロジェクトライセンスの入力時、ログを同じリージョンの希望するプロジェクトに転送可能
# pcode = プロジェクトコード
# license = "プロジェクトライセンス"

[[inputs.logsink.file]]
# whatap_host_ip = "収集サーバーIP1/収集サーバーIP2"
# whatap_host_port = 6600

[[inputs.logsink.file]]
path = "/tmp/mixed.log"
disabled = false
encoding = "utf-8"
## [{"A":1,"B":2},{"C":3,"D":4},...] 形態をそれぞれのログに分離することができる。
{"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"
```

オプション設定

- **stats_enabled**：収集ステータスデータの収集有無を設定します。 (既定値: false)
  - true に設定する必要があり、stats_category で設定したカテゴリ（ logsink_stats ）に統計データが生成されます。
- **stats_category**：収集ステータスデータを保存するメトリクスカテゴリを設定します。 logsink_stats に設定す

る必要があります。 統計データのフィールドは次のとおりです。

保存される統計データフィールド: `file` `checkInterval` `encoding` `filepos` `checkedLocalTime` `lastupdatedLocalTime` `fileSize` `error` `firstCheck` `transferBytes`

- **excludeNames** ：ログファイルのパス（ path ）にアスタリスク（ * ）が含まれている場合、ログ収集対象外のファイル名を設定して除外することができます。 カンマ（ , ）を区切り文字として使用することで、複数の値を設定できます。

Example

```
excludeNames = [ ".gz",".zip" ]
```

- **nowrap_keywords** : 行単位のログ検索時に、このオプションで指定されたキーワードが検出されると、前のログと結合されます。
- **pcode** : ログを送信する同じリージョンのプロジェクトコードを入力します。
- **license** : ログを送信する同じリージョンのプロジェクトのライセンスコードを入力します。

> ⓘ pcode 、 license オプションの場合、サーバーエージェント2.7.4バージョン以降に対応します。

Windowsイベントログのオプション設定

Windowsイベントログ収集時、次のようにオプションを設定することができます。

```
# 管理者権限が必要
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 = ""
    #プロジェクトコード及びプロジェクトライセンスの入力時、ログを同じリージョンの希望するプロジェクトに転送可能
    #pcode = プロジェクトコード
    #license = "プロジェクトライセンス"
    #whatap_host_ip = "収集サーバーIP1/収集サーバーIP2"
    #whatap_host_port = 6600

"@

New-Item -path "C:\Program Files\WhatapInfra\extension" -name "win_eventlog.conf" -type "file" -value $contentToAdd -Force

Restart-Service "Whatap Infra"
```

- カテゴリ指定(`category`) 必須

  例） `win_event_log`

- 統計カテゴリ指定(`stats_category`) 必須

  例） `win_event_log_stats`

- 統計カテゴリOn/Off(`stats_enabled`) 必須

  例） `true` または `false`

- 収集機能On/Off(`enabled`) 必須

  例） `true` または `false`

- ファイル別収集機能On/Off(`enabled`) 必須

  例） `true` または `false`

- ファイル(`file`) 必須

例） `Application` 、 `Security` 、 `Setup` 、 `System` 、 `Forwarded`

- イベントタイプ(`event_type`) 非必須

  例） `1` 、 `2` 、 `3` 、 `4` 、 `5`

> **イベントタイプ**
>
> 1. Information
> 2. Warning
> 3. Error
> 4. Audit Success
> 5. Audit Fail

- イベントID(`event_id`) 非必須
- イベントソース名(`source_name`) 非必須
- プロジェクトコード(`pcode`)非必須
- プロジェクトライセンス(`license`)非必須

> **エージェント対応バージョン**
>
> - ウィンドウイベントログ収集の場合、サーバーエージェント2.5.2バージョン以降に対応します。
> - ウィンドウイベントログオプションの中で `pcode` 、 `license` オプションの場合、サーバーエージェント2.7.4バージョン以降に対応します。

複数のファイルを指定および分離

複数のファイル指定

複数のログファイルを同じカテゴリに設定するには、各ログファイルのパスを `[[inputs.logsink.file]]` 項目に追加して、 `path` 設定により、ファイルパスを指定します。 また、 `category` 、 `stats_enabled` 、 `stats_category` などのオプションを使用して、ログ発生量の統計設定などの付加的な設定が可能です。

/root/test1 および /root/test2 ファイルを同じカテゴリ（ serverlog ）に指定した例です。

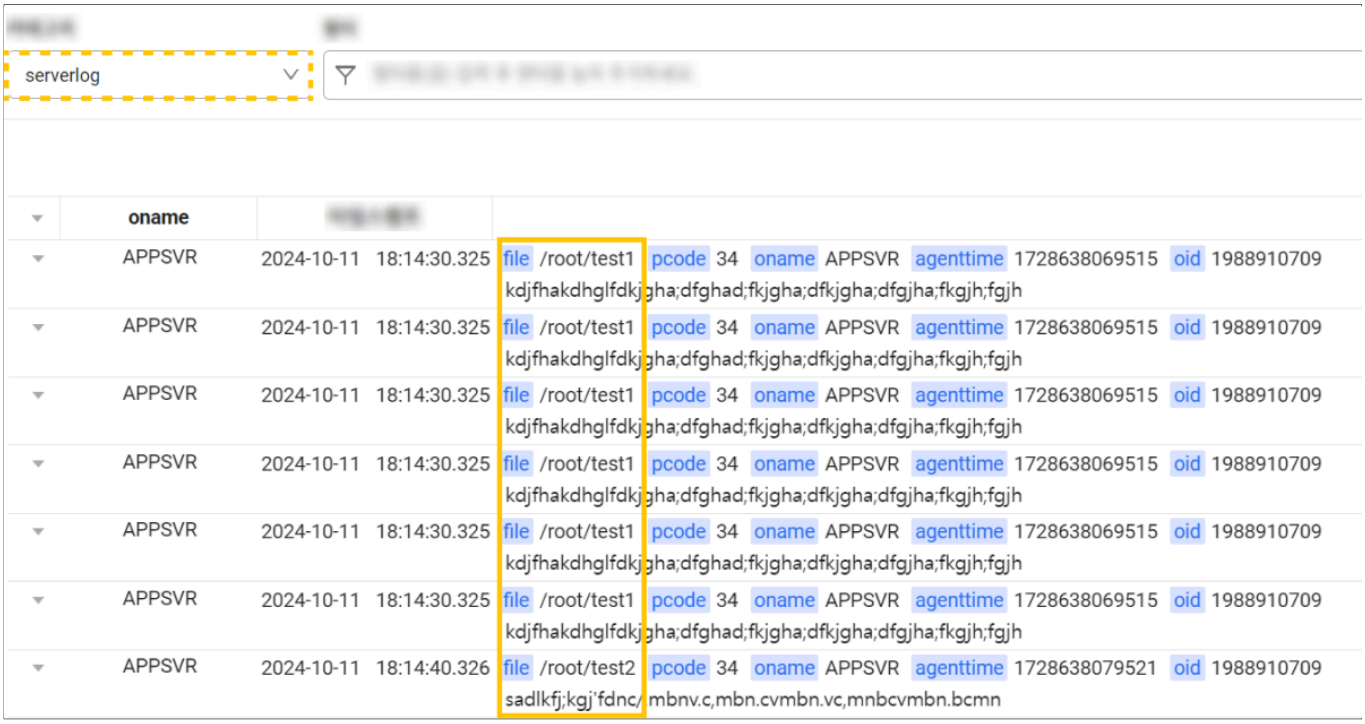
```
[[inputs.logsink]]
 category = "serverlog"
 stats_enabled = true
 stats_category = "logsink_stats"
 excludeNames = [ ".gz", ".zip" ]

[[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 などのオプションに関する詳細は、オプション設定 を参照してください。

この例は、**ログTail** メニューで次のように確認できます。



ファイル分離

異なるカテゴリにログファイルを割り当てるには、logsink{**数字**}.conf ファイルをそれぞれ作成し、各ファイルの `category` 項目に異なる値を設定してください。

logsink.conf と logsink**1**.conf を使用して、異なるログカテゴリを設定した例です。

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"
```

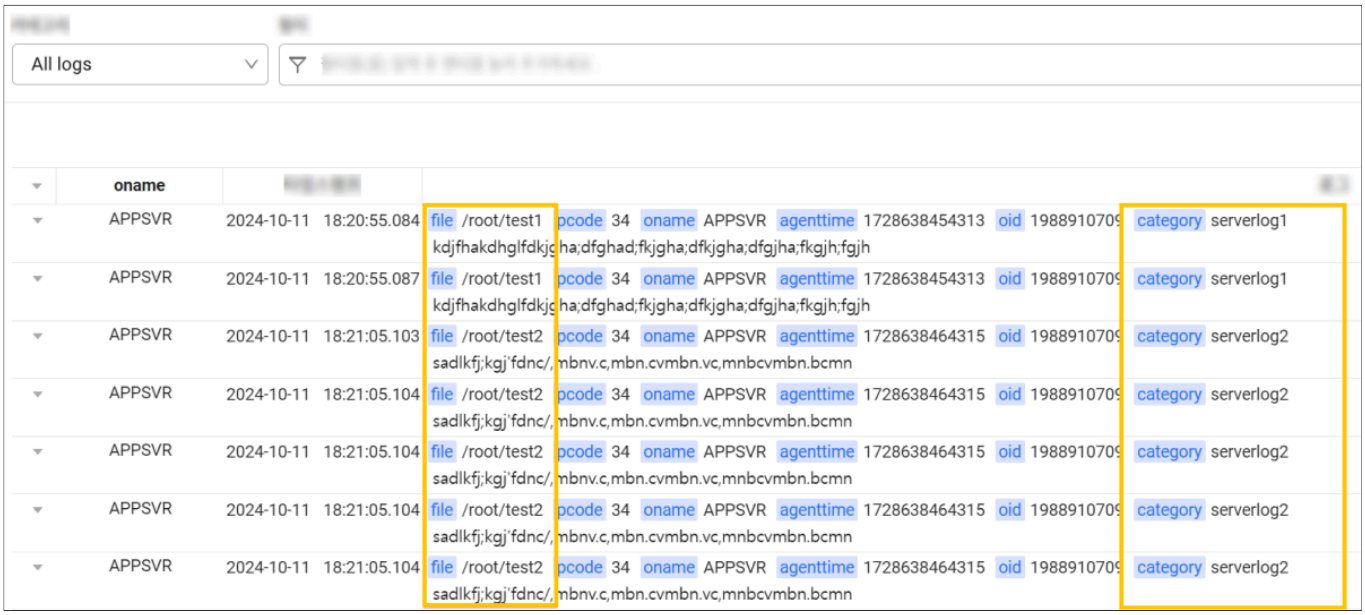
logsink1.conf

```
#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 などのオプションに関する詳細は、オプション設定 を参照してください。

この例は、**ログTail** メニューで次のように確認できます。



ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始するセクションの**ログモニタリングを有効にする**タブのトグルボタンで、WhaTapログモニタリングの有効または無効にできます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

> **ログモニタリングの有効化権限**
>
> エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

Kubernetes

Kubernetesコンテナおよびコンテナ内のアプリケーションのログを収集するための設定方法を案内します。

エージェントのアップデート

Kubernetesエージェント1.1.35バージョン以降で使用可能です。アップデート方法は次の文書を参照してください。

エージェント設定の確認

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

1. WhaTapモニタリングサービスの初期画面でプロジェクトを選択した後、**ログ** > **ログ設定** メニューに移動してください。
2. **ログモニタリングを開始する** タブの **エージェント設定およびログモニタリングの有効化** にある **1. エージェント設定の確認** の案内を参考にして、エージェント設定を確認してください。

コンテナログ収集の有効化

Kubernetesコンテナでログを収集するには、**1. エージェント設定の確認** で **ログ設定を適用** ボタンをクリックしてください。

コンテナの内部アプリケーションログ収集の有効化

Java 2.1.1、Python 1.2.2以降のバージョンで利用可能です。Kubernetesコンテナ上で実行されるアプリケーションのログを収集するには、以下を参考にしてください。

1. **ログ > ログ設定 > ログモニタリングを開始する** タブに移動してください。
2. **1. エージェント設定の確認** の下部にある **エージェント設定** ボタンをクリックしてください。
3. **エージェント設定** で、**オプション作成** のオプション選択ドロップダウンリストから **直接入力** をクリックしてください。

4. エージェント設定コマンドのキー `logsink_enabled` と値 `true` を入力欄に入力してください。

```
logsink_enabled=true
```

5. **適用**ボタンを選択します。Kubernetesコンテナ内のアプリケーションからログを収集できます。

> ⓘ
> - Javaアプリケーションのログ収集に関する詳細は、次のドキュメントを参考にしてください。
> - Pythonアプリケーションのログ収集に関する詳細は、次のドキュメントを参考にしてください。
> - Goアプリケーションのログ収集に関する詳細は、次のドキュメントを参考にしてください。

ログモニタリングの活性化

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログモニタリングを開始するセクションの**ログモニタリングを有効にする**タブのトグルボタンで、WhaTapログモニタリングの有効または無効にできます。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

> ⓘ **ログモニタリングの有効化権限**
>
> エージェントをインストールした後、プロジェクトに対して**編集権限**を持つユーザーのみがログモニタリングを有効化できます。 権限の詳細については、次のドキュメントを参照してください。

Kubernetesログモニタリングのカテゴリガイド

Kubernetesに関するさまざまなログを確認できます。WhaTap Kubernetesでは、モニタリングカテゴリを提供しています。

設定によって重複したログ内容が保存される場合があるため、重複がないか必ず確認してください。

カテゴリ	説明
#K8sEvent	• Kubernetesで発生したイベントがデフォルトで保存および作成されるログ • ユーザー設定に関係なく基本作成
#WhatapEvent	• WhaTapイベント設定によって発生したイベントが格納されるログ • ユーザー設定に関係なく、デフォルトで作成されます
containerStdout	• コンテナ標準出力ログ • ユーザー設定時に作成 • ノードエージェントに logsink_enabled=true 設定を追加したした場合
AppLog	• コンテナ内アプリケーションログ • ユーザー設定時に作成 • アプリケーションエージェントに logsink_enabled=true 設定の追加時

containerStdOutログフィルターの設定

ホーム画面 > プロジェクト選択 > **管理** > **エージェント設定**

Kubernetesログで必要な情報を選別するためのフィルタリングオプションを案内します。 フィルタリングにより、許可または除外するログを設定できます。

1. **管理** > **エージェント設定** に移動してください。
2. **プロジェクトデフォルト** タブで、**オプション作成** 方式から **直接入力** をクリックしてください。



ログネーミングのルール

Kubernetes containerStdOutログはノードの*/var/log/containers/*パスの下位に生成されます。 次のようなネーミングルールに従っています。

```
<podName>_<namespace>_<containerName>-<containerId>.log
```

ログフィルターオプション

- **log_filter_enabled** bool

  既定値 false

  ログフィルターの使用有無を設定します。

- **log_filter_allow_list** list

  許可するログの一覧を設定します。 フィルタリングの適用時に含める項目を意味します。

- **log_filter_deny_list** list

  除外するログの一覧を設定します。 フィルタリングの適用時に除外する項目を意味します。

ログフィルター動作

`log_filter_enabled` オプションが有効になっている場合にのみ、ログフィルターが動作します。 該当オプションの値が `true` に設定された状態で、 `log_filter_allow_list` および `log_filter_deny_list` を通じてログを選択的に収集できます。

ログフィルターはブラックリストベースで動作します。 例えば、 `log_filter_enabled` 活性化した後 `log_filter_allow_list` オプションに追加する項目を設定しない場合、デフォルトではすべてのコンテナのログ収集がブロックされます。 また、ログフィルターはallowルールよりdenyルールの優先順位が高いです。 同一の項目がallowとdenyの両方に設定されている場合、denyルールが優先的に適用され、そのログは収集されません。

ログフィルターの例

単一ネームスペースログの収集

- 許容 **log_filter_allow_list** `*_infra_*`

`infra` ネームスペースに存在するすべてのコンテナが収集対象として特定されます。

マルチネームスペースログの収集

- 許容 **log_filter_allow_list** `*_infra_*`、`*_monitoring_*`

`infra` 、 `monitoring` ネームスペースに存在するすべてのコンテナが収集対象として特定されます。

特定の単語を含むPodを除く

- 許容 **log_filter_allow_list** `*_*_*`
- 除外 **log_filter_deny_list** `*prod*_*_*`

すべてのコンテナが収集対象として特定されますが、 `podName` に `prod` が含まれているログは除外されます。

特定の単語を含むPodログの収集

- 許容 **log_filter_allow_list** `*prod*_*_*`

`podName` に `prod` が含まれるPodのログのみが収集されます。

マルチネームスペースログの収集と特定の単語を含むPodを除く

- 許容 **log_filter_allow_list** `*_infra_*、*_monitoring_*`
- 除外 **log_filter_deny_list** `*prod*_*_*`

`infra`、`monitoring`ネームスペースのログが収集され、`podName`に`prod`が含まれているログは除外されます。

コンテナ名内の特定の単語で始まるコンテナログを除外

- 許容 **log_filter_allow_list** `*_*_db-*`

すべてのネームスペースで`containerName`が`db`で始まるコンテナのログは除外されます。

ライブテール

> **ログ参照**権限がない場合は、そのメニューにアクセスできません。

ホーム画面 > プロジェクト選択 > **ログ** > **ログTail**

ログTail メニューでは、サーバーコンソールに接続せずに、ログデータストリームをモニタリング画面上で直接確認できます。複雑で大量のログも、**フィルター** と **ハイライト** 機能を活用して絞り込み、素早く把握できます。ログデータの取得周期は2秒です。

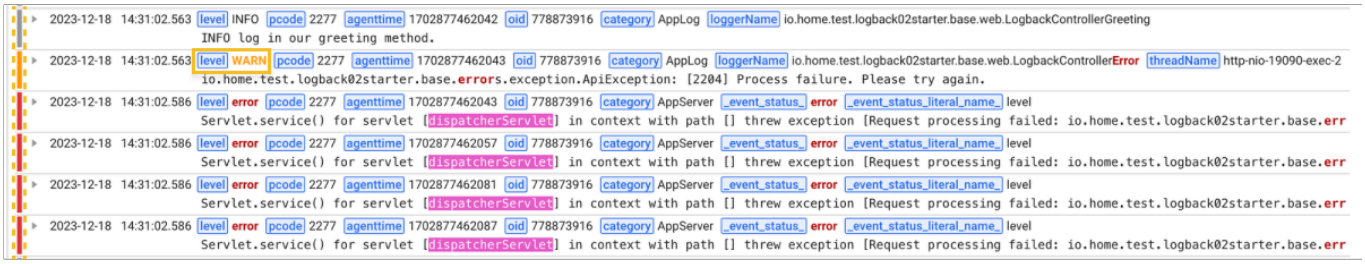
主な用語は次のとおりです。

- **Category**：ログの収集および照会単位
- **Content**：ログメッセージ
- **Search Key**：ログパーサーの設定によって生成される検索キー
- **Tag**：収集されたログを検索できるタグ

> ログテーブルカラムの端をドラッグしてカラムの幅を修正できます。

エージェントオプション

エージェントオプションが設定されている場合、ログレベルを収集し、各ログの左側に色付きの縦線が次のように表示されます。



エージェントのオプション設定

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- Javaエージェント2.2.22バージョンからウィービング設定にlog4j-2.17またはlogback-1.2.8設定時に使用できます。 エージェントの再起動が必要です。
- ログレベルは、パーシングされたキーワードの中で `level`、`type` 基準で判別します。 `level`、`type` にパーシングされたキーが存在し、パーシングの値がFATAL, CRITICAL、ERROR、WARN、WARNING、INFOを含む場合は、ログレベルの色を表示します。

1 フィルター領域

フィルター適用

フィルターを適用すると、入力した条件に合ったログをフィルタリングします。 複数のフィルターを入力できます。 フィルターのタグが同じ場合、OR(`||`)、 そうでない場合はAND(`&&`)が適用されます。

入力欄に文字を直接入力するか、**フィルター**入力欄をクリックしてフィルターを指定します。 フィルタータグは `検索キー`、`オペレーター`、`検索値` の順に入力します。 **検索** ボタンを選択すると、フィルターが適用されたデータをログ一覧で確認できます。

ガイドUI

次のように入力ウィンドウの下に**ガイドUI**を提供します。 入力画面またはフィルタータグにマウスカーソルがある場合、**ガイドUI**を閉じる時はESCキーで操作できます。



検索キー、オペレーター、検索値の入力

- 検索キー 入力時、一般インデックス、予約語インデックス、数字のみ入力できるインデックスを区分して推奨値を提供します。
- オペレーター 入力時、一般インデックス検索キーの場合は、 == 、 != オプションが下に表示されます。 数字のみ入力できるインデックスの場合、 > 、 < 、 <= 、 >= 、 == 、 != オプションを提供します。
- 検索値 入力時の一致検索(> 、 < 、 <= 、 >= 、 ==)の時は青色で、除外検索(!=)の時は赤色でハイライトします。 検索値 入力時に大文字と小文字の区分オプションを活用して検索することができます。

- フィルタータグが2行以上長くなる場合へ **折りたたみ**アイコンを選択して折りたたむことができます。
- フィルタータグを入力した後、入力画面でShiftとTabキーを同時に使用して、以前のフィルタータグに移動できます。
- フィルタータグを入力した後、入力画面でTABキーを通じて次のフィルタータグに移動できます。

フィルタータグ追加

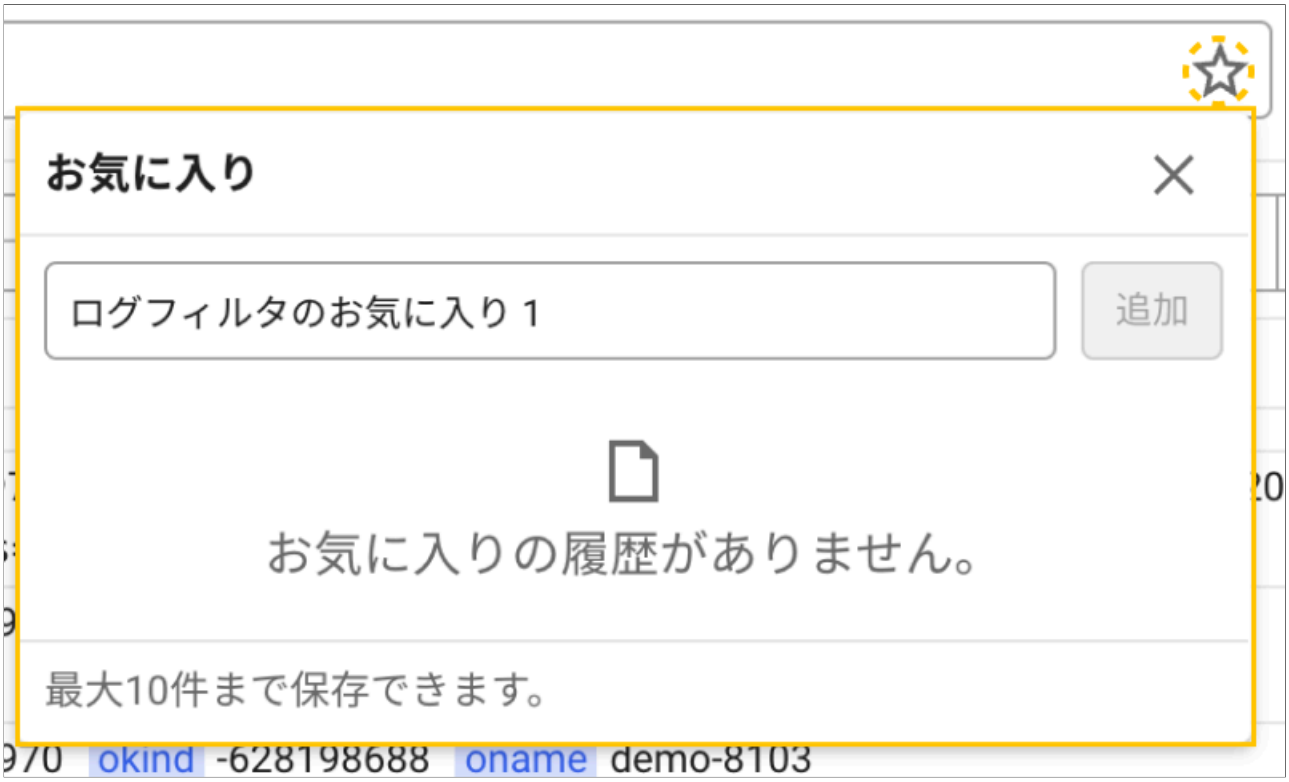
入力フィールドにテキストを入力し、Enter または Tab キーを押すか、入力欄下のガイド UI で推奨値をクリックまたは矢印キーで選択して追加できます。

フィルタータグの削除

単一のタグを削除するには、**Backspace** キーを押すか、タグ上の **X** アイコンをクリックします。 すべてのタグを一度に削除するには、入力フィールド右側の **X** アイコンをクリックします。

フィルターのお気に入り

フィルター**お気に入り**機能を提供します。 入力画面の右側にある☆ **お気に入り**アイコンを選択して、目的のフィルター検索条件をお気に入りに**追加**、**削除**および既存のお気に入りを選択できます。 **お気に入り**は最大50個まで保存されます。



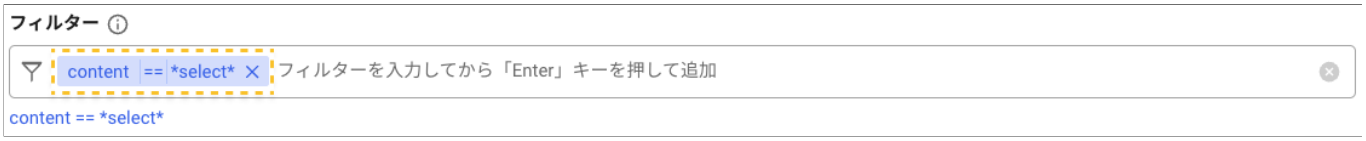
フィルター適用の例外状況

- 数字のみ入力できるインデックス(.n で終わる 検索キー)を入力したタグで、 検索値 は数字のみ入力できます。
- 重複した 検索キー 、 検索値 は入力できません。
- 検索キー 、 検索値 いずれかのタグが存在しない場合は、検索できません。 有効でないタグの場合、灰色と表示します。

> ⓘ **ログTail** 検索キー として category は入力できません。
> 入力フィルター値の下にある数式(expression)は、ログデータを表示するときに適用されるフィルター式のプレビューです。

パーシングされていないキーワードのフィルター適用

ログからパーシングされていない、つまりインデックスが作成されていないキーワードを含むログを照会できます。この場合、指定範囲内のすべてのログをFull Scanします。 そのため、インデックスが生成されたキーに比べて検索速度が多少落ちる可能性があります。 定型化されたログデータの場合、**ログパーサの設定**を通じてインデックスキーを活用した検索をお勧めします。



1. **カテゴリ**を選択します。
2. **フィルター**入力欄に content を選択後、検索キーワードを入力します。
   例） content *select*
3. **検索**ボタンをクリックしてログを照会します。

> ⓘ
> - **ログTail**の場合、すべてのログ検索が可能であるため、カテゴリを指定する必要はありません。
> - パーサ設定の詳細については、次の文書を参考にしてください。

フィルター修正

フィルターに値を入力し、入力した値をクリックすると、その値を修正できます。



- フィルター入力欄にテキストを書き換えて修正できます。
- フィルター入力欄の下に表示される**ガイドUI**から推奨値を選択して修正できます。

検索キー(Search Key)

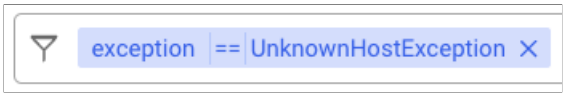
次の画像で青いボックス部分はパーシング(parsing)された検索キーです。 検索キーは**ログ設定**の**ログパーサの設定**タブでパーシングロジックを登録し、設定できます。

検索キー

フィルター入力の文法

タグは検索キーと検索値で構成されています。 次の例で、検索キーは `exception` 、検索値は `UnknownHostException` となります。 この例は、収集したログデータのうち、IPアドレスとドメインアドレスが一致せず、サーバーをホストに接続できない場合に発生する例外(`UnknownHostException`)が含まれたログデータを照会

します。



検索キーの種類

検索キーの種類	検索キーフォーマット	意味	検索キーと検索値の例	検索例
文字列キーワード	keyword	ファイル名	- キー：fileName - 値：/data/whatap/logs/yard.log	fileName:/data/whatap/logs/yard.log
数字キーワード	keyword.n	レスポンス時間	- キー: response_time.n - 値: 2945	response_time.n>=2945
予約語キーワード (辞書定義キーワード)	@keyword	TXID	- キー：@txid - 値：85459614215434144	-
ログ本文キーワード	content	ログ本文	- キー: content - 値: ユーザー入力値	content: `*ERROR*`

Content検索キー

- Content検索キーは、インデックスされていないログの本文を対象に検索します。 例えば、`content *ERROR*` のように入力する場合は、ログ本文中で `ERROR` を含めたログを検索します。
- どのキーワードでインデックスをかけるべきかわからない場合は、Content検索キーを活用して問題となるキーワードを含むログを識別します。 その後、**ログ設定**メニューのログパーサ設定を通じて、該当キーワードでパーサを設定し、インデックスを生成する方式で、検索速度を早くできます。

共通文法

文法の種類	説明	例
==searchValue	検索値と一致するログを検索します。	exception==RuntimeExceptionexception
!=searchValue	検索値を除いたログを検索します。	exception!=RuntimeException
*searchValue	検索値で終わるログを検索します。	word==*hello
searchValue*	検索値で始まるログを検索します。	word==hello*
searchValue	検索値として中間に含まれるログを検索します。	word==*hello*
*search*Value*	検索値として含まれるログを検索します。	word==*he*llo*
re:{正規表現}	正規表現にマッチングされるログを検索します。	caller==re:^i\.w\.a\.w\.s\.v\.r\.
**	検索語に該当するすべてのログを検索します。	

検索キーが数字キーワード(keyword.n)の場合の文法

次の文法は検索キーが、 keyword.n 形式の場合のみに対応します。

- 検索値は数字のみ使えます。
- .n キーワードの値にはPrefixを付けません。 .n 以外のキーワードは、すべてPrefixを付ける必要があります。

  例) +>searchValue は有効ではありません。

文法の種類	説明	例
>searchValue	検索値よりも大きな値を含むログを照会します。	response_time.n>3000
>=searchValue	検索値よりも大きいか、同じ値を含むログを照会します。	response_time.n>=3000
==searchValue	検索値と同じ値を含むログを照会します。	response_time.n==3000

文法の種類	説明	例
!=searchValue	検索値とは異なる値を含むログを照会します。	response_time.n!=3000
<searchValue	検索値より小さい値を含むログを照会します。	response_time.n<3000
<=searchValue	検索値より小さいか、同じ値を含むログを照会します。	response_time.n<=3000

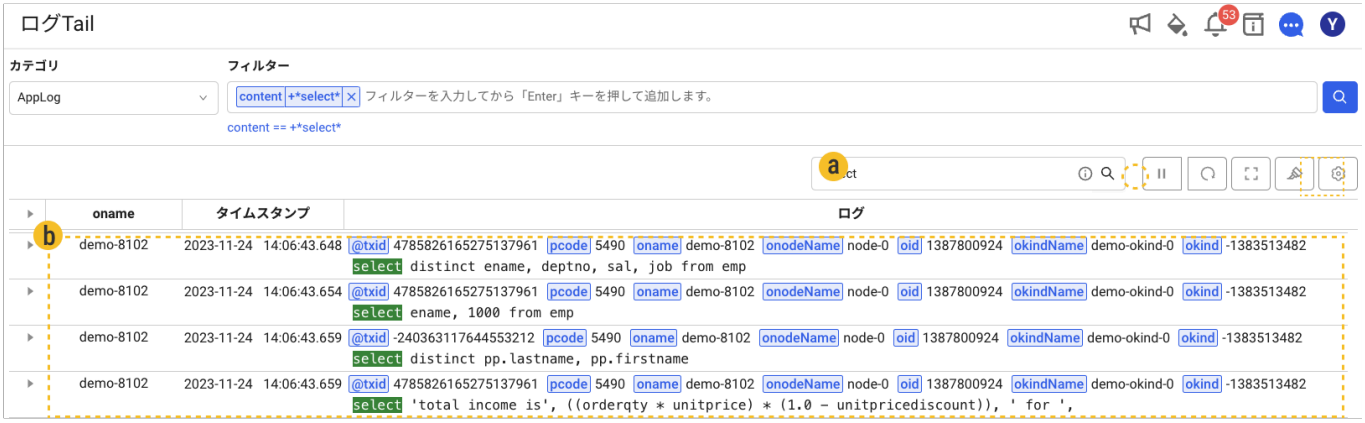
個人情報の照会

ログの個人情報照会権限がある場合**個人情報の表示**トグルボタンの活性化を通じて非識別化処理した個人情報を表示することができます。

> ⓘ
> - **ログの個人情報照会**権限についての詳細は次の文書を参照してください。
> - **ログ個人情報の非識別化**機能についての詳細は次の文書を参照してください。

2 コンテンツハイライト領域

ログのコンテンツ内で目的のキーワードを簡単に識別できるよう、ハイライト機能を提供します。単一または複数のキーワードでフィルタリングできます。



1. ハイライトしたいキーワードをキーワード入力欄に入力します。

- 例: `select`

2. 検索したキーワードがログリストのContent内でハイライト表示されます。
   - ⛶ **拡大表示** アイコンをクリックすると、**ログ** と **タイムスタンプ** を全画面で確認できます。
   - ハイライトされたキーワード間の移動はショートカットで操作できます。
     - `Enter` : 次の行へ移動
     - `Shift+Enter` : 前の行へ移動
     - ∧ / ∨ アイコン: 行の移動
     - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): キーワード検索

複数キーワード条件

複数キーワードでハイライトする場合は、次のように作成します。

入力文字列	説明	結果
`a b c`	各キーワードをスペースで区切ります。	a, b, c
`"Whatap is good."`	キーワードにスペースを含める場合は、`''`または`""`で囲みます。	Whatap is good.
`"Whatap\\ is good."`	""で囲まれたキーワードに、`\`を含める場合は、`\\`と入力する必要があります。	Whatap\ is good.

ハイライトカラー設定

ハイライトするキーワードおよび色を設定できます。

- デフォルトでは、ログレベルに応じたハイライト（FATAL、ERROR、WARN、SEVERE、CRITICAL）が適用されています。
- 設定した内容は**プロジェクト単位**で保存されます。

1. アイコンをクリックします。

2. **ハイライト** ウィンドウで、色を設定するキーワードを入力欄に追加してください。
3. 入力欄の左側の色をクリックすると、色を選択できます。

テーブル設定

カラム設定

2領域の右で **カラム設定**ボタンを選択します。カラムを追加したり、順序を設定できます。

- **カラム追加**

  タグを選択してテーブルにカラムを追加できます。**log**カラム選択を解除する場合**ログ表示詳細設定**が確認できません。必ず1つ以上のカラムを選択します。

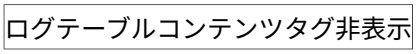
- **カラム順序設定**

  カラムを追加すると**表示項目**に該当カラムが追加されます。希望のカラムをドラッグしてカラムの順序を変更します。

ログ表示詳細設定

2領域右で **ログ表示詳細設定** ボタンを選択します。デフォルトで**content**、**タグ**すべてチェックされており、両方の項目を表示します。**content**と**タグ**どちらかは必ず選択します。

チェックが解除されたアイテムはテーブルに表示されません。次のように**タグ**を解除する場合、テーブルでログの**タグ**は表示されません。



タグ管理一覧にタグを追加すると、追加した順にログのタグがソートされます。タグの順序はドラッグして変更できます。追加したタグを無効にすると、無効にしたタグはログのタグに表示されません。

> **カラムおよびログ表示設定の適用範囲**
>
> - **カラム設定**と**ログ表示詳細設定**は、**ログTail**、**ログ検索**、**ログトレンド**で使用できます。
> - 同一プロジェクト内の**ログTail**、**ログ検索**、**ログトレンド**は、**カラム設定**と**ログ表示詳細設定**を共有します。

ログファイルのダウンロード

検索したログの結果を CSV または TXT ファイル形式でダウンロードできます。

1. 画面中央右側の **ダウンロード** ボタンをクリックします。
2. **CSVダウンロード** または **Logダウンロード** をクリックして、希望の形式のファイルをダウンロードします。

ログエクスプローラー

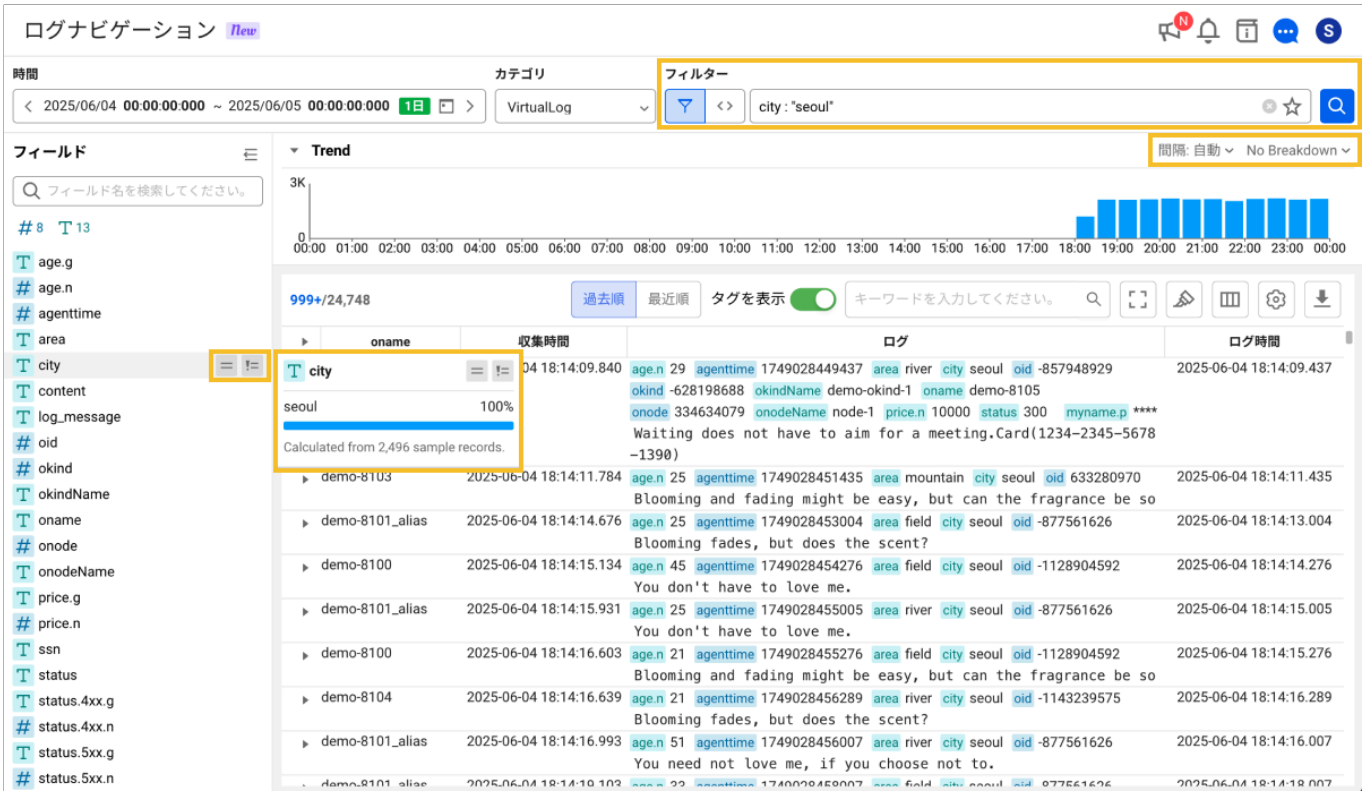
> **ログ参照**権限がない場合は、そのメニューにアクセスできません。

ホーム画面 > プロジェクト選択 >**ログ**> **ログエクスプローラー** *New*

ログエクスプローラー（Log Explorer） 機能は、ログデータを迅速に検索、フィルタリング、分析します。 フィールド構造に関する情報を把握し、クエリベースのフィルタリングによって必要なログを効率的に検索し、結果をデータの可視化を通じて確認できます。 本機能はLuceneインデックスをベースに動作し、主な特徴は以下のとおりです。

1. **ログトレンド分析および検索**: ログのトレンドと検索機能を同時にサポートし、時間経過によるログの変化傾向を把握できます。
2. **高速検索およびフィルタリング**: ユーザーは簡単なキーワード入力だけで、問題のあるログを迅速に検索できます。 WhaTapログ検索クエリおよびLuceneクエリ言語に対応しており、さまざまな検索条件を設定できます。
3. **オートコンプリート機能**：クエリ入力時にオートコンプリート機能が提供され、ユーザーが簡単にクエリを作成し、習得できるようサポートします。
4. **データ可視化**：スタックグラフを活用して時系列のログ件数データを視覚的に表示し、ドリルダウン機能を通じて特定のログを深く分析できます。
5. **お気に入りと共有**：よく使うフィルターはお気に入りとして保存でき、URLを通じて他のユーザーと簡単に共有できます。

基本画面ガイド



時間設定

上部のオプションバーにある**タイムセレクター**を使用して、ログ検索の時間範囲を設定できます。

カテゴリ

上部のオプションバーにある**カテゴリ選択**で、探索したいログの種類を選択します。（例：AppLogなど）

フィルター

上部オプションバーの **フィルター** でクエリ言語を選択し、検索条件を入力してフィルターを適用します。
検索クエリ構文は、WhaTapログ検索クエリとLuceneクエリの両方をサポートしています。

- : WhaTapログ検索クエリ
- : Luceneクエリ

検索クエリ構文

WhaTap log search query

WhaTapのログ検索クエリはオートコンプリート機能を提供しており、簡単なキーワード入力だけで問題のログを迅速に検索できます。 and、orなどのさまざまな構文を活用して、さまざまな検索条件を入力できます。

- 例、検索フィルターに `error` と入力するだけで、content（ログ本文）を検索できます。

    1. 上部オプションバーでフィルターアイコンをクリックした後、クエリ検索ウィンドウでフィールド（選択した時間とカテゴリに応じたリストが表示されます）を選択してください。
    2. 演算子を選択してください。

    a. 基本: `:` , `:*` 、 `and` 、 `or`

    b. 選択したフィールドが `Number` 型の場合（基本 + `<=` 、 `>=` 、 `<` 、 `>` ）

    c. 選択したフィールドが `String` の場合（基本のみ）

    3. 値（選択した時間、カテゴリ、フィールドに一致するリストが表示されます）を選択してください。

検索インデックスフィールドタイプ

サポートされているフィールドタイプは、 `Number` 、 `String` 、 `Boolean` の3種類のみです。

- `Number` : Long、double、float 型の数値
- `String` : メール本文のような全文テキスト
- `Boolean` : True または False の値

> WhaTapログ検索クエリの構文に関する詳細は次の文書を参照してください。

Lucene

Apache Lucene 8.11.1バージョンのオープンソースの構文がそのまま利用可能です。 ただし、自動補完機能は提供していません。

> LuceneのQueryParserクラシックパッケージに関する詳細は公式ドキュメントを参照してください。

フィールド一覧

ドリルダウン分析に活用できます。

フィールド情報のフィルターを反映する

フィールド一覧でフィールド名を確認し、 = または != ボタンをクリックして条件をフィルターに適用してください。

- フィールドにマウスオーバーすると、 = 、 != ボタンが表示され、クリックするとフィルターに反映されます。
- フィールド検索機能を使って、目的のフィールドを素早く見つけることができます。

Trendチャート

Trendチャートで**インターバル**や**ブレイクダウン**を適用して、ログをグラフで確認できます。

- **グラフインタラクション**: マウスドラッグや 🔍 ボタンで時間のズームイン、バークリックで該当領域のログを確認可能
- **インターバル**: 自動、1秒、5秒、30秒、1分、5分、10分、30分、1時間から選択可能
- **Breakdown**: 特定のフィールドを基準にTop 3 + Othersの値を分布として可視化し、ドリルダウンフィルターの適用が可能

breakdown

Breakdownは、ログ件数の推移を特定のフィールドで分析する可視化機能です。

- 特定のフィールドのTop 3の値とOthers（その他の値）を区別して、スタックグラフで分布を表示します。
- ドリルダウン分析に活用でき、凡例やスタックグラフの特定領域（部分グラフ）をクリックして、特定の値のみを

強調したり、フィルター条件を追加することができます。

ログの確認

検索条件に一致するログの一覧を確認でき、**フィールド一覧**ではフィールド情報とログタグが同じ色で表示されます。

コンテンツハイライト

ログのコンテンツ内で目的のキーワードを簡単に識別できるよう、ハイライト機能を提供します。単一または複数のキーワードでフィルタリングできます。

1. ハイライトしたいキーワードをキーワード入力欄に入力します。
   - 例: `select`
2. 検索したキーワードがログリストのContent内でハイライト表示されます。
   - ⛶ **拡大表示** アイコンをクリックすると、**ログ** と **タイムスタンプ** を全画面で確認できます。
   - ハイライトされたキーワード間の移動はショートカットで操作できます。
     - `Enter` : 次の行へ移動
     - `Shift+Enter` : 前の行へ移動
     - ∧ / ∨ アイコン: 行の移動
     - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): キーワード検索

複数キーワード条件

複数キーワードでハイライトする場合は、次のように作成します。

入力文字列	説明	結果
`a b c`	各キーワードをスペースで区切ります。	a, b, c
`"Whatap is good."`	キーワードにスペースを含める場合は、`''`または`""`で囲みます。	Whatap is good.

入力文字列	説明	結果
`"Whatap\\ is good."`	""で囲まれたキーワードに、`\`を含める場合は、`\\`と入力する必要があります。	Whatap\ is good.

ハイライトカラー設定

ハイライトするキーワードおよび色を設定できます。

- デフォルトでは、ログレベルに応じたハイライト（FATAL、ERROR、WARN、SEVERE、CRITICAL）が適用されています。
- 設定した内容は**プロジェクト単位**で保存されます。

1. アイコンをクリックします。
2. **ハイライト** ウィンドウで、色を設定するキーワードを入力欄に追加してください。
3. 入力欄の左側の色をクリックすると、色を選択できます。

カラム設定

ボタンをクリックして、カラムを追加したり順序を設定することができます。

- **カラム追加**

  カラム追加：タグを選択して、テーブルにカラムを追加できます。

  > ログカラムの選択を解除すると、**ログ表示詳細設定**を確認できません。 必ず1つ以上のカラムを選択します。

- **カラム順序設定**

  カラム設定でカラムを追加すると、**表示項目**に該当カラムが追加されます。 希望するカラムをドラッグして、カラムの順序を変更してください。

ログ表示詳細設定

ボタンをクリックして、ログの表示を詳細に設定します。 デフォルトでは **content**、**タグ** の両方がチェックされており、両方の項目が表示されます。

- **content**と**タグ**のいずれかを必ず選択してください。
- **タグ管理**リストにタグを追加すると、追加した順にログのタグが表示されます。 タグの順序はドラッグで変更できます。 追加したタグを無効にすると、無効にしたタグはログのタグに表示されません。
- チェックが解除された項目は、テーブルに表示されません。 次のように**タグ**を解除する場合、テーブルからログの**タグ**は表示されません。

  log table contents remove tag sc

> **カラムおよびログ表示設定の適用範囲**
>
> - **カラム設定**と**ログ表示詳細設定**は、**ログTail**、**ログ検索**、**ログトレンド**で使用できます。
> - 同一プロジェクト内の**ログTail**、**ログ検索**、**ログトレンド**は、**カラム設定**と**ログ表示詳細設定**を共有します。

ログファイルのダウンロード

検索したログの結果を CSV または TXT ファイル形式でダウンロードできます。

1. 画面中央右側の **ダウンロード** ボタンをクリックします。
2. **CSVダウンロード** または **Logダウンロード** をクリックして、希望の形式のファイルをダウンロードします。

WhaTap log search query 文法

> これは、WhaTapログエクスプローラーで使用できるWhaTapログ検索クエリ構文に関するドキュメントです。

1. **Equal**
   - `field : value`
2. **Not**
   - `not field : value`
3. **Wildcard** (`String` タイプのフィールドのみ使用可能、検索値の前後に `"` は不)
   - `field : val*`
4. **Exist**
   - `field : *`
5. **Range** (`Number` タイプのフィールドのみ使用可能)
   - `<` , `>` , `<=` , `>=` が使用可能
   - 例: `field < 30 and field >= 10`
6. **Combining Multiple Queries**
   - AND 例: `field < 30 and field >= 10`
   - OR 例: `field < 30 or field >= 10`
7. **Grouping**
   - 例: `(field < 30 or field >= 10) and other : 70`
8. **Escaping Special Characters**
   - Luceneでエスケープが必要な特殊文字一覧: `&&` `||` `!` `(` `)` `{` `}` `^` `"` `~` `*` `?` `:` `\` `/`
   - 文字の前に `\` を付けてエスケープする
   - 例: `(1+1):2` を検索するには `\(1\+1\)\:2`

検索インデックスフィールドタイプ

サポートされるフィールドタイプは `Number` と `String` の2種類のみです。

- `Number` : Long、double、floatタイプの数値値
  - 使用例： `field : "value"`
- `String` : メール本文のような全文テキスト
  - 使用例： `field : value`
- `Boolean` : True または False
  - `field : true` または `field : false` の形式で使用します

ログトレンド

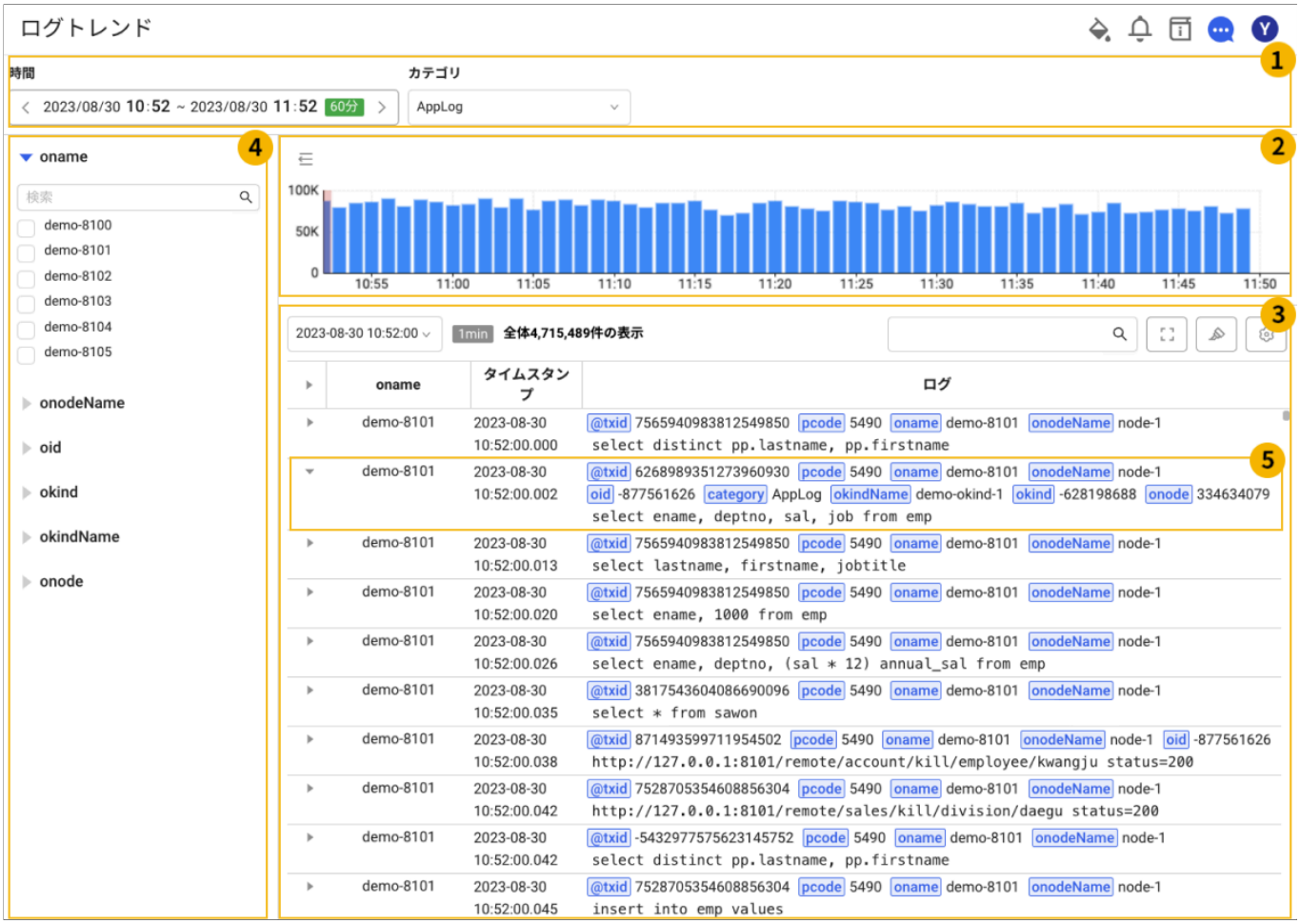
> **ログ参照**権限がない場合は、そのメニューにアクセスできません。

ホーム画面 > プロジェクト選択 > **ログ** > **ログトレンド**

ログトレンド では、タイプ別に分類されたログの発生件数の推移を通じて、特定エラータイプの発生パターンを確認し、時間別の詳細ログデータを確認できます。ハイライト機能により、目的のログをすばやく識別できます。カテゴリ別に収集されたログの推移を照会できます。

主な用語は次のとおりです。

- **Category**：ログの収集および照会単位
- **Content**：ログメッセージ
- **Search Key**：ログパーサーの設定によって生成される検索キー
- **Tag**：収集されたログを検索できるタグ

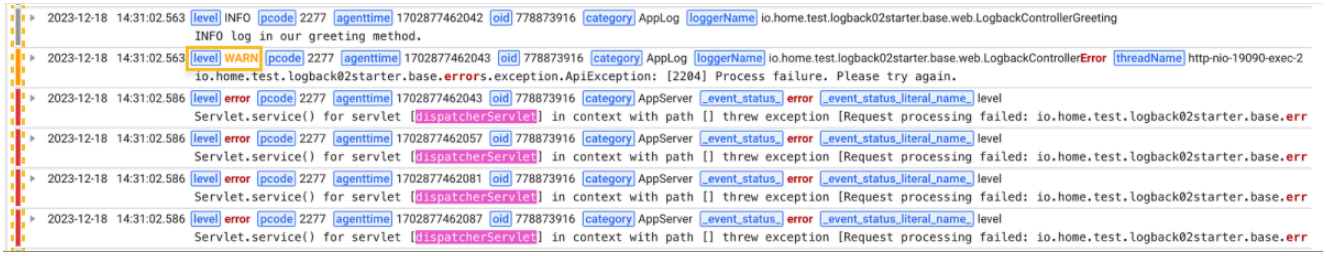


データ参照

- 下部までスクロールすると、次のデータを表示します。
- 1で検索するログデータの**時間**と収集単位として**カテゴリ**を指定します。
- **カテゴリ**を変更すると、選択されたカテゴリに対応するログを表示します。2バーチャートと3ログテーブルで確認できます。
- 2棒グラフのバーをクリックすると、バーの時間範囲に対応するログを検索します。
- 3ログテーブルの左上から照会されたログの総数を確認できます。
- 3 ログテーブル上段の右の [ ] **全画面** アイコンを選択すると、 **ログ**と**収集時間**を全画面で確認できます。
- 4 サイドメニューからタグでフィルタリングすることでログを確認できます。 検索キーは最大2つまで選択でき、

検索値は複数選択できます。

- **エージェントオプションが設定されている場合**、ログレベルを収集し、各ログの左側に色付きの縦線が次のように表示されます。



> **エージェントのオプション設定**
>
> ```
> # whatap.conf
> weaving=log4j-2.17
> weaving=logback-1.2.8
> ```
>
> - Javaエージェント2.2.22バージョンからウィービング設定にlog4j-2.17またはlogback-1.2.8設定時に使用できます。 エージェントの再起動が必要です。
> - ログレベルは、パーシングされたキーワードの中で `level` 、 `type` 基準で判別します。 `level` 、 `type` にパーシングされたキーが存在し、パーシングの値がFATAL, CRITICAL、ERROR、WARN、WARNING、INFOを含む場合は、ログレベルの色を表示します。

> ログテーブルカラムの端をドラッグしてカラムの幅を修正できます。

個人情報の照会

ログの個人情報照会権限がある場合**個人情報の表示** トグルボタンの活性化を通じて非識別化処理した個人情報を表示することができます。

> - **ログの個人情報照会**権限についての詳細は次の文書を参照してください。
> - **ログ個人情報の非識別化**機能についての詳細は次の文書を参照してください。

ログContentの確認

Content はログメッセージです。 ログカラムの1行目には、パースされたキーと値が表示され、2行目にはContent（ログメッセージ）が表示されます。

- 3 ログテーブルの行(ログ)ごとに ▸ **詳細**ボタンがあります。 ▸ **詳細**ボタンを選択すると5のように、ログの全体contentを確認できます。

ハイライト設定

目的のキーワードを簡単に識別できるように、ハイライト機能を提供しています。 アイコンをクリックし、キーワードを入力してからEnterキーを押して追加してください。

コンテンツハイライト

ログのコンテンツ内で目的のキーワードを簡単に識別できるよう、ハイライト機能を提供します。単一または複数のキーワードでフィルタリングできます。

1. ハイライトしたいキーワードをキーワード入力欄に入力します。
   - 例: `select`
2. 検索したキーワードがログリストのContent内でハイライト表示されます。
   - **拡大表示** アイコンをクリックすると、**ログ** と **タイムスタンプ** を全画面で確認できます。
   - ハイライトされたキーワード間の移動はショートカットで操作できます。
     - `Enter` : 次の行へ移動
     - `Shift+Enter` : 前の行へ移動
     - ∧ / ∨ アイコン: 行の移動
     - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): キーワード検索

複数キーワード条件

複数キーワードでハイライトする場合は、次のように作成します。

入力文字列	説明	結果
`a b c`	各キーワードをスペースで区切ります。	a, b, c
`"Whatap is good."`	キーワードにスペースを含める場合は、`''`または`""`で囲みます。	Whatap is good.
`"Whatap\\ is good."`	""で囲まれたキーワードに、`\`を含める場合は、`\\`と入力する必要があります。	Whatap\ is good.

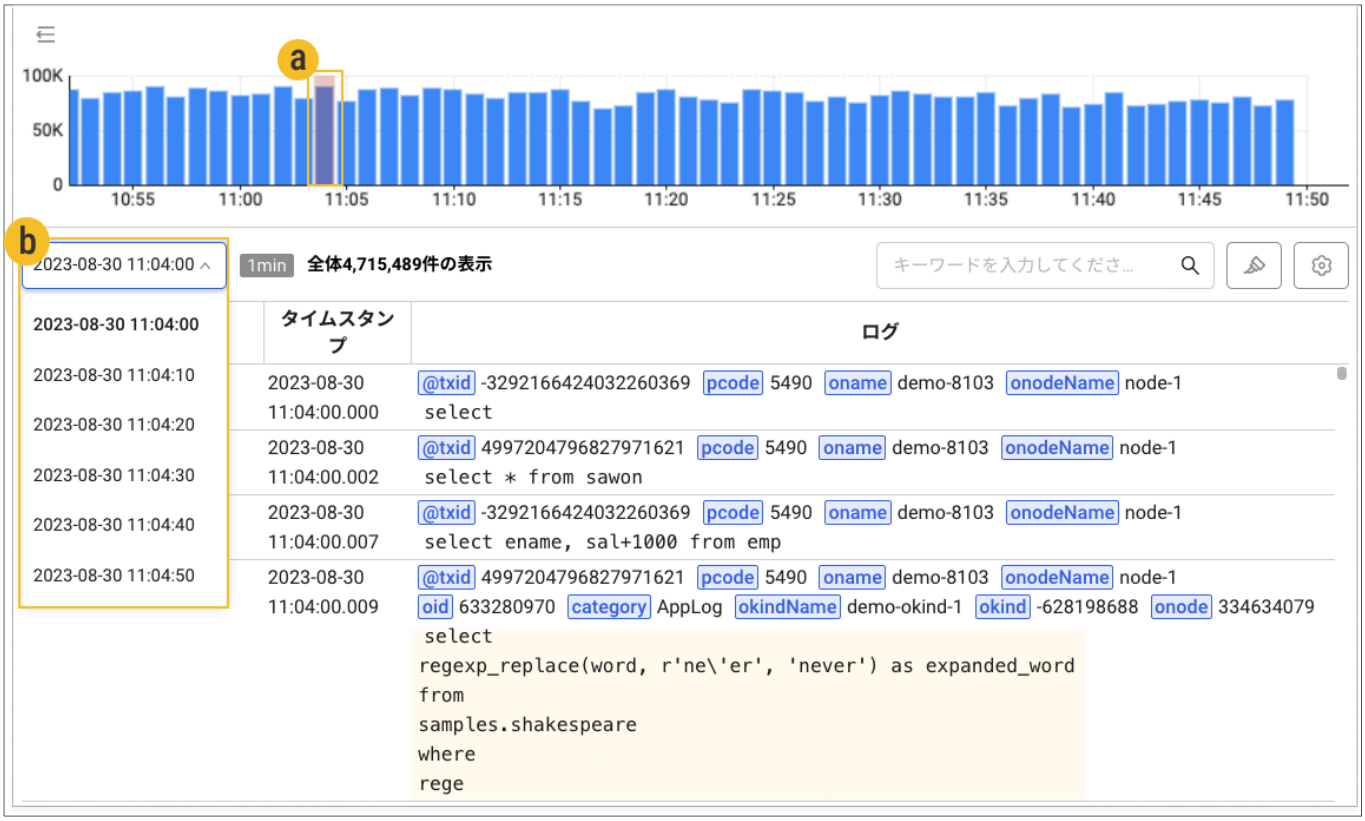
ハイライトカラー設定

ハイライトするキーワードおよび色を設定できます。

- デフォルトでは、ログレベルに応じたハイライト（FATAL、ERROR、WARN、SEVERE、CRITICAL）が適用されています。
- 設定した内容は**プロジェクト単位**で保存されます。

1. アイコンをクリックします。
2. **ハイライト** ウィンドウで、色を設定するキーワードを入力欄に追加してください。
3. 入力欄の左側の色をクリックすると、色を選択できます。

ログをチャートで検索



- バーチャートでa目的の時間領域バーをクリックまたはドラッグして、その時間範囲のログを確認できます。
- 棒チャートの下のログテーブル上部左側のb時間選択オプションを使用して、次のように選択した時間帯でより詳細なログ検索が可能です。
  - 1min：interval（チャート内の棒の間隔）
  - 時間選択オプション：選択された時間範囲を6つの間隔に分割した時間帯

テーブル設定

カラム設定

3領域の右で **カラム設定**ボタンを選択します。カラムを追加したり、順序を設定できます。

- **カラム追加**

  タグを選択してテーブルにカラムを追加できます。**log**カラム選択を解除する場合**ログ表示詳細設定**が確認できません。必ず1つ以上のカラムを選択します。

- **カラム順序設定**

  カラムを追加すると**表示項目**に該当カラムが追加されます。希望のカラムをドラッグしてカラムの順序を変更します。

ログ表示詳細設定

3領域右で **ログ表示詳細設定** ボタンを選択します。デフォルトで**content**、**タグ**すべてチェックされており、両方の項目を表示します。**content**と**タグ**どちらかは必ず選択します。

チェックが解除されたアイテムはテーブルに表示されません。次のように**タグ**を解除する場合、テーブルでログの**タグ**は表示されません。

ログテーブルコンテンツタグ非表示

タグ管理一覧にタグを追加すると、追加した順にログのタグがソートされます。タグの順序はドラッグして変更できます。追加したタグを無効にすると、無効にしたタグはログのタグに表示されません。

> **カラムおよびログ表示設定の適用範囲**
>
> - **カラム設定**と**ログ表示詳細設定**は、**ログTail**、**ログ検索**、**ログトレンド**で使用できます。
> - 同一プロジェクト内の**ログTail**、**ログ検索**、**ログトレンド**は、**カラム設定**と**ログ表示詳細設定**を共有します。

ログファイルのダウンロード

検索したログの結果を CSV または TXT ファイル形式でダウンロードできます。

1. 画面中央右側の **ダウンロード** ボタンをクリックします。
2. **CSVダウンロード** または **Logダウンロード** をクリックして、希望の形式のファイルをダウンロードします。

ログの検索

> **ログ参照**権限がない場合は、そのメニューにアクセスできません。

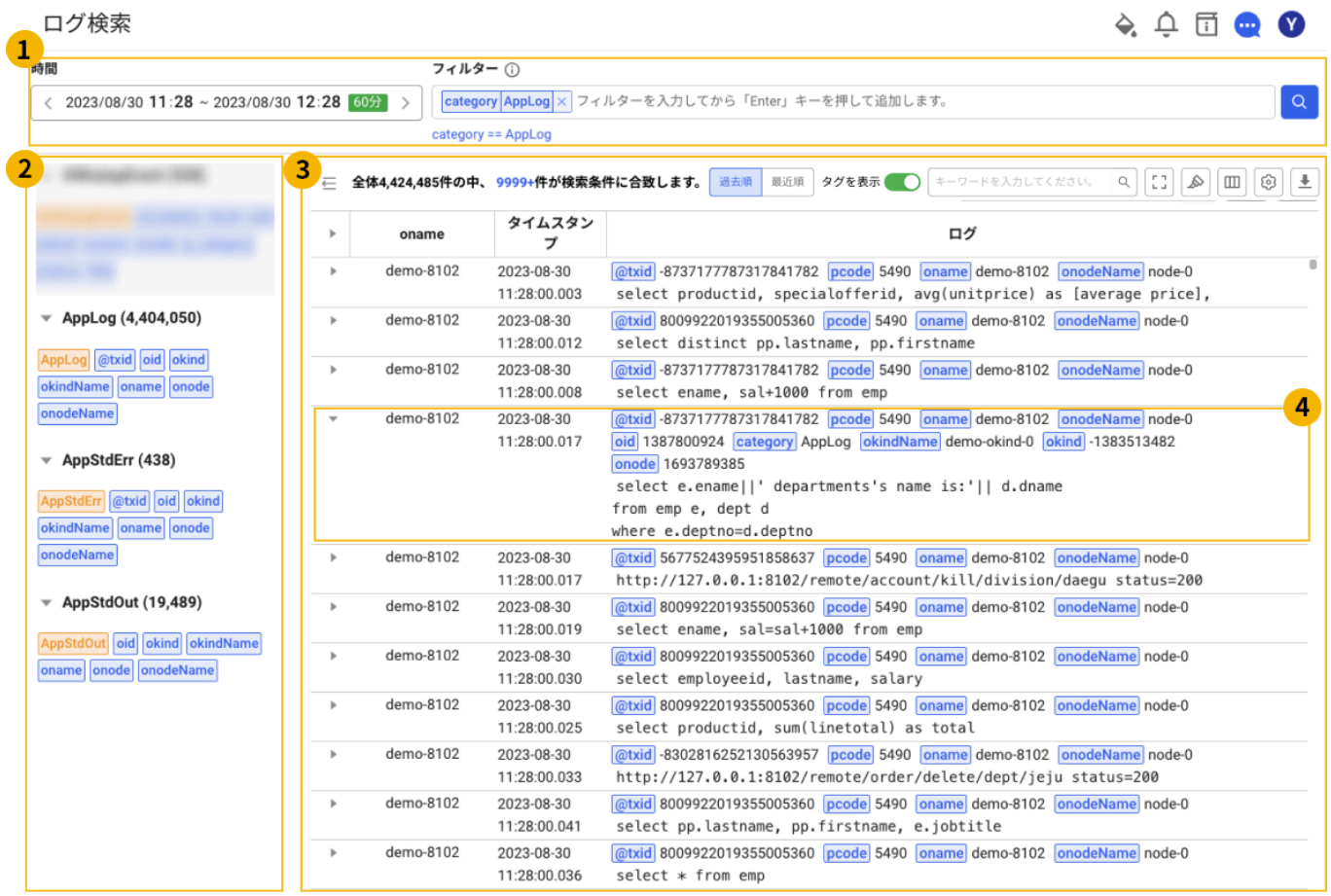
ホーム画面 > プロジェクト選択 > **ログ** > **ログ検索**

ログ検索メニューから収集された大量のログをさまざまな条件で検索し、ユーザーが必要なログを指定できます。複数の検索条件をパーシングされたキーとバリューで指定できるため、目的の条件に一致するログデータのみを抽出します。

動的ページで取得したログデータは、設定された行単位で取得され、スクロールで画面の下まで移動すると自動的に次のデータが取得されて表示されます。

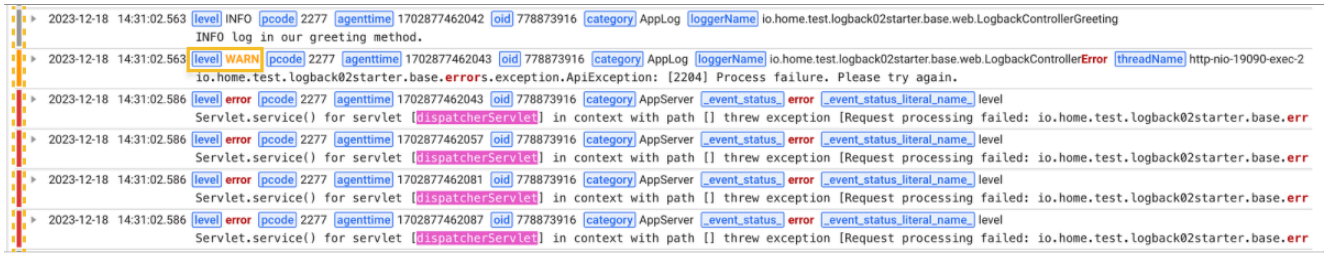
主な用語は次のとおりです。

- **Category**：ログの収集および照会単位
- **Content**：ログメッセージ
- **Search Key**：ログパーサーの設定によって生成される検索キー
- **Tag**：収集されたログを検索できるタグ



データ参照

- 画面の下までスクロールすると、次のデータを検索します。一度に、10,000個のログを参照できます。
- 3ログテーブルの左上に検索されたログの総数を確認することができます。
- ログデータを時間の順と逆順で照会できます。3ログテーブルの右上の**古い順**と**新しい順**の中から照会方式を選択します。
- 時間範囲を指定した後**適用**ボタンを選択して照会時間を設定し、**検索**ボタンを選択してデータを照会します。
- 3ログテーブルの右上の**拡大表示**アイコンを選択すると、**ログ**と**タイムスタンプ**を全画面で確認できます。
- **エージェントオプションが設定されている場合**、ログレベルを収集し、各ログの左側に色付きの縦線が次のように表示されます。



エージェントのオプション設定

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- Javaエージェント2.2.22バージョンからウィービング設定にlog4j-2.17またはlogback-1.2.8設定時に使用できます。 エージェントの再起動が必要です。
- ログレベルは、パーシングされたキーワードの中で `level` 、 `type` 基準で判別します。 `level` 、 `type` にパーシングされたキーが存在し、パーシングの値がFATAL, CRITICAL、ERROR、WARN、WARNING、INFOを含む場合は、ログレベルの色を表示します。

ログテーブルカラムの端をドラッグしてカラムの幅を修正できます。

ログContentの確認

Content はログメッセージです。 ログカラムの1行目には、パースされたキーと値が表示され、2行目にはContent（ログメッセージ）が表示されます。

- 3 ログテーブルの行(ログ)ごとに ▸ **詳細**ボタンがあります。 ▸ **詳細**ボタンを選択すると4のように、ログの全体Contentを確認できます。
- ログのタグを選択すると、**コピー, 検索, 除外検索, 隣接ログ**を検索できるドロップダウンメニューが表示されます。

フィルター

フィルター適用

1左側の**時間選択バー**で時間範囲を指定できます。 右側から、フィルターを適用すると、入力した条件に合ったログをフィルタリングします。 複数のフィルターを入力できます。 フィルターのタグが同じ場合、OR(`||`)、 そうでない場合はAND(`&&`)が適用されます。

入力欄に文字を直接入力するか、**フィルター**入力欄をクリックしてフィルターを指定します。 フィルタータグは `検索キー` 、 `オペレーター` 、 `検索値` の順に入力します。 **検索** ボタンを選択すると、フィルターが適用されたデータをログ一覧で確認できます。

> **ガイドUI**
>
> 次のように入力ウィンドウの下に**ガイドUI**を提供します。 入力画面またはフィルタータグにマウスカーソルがある場合、**ガイドUI**を閉じる時はESCキーで操作できます。



検索キー、オペレーター、検索値の入力

- 検索キー 入力時、一般インデックス、予約語インデックス、数字のみ入力できるインデックスを区分して推奨値を提供します。
- オペレーター 入力時、一般インデックス検索キーの場合は、 == 、 != オプションが下に表示されます。 数字のみ入力できるインデックスの場合、 > 、 < 、 <= 、 >= 、 == 、 != オプションを提供します。
- 検索値 入力時の一致検索(> 、 < 、 <= 、 >= 、 ==)の時は青色で、除外検索(!=)の時は赤色でハイライトします。 検索値 入力時に大文字と小文字の区分オプションを活用して検索することができます。

- フィルタータグが2行以上長くなる場合へ **折りたたみ**アイコンを選択して折りたたむことができます。
- フィルタータグを入力した後、入力画面でShiftとTabキーを同時に使用して、以前のフィルタータグに移動できます。
- フィルタータグを入力した後、入力画面でTABキーを通じて次のフィルタータグに移動できます。

フィルタータグ追加

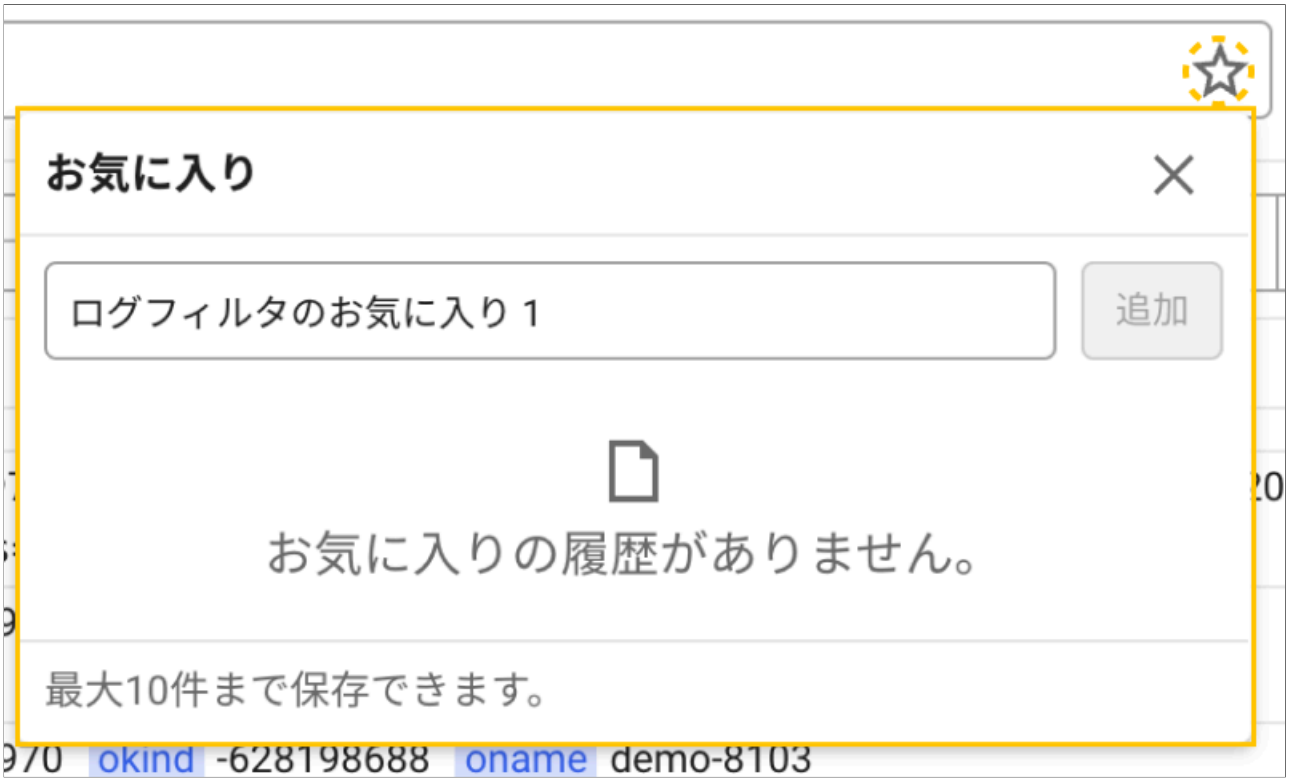
入力フィールドにテキストを入力し、Enter または Tab キーを押すか、入力欄下のガイド UI で推奨値をクリックまたは矢印キーで選択して追加できます。

フィルタータグの削除

単一のタグを削除するには、**Backspace** キーを押すか、タグ上の **X** アイコンをクリックします。 すべてのタグを一度に削除するには、入力フィールド右側の **X** アイコンをクリックします。

フィルターのお気に入り

フィルター**お気に入り**機能を提供します。 入力画面の右側にある☆ **お気に入り**アイコンを選択して、目的のフィルター検索条件をお気に入りに**追加**、**削除**および既存のお気に入りを選択できます。 **お気に入り**は最大50個まで保存されます。

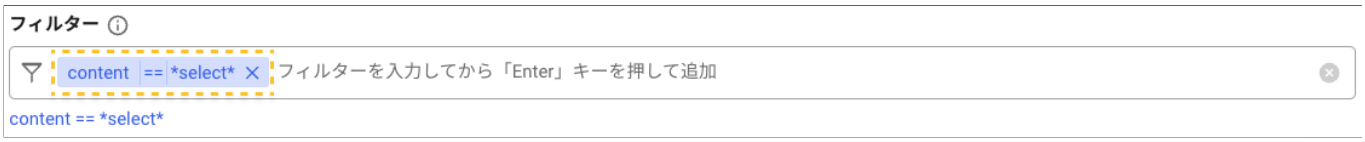


フィルター適用の例外状況

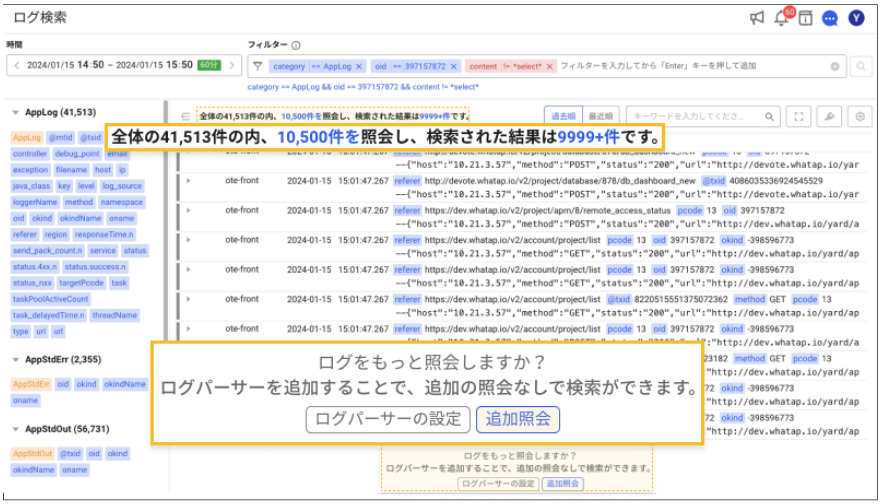
- 数字のみ入力できるインデックス(.n で終わる 検索キー)を入力したタグで、 検索値 は数字のみ入力できます。
- 重複した 検索キー 、 検索値 は入力できません。
- 検索キー 、 検索値 いずれかのタグが存在しない場合は、検索できません。 有効でないタグの場合、灰色と表示します。

構文解析されていないキーワードのフィルター適用

ログからパーシングされていない、つまりインデックスが作成されていないキーワードを含むログを照会できます。この場合、指定範囲内のすべてのログをFull Scanします。 そのため、インデックスが生成されたキーに比べて検索速度が多少落ちる可能性があります。 定型化されたログデータの場合、**ログパーサの設定**を通じてインデックスキーを活用した検索をお勧めします。



1. **カテゴリ**を選択します。 カテゴリ指定が必須です。
2. **フィルター**入力欄に content を選択後、検索キーワードを入力します。
   例） content *select*
3. **検索**ボタンをクリックしてログを照会します。 ログ全体の一部を先に照会します。 1回あたりの検索結果は最大1万件です。
4. スクロールを下げて下段の**追加照会**ボタンを選択すると、つづけて照会できます。



- 全体ログのうち、サーバー照会範囲当たり1万件ずつ照会します。 サーバー照会範囲の場合、基本20万件ですが、ログ全体の量によって比率が変わることがあります。
- パーサ設定の詳細については、次の文書を参考にしてください。

フィルター修正

フィルターに値を入力し、入力した値をクリックすると、その値を修正できます。



- フィルター入力欄にテキストを書き換えて修正できます。
- フィルター入力欄の下に表示される**ガイドUI**から推奨値を選択して修正できます。

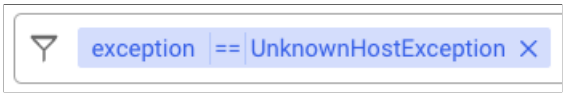
検索キー(Search Key)

検索キーは、ログデータ内で希望する特定の値にアクセスするための識別子を意味します。 検索キーに該当する実際のデータが検索値です。 左側の2領域のタグはカテゴリ別にパーシング(parsing) された検索キーです。 タグを選択してフィルターを入力します。 オレンジ色タグはカテゴリで、青色タグは検索キーです。

例えば2領域のAppLogとAppStdOutはカテゴリであり、その下のoidなどのタグは、パーシング(parsing) された検索キーです。 検索キーは**ログ** > **ログ設定**の**ログパーサの設定**タブでパーシングロジックを登録し、設定できます。

フィルター入力文法

タグは検索キーと検索値で構成されています。 次の例で、検索キーは `exception` 、検索値は `UnknownHostException` となります。 この例は、収集したログデータのうち、IPアドレスとドメインアドレスが一致せず、サーバーをホストに接続できない場合に発生する例外(`UnknownHostException`)が含まれたログデータを照会します。



検索キーの種類

検索キーの種類	検索キーフォーマット	意味	検索キーと検索値の例	検索例
文字列キーワード	keyword	ファイル名	- キー：fileName - 値：/data/whatap/logs/yard.log	fileName:/data/whatap/logs/yard.log
数字キーワード	keyword.n	レスポンス時間	- キー: response_time.n - 値: 2945	response_time.n>=2945
予約語キーワード (辞書定義キーワード)	@keyword	TXID	- キー：@txid - 値：85459614215434144	-

共通文法

文法の種類	説明	例
`==searchValue`	検索値と一致するログを検索します。	`exception==RuntimeExceptionexception`
`!=searchValue`	検索値を除いたログを検索します。	`exception!=RuntimeException`
`*searchValue`	検索値で終わるログを検索します。	`word==*hello`
`searchValue*`	検索値で始まるログを検索します。	`word==hello*`
`*searchValue*`	検索値として中間に含まれるログを検索します。	`word==*hello*`
`*search*Value*`	検索値として含まれるログを検索します。	`word==*he*llo*`

文法の種類	説明	例
`re:{正規表現}`	正規表現にマッチングされるログを検索します。	`caller==re:^i\.w\.a\.w\.s\.v\.r\.`
`**`	検索語に該当するすべてのログを検索します。	

検索キーが数字キーワード(keyword.n)の場合の文法

次の文法は検索キーが、`keyword.n`形式の場合のみに対応します。

- 検索値は数字のみ使えます。
- `.n`キーワードの値にはPrefixを付けません。`.n`以外のキーワードは、すべてPrefixを付ける必要があります。

  例) `+>searchValue`は有効ではありません。

文法の種類	説明	例
`>searchValue`	検索値よりも大きな値を含むログを照会します。	`response_time.n>3000`
`>=searchValue`	検索値よりも大きいか、同じ値を含むログを照会します。	`response_time.n>=3000`
`==searchValue`	検索値と同じ値を含むログを照会します。	`response_time.n==3000`
`!=searchValue`	検索値とは異なる値を含むログを照会します。	`response_time.n!=3000`
`<searchValue`	検索値より小さい値を含むログを照会します。	`response_time.n<3000`
`<=searchValue`	検索値より小さいか、同じ値を含むログを照会します。	`response_time.n<=3000`

個人情報の照会

ログの個人情報照会権限がある場合**個人情報の表示**トグルボタンの活性化を通じて非識別化処理した個人情報を表示することができます。

> ⓘ • **ログの個人情報照会**権限についての詳細は次の文書を参照してください。

- **ログ個人情報の非識別化**機能についての詳細は次の文書を参照してください。

ログのタグオプション

ログのタグを選択すると、ドロップダウンメニューが表示されます。**検索**、 **除外検索**、**隣接ログ**、**トランザクション情報**オプションが確認できます。



ログタグオプションメニュー	説明
コピー	該当するタグのログをコピーします。
検索	**フィルター**には、該当のタグを含む（'=='）条件で入力されます。
除外検索	**フィルター**には、該当のタグを除外する（'!='）条件で入力されます。
隣接ログ	**隣接ログ** の詳細画面では、選択したログのサーバーを対象に、選択したログと隣接する時間帯のログを照会できます。時間を選択して、周辺の時間帯に発生したログを確認することも可能です。基準となるログは青い背景で表示されます。
トランザクション情報	ログに `@txid` や `@mtid` タグが含まれている場合は、**トランザクション情報** をクリックすると、**トランザクション情報** の詳細画面で内容を確認できます。

コンテンツハイライト

ログのコンテンツ内で目的のキーワードを簡単に識別できるよう、ハイライト機能を提供します。単一または複数のキーワードでフィルタリングできます。

1. ハイライトしたいキーワードをキーワード入力欄に入力します。
   - 例: `select`
2. 検索したキーワードがログリストのContent内でハイライト表示されます。
   - ⛶ **拡大表示** アイコンをクリックすると、**ログ** と **タイムスタンプ** を全画面で確認できます。
   - ハイライトされたキーワード間の移動はショートカットで操作できます。
     - `Enter` : 次の行へ移動
     - `Shift+Enter` : 前の行へ移動
     - ∧ / ∨ アイコン: 行の移動
     - `Cmd` + `F` (Mac) / `Ctrl` + `F` (Windows): キーワード検索

複数キーワード条件

複数キーワードでハイライトする場合は、次のように作成します。

入力文字列	説明	結果
`a b c`	各キーワードをスペースで区切ります。	a, b, c
`"Whatap is good."`	キーワードにスペースを含める場合は、`''` または `""` で囲みます。	Whatap is good.
`"Whatap\\ is good."`	""で囲まれたキーワードに、`\` を含める場合は、`\\` と入力する必要があります。	Whatap\ is good.

ハイライトカラー設定

ハイライトするキーワードおよび色を設定できます。

- デフォルトでは、ログレベルに応じたハイライト（FATAL、ERROR、WARN、SEVERE、CRITICAL）が適用されています。
- 設定した内容は**プロジェクト単位**で保存されます。

1. アイコンをクリックします。
2. **ハイライト** ウィンドウで、色を設定するキーワードを入力欄に追加してください。
3. 入力欄の左側の色をクリックすると、色を選択できます。

テーブル設定

カラム設定

3領域の右で **カラム設定**ボタンを選択します。カラムを追加したり、順序を設定できます。

- **カラム追加**

  タグを選択してテーブルにカラムを追加できます。**log**カラム選択を解除する場合**ログ表示詳細設定**が確認できません。必ず1つ以上のカラムを選択します。

- **カラム順序設定**

  カラムを追加すると**表示項目**に該当カラムが追加されます。希望のカラムをドラッグしてカラムの順序を変更します。

ログ表示詳細設定

3領域右で **ログ表示詳細設定** ボタンを選択します。デフォルトで**content**、**タグ**すべてチェックされており、両方の項目を表示します。**content**と**タグ**どちらかは必ず選択します。

チェックが解除されたアイテムはテーブルに表示されません。次のように**タグ**を解除する場合、テーブルでログの**タグ**は表示されません。

ログテーブルコンテンツタグ非表示

タグ管理一覧にタグを追加すると、追加した順にログのタグがソートされます。タグの順序はドラッグして変更できます。追加したタグを無効にすると、無効にしたタグはログのタグに表示されません。

> **カラムおよびログ表示設定の適用範囲**
>
> - **カラム設定**と**ログ表示詳細設定**は、**ログTail**、**ログ検索**、**ログトレンド**で使用できます。
> - 同一プロジェクト内の**ログTail**、**ログ検索**、**ログトレンド**は、**カラム設定**と**ログ表示詳細設定**を共有します。

ログファイルのダウンロード

検索したログの結果を CSV または TXT ファイル形式でダウンロードできます。

1. 画面中央右側の **ダウンロード** ボタンをクリックします。
2. **CSVダウンロード** または **Logダウンロード** をクリックして、希望の形式のファイルをダウンロードします。

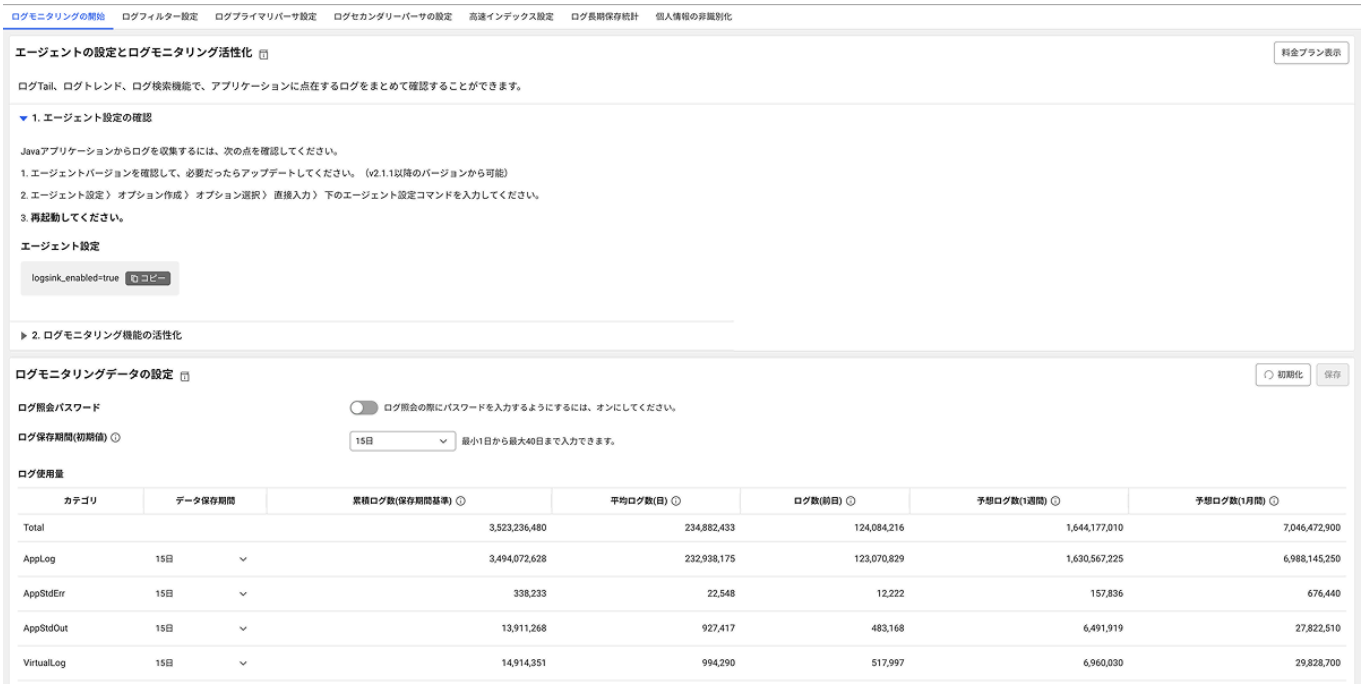
ログの設定

ホーム画面 > プロジェクト選択 > **ログ** > **ログ設定**

ログ設定 の上部タブからログモニタリングを設定できます。 エージェント設定の確認、ログモニタリングの有効化設定、ログデータの保持期間および照会パスワードの設定、ログパーサーの登録、クイックインデックスの設定、個人情報のマスキング管理などを行うことができます。

> **権限**
>
> - **ログモニタリング機能の活性化** を使用するには、**プロジェクト修正** 権限が必要です。
> - **ログエディター** 権限があれば、**ログモニタリング機能の活性化** のほか、**ログ設定** メニューを編集できます。

ログモニタリングを開始する



1. **ログ** > **ログ設定** の **ログモニタリングの開始** タブをクリックしてください。
2. **ガイド表示**アイコンと **料金プラン表示**ボタンを選択すると、案内画面に移動します。

エージェント設定とログモニタリングの有効化

1 領域でエージェント設定を確認し、ログモニタリングの有効化または無効化を設定できます。

エージェントの設定確認

ログモニタリングを開始するために、エージェントのバージョンと設定情報を確認してください。 **1. エージェント設定の確認** の案内に従って進めてください。

- **アプリケーションモニタリング**

  アプリケーション別の適用方法については、Java、PHP、Node.js、Python、Go のドキュメントを参照してください。

- **サーバーモニタリング**

  サーバーモニタリングの適用方法については、Server ドキュメントを参照してください。 指示に従って whatap.confにログモニタリング対象ファイルの設定を追加します。

- **Kubernetesモニタリング**

  Kubernetesモニタリングの適用方法については、Kubernetes ドキュメントを参照してください。

ログモニタリングの有効化

ログモニタリングを有効化または無効化できます。 **2. ログモニタリング機能の活性化** でログモニタリングの有効／無効を設定してください。

- ログモニタリングの有効化：トグルボタンをオンにするとログモニタリングが有効になり、有効化した日から15日間は無料でお試しいただけます。
- ログモニタリングの無効化：トグルボタンをオフにするとログモニタリングが無効になり、以降ログは保存されません。

ログモニタリングのデータ設定

2領域で**ログ使用量**が確認できます。 また**ログ保存期間**と**ログ照会パスワード**設定を変更できます。

ログ参照パスワード

セキュリティを強化するため、**ログ照会パスワード**を設定してください。 ログ検索パスワードの指定はオプションです。 ログ検索パスワードを使用中の場合は、ログ画面に入るときは必ずパスワードを入力します。

> **パスワードを忘れた場合**
>
> ログ編集権限を持つユーザーは、**ログ設定** で新しいパスワードに変更できます。

ログ保存期間

共通に適用する基本(default) データ保持期間です。

- 最小1日から最大40日まで入力できます。 未指定の場合、既定値は1日です。 ログ保存期間の選択オプションの他に、ユーザーが希望する期間を入力できます。
- **ログ使用量**一覧で別に設定しない場合、このログデータの保存期間がデフォルトで適用されます。 **ログ使用量** リストでカテゴリごとのデータ保持期間を設定し、**初期化** ボタンをクリックすると、デフォルトのデータ保持期間に初期化されます。

ログ使用量

ログ使用量一覧でカテゴリ別のログデータの保存期間を指定できます。 ログ数は、当該期間中に蓄積されたログの件数を意味します。 例えば、**日間ログ数**は、1日にたまったログ件数、**予想ログ数**は、データ保持日に今日のログ数を掛けたログ件数を意味します。

ログデータの保存期間を次のように指定できます。 期間指定により、古いデータを削除してスペースを確保できます。

- **トライアルプロジェクト**

  データ保存期間として1日、2日、3日を選択できます。

- **有料プロジェクト**

データの保存期間として1日、2日、3日、4日、5日、6日、7日、10日、30日、40日が選択できます。

- **保存量基準課金**

  データの保存期間によってコストが異なります。

  例、1日平均200万ログ件数が蓄積され、データ保存期間を3日に指定した場合、平均600万ログ件数が収集サーバーに保持され、課金の対象となります。

ログフィルター設定

ログフィルター設定は、ユーザーが指定したフィルター条件に一致するログのみを保存する機能です。必要なログだけを保存することでコストを削減できます。

> **権限**
>
> ログ編集権限を持つユーザーのみ設定できます。

ログフィルター設定の追加

新しいフィルター設定を追加する方法です。ログフィルターを設定する際は、**フィルター名、ライブテール表示有無、カテゴリ、フィルター条件**が必須入力項目です。

保存しないログであっても、**ライブテール表示有無**トグルを有効にすると**ライブテール**で確認できます。

> カテゴリごとに設定できるフィルターは1つのみです。

1. **ログ > ログ設定 > ログフィルター設定** タブに移動します。
2. **+ 追加** ボタンをクリックして新しいフィルター設定を作成します。
3. **ログフィルター設定追加**で **フィルター名、ライブテール表示有無、カテゴリ、フィルター条件** を入力します。
   - **フィルター名**: フィルターを識別する名前
   - **ライブテール表示有無**: 保存の有無に関係なくライブテールに表示するかを選択
   - **カテゴリ**: フィルターを適用するログカテゴリ **(複数選択可能)**
   - **フィルター条件**: フィルターに保存するログの条件

   > フィルター条件の詳細については WhaTap log search query 構文 を参照してください。

- **例**
  - **含む:** `oid : 123`
    - oid が 123 のログだけを収集します。
  - **含まない:** `not oid : 123`
    - oid が 123 **ではない** ログだけを収集します。
  - **すべて含む**: フィルター条件を未入力
  - **すべて含まない:** `not content : *`

4. **追加** ボタンをクリックすると、**登録結果プレビュー**で各フィルター設定を確認・修正できます。
5. 確認が完了したら、**適用** ボタンをクリックしてログフィルター設定を保存します。

シミュレーション

入力したカテゴリとフィルター条件でログを事前に検索し、ログフィルターを適用する前に結果を確認できます。

シミュレーション ボタンをクリックすると、フィルターが適用された **ログ検索** 画面が表示されます。

ログフィルター設定の修正

登録済みのログフィルターは編集アイコンをクリックして修正できます。
ログフィルター設定修正 画面では既存の値が表示され、必要な項目を変更して適用できます。

> ログフィルターを修正する際、選択できるカテゴリは1つのみです。

ログフィルター設定の削除

登録したフィルターは削除アイコンをクリックして削除できます。削除後は必ず **保存** ボタンをクリックしてください。

変更を保存しないとフィルターは削除されません。

ログフィルター設定のJSONインポート/エクスポート

ログフィルター設定はJSON形式でインポート/エクスポートできます。
他のプロジェクトに同じログフィルターを適用したい場合に便利です。

JSONエクスポート

1. 画面右上の JSON をクリックします。
2. **JSONエクスポート** ウィンドウが開いたら、**エクスポート** ボタンをクリックします。
3. ログフィルター設定がJSONファイルとしてダウンロードされます。

JSONインポート

1. 画面右上の をクリックします。
2. インポートするJSONファイルを選択します。
3. **JSONインポート** ウィンドウで **リストに追加** または **上書き** を選択してログフィルター設定をインポートします。
4. **保存** ボタンをクリックして変更を反映します。

バリデーション条件

設定を保存する際のバリデーションは次の通りです。

- **type**: `GENERAL` のみサポート
- **category**: 必須、 `string` タイプのみサポート
- **condition**
  - **language**: `jsonDsl` , `lucene` のみサポート
  - **description**: 無効なフィルター条件は保存できません

- **showLiveTail**: `boolean` タイプのみサポート
- **name**: 必須、 `string` タイプのみサポート

ログパーシング

ログパーサを使用すると、不規則な形のログをクエリが可能な構造化された形に変更できます。 WhaTapログモニタリングは、次の2つのタイプのパーサを提供します。

- **GROKパーサ**：任意の形で収集されるログを正規表現とGROKを活用してパーシングします。
- **JSONパーサ**：JSON形式で収集されるログをパーシングします。

> ⓘ **共通の注意事項**
>
> - 同じカテゴリに複数のパーサが登録されている場合、最初にマッチングされるパーサのみが適用されます。
> - **WhaTapはWhaTapサービスの安定性に影響を与えるパーサを無効にする場合があります。。**

GROKパーサ

ログが不規則な形で収集される場合、GROKパーサを使用してログをパーシングできます。 GROKパターンは、named regular expressionsを提供し、正規表現をより簡単かつ便利に使用できます。

GROKを開始する

GROKは2種類の文法を提供します。

1. `%{SYNTAX:SEMANTIC}`：GROKライブラリーが提供するパターンです。 **named regular expressions**を活用してタグを抽出できます。 活用例は次のリンクを参照してください。
   - **SYNTAX**：GROKが提供するnamed regular expressionsを指定します。
   - **SEMANTIC**：マッチングされる値に付ける名前を指定します。

   > ⓘ **named regular expressions**
   >
   > GROKが提供するパターンです。 複雑な正規表現に名前を付けて使用できるよう、GROKが提供す

る機能です。

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}

WhaTapで提供するすべてのnamed regular expressionsを確認したい場合は、次のリンクを参考にしてください。

2. (?<SEMANTIC>REGX)：正規表現の**named capturing group**文法です。 正規表現を活用して、ユーザーの意図に合わせてタグを抽出できます。 活用例は次のリンクを参照してください。

- **SEMANTIC**：マッチングされる値に付ける名前を指定します。
- **REGX**：マッチングに使用する正規表現を入力します。

named capturing group

正規表現の文法です。

- capturing group: 複数のトークンを1つにまとめて1つのマッチング単位で使用する機能を意味します。
- named capturing group: capturing groupに名前を付けたものです。
- 文字列マッチングの例を見てみましょう。 dev@whatap.io
  - 例1：(\w+)@(\w+\.\w+)
  - 例2：Eメール全体のマッチング、およびusernameとdomainを追加でマッチングする場合 (?<username>\w+)@(?<domain>\w+\.\w+)

%{SYNTAX:SEMANTIC} 活用の例

次は %{SYNTAX:SEMANTIC} を活用する例です。

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
```

サンプルログを見て、各単語が意味する内容を推測できます。 各部分をsemanticな単語に置き換えると、次のように表現できます。

semantic replace

```
[date][logLevel][caller] projectCode logCategory dexBuildStartTime {area=areaEnum, city=cityEnum} dexBuildElapsed
```

semanticな単語すべて、正規表現に書き換えできます。 GROKパーサを使用すると、事前に定義されたnamed regular expressionsを活用できます。 ここで使われた TIMESTAMP_ISO8601 、 LOGLEVEL 、 DATA は、GROKが提供するnamed regular expressionsです。 これらの値は、それぞれ次の正規表現に置き換えられてマッチングされます。

- 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)?|[Ee]rr?(?:or)?|ERR?(?:OR)?|[Cc]rit?(?:ical)?|CRIT?(?:ICAL)?|[Ff]atal|FATAL|[Ss]evere|SEVERE|EMERG(?:ENCY)?|[Ee]merg(?:ency)?)
- name: DATA
  - regular expression: .*?

GROK parsing pattern

```
\[%{TIMESTAMP_ISO8601:date}\sGMT\]\[%{LOGLEVEL:level}\s\]\[%{DATA:caller}\]
```

上記の内容でパーシングすると、次のようにタグを抽出できます。 このようにGROKの %{SYNTAX:SEMANTIC} パター

ンは、複雑で長い正規表現を簡単に適用できる機能を提供します。

Tag extraction

```
- date : 2023-08-08 02:02:30,101
- caller : i.w.y.l.c.LogSinkDexScheduleThread.realProcess(159)
- level : LEVEL
```

(?<SEMANTIC>REGX) 活用の例

named regular expressionsでマッチングされない部分は (?<SEMANTIC>REGX) パターンを使ってパーシングできます。 上記のサンプルログから %{SYNTAX:SEMANTIC} パターンだけでパーシングされない領域は次の通りです。

Unparsed area

```
8 VirtualLog 20230808 02:01:00.000 {area=4, city=5} 56ms
```

該当ログの各部分をsemanticな単語に置き換えると、次のように表現できます。

semantic replace

```
projectCode logCategory dexBuildStartTime {area=areaEnum, city=cityEnum} dexBuildElapsed
```

このように不規則な形の文字列は、次のような (?<SEMANTIC>REGX) パターンを使ってパーシングできます。

サンプルログパーシングキーワード別にマッチングされる正規表現

パーシングキーワード	(?<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)

パーシングキーワード	(?<SEMANTIC>REGX)
56 ms	(?<dexBuildElapsed>\d{2})ms

正規表現の基本文法

文法	意味	別名
?	0 or 1	-
+	1 or more	-
*	0 or more	-
a{5}	exactly 5	-
\w	word character	[a-zA-Z_0-9]
\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 (共通集合)	-
[0-9&&[^2468]]	subtraction (差集合)	-
a{2,}	2 or more	-
a{1,3}	between 1 and 3	-

文法	意味	別名
a+?	match as few as possible	-
{2,3}?	match as few as possible	-
(abc)	capturing group (複数の文字列をsingle unitで処理する)	-
\d	digit	[0-9]
\D	non-digit	[^0-9]
\W	non-word character	-
\S	non-white space	-

このようにパーシングされたキーワードをスペース(\s)と、特殊文字escape(\{ 、 \, 、 \})で接続すると、次のようにパターンを適用できます。

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
```

上記の内容でパーシングすると、次のようにタグを抽出できます。

Tag extraction

```
- projectCode : 8
- logCategory : VirtualLog
- dexBuildStartTime : 20230808 02:01:00.000
- areaEnum : 4
- cityEnum : 5
- dexBuildElapsed : 56
```

GROKを適用する

ログ設定 > ログプライマリパーサ設定

1. GROKパターンパーサを適用するには、**ログ設定**メニューの**ログプライマリパーサ設定**タブに移動します。
2. **+ 追加する**を選択した後**パーサ**入力欄で**GROK**パーサを選択します。
3. **パターン登録**ボタンを選択すると、右側にパターン登録とシミュレーション画面が表示されます。
4. **パターン**と**ログ**入力後、**シミュレーション**ボタンをクリックして、適用するパターンでパーシングに成功することを確認します。

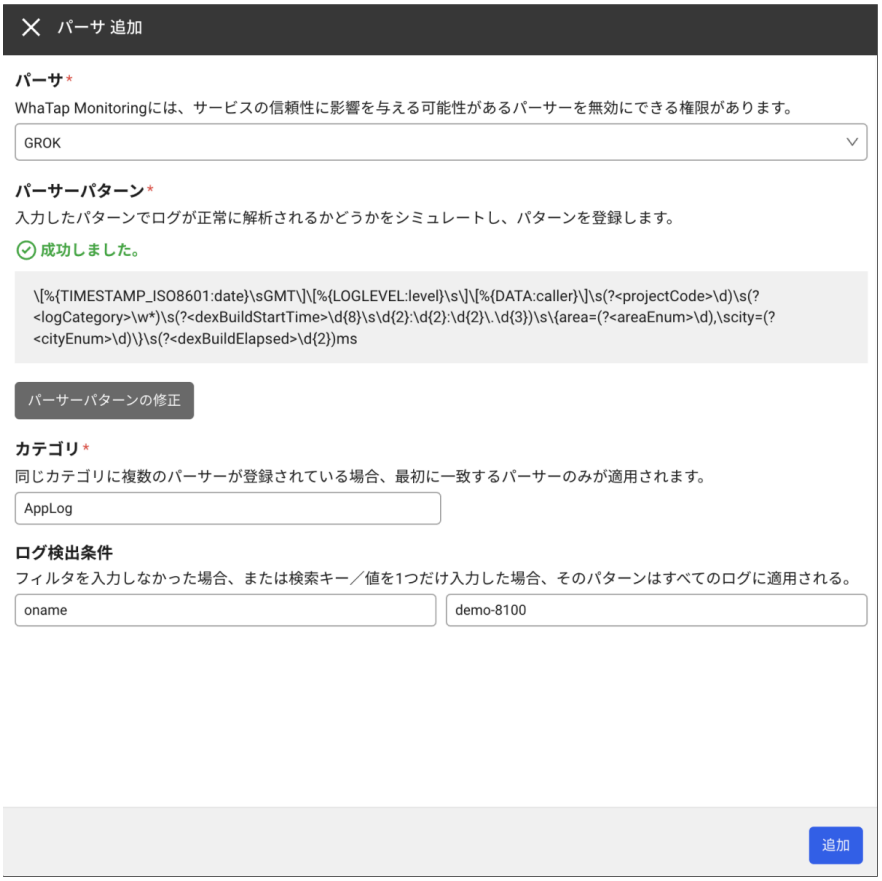
   パターン 例:

   ```
   [%{TIMESTAMP_ISO8601:date}\sGMT][%{LOGLEVEL:level}\s][%{DATA:caller}]\s(?<projectCode>\d)\s(?<logCatego
   d{2}:\d{2}:\d{2}\.\d{3})\s\{area=(?<areaEnum>\d),\scity=(?<cityEnum>\d)\}\s(?<dexBuildElapsed>\d{2})ms
   ```

   ログ例:

   ```
   [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. シミュレーションの成功時**シミュレーション結果**と**性能測定の結果**を照会できます。
6. シミュレーション後、**パターン適用**ボタンをクリックすると、選択したパーサーに入力したパターンが適用されます。
7. パターン適用後、**カテゴリ**と**ログ検出条件**、**パターン**を入力します。



- **カテゴリ**

  ログカテゴリを選択します。 **カテゴリ**は必須入力項目です。

- **ログ検出条件**

  - 条件を満たすログのみパーサが適用されます。
  - **検索キー**と**検索値**を選択するか、直接入力します。
  - **ログ検出条件**は、すべてのパーサが実行される前に適用されます。 つまり、パーサの結果として追加される**タグ**を使用できません。

8. **追加**ボタンを選択してパーサを登録します。

- ログパーサー一覧から該当パーサの**優先順位**を変更するか、**活性化**および**修正**、**削除**できます。
- **パーサーシミュレーション**後、パターンを登録できます。

GROKパーサの注意事項

- GROKパーサは `%{SYNTAX:SEMANTIC}` 、 `%{SYNTAX:SEMANTIC}` 二つのパターンをサポートします。
- `%{SYNTAX:SEMANTIC}` パターンを使用する時、 `SEMANTIC` を必ず入力します。
- `%{SYNTAX:SEMANTIC}` パターンを使用する時、 `SEMANTIC` は一つのパーサの中でユニークである必要があります。
- `(?<SEMANTIC>REGX)` パターンを使用する時、 `SEMANTIC` は文字(a-z、A-Z)、数字(0-9)、および指定された特殊文字(`.` 、 `_` 、 `-`)のみが使えます。
- `SEMANTIC` は文字(a-z、A-Z)で始めなければなりません。
- `SEMANTIC` は文字(a-z、A-Z)または数字(0-9)で終わる必要があります。

GROK追加パターンの選択

GROKパーサのシミュレーション時に、基本パターン(`WORD` 、 `NUMBER` 、 `IP` など)以外に特定のログ形式に合った**追加パターン**を選択できます。 Java、Nginx、MySQLなど32個のパターンタイプが11個のカテゴリにグループ化されて提供されます。

追加パターンカテゴリのドロップダウン

1. パーサのシミュレーション画面で**追加パターン (選択)**領域のドロップダウンをクリックしてください。
2. カテゴリ別にグループ化されたパターンタイプ一覧から、必要なパターンタイプを選択してください。 検索入力欄にパターンタイプ名または説明を入力して検索することもできます。
3. 選択したパターンタイプは、シミュレーション時に基本パターンとともにロードされます。

基本パターン(Core)は常に自動的に含まれます。 追加パターンの選択は任意です。

カテゴリ別パターンタイプ一覧

カテゴリ	パターンタイプ
Web Servers	haproxy, httpd, iis, nginx, squid

カテゴリ	パターンタイプ
Application Servers & Frameworks	java, log4j, rails, ruby, springboot
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

活用例

Tomcat Catalina 8のログをパーシングするには、**java**パターンタイプを追加で選択した後、`%{CATALINA8_LOG}` パターンを使用してください。

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}
```

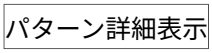
Tag extraction

```
- 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
```

パターン詳細表示

選択した追加パターンタイプにどのパターンが含まれているかを詳細画面で確認できます。 パターン定義(正規表現)とサンプルログを照会し、シミュレーションにすぐに適用できます。



追加パターンタイプを選択した後、選択されたパターンタイプのタグをクリックすると、右側に詳細画面が開きます。詳細画面はシミュレーションフォームを隠さずに並んで表示されるため、パターン定義を確認しながらシミュレーションを進めることができます。

各パターン項目で次の機能を使用できます。

ボタン	機能	説明
サンプル	サンプルログ入力	該当パターンにマッチングする例ログを**シミュレーションパーシングログ**入力欄に自動的に入力します。
選択	パターン挿入	`%{パターン名}`形式で**パターン**入力欄に挿入します。 該当パターンタイプも自動的に選択されます。
コピー	クリップボードコピー	`%{パターン名}`形式をクリップボードにコピーします。

- パターン項目をクリックすると、パターン定義(正規表現)が展開されます。 複数のパターンを同時に展開して定義を比較できます。
- パターン定義に含まれる `%{PATTERN:field}` 参照は、パターン名、フィールド名、リテラル文字がそれぞれ異なる色で区別されて表示されます。
- **サンプル**ボタンで入力されたログでそのまま**シミュレーション**を実行してパーシング結果を確認できます。

シミュレーションサンプルログの保存

シミュレーションに使用したログは、パーサ設定とともに自動的に保存されます。 パーサを修正する際に、以前に使用したサンプルログがシミュレーション画面に自動的に復元され、毎回ログを再入力する必要がありません。

JSONパーサ

ログがJSONフォーマットで収集される場合、JSONパーサを使用して簡単かつ便利にパーシングできます。

JSONを適用する

ログ設定 > ログプライマリパーサ設定

1. JSONパターンパーサを適用するには**ログ設定**メニューの**ログプライマリパーサ設定**タブに移動します。
2. + **追加する**を選択した後**パーサ**入力欄でJSONパーサを選択します。
3. **パターン登録**ボタンを選択すると、右側にパターン登録とシミュレーション画面が表示されます。
4. **パターン**と**ログ**入力後、**シミュレーション**ボタンをクリックして、適用するパターンでパーシングに成功することを確認します。

   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"} --
   ```

   例のログで**Prefix**と**Postfix**を `--` に指定し、**Ignore**に `url` を指定時 `host` 、 `method` 、 `status` 3つ**タグ**のみ生成します。

   - **Prefix** ログでJSONフォーマットが開始する場所を指定します。 ログ全体がJSONフォーマットの場合は、空の値に設定します。
   - **Postfix** ログでJSONフォーマットが終わる位置を指定します。 ログ全体がJSONフォーマットの場合は、空の値に設定します。
   - **Ignore** JSONフォーマットの中でタグを生成しないキーを指定します。
5. シミュレーションの成功時**シミュレーション結果**と**性能測定の結果**を照会できます。



6. シミュレーション後、**パターン適用**ボタンをクリックすると、選択したパーサーに入力したパターンが適用されます。
7. パターン適用後、**カテゴリ**と**ログ検出条件**、**パターン**を入力します。



- **カテゴリ**

  ログカテゴリを選択します。 **カテゴリ**は必須入力項目です。

- **ログ検出条件**

  - 条件を満たすログのみパーサが適用されます。
  - **検索キー**と**検索値**を選択するか、直接入力します。
  - **ログ検出条件**は、すべてのパーサが実行される前に適用されます。 つまり、パーサの結果として追加される**タグ**を使用できません。

8. **追加**ボタンを選択してパーサを登録します。

- ログパーサー一覧から該当パーサの**優先順位**を変更するか、**活性化**および**修正**、**削除**できます。
- パーサを登録する前に**シミュレーション**を通じて登録するパターンが正常であるか確認できます。 GROK

パーサ登録シミュレーションプロセスと同じです。 次の文書を参考にしてください。

JSONフォーマット活用例

Sample log

```
{"host":"10.21.3.24","method":"POST","status":"200","url":"http://dev.whatap.io/yard/api/flush"}
```

上記のようなサンプルログが収集された場合**パーサ 追加**画面で**JSON**パーサを選択します。 複雑なパーシングロジックを作成する必要なく、ログ分析に必要な**タグ**を次のように抽出できます。

Tag extraction

```
- host : 10.21.3.24
- method : POST
- status : 200
- url : http://dev.whatap.io/yard/api/flush
```

JSONフォーマットを一部構成する時の活用例

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"}
--
```

もし、例のようにログの一部のみがJSONフォーマットで構成されている場合は**Prefix**と**Postfix**を指定してください。 WhaTapログモニタリングは**Prefix**と**Postfix**間の領域をJSONフォーマットで認識してからパーシングします。

Tag extraction

```
- host : 10.21.3.24
- method : POST
- status : 200
- url : http://dev.whatap.io/yard/api/flush
```

ログ高速インデックス設定

高速インデックス設定

ログ設定メニューの上段から**高速インデックス設定**タブを選択します。 大量のログを収集すると、ログ検索性能が低下する場合があります。 よく使う検索条件は、**インデックス**(index)で生成すると、ログ検索がより速くなります。 設定項目は次のとおりです。

設定値	必須の有無	説明
カテゴリ	必須	高速インデックスを設定するカテゴリ
検索キー	必須	高速インデックスを設定する検索キー
大文字と小文字を区別しない	オプション	大文字と小文字の区別するかどうか
ルール	必須	`*` を少なくとも１つ以上を含める必要あり
活性化有無	必須	活性化または非活性化の設定(既定値`true`)

ログ長期保管統計設定

ログ長期保存統計の設定

ログ設定メニューの上段から**ログ長期保存統計**タブを選択します。 ログデータは、非常に大容量のため、長期間保存することは難しいです。 したがって、ログ統計データの設定機能を使用して**特定の条件を満たすログデータが5分ごとに何件収集されたかに関する情報を保存**できます。 長期間にあたって、実際のログデータが削除されても、条件を満たすログがどれだけ収集されているかの推移は確認できます。

ログ長期保存統計の追加



ログ長期保存統計タブで **+ 追加する**ボタンを選択すると**ログ長期保存統計 追加**画面が表示されます。 **+ 追加する**ボタンでルールを追加したり、作成したルールを**-**アイコンで削除できます。

設定項目

フィールド	説明
カテゴリ	ルールが適用されるカテゴリです。
統計キー(key)	ルールを満たすログが発生した場合、保存するキーとして同じキーを重複で設定できません。
ログ検出条件	ログ統計データを生成する条件です。 この条件を満たすログがどのくらい収集されたかで統計データを生成します。
除外	除外をチェックすると、入力した条件を満たさない値で統計データを作成します。
大小文字を区別する	入力したログ検出条件の値に対して大文字と小文字の区別有無を指定します。
活性化有無	活性化または非活性化の設定(既定値`true`)

例

次のように設定を追加した場合、収集されたログのうち、statusが`200`, `300`のログを対象に**TotalCount**のキーで統計データを作成します。

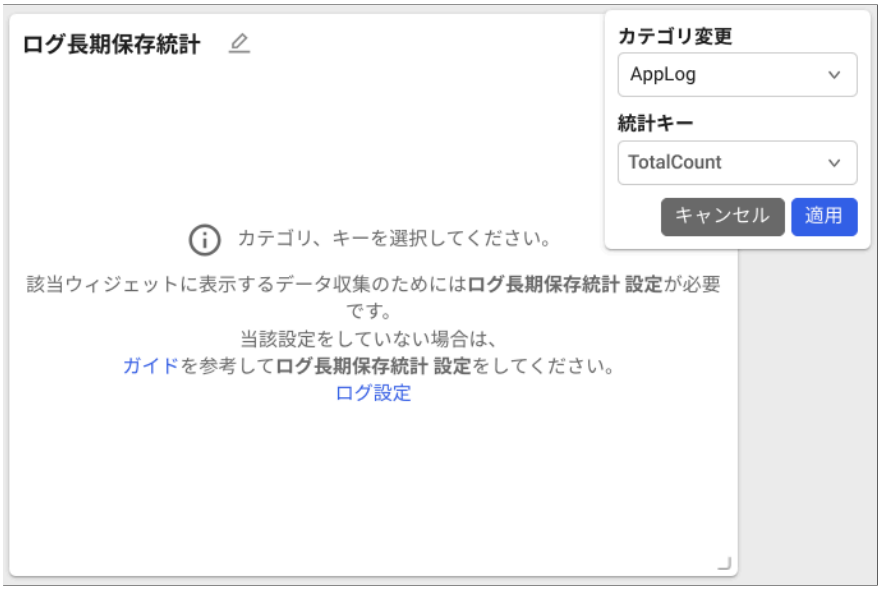


データ参照

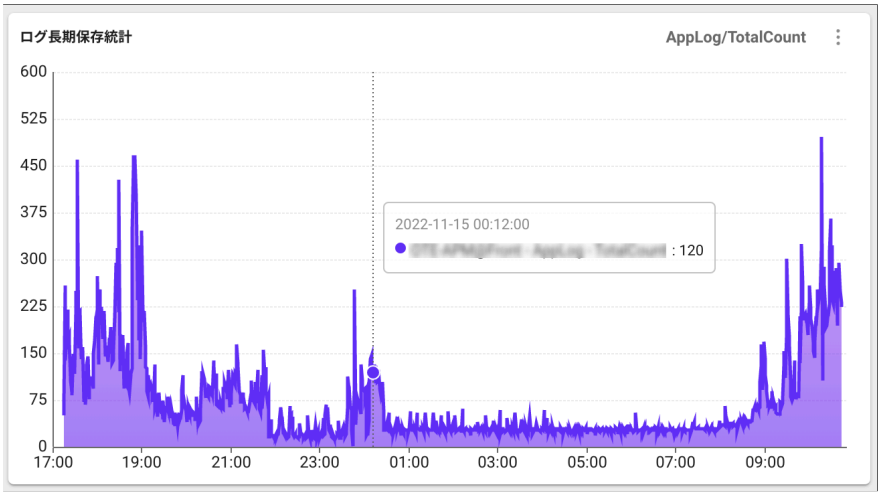
1. **統合Flexボード**の**ウィジェットテンプレート**で**ログ長期保存統計**を検索し、ウィジェットを作成します。



2. データを照会する**カテゴリ**と**統計キー**を指定して**適用**ボタンを選択します。



3. 追加した設定で**ログ長期保存統計**データを次のように確認できます。


ログ長期保存統計
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

ログ個人情報非識別化設定

個人情報の非識別化

個人情報の非識別化ではログデータ内の個人情報をマスキング処理したり、安全な値に置換できます。

- 暗号化の保存：指定された機密な情報は暗号化され、データベースに保存されます。
- マスキング処理：ログ画面に表示される時、機密な情報は基本的にマスキング(`***`)処理されて表示されます。
- 置換処理：機密な情報をユーザーが指定した安全な値に変更します。

個人情報の分類

一般的に個人情報として分類する項目は次のとおりです。 一般の個人情報以外にも機密な情報として処理すべき項目があれば**個人情報の非識別化**機能を通じて、ログ内で必要な項目をマスキング処理または置換処理してください。

- 個人識別番号
- パスポート番号
- 運転免許証番号
- 外国人登録番号
- クレジットカード番号
- 銀行口座番号
- 生体情報

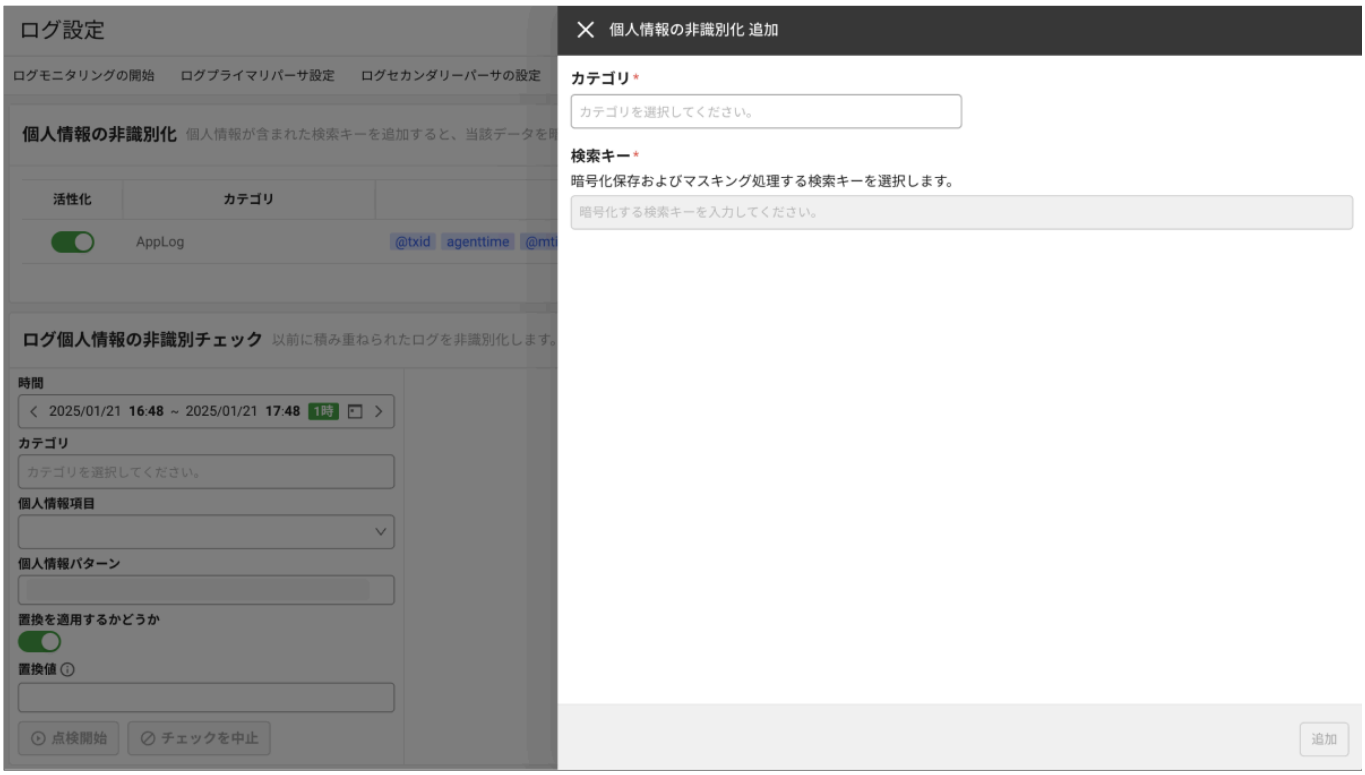
要求権限

非識別化された個人情報はマスキング処理されて表示されます。 **ログの個人情報照会**権限が付与されたユーザーのみ**個人情報の表示**トグルを通じて元のデータを確認できます。 非識別化の対象検索キーの追加及び修正は、**ログ編集**権限が必要です。

> ⓘ **ログの個人情報照会**権限及び**ログ編集**権限の詳細は次の文書を確認してください。

個人情報の非識別化設定

新規のログデータにマスキング処理を適用して機密な情報を保護できます。



1. **個人情報の非識別化**タブで**追加する**ボタンを選択します。
2. 右側の**個人情報の非識別化 追加**画面で**カテゴリ**を選択し、マスキング処理する**検索キー**を入力します。
3. **追加**ボタンをクリックして設定を追加します。
4. **個人情報の非識別化**一覧で非識別化項目を確認した後、右上の**保存**ボタンを押して現在の設定を保存してください。



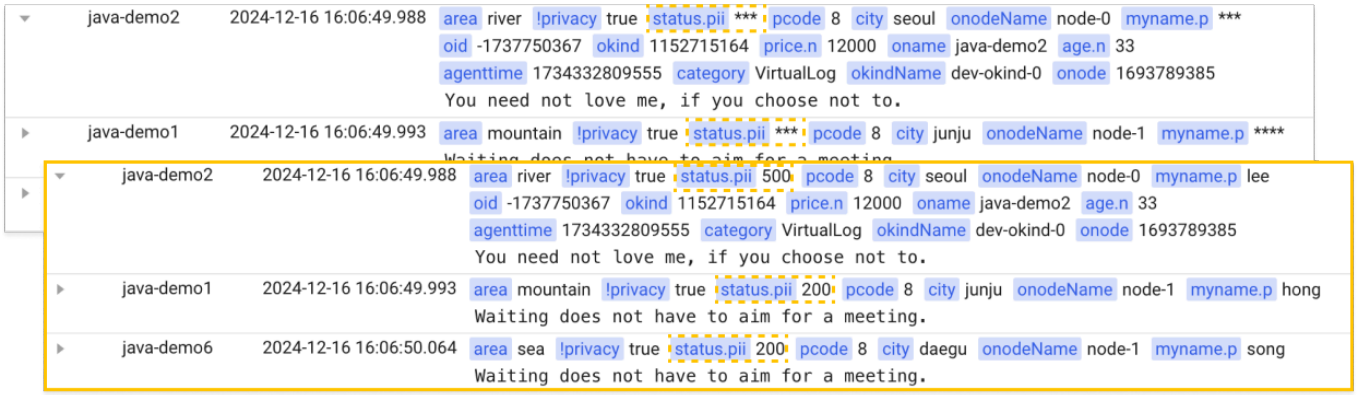
- 非識別化処理された検索キーの場合、ログコンテンツ内の `検索キー.pii` 形式で出力されます。

- **個人情報の非識別化**一覧内の**活性化**カラムで非識別化項目を**活性化**にできます。
- **個人情報の非識別化**一覧内の**編集**カラムで 修正及び 除去できます。

> **ログプライマリパーサ設定**タブで検索キー名を 検索キー.p に指定すると、非識別化対象として自動登録されます。自動登録された項目は**個人情報の非識別化**タブで**カテゴリ**選択後、**指定された検索キー**で確認または解除できます。
>
> パーサーで .p 接尾辞を指定する方式よりも、**個人情報の非識別化**タブで直接設定することを推奨します。

個人情報の表示

非識別化されたデータは**ログTail**、**ログトレンド**、**ログ検索**メニューでマスキング状態で表示されます。 **ログの個人情報照会**権限のあるユーザーはログ一覧内の**個人情報の表示**ボタンをクリックすると元のデータを確認できます。



> **個人情報の非識別化**タブで指定した検索キーは、ログメニュー内のログコンテンツから 検索キー.pii 形式で表示されます。
>
> 例: status.pii 例 ***

ログの個人情報の非識別化点検

> **ログ個人情報の非識別チェック**機能を通じて既存のデータを置き換える場合、そのデータは元に戻すことが

できません。

ログ個人情報の非識別チェックは既存のログデータ内の個人情報を識別し、ユーザーが指定した**置換値**に置き換えます。

対象キー指定による置換

対象キーを指定すると、該当キーに保存された値のみを置換し、残りのログ内容はそのまま維持します。対象キーはログの `field` 値と `tag` 値の両方に適用されます。対象キーを指定しない場合、従来と同様にログコンテンツ全体から個人情報を識別して置換します。

置換は保存されたログデータにのみ適用されます。検索インデックスには、置換後も元の値が残っている場合があります。

対象キーは、ログ個人情報の非識別化点検フォームの**対象キー**入力欄で指定し、画面では `field` と `tag` を区別せず、1つの対象キーとして入力します。



1. 開始時間と終了時間を設定してください。
2. **カテゴリ**を設定します。
3. **個人情報項目**を例の中から選択または**直接入力**を選択します。

> ⓘ **直接入力**選択時に個人情報を識別するための正規表現式パターンを直接入力します。

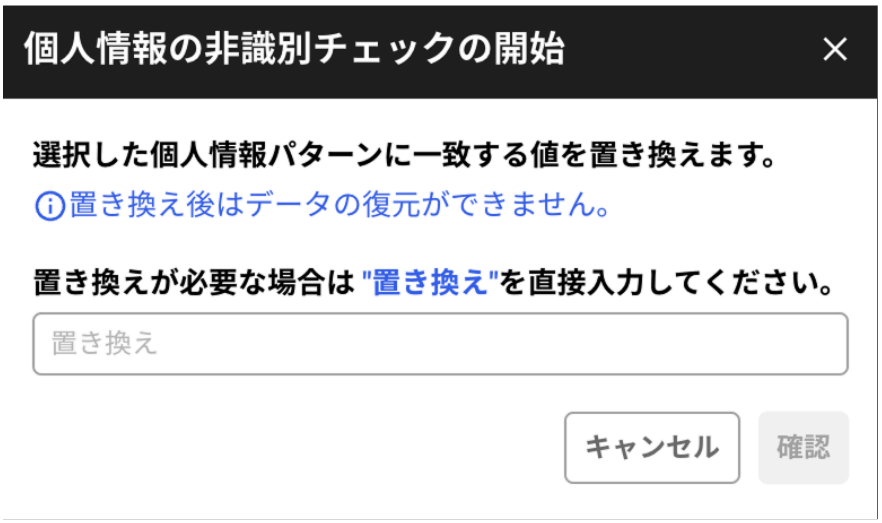
4. **置換の適用有無**トグルで個人情報が識別された場合、該当情報の置換有無を選択します。
5. 既存のログで正規表現式パターンに識別された文字列を変更する**置換値**を入力します。

> ⓘ
> - **置換値**は元の値より長くしてはなりません。
> - **置換の適用有無**トグルの活性化時に、必ず**置換値**を入力する必要があります。

6. 条件を入力した後、**チェック開始**ボタンを選択します。

> ⓘ 点検中の場合はいつでも**チェック中止**ボタンで点検過程を中止できます。

7. 置き換えを適用する時**個人情報の非識別チェックの開始**確認画面で**置き換え**入力後**確認**ボタンをクリックします。



8. 点検作業の完了後、例の画像のように右側から**チェック作業の結果**を確認できます。
   - **カテゴリ**：点検されたカテゴリ情報を表示します。
   - **検索開始時刻**及び**検索終了時刻**：点検が実行された時間範囲を表示します。
   - **個人情報項目**：点検された個人情報の項目を表示します。

- **個人情報パターン**：使用された正規表現式のパターンを表示します。
- **置換値**：適用された置換値を表示します。
- **パターンのマッチング結果**: 識別された個人情報の数を表示します。 **結果の確認**クリックするとサンプルログを確認できます。
- **チェック作業の結果**：個人情報が識別された結果を表示します。

  `SUCCESS` (点検成功)、 `FAILED` (点検失敗)、 `STOP` (点検中止)、 `TIMED_OUT` (時間超過による点検終了)

チェック履歴の確認

ログ個人情報の非識別化点検を行った後、過去に行われた点検結果を照会するには、上段右側で**チェック履歴の確認**ボタンをクリックします。 置き換え履歴、点検時点、適用された個人情報項目などを追跡できます。 また**チェック履歴の確認**画面の上段右側の CSVボタンをクリックすると_CSV_ファイル形式でそのリスト情報をダウンロードできます。



個人情報のパターン

既存のログデータから個人情報を識別する際に使用される正規表現式のパターンの例は以下の通りです。

個人情報のパターン	個人情報の例	正規表現式パターン	置換される値
個人識別番号	123456-1234567	(?<!w)d{2}(0[1-9]|1[0-2])([0-2][0-9]|3[0-1])-[1-4]d{6}(?!w)	`******_*******`
パスポート番号	M1234567	([MSRODTC][0-9]{7}\b)	`********`
運転免許証番号	01-23-456789-00	(?<!w)d{2}-d{2}-d{6}-d{2}(?!w)	`**_**_******_**`
外国人登録	987654-1234567	(?<!w)d{6}-?[5-8]d{6}(?!w)	`******_*******`

個人情報のパターン	個人情報の例	正規表現式パターン	置換される値
番号			
クレジットカード番号	1234-5678-9012-3456	(?<!w)d{4}-d{4}-d{4}-d{4}(?!w)	`****-****-****-****`
携帯電話番号	010-1234-5678	(?<!w)(+82-?1[016789]-?d{3,4}-?d{4}|01[016789]-?d{3,4}-?d{4})(?!w)	`***-****-****`
メールアドレス	`example@domain.com`	([a-zA-Z0-9][a-zA-Z0-9._%+-]*@[a-zA-Z0-9.-]+.[a-zA-Z]{2,})	`***@***.***`

個人情報のパターン	個人情報の例	正規表現式パターン	置換される値
銀行口座番号	123456-01-654321	(?<!w)d{6}-d{2}-d{6}(?!w)	`******_**_******`
事業者登録番号	111-11-01111	(?<!w)d{3}-d{2}-d{5}(?!w)	`***_**_*****`
法人登録番号	111111-1111111	(?<!w)d{6}-?d{7}(?!w)	`******_*******`

ログ設定のインポート/エクスポート

ログ設定のインポート・エクスポート

パーサ設定、高速インデックス設定、フィルター設定の内容をJSONファイル形式で保存し、他のプロジェクトからJSONファイルをインポートして適用できます。 プロジェクトごとに何度も繰り返し設定を作成する手間を減らします。

1. 1つのプロジェクトにパーサ設定、高速インデックス設定、フィルター設定を追加します。
2. 各設定タブで画面右上に**JSON**ボタンを選択します。
3. **JSONエクスポート**画面が表示されると、画面右上にある**抽出**ボタンを選択します。
4. JSON設定ファイルがユーザーPCに保存されます。
5. 他のプロジェクトに移動してから、**ログ** > **ログ設定**メニューに移動してください。
6. まず、JSON設定ファイルをエクスポートした設定タブを選択し、ボタンを選択します。
7. ファイル選択画面が表示されたら、ユーザーのPCに保存したJSON設定ファイルを選択します。
8. **JSONインポート**画面が表示されたら、設定内容を確認し、**リストに追加する**または **上書き**ボタンを選択します。
9. 画面右上の**保存**ボタンを選択します。

> JSON設定ファイルをインポートして、**保存**ボタンを選択しないと、インポートされた設定は保存されません。

警告アラート

監視対象のシステムまたはアプリケーションで問題が発生した際に、あらかじめ設定した条件に基づいてユーザーへ通知を送信する機能です。通知は、メール、SMS、メッセンジャー、アプリのプッシュ通知など、さまざまなチャネルを通じて配信されます。

たとえば、CPU 使用率、メモリ使用量、ディスク I/O、ネットワークトラフィック、トランザクション数、エラー発生、レスポンスタイムの遅延などの主要なパフォーマンス指標にアラート基準を設定しておくと、その基準を超過した場合や異常な兆候が検出された際に通知が送信されます。

WhaTap の AI ベースのアラート機能を使用すると、複雑な条件を手動で設定する必要はありません。AI がアプリケーションの正常な動作パターンを学習し、異常パターンを自動的に検知して通知します。また、タグカウントアラート機能を活用することで、特定のタグを基準に集計された値に条件を設定でき、複雑な状況でも高精度な検知が可能です。

主な機能

- **イベント設定**: 閾値ベースのアラート条件および通知チャネルの設定
- **通知設定**: ユーザーごとの通知チャネルおよび時間帯設定
- **イベント履歴**: アラート発生履歴の確認

権限

イベントを追加または設定するには、**アラート設定権限** が必要です。

- **閲覧**: メトリクスイベント設定を確認するには、**閲覧・分析権限** が必要です。
- **追加／編集／削除**: 設定を変更するには、**アラート設定権限** が必要です。
- **通知受信**: 通知を受信するには、**通知受信権限** が必要です。

> **参考資料**
>
> 詳細については、メンバー権限 を参照してください。

イベント設定

ホーム画面 > プロジェクト選択 > **アラート通知** > **イベント設定**

イベント設定 では、監視対象のシステムやアプリケーションで発生する可能性のある異常兆候を基準に、イベントを設定・管理できます。設定した条件が満たされると、メール、SMS、メッセンジャー、アプリのプッシュ通知など、希望するチャネルで通知を受信できます。監視の目的に応じて、イベントを柔軟に構成できます。

メトリクス

メトリクスイベントは、リアルタイムデータを基にシステムおよびアプリケーションのパフォーマンス指標を監視できます。

リアルタイムログ

リアルタイムで収集されたログから設定したイベント条件が検出された場合、警告アラートを送信します。

複合ログ

ログの内容を基準に警告通知を発生させる複合ログイベントを設定します。

ホーム画面 > プロジェクト選択 > **アラート通知** > **イベント設定**

イベント設定 では、監視対象のシステムやアプリケーションで発生する可能性のある異常兆候を基準に、イベントを設定・管理する機能を提供します。設定した条件を満たすと、メール、SMS、メッセンジャー、アプリプッシュなど、希望するチャネルで通知を受信できます。監視目的に応じてイベント構成を設計できます。

メトリクス

メトリクスイベントは、リアルタイムデータを基にシステムおよびアプリケーションのパフォーマンス指標を監視できます。

リアルタイムログ

リアルタイムで収集されたログから設定したイベント条件が検出された場合、警告アラートを送信します。

複合ログ

ログの内容を基準に警告通知を発生させる複合ログイベントを設定します。

参考

- イベント設定時は、不要なイベント発生を防ぐために適切なしきい値を設定してください。
- 通知設定では、問題解消時に通知される回復通知もあわせて検討すると、運用対応に役立ちます。

共通機能

すべてのイベント設定は **JSON または Excel 形式** でダウンロードできます。同じ種類のプロダクト設定を共有することで、繰り返し発生するイベント設定作業を削減できます。

- **JSON 一括編集**: すべてのイベント条件を JSON 形式で一括編集できます。
- **JSON 一括ダウンロード**: 現在登録されているイベント条件を JSON ファイルとして保存できます。
- **Excel 一括ダウンロード**: イベント一覧を Excel ファイルとしてダウンロードし、外部レポートやバックアップに活用できます。

JSON 一括編集

1. 画面上部の [JSON 一括編集] ボタンをクリックします。
2. **JSON 一括ダウンロード** 機能でダウンロードした JSON ファイルを読み込むか、直接編集します。
3. [上書き] ボタンをクリックして保存します。
4. 確認メッセージが表示されたら、[上書き] ボタンをクリックします。

上書き後は、以前のイベント設定に戻すことはできません。

- `productType` または `platform` プロパティの値を任意に変更すると、他のプロジェクトでイベントが正常に動作しなくなる可能性があります。
- JSON 編集画面には検証機能が組み込まれています。形式や値が正しくない場合、編集画面下部にエラーメッセージが表示され、[上書き] ボタンは無効になります。



JSON データ構造

```
{
 "productType": "{productType}",
 "platform": "{platform}",
 "events": {
  "HITMAP": {
   ...
  },
  "LOG_REALTIME": [
   {...}
  ],
```



```
  "METRICS": [
   {...}
  ],
  "COMPOSITE_LOG": [
   {...}
  ],
  "BASIC": [
   {...}
  ],
  "COMPOSITE_METRICS": [
   {...}
  ],
  "ANOMALY": [
   {...}
  ]
 }
}
```

JSON フィールド	説明
productType	プロダクトタイプ（例: APM、Infra など）
platform	プラットフォーム情報（例: java、node など）
events	イベントデータ
ㄴ HITMAP	ヒットマップパターン
ㄴ LOG_REALTIME	リアルタイムログイベント
ㄴ METRICS	メトリクス
ㄴ COMPOSITE_LOG	複合ログイベント
ㄴ BASIC	基本イベント
ㄴ COMPOSITE_METRICS	複合メトリクス

JSON フィールド	説明
ㄴ ANOMALY	異常値

JSON 一括ダウンロード

画面上部の [JSON 一括ダウンロード] ボタンをクリックします。

- JSON ファイル名は integrated-event-`YYYYMMDD`.json 形式です。
- JSON データの構造や `events` 属性配下のサブ属性は、プロダクト（ `platform` ）またはプロダクトタイプ（ `productType` ）によって異なる場合があります。

JSON データ内の `productType` または `platform` プロパティの値を任意に変更すると、他のプロジェクトでイベントが正常に動作しなくなる可能性があります。

Excel 一括ダウンロード

画面上部の [Excel 一括ダウンロード] ボタンをクリックします。

メトリクス

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **メトリクス** タブ

プロジェクトに設定されているすべてのメトリクスベースのイベント一覧を確認できます。設定済みのメトリクスイベントを確認、修正、削除したり、新しいイベントを追加したりできます。

基本オプション

- **イベントレベル**: イベントの深刻度を示し、正常（Info）、警告（Warning）、危険（Critical）の3段階に分類されます。
- **エージェント重複アラート抑止**: 同じ名前のエージェントが同時に2つ以上動作している場合、選択した時間（30秒、1分、3分、5分、10分）の間はイベントが発生しません。
- **検索**: イベント名を基準に検索します。
- **JSON ダウンロード／アップロード**: JSON エディタでイベントルールを直接編集するか、ファイルとしてダウンロードして編集後に再アップロードできます。
- **+ イベント追加**: 新しいメトリクスイベントを追加します。
- **カラムサイズ調整**: 各カラムの境界をドラッグして幅を調整できます。

表 | メトリクスイベント一覧構成

項目	説明
No.	イベントの番号
	イベントの有効化状態 – 有効: 条件を満たすとイベントが発生 – 無効: イベントは動作せず、条件も評価されません
	イベント設定の編集および削除
イベント名	ユーザーが指定したイベント名 – 最大255文字

項目	説明
ルール	イベント発生条件
対象	イベントが適用される監視対象
発生回数	イベント発生の基準となる期間と回数 – 選択した時間内に設定回数分条件を満たすとイベントが発生
一時停止	アラート発生後、同一イベントの発生を一定時間停止 – 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 – 回復通知が有効な場合、RECOVERED（正常）通知後、選択した時間は同一イベントが発生しても通知されません
回復通知	Critical および Warning レベルのイベントが解消された際に、RECOVERED（正常）通知を受信するかどうか – 有効: 進行中から正常状態に遷移すると RECOVERED 通知を送信し、イベント履歴で進行中／解消状態を区別して確認可能 – 無効: しきい値を超えるたびに単発イベントが発生
イベント受信	該当イベント通知を受信するユーザーまたはグループ（受信タグ）

イベント設定

メトリクスイベントは、監視対象で発生する事象を条件に基づいて定義し、効率的に通知を受信できるよう設定するための機能です。

イベント追加

メトリクスイベントを追加するには、**イベント設定** > **メトリクス** タブに移動し、右上の [+ イベント追加] ボタンをクリックします。

メトリクスイベント追加手順
Step 1. テンプレート選択: イベントの基本構成を選択します。
Step 2. イベント条件定義: どの条件でイベントを発生させるかを設定します。
Step 3. イベント対象選択: イベントが適用される監視対象を選択します。
Step 4. 基本情報および通知設定: イベント名、メッセージ、受信対象を設定します。

テンプレート選択

1. イベントの構成方法に応じて、画面左側の 3 つのテンプレートのいずれかを選択します。
   - **新規作成**: カテゴリやメトリクスを一から自由に構成できます。複雑な条件や詳細な監視ルールが必要な上級ユーザーに適しています。
   - **クイック設定**: あらかじめ定義されたカテゴリ、メトリクス、基本しきい値が提供され、しきい値のみを調整して素早く適用できるため、初心者に適しています。
   - **高度な設定**: あらかじめ定義されたカテゴリやメトリクスを基に、条件を追加したり自由に変更したりできます。

> ⓘ メトリクスイベントでは、さまざまな設定方法をサポートしています。利用する監視プラットフォームによって、提供されるテンプレートは異なる場合があります。

イベント条件定義

イベント条件定義 のステップでは、イベント発生の判断に使用するメトリクス設定、発生回数、一時停止、回復通知、イベント動作時間を設定します。

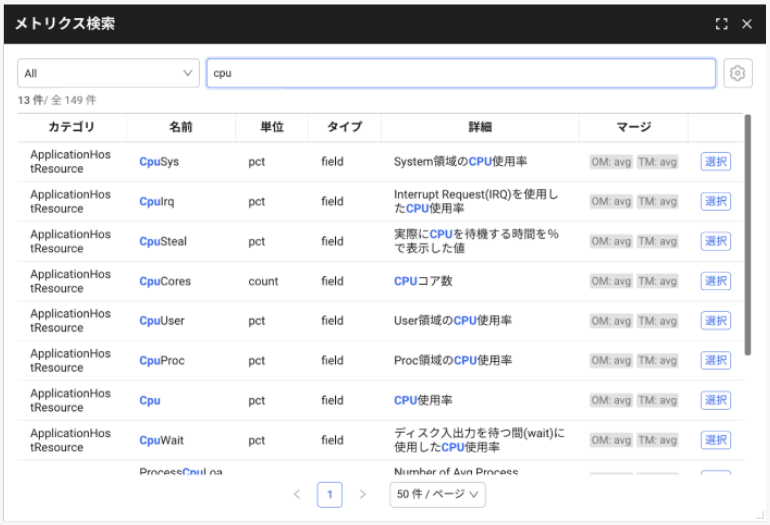
指標設定

1. 検索欄にカテゴリを入力するか、カテゴリー覧からカテゴリ（指標）を選択します。
   指標設定を進めるには、カテゴリの選択が必要です。

- カテゴリ一覧では、カテゴリ名、データ収集間隔、キーを確認できます。直近 3 時間にプロジェクトで収集されたメトリクスデータが表示されます。

2. 一覧にない指標を使用する場合は、**直接入力** を選択してカテゴリを手動で入力します。

> **カテゴリや指標が不明な場合は、メトリクス照会を活用してください。**
>
> 初めて指標を設定する場合、メトリクス照会を利用すると効率的に指標を見つけられます。
>
> - メトリクス照会で指標を検索すると、関連するカテゴリや詳細情報を確認できます。
> - 検索結果から必要なメトリクスの [選択] ボタンをクリックすると、選択した値がイベント設定画面に自動的に反映されます。



3. イベントレベル（Critical、Warning、Info）を選択します。1 つの指標に対して複数のレベルを設定できます。

> **イベントレベルの動作**
>
> 複数の条件を満たしても、最も優先度の高いレベル（Critical → Warning → Info）のみがイベントとして発生します。詳細は レベル動作ガイド を参照してください。
>
> - イベントレベルは、危険（Critical）、警告（Warning）、正常（Info）に分類されます。
> - 同一指標に対して段階的なしきい値でレベルを設定することを推奨します。

4. 選択したイベントレベルのフィールド、演算子、値を入力します。

   a. **+ 追加** をクリックすると条件を追加できます。

   - `&&`: **すべての条件を同時に満たした場合** にイベントが発生
   - `||`: **いずれか 1 つ以上の条件を満たした場合** にイベントが発生

   b. — をクリックすると、その条件が削除されます。

   > ⓘ 条件式の書き方や対応する演算子については、条件設定ガイド を参照してください。

発生回数

条件が何回満たされた場合にイベントを発生させるかを設定します。継続的な問題のみを検知したい場合に有効です。

1. 発生頻度を選択します。
   - **連続**: 設定した回数分、イベントが連続して発生した場合にイベントを発生
   - **直近**: 選択した時間内に、設定した回数分イベントが発生した場合にイベントを発生
     - 選択可能時間: `5秒`, `10秒`, `15秒`, `30秒`, `1分`, `2分`, `3分`, `4分`, `5分`, `10分`, `30分`, `1時間`
2. 発生回数を入力します。
   - **Interval** の値は、選択したカテゴリのデータ収集間隔を示します。（Interval: 10s）

一時停止

過剰なイベント発生を防ぐため、イベント動作を一時停止する時間を選択してください。イベント発生後、設定した時間が経過するまで同一イベントは発生しません。

- 選択可能時間: `使用しない`, `5分`, `10分`, `15分`, `20分`, `30分`, `1時間`, `2時間`, `3時間`, `6時間`, `12時間`, `1日`

> ⓘ **解消通知が有効な場合**
>
> 正常状態の通知（**RECOVERED**）を受信した時点から、設定した時間が経過するまで、同一イベントが再発しても通知は送信されません。

解消通知

解消通知機能を有効にすると、イベント発生後の状態変化（進行中 → 解消）を追跡できます。**イベント履歴** メニューでは、進行中のイベントとして表示されます。

- 解消通知は Critical および Warning レベルにのみ適用されます。Info レベルは常に単発イベントとして動作します。
- Critical または Warning レベルのイベントが解消されると、正常（RECOVERED）状態の通知が送信されます。
- トグルボタンをクリックして **有効化** または **無効化** してください。
  - **有効**: イベントは状態ベースで動作します。しきい値を超えると進行中となり、値がしきい値未満に回復すると解消状態に遷移します。
  - **無効**: しきい値を超えるたびに単発イベントが発生します。

イベント動作時間

イベントが特定の時間帯（勤務時間／非勤務時間／点検時間など）にのみ動作するよう、タグを設定してください。タグを設定しない場合、イベントが有効な状態では 24 時間常時動作します。

1. **+ 追加** をクリックします。
2. イベント動作時間で **+ 新しいタグを作成** をクリックします。
3. **イベント動作時間タグ作成** で **タグ名、曜日、時間、色** を選択し、[適用] ボタンをクリックします。
4. 作成したタグはタグ一覧で確認でき、チェックボックスを選択すると適用されます。
5. タグを修正または削除するには、 をクリックし、**イベント動作時間タグ修正** で修正または削除してください。

> タグを削除すると、そのタグが適用されているすべてのユーザーからタグが削除されます。ただし、イベントルールで使用中のタグは削除できません。

シミュレーション

シミュレーション機能では、設定したイベント条件を過去データに基づいて事前にテストできます。想定されるイベント件数が多すぎる、または少なすぎる場合は、条件を調整して最適なしきい値を見つけることができます。

- **ルール検証**: 新しく作成したイベントルールを過去データに適用した場合の結果を確認できます。
- **しきい値チューニング**: イベントが頻発する、またはまったく発生しない場合に、適切なレベルへしきい値を調整します。

シミュレーション開始

1. イベント条件の設定が完了したら、[シミュレーション] ボタンをクリックします。
2. 想定イベント件数と可視化された結果を確認します。

シミュレーション結果の理解

1. **時系列データ**

   指標の時間帯別変化と各レベルのしきい線がグラフで表示され、イベント発生タイミングを直感的に確認できます。

2. **レベル別発生状況**

   画面右側に Critical、Warning、Info レベルごとのイベント発生回数が表示され、全体の分布や頻度を把握できます。

3. **対象別詳細情報**

   グラフ上の特定ポイントにマウスを合わせると、その時点の正確な指標値と監視対象が表示されます。複数の対象がある場合は色で区別され、対象ごとにパターンが異なる場合はフィルタリング機能で個別分析が可能です。

4. **ピーク区間の特定**

   指標が急激に上昇する時間帯を確認し、問題が頻発する時間帯やパターンを把握できます。

> **シミュレーション活用のヒント**
>
> - **複数時間帯でのシミュレーション**
>
>   平日／週末、ピーク／非ピーク時間帯をそれぞれシミュレーションし、時間帯ごとの最適なしきい値を設定してください。

- **実際の障害期間を含める**

  過去に障害が発生した期間を含めてシミュレーションを実行し、現在のルールが障害を正しく検知できるか確認しましょう。

- **対象フィルタリングの追加**

  監視対象が多い場合、結果が複雑になります。フィルタリング機能を使って特定の対象のみを選択し、明確な結果を確認してください。

- **しきい値を段階的に調整**

  一度に大きく変更するのではなく、10 パーセント単位で段階的に調整しながら最適値を探すことをおすすめします。

イベント対象選択

イベント対象を指定しない場合、プロジェクト内のすべての対象に対してイベントが発生し、多数のイベントが生成される可能性があります。そのため、イベントを適用する対象を明確に設定することを推奨します。

イベント対象を変更すると、イベント件数が変わる場合があります。**シミュレーション** ボタンをクリックして再実行し、想定イベント件数を確認してください。

対象選択

イベントを適用する対象は、**選択入力** または **直接入力** の方法で指定できます。

- 選択入力　直接入力

1. タグ、演算子、値を選択または入力してください。
   a. タグが見つけにくい場合は、**メトリクス照会** を使ってタグを検索できます。
2. **+ 追加** をクリックして、条件（ `&&` , `||` ）でイベント対象を追加してください。

監視対象のタグ、演算子、値を直接入力してください。

- 直接入力を選択すると、入力欄の下に使用可能なタグ一覧（**タグ選択入力**）が表示されます。

```
ex. endsWith(okindName, 'example_name') && container == 'prod.billing'
ex. ${4xxErrorType} == '401'
```

> 入力方法を変更すると、作成中の内容は初期化されます。

基本情報および通知設定

イベント名、メッセージ、受信対象 を設定します。設定した内容はイベント履歴や実際の通知メッセージにそのまま表示されるため、識別しやすく分かりやすい名称にすることが重要です。

イベント有効化

設定した条件を満たした際にイベントを発生させるかどうかを設定します。

- **有効**: 設定した条件を満たすとイベントが発生します。
- **無効**: 条件を判定しません。

イベント名

イベントのタイトルとして使用するイベント名を入力してください。設定したイベント名は **イベント一覧** および **イベント履歴** メニューに表示され、この名前で検索することができます。

メッセージ

イベントメッセージを入力してください。入力したメッセージは、イベント履歴および通知に使用されます。 をクリックすると、過去に作成したメッセージ履歴を確認できます。メッセージには変数を使用して、イベント発生時点の実際の値を含めることができます。

- **変数の使用ルール**
  - 変数形式: `${指標名}`
  - 設定した **カテゴリと同一カテゴリ内** の指標のみを変数として使用できます。
  - 複数の変数を組み合わせて、より詳細なメッセージを作成できます。
  - 使用可能な変数一覧は **メトリクス照会** メニューで確認できます。

- **変数の使用例**

	イベント設定	イベント発生時
タイトル	Disk 使用量増加	Disk 使用量増加
メッセージ	`Disk = ${disk}%`	`Disk = 89.9%`

イベント受信

イベント通知を受信するメンバーを指定します。

- **全体受信** **タグ選択受信**

プロジェクトで通知設定が有効になっているすべての対象（メンバー、チャンネル）に通知が送信されます。

指定したタグを持つプロジェクトメンバーに通知が送信されます。

1. **タグ選択受信** を選択します。
2. **受信タグ** 項目で [+ タグ追加] または [+] ボタンをクリックします。
3. **イベント受信タグ** 画面の下にある **+ 新しいタグを作成** をクリックします。
4. **タグ作成** 画面でタグ名を入力し、色を選択した後、[タグ作成] ボタンをクリックしてタグを作成します。
5. 作成したタグは、タグ一覧の をクリックして修正または削除できます。
6. **イベント受信タグ** 画面の **タグ一覧** から希望するタグを選択すると適用されます。

> **受信タグ**
>
> 受信タグは、通知を受け取るユーザーやチャンネルを **グループ単位で管理するための機能** です。イベントに受信タグを設定すると、そのタグが付与されたユーザーにのみ通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

イベント修正／削除

1. **警告アラート** > **イベント設定** の **メトリクス** タブに移動します。

2. イベント一覧で、修正または削除したいイベントの をクリックします。
3. **イベントルール修正** 画面でオプションを修正し、[保存] ボタンをクリックします。

   a. 選択したイベントを削除する場合は、**イベントルール修正** 画面右上の [削除] ボタンをクリックします。

イベント共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules-`YYYYMMDD`.json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
   - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

> 同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。

2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

> ⓘ 修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。
>
> JSON error

JSON データ構造


```
{
  "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"
      ]
  }
}
```

表 | JSON データ構造

JSON フィールド	型	説明	備考
version	Integer	イベントルールのバージョン	変更制限
eventId	String	イベントルールの一意識別子	
enabled	boolean	イベントの有効／無効	
stateful	boolean	状態ベースイベントかどうか	
title	String	ユーザーが設定したイベントルール名	
message	String	ユーザーが設定したイベントルールメッセージ	
category	String	データカテゴリ	
alertLabel	List<String>	イベント状態管理のため、カテゴリごとに指定される基本識別子（プライマリキー）値	変更制限
repeatCount	Integer	繰り返し回数	
repeatDuration	Long	繰り返し期間	
silent	Integer	一時停止時間	
receiver	List<String>	受信タグのキー値一覧	

JSON フィールド	型	説明	備考
timeTag	`List<String>`	動作時間タグのキー値一覧	
selectString	String	対象選択	
conditions	`List<Object>`	レベル別の発生条件一覧	
ㄴ level	byte	レベル値	`Critical`: 30、`Warning`: 20、`Info`: 10
ㄴ enabled	boolean	レベルの有効／無効	
ㄴ rule	String	レベル発生条件	
createTime	Long	イベントルールの初回作成時刻	
lastModifiedTime	Long	イベントルールの最終修正時刻	
lastModifiedUser	String	イベントルールを最後に修正したアカウント	
basic	boolean	クイック設定で作成されたイベントかどうか	変更制限
metaId	String	クイック設定テンプレートの一意識別子	変更制限: この値はクイック設定で作成されたイベントにのみ存在します（basic=true）。
selectCondition	Object	クイック設定の対象選択値	この値はクイック設定で作成されたイベントにのみ存在します（basic=true）。

「変更制限」と表示されたフィールドはイベント動作に影響を与える可能性があるため、値を変更しないことを推奨します。

付録

発生条件・対象選択ガイド

メトリクスアラートにおけるイベントの発生条件とイベント対象の選択は、同一の構文を使用します。ただし、イベント発生条件ではフィールド（Field）のキーを変数として使用し、イベント対象の選択ではタグ（Tag）のキーを変数として使用します。

レベル動作ガイド

イベントレベルは、危険（Critical）、警告（Warning）、正常（Info）の3段階に分類されます。

1. **優先順位に基づくイベント発生**

   複数のレベル条件を同時に満たした場合、最も優先度の高いイベントのみが発生します。

   例

   ```
   設定:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   現在の状態: CPU 95%
   → 結果: Critical イベントのみ発生（Warning は抑制される）
   ```

2. **レベル上昇**

   進行中状態を持つイベントにおいて、低いレベルのイベントが進行中の状態で高いレベルの条件を満たすと、両方のレベルが進行中状態になります。

   例

   ```
   設定:
   - Warning: CPU > 70%
   - Critical: CPU > 90%

   シナリオ:
   ```

```
1) CPU 80% → Warning 発生（進行中）
2) CPU 95% → Warning（進行中）、Critical（進行中）
```

3. **レベル下降**

   進行中状態を持つイベントで、高いレベルから低いレベルへ遷移した場合、高いレベルは解消され、低いレベルが維持されます。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%

シナリオ:
1) CPU 97% → Critical 発生
2) CPU 85% → Critical 解消、Warning 維持
3) CPU 65% → Warning 解消
```

4. **動作フロー**

```
- Warning: CPU > 70%
- Critical: CPU > 90%

メトリクス値: 60%  →  75%  →  92%  →  85%  →  60%
状態:     正常     Warning    Critical    Warning    正常
                （発生）      （発生）      （維持）      （解消）
                          Warning
                          （維持）
```

> ⓘ 進行中状態を持つイベントは、条件が満たされている間は継続して有効となり、条件が解除されるまで維持されます。

条件設定ガイド

メトリクス名が数字で始まる場合、または特殊文字を含む場合は、 `${メトリクス名}` の形式で記述する必要があります。

- 括弧 `()`, `[]`
- 演算子 `+`, `-`, `*`, `/`, `%`
- 区切り文字 `:`, `@`, `#`, `,`, (空白)
- その他の特殊文字 `!`, `^`, `&`, `|`, `~`, `` ` ``, `=`

`${}` **使用例:**

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` **なしで使用可能:** 英字、アンダースコア（ `_` ）、ドット（ `.` ）のみを含む場合

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. **比較演算子**

   数値比較

```
より大きい:     cpu > 80
以上:          cpu >= 80
より小さい:     memory < 1000
以下:          memory <= 1000
等しい:        status == 200
等しくない:     error != 0
```

文字列比較

```
status == 'OK'
region == "us-east-1"
```

2. 算術演算子

- 基本的な算術演算

```
加算:    cpu + 10 >= 90
減算:    memory - 100 >= 500
乗算:    cpu * 2 >= 100
除算:    disk / 1024 >= 100
剰余:    value % 10 == 0
```

- 括弧による優先順位の制御

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 負の値の表現

```
cpu > -100
```

3. 論理演算子

- AND 演算（&&）

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 演算（||）

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 複合論理演算

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. パターンマッチング演算子

- LIKE 演算子

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 演算子

```
oname not like 'test-*'
url not like '*debug*'
```

5. 組み込み関数

- Null チェック関数

```
isNull(value)          # null かどうかを確認
isNotNull(value)       # null でないかを確認
nvl(value, 0)          # null の場合に既定値を返す
isEmpty(str)           # 空文字列かどうかを確認
isNotEmpty(str)        # 空文字列でないかを確認
```

- 集計関数

```
sum(cpu, memory, disk)          # 合計
avg(cpu, memory)                # 平均
max(cpu, memory, disk)          # 最大値
min(cpu, memory, disk)          # 最小値
count(value1, value2, value3)   # 件数
```

- 数学関数

```
round(cpu, 2)         # 四捨五入（小数点以下2桁）
```

- 文字列関数

```
length(str)           # 文字列の長さ
startsWith(str, 'prefix')  # 接頭辞の確認
endsWith(str, 'suffix')    # 接尾辞の確認
indexOf(str, 'search')     # 文字列の位置
substring(str, 0, 10)      # 部分文字列
trim(str)             # 空白の削除
replace(str, 'old', 'new') # 文字列の置換
hasStr(str, 'search')      # 文字列の包含判定
```

- 条件関数
  - **if**: 条件式が `true` の場合は `true値` を返し、 `false` の場合は `false値` を返す条件関数

    形式

```
if(条件, true値, false値)
```

    例

```
if(value >= 90, 'High', 'Medium') == 'High'
```

  - **decode**: 値と条件を比較し、一致した条件の結果を返します。一致しない場合は既定値を返します

    形式

```
decode(値, 条件1, 結果1, 条件2, 結果2, ..., 条件N, 結果N, 既定値)
```

    例

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

  - **in**: 値が指定した候補値のいずれかと一致した場合は `true` 、それ以外は `false` を返します

形式

```
in(値, 候補値1, 候補値2, ..., 候補値N)
```

例

```
in(status, 200, 201, 204) == false
```

記述時の注意事項

- 推奨事項
  - 簡単な変数名は `${}` を使わずに記述（例: cpu, memory）
  - 特殊文字を含む場合は必ず `${}` を使用
  - 複雑な演算は括弧で明確に区切る
  - 関数名は正確に入力（大文字と小文字を区別）
- 避けるべき事項
  - 不完全な式の記述（例: `cpu >`, `memory &&` ）
  - 括弧の不一致（例: `(cpu > 80 && cpu > 70)` ）
  - 演算子の連続使用(例: `cpu >> 80`, `cpu >< 80`)

リアルタイムログイベント

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **リアルタイムログ** タブ

収集したログデータを条件に合わせてフィルタリングして警告通知を設定できます。ログイベントを使用するには、ログモニタリングを有効化する必要があります。**ログモニタリングの有効化**については、各製品のログドキュメントを参照してください。

- **リアルタイムログイベント**: リアルタイムで収集したログで、設定したイベント条件が満たされるとイベントが発生します。

基本オプション

- **検索**: イベント名またはイベント受信タグを選択して目的のイベントを検索
- **JSONダウンロード/アップロード**: イベントルールはJSONエディタで直接編集するか、ファイルにダウンロードして修正後に再アップロード可能
- **通知メッセージ設定**: 受信するメッセージの詳細項目を選択し、項目ごとに内容を編集
- **+ イベント追加**: 新しいログイベントを追加

表 | リアルタイムログイベント一覧構成

項目	説明
イベント名	通知タイトルとして使用されるイベント名 - 最大255バイトまで入力可能
カテゴリ	ログ分類名（ログフォルダ名）
検索キー	ログデータで検索したい項目 - 例: HTTPレスポンスステータスコードの値にアクセスする場合、検索キー `status`
検索値	検索キーに対応する実際のデータ。ログに入力した文字列が含まれる場合に警告通知を送信 - 例: 検索キー `status`、検索値 `200` を設定した場合、HTTPステータスコード200を含むログデータ収集時に警告通知が発生

項目	説明
イベント発生一時停止	イベント通知発生後、同じイベントの発生を一定時間停止 - 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 - 解消通知が有効な場合、正常（RECOVERED）状態の通知受信後、選択した時間中は同じイベントが発生しても通知されない
イベント対象フィルタリング	イベントが適用される対象の条件 - 対象条件を設定しない場合、収集されるすべてのデータに対して通知を判断 - 選択入力: タグ、演算子、値を選択して条件を構成 - 直接入力: 条件を直接入力して対象を指定
イベント受信タグ	イベント通知を受け取るユーザーまたはグループ（受信タグ）
	イベントの編集と削除
	イベントの有効化状態 - 有効: 条件を満たすとイベント発生 - 無効: イベントが動作せず、条件をチェックしない

イベント設定

ログイベントは、リアルタイムで収集されるログ、または特定条件を満たすログが一定数蓄積された際に警告通知を設定できる機能です。

リアルタイムログイベントの追加

1. **アラート通知** > **イベント設定**の**リアルタイムログ**タブをクリックしてください。
2. 画面右側の [+ イベント追加] ボタンをクリックしてください。
3. **イベントルール追加**ウィンドウでログイベント条件を設定してください。
    - **イベント名**: イベントタイトルとして使用する名前を入力してください。

- **イベント有効化**: トグルボタンを有効にすると、条件を満たしたときにイベントが発生します。
- **レベル**: イベントレベル（Critical、Warning、Info）を選択してください。
- **メッセージ**: イベント発生時に受け取る通知メッセージを入力してください。メッセージに変数を使用して、イベント発生時点の実際の値を含めることができます。

  > **変数使用ルール**
  >
  > - 変数形式: `${指標名}`
  > - 設定した**カテゴリと同じカテゴリ**内の指標のみ変数として使用できます。
  > - 複数の変数を組み合わせて詳細なメッセージを構成できます。
  > - 使用可能な変数一覧は**メトリクス照会**で確認できます。

- **カテゴリ**: ログ分類名（ログフォルダ名）をリストから選択するか、直接入力してください。カテゴリを選択してから検索キーと検索値を入力できます。
- **検索キー**: ログデータで検索したい項目を入力してください。
  - 例: HTTPレスポンスステータスコードの値にアクセスする場合、**検索キー** `status`
- **検索値**: **検索キー**に対応する実際のデータを入力してください。ログに入力した文字列が含まれる場合に警告通知を送信します。
  - 例: **検索キー** `status` 、**検索値** `200` を設定した場合、HTTPステータスコード200を含むログデータ収集時に警告通知が発生
- **イベント対象フィルタリング**: イベントが適用される対象の条件を入力してください。入力値がない場合、収集されるすべてのデータに対して通知を判断します。
  - **選択入力**: タグ、演算子、値を選択して対象条件を設定してください。

    > 条件式の記述方法とサポートされる演算子については、条件設定ガイドを参照してください。

  - **直接入力**: 条件を直接入力してください。
- **イベント発生一時停止**: 通知を一時停止する時間を選択してください。イベント発生後、設定した時間中は同じイベントが発生しません。

    - 選択可能時間: `使用しない`、`1分`、`2分`、`3分`、`5分`、`10分`、`20分`、`30分`、`1時間`、`3時間`、`6時間`、`12時間`

- **イベント動作時間**: 特定の時間帯（業務/非業務/メンテナンス時間など）にイベントが動作するようにタグを設定してください。タグを設定しない場合、イベントが有効な状態で24時間常時動作します。

    1. **+ 追加**をクリックしてください。
    2. **イベント動作時間**で**+新しいタグを作成**をクリックしてください。
    3. **イベント動作時間タグの作成**で**タグ名、曜日、時間、色**を選択し、[適用] ボタンをクリックしてください。
    4. 作成したタグはタグリストで確認でき、チェックボックスを選択すると適用されます。
    5. 変更または削除するタグの✎アイコンをクリックして、**イベント動作時間タグの編集**でタグを編集または削除できます。

    > タグを削除すると、そのタグが適用されているすべてのユーザーのタグが削除されます。ただし、イベントルールで使用中のタグは削除できません。

- **イベント受信タグ**: イベントに受信タグを設定すると、そのタグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

    1. **+ タグ追加**または+をクリックしてください。
    2. **イベント受信タグ**ウィンドウ下部の**+ 新しいタグを作成**をクリックしてください。
    3. **タグ作成**ウィンドウでタグ名を入力し、色を選択してから、[タグ作成] ボタンをクリックしてタグを作成してください。
    4. 作成されたタグはタグリストの✎アイコンをクリックして編集または削除してください。
    5. **イベント受信タグ**ウィンドウの**タグリスト**から目的のタグを選択すると適用されます。

    > イベント設定時にイベント受信タグを選択して、そのタグを持つプロジェクトメンバーとサードパーティプラグインに通知を送信できます。**アラート通知 > 通知設定**でプロジェクトメンバーとサードパーティプラグインにそれぞれタグを指定できます。

4. **イベントルール追加**ウィンドウで設定が完了したら、[保存] ボタンをクリックしてください。

イベントの編集/削除

1. **アラート通知** > **イベント設定**の**リアルタイムログ**タブに移動してください。
2. ログイベントリストで編集または削除するイベントの アイコンをクリックしてください。
3. **イベント設定**画面でオプションを変更し、[保存] ボタンをクリックしてください。

   a. 選択したイベントを削除するには、**イベント設定**画面の右上の [削除] ボタンをクリックしてください。

イベントの共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules-`YYYYMMDD`.json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
   - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

> 同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。
2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

> 修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。
>
> JSON error

付録

発生条件・対象選択ガイド

メトリクスアラートにおけるイベントの発生条件とイベント対象の選択は、同一の構文を使用します。ただし、イベント発生条件ではフィールド（Field）のキーを変数として使用し、イベント対象の選択ではタグ（Tag）のキーを変数として使用します。

レベル動作ガイド

イベントレベルは、危険（Critical）、警告（Warning）、正常（Info）の3段階に分類されます。

1. **優先順位に基づくイベント発生**

   複数のレベル条件を同時に満たした場合、最も優先度の高いイベントのみが発生します。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%
```

```
現在の状態: CPU 95%
→ 結果: Critical イベントのみ発生（Warning は抑制される）
```

2. **レベル上昇**

   進行中状態を持つイベントにおいて、低いレベルのイベントが進行中の状態で高いレベルの条件を満たすと、両方のレベルが進行中状態になります。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%

シナリオ:
1) CPU 80% → Warning 発生（進行中）
2) CPU 95% → Warning（進行中）、Critical（進行中）
```

3. **レベル下降**

   進行中状態を持つイベントで、高いレベルから低いレベルへ遷移した場合、高いレベルは解消され、低いレベルが維持されます。

   例

```
設定:
- Warning: CPU > 70%
- Critical: CPU > 90%

シナリオ:
1) CPU 97% → Critical 発生
2) CPU 85% → Critical 解消、Warning 維持
3) CPU 65% → Warning 解消
```

4. **動作フロー**

```
- Warning: CPU > 70%
- Critical: CPU > 90%
```

```
メトリクス値: 60%  →  75%  →  92%  →  85%  →  60%
状態:     正常    Warning   Critical   Warning   正常
                （発生）    （発生）    （維持）    （解消）
                        Warning
                          （維持）
```

進行中状態を持つイベントは、条件が満たされている間は継続して有効となり、条件が解除されるまで維持されます。

条件設定ガイド

メトリクス名が数字で始まる場合、または特殊文字を含む場合は、`${メトリクス名}` の形式で記述する必要があります。

- 括弧 `()`, `[]`
- 演算子 `+`, `-`, `*`, `/`, `%`
- 区切り文字 `:`, `@`, `#`, `,`, (空白)
- その他の特殊文字 `!`, `^`, `&`, `|`, `~`, `` ` ``, `=`

`${}` **使用例:**

```
${cpu(xos)} > 50
${mem[0]} >= 100
${cpu-usage} > 80
${namespace:cpu} > 70
${metric name} > 60
${4xx_error} > 10
```

`${}` **なしで使用可能:** 英字、アンダースコア（`_`）、ドット（`.`）のみを含む場合

```
cpu > 80
cpu_usage > 50
CPUUtilization.Average > 0.8
```

1. 比較演算子

数値比較

```
より大きい:      cpu > 80
以上:           cpu >= 80
より小さい:      memory < 1000
以下:           memory <= 1000
等しい:          status == 200
等しくない:      error != 0
```

文字列比較

```
status == 'OK'
region == "us-east-1"
```

2. 算術演算子

- 基本的な算術演算

```
加算:      cpu + 10 >= 90
減算:      memory - 100 >= 500
乗算:      cpu * 2 >= 100
除算:      disk / 1024 >= 100
剰余:      value % 10 == 0
```

- 括弧による優先順位の制御

```
(cpu + memory) * 2 >= 200
(disk - used) / total >= 0.2
((cpu + memory) / 2) >= 50
```

- 負の値の表現

```
cpu > -100
```

3. 論理演算子

- AND 演算（&&）

```
cpu > 80 && memory > 1000
${cpu(xos)} > 50 && ${mem(xos)} > 60
```

- OR 演算（||）

```
cpu > 90 || memory > 90
disk < 10 || network > 1000
```

- 複合論理演算

```
(cpu > 50 && memory > 50) || disk < 20
cpu > 80 && (memory > 1000 || disk > 500)
```

4. パターンマッチング演算子

- LIKE 演算子

```
oname like 'prod-*'
url like '*error*'
message like 'WARN%'
```

- NOT LIKE 演算子

```
oname not like 'test-*'
url not like '*debug*'
```

5. 組み込み関数

- Null チェック関数

```
isNull(value)          # null かどうかを確認
isNotNull(value)       # null でないかを確認
nvl(value, 0)          # null の場合に既定値を返す
isEmpty(str)           # 空文字列かどうかを確認
isNotEmpty(str)        # 空文字列でないかを確認
```

- 集計関数

```
sum(cpu, memory, disk)          # 合計
avg(cpu, memory)                # 平均
max(cpu, memory, disk)          # 最大値
min(cpu, memory, disk)          # 最小値
count(value1, value2, value3)     # 件数
```

- 数学関数

```
round(cpu, 2)         # 四捨五入（小数点以下2桁）
```

- 文字列関数

```
length(str)             # 文字列の長さ
startsWith(str, 'prefix')  # 接頭辞の確認
endsWith(str, 'suffix')   # 接尾辞の確認
indexOf(str, 'search')    # 文字列の位置
substring(str, 0, 10)     # 部分文字列
trim(str)               # 空白の削除
replace(str, 'old', 'new') # 文字列の置換
hasStr(str, 'search')     # 文字列の包含判定
```

- 条件関数
  - **if**: 条件式が `true` の場合は `true値` を返し、 `false` の場合は `false値` を返す条件関数

    形式

    ```
    if(条件, true値, false値)
    ```

    例

    ```
    if(value >= 90, 'High', 'Medium') == 'High'
    ```

  - **decode**: 値と条件を比較し、一致した条件の結果を返します。一致しない場合は既定値を返します

    形式

```
decode(値, 条件1, 結果1, 条件2, 結果2, ..., 条件N, 結果N, 既定値)
```

例

```
decode(value, 1, 'Low', 2, 'Medium', 3, 'High', 'Unknown') == 'Unknown'
```

- **in**: 値が指定した候補値のいずれかと一致した場合は `true`、それ以外は `false` を返します

形式

```
in(値, 候補値1, 候補値2, ..., 候補値N)
```

例

```
in(status, 200, 201, 204) == false
```

記述時の注意事項

- 推奨事項
  - 簡単な変数名は `${}` を使わずに記述（例: cpu, memory）
  - 特殊文字を含む場合は必ず `${}` を使用
  - 複雑な演算は括弧で明確に区切る
  - 関数名は正確に入力（大文字と小文字を区別）
- 避けるべき事項
  - 不完全な式の記述（例: `cpu >`, `memory &&` ）
  - 括弧の不一致（例: `(cpu > 80 && cpu > 70)` ）
  - 演算子の連続使用(例: `cpu >> 80`, `cpu >< 80`)

複合ログイベント設定

ホーム > プロジェクト選択 > **アラート通知** > **イベント設定** > **複合ログ** タブ

収集したログデータを条件に合わせてフィルタリングして警告通知を設定できます。ログイベントを使用するには、ログモニタリングを有効化する必要があります。**ログモニタリングの有効化**については、各製品のログドキュメントを参照してください。

- **複合ログイベント**: 設定した**データ照会範囲**内のログを基準に、ログ検索条件を満たすログ数が閾値以上の場合にイベントが発生します。

> **権限**
>
> 複合メトリクスイベントを設定するには**通知設定**権限が必要です。 通知設定権限は、**ホーム画面 > プロジェクト選択 > 管理 > 統合メンバー管理**で、権限を付与するメンバーの アイコンをクリックして設定できます。

基本オプション

- **検索**: イベント名またはイベント受信タグを選択して目的のイベントを検索
- **JSONダウンロード/アップロード**: イベントルールはJSONエディタで直接編集するか、ファイルにダウンロードして修正後に再アップロード可能
- **通知メッセージ設定**: 受信するメッセージの詳細項目を選択し、項目ごとに内容を編集
- **+ イベント追加**: 新しいログイベントを追加

表 | 複合ログイベント一覧構成

項目	説明
イベント名	通知タイトルとして使用されるイベント名
テンプレート	複合ログテンプレート

項目	説明
カテゴリ	ログ分類名（ログフォルダ名）
ルール	イベントが発生する条件
イベント状態が解消されたら追加通知	CriticalとWarningレベルのイベントが解消されたとき、正常（RECOVERED）状態の通知を受信するかどうか - 有効: 進行中 → 正常状態に移行したときRECOVERED通知を送信し、イベント履歴で進行中/解消状態を区別して確認可能 - 無効: 閾値を超えるたびに単発のイベントが発生
イベント発生一時停止	イベント通知発生後、同じイベントの発生を一定時間停止 - 選択可能時間: 使用しない、5分、10分、15分、20分、30分、1時間、2時間、3時間、6時間、12時間、1日 - 解消通知が有効な場合、正常（RECOVERED）状態の通知受信後、選択した時間中は同じイベントが発生しても通知されない
インターバル	ログデータを照会する時間範囲
イベント受信タグ	イベント通知を受け取るユーザーまたはグループ（受信タグ）
	イベントの編集と削除
	イベントの有効化状態 - 有効: 条件を満たすとイベント発生 - 無効: イベントが動作せず、条件をチェックしない

イベント設定

設定した**データ照会範囲**内のログを基準に、ログ検索条件を満たすログ数が閾値以上の場合に警告通知を設定できる機能です。

複合ログイベントの追加

1. **アラート通知** > **イベント設定**の**複合ログ** タブをクリックしてください。
2. 画面右側の [+ イベント追加] ボタンをクリックしてください。
3. **イベントルール追加**ウィンドウでログイベント条件を設定してください。詳細については、基本設定、詳細設定、追加設定を参照してください。

   複合ログイベントの追加
4. **イベントルール追加**ウィンドウで設定が完了したら、[保存] ボタンをクリックしてください。

基本設定

- **イベント名**: イベントタイトルとして使用する名前を入力してください。
- **イベント有効化**: トグルボタンを有効にすると、条件を満たしたときにイベントが発生します。
- **レベル**: イベントレベル（Critical、Warning、Info）を選択してください。

詳細設定

テンプレート（プリセット）、メッセージ、カテゴリ、**フィルター**、イベント発行条件を設定します。

テンプレート（プリセット）

定義済みのプリセットを選択すると、**フィルター、クエリタイプ、イベント発行条件**が自動的に設定されます。プリセットを選択後、必要に応じて変更できます。

複合ログテンプレートの種類

プリセット	クエリタイプ	デフォルトルール
使用しない	none	`rows > 10`
2xxステータスコード件数	count	`count > 10`

プリセット	クエリタイプ	デフォルトルール
3xxステータスコード件数	count	`count > 10`
4xxステータスコード件数	count	`count > 10`
5xxステータスコード件数	count	`count > 10`
正常ステータスコード(2xx, 3xx)件数	count_v2	`include_minus_exclude_count > 10`
エラーステータスコード(4xx, 5xx)件数	count_v2	`include_minus_exclude_count > 10`
エラー受信件数	count	`count > 10`
例外受信件数	count	`count > 10`
毎秒ログ発生件数	rps	`rps > 100`
フィールド集計（Aggregation）	aggr	`avg > 1000`
比率（Rate）	rate	`rate > 10`
ユニーク値数（Cardinality）	cardinality	`cardinality > 100`
パーセンタイル（Percentile）	percentile	`p99 > 3000`

メッセージ

イベント発生時に受け取る通知メッセージを入力してください。メッセージに変数を使用して動的なメッセージを構成できます。

- **例示:** メッセージ `${host}` でエラー `${count}` 件発生 → 通知メッセージ web-01でエラー15件発生

表 | メッセージ変数

変数	説明	使用条件
${count}	条件に一致するログ件数	常に使用可能
${groupField}	グループ化フィールド名	グループ化通知のみ
${groupValue}	グループ化フィールド値	グループ化通知のみ
${フィールド名}	グループ化フィールド値（フィールド名で直接参照）	グループ化通知のみ

> ${count} 変数はグループ化の有無に関わらず、すべての通知で使用できます。 ${groupField} 、 ${groupValue} 、 ${フィールド名} 変数はグループ化フィールドが設定されている場合のみ置換されます。

カテゴリ

ログ分類名（ログフォルダ名）をリストから選択するか、直接入力してください。カテゴリを選択してから検索キーと検索値を入力できます。

フィルター

Luceneクエリ構文でログ検索条件を入力します。2種類の入力モードをサポートします。

- **フィルターモード**: フィールドと値を選択してクエリを構成します。
- **コードモード**: Luceneクエリを直接入力します。

表 | Luceneクエリ例

クエリ	説明
level:ERROR	levelフィールドがERRORのログ
status_nxx:status_4xx OR status_nxx:status_5xx	4xxまたは5xxステータスコード
appender:accessLog	accessLog appenderのログ
response_time.n:*	response_time.nフィールドが存在するログ

クエリ	説明
host:web-01 AND level:ERROR	web-01ホストのエラーログ

イベント発行条件

クエリタイプ別に設定します。選択したテンプレート（プリセット）またはクエリタイプに応じて、追加設定項目が表示されます。

表 | イベント発行条件キーの意味

キー	意味
rows / count	フィルター条件に一致するログ件数
rps	毎秒ログ発生件数（件数 / インターバル秒）
avg	対象フィールド値の平均
sum	対象フィールド値の合計
min	対象フィールド値の最小値
max	対象フィールド値の最大値
rate	分子件数 / 分母件数 × 100（%）
cardinality	対象フィールドのユニーク値数
p99 / p95 / p90 / p75 / p50	対象フィールドのパーセンタイル値
include_minus_exclude_count	含むフィルター件数 - 除外フィルター件数

- **件数（count / none）**: 設定したフィルター条件に一致するログの件数をカウントします。

  表 | 件数設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
イベント発行条件	`件数` または `条件に一致するログ` `{演算子}` `{閾値}`（例: `count > 10`）

- **毎秒ログ発生件数（rps）**: 設定したフィルター条件に一致するログの毎秒発生件数（Requests Per Second）を計算します。インターバル時間中の総件数をインターバル秒数で割って算出します。

表 | 毎秒ログ発生件数設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
イベント発行条件	`RPS` `{演算子}` `{閾値}`（例: `rps > 0.5`）

- **フィールド集計（aggr）**: 指定した数値フィールドの集計値（平均、合計、最小値、最大値）を計算します。イベント発行条件で集計方式をドロップダウンから選択できます。

表 | フィールド集計設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
対象フィールド	集計対象の数値フィールド（例: `response_time.n`）
イベント発行条件	`平均` / `合計` / `最小値` / `最大値` `{演算子}` `{閾値}`（例: `avg > 1000`）

- **比率（rate）**: 分子フィルターと分母フィルターの件数比率（%）を計算します。全ログに対する特定条件の比率を監視する際に有用です。

表 | 比率設定項目

設定項目	説明
フィルター（分子）	比率計算の分子となるログ条件（例: `level:ERROR`）
分母フィルター	比率計算の分母となるログ条件（例: `level:*`）
イベント発行条件	`比率（%）` `{演算子}` `{閾値}`（例: `rate > 10`）

  - **例示**: フィルター `level:ERROR`、分母フィルター `level:*`、ルール `rate > 10` → 全ログ中のエラー比率が10%を超えると通知が発生

- **ユニーク値数（cardinality）**: 指定したフィールドのユニーク値数を計算します。異常に多くの種類の値が発生する場合を検出する際に有用です。

表 | ユニーク値数設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
対象フィールド	ユニーク値を計算するフィールド（例: `host`）
イベント発行条件	`ユニーク値数` `{演算子}` `{閾値}`（例: `cardinality > 100`）

- **パーセンタイル（percentile）**: 指定した数値フィールドのパーセンタイル値を計算します。レスポンスタイムのテールレイテンシを監視する際に有用です。パーセンタイルの選択を変更すると、イベント発行条件のキーが自動的に更新されます。

表 | パーセンタイル設定項目

設定項目	説明
フィルター	ログ検索条件（Luceneクエリ）
対象フィールド	パーセンタイルを計算する数値フィールド（例: `response_time.n`）
パーセンタイル	P99、P95、P90、P75、P50から選択

設定項目	説明
イベント発行条件	`P99` / `P95` /... `{演算子}` `{閾値}`（例: `p99 > 3000`）

表 | パーセンタイルの意味

パーセンタイル	意味
P50	中央値。全データの50%がこの値以下
P75	全データの75%がこの値以下
P90	全データの90%がこの値以下
P95	全データの95%がこの値以下
P99	全データの99%がこの値以下

- **含む/除外件数（count_v2）**: 含むフィルターの件数から除外フィルターの件数を引いた純件数を計算します。

表 | 含む/除外件数設定項目

設定項目	説明
フィルター（含む）	含むログ条件（例: `status_nxx:status_4xx OR status_nxx:status_5xx`）
フィルター除外	除外するログ条件（例: `status_nxx:status_4xx`）
分母フィルター	比率計算用の分母（任意）
イベント発行条件	`含む-除外件数` `{演算子}` `{閾値}`（例: `include_minus_exclude_count > 10`）

  - **例示**: フィルター `status_nxx:status_4xx OR status_nxx:status_5xx`、フィルター除外 `status_nxx:status_4xx`
    → 4xx+5xxから4xxを除いた純5xx件数が10を超えると通知が発生

グループ化フィールド（Group By）

グループ化フィールドを設定すると、そのフィールドの値ごとに独立して通知条件が判定されます。ドロップダウンからグループ化するフィールドを選択します。ドロップダウンをクリックすると、現在のカテゴリで使用可能なフィールド一覧がオートコンプリートで表示されます。

- **例示**: フィルター `level:ERROR`、グループ化フィールド `host`、ルール `rows > 3`

表 | グループ化設定の比較

項目	グループ化なし	グループ化あり
判定単位	全ログを1つに集計	グループ値ごとに独立して集計
通知発生	1件	条件を満たすグループ数分
通知タイトル	`イベントタイトル`	`[グループ値] イベントタイトル`
Stateful	全体基準での状態遷移	グループごとに独立した状態遷移

  - `[web-01] エラー通知` — web-01でエラー8件 → 通知発生
  - `[web-02] エラー通知` — web-02でエラー5件 → 通知発生
  - `[api-01] エラー通知` — api-01でエラー2件 → 条件未達、通知なし

> ⓘ **イベント状態が解消されたら追加通知**を有効にした場合、グループごとに独立してABNORMAL → RECOVERED状態が遷移します。例えば `web-01` のエラーが解消されると `[web-01] RECOVERED` 通知が発生し、他のホストの状態には影響しません。

追加設定

- **インターバル**: 選択した時間（`unset`、`1分`、`5分`、`10分`、`30分`、`1時間`、`12時間`）間隔で通知条件を確認します。
- **無音**: イベント発生後、選択した時間（`unset`、`1分`、`5分`、`10分`、`30分`、`1時間`、`12時間`）中は同じイベントが発生しません。

- **イベント動作時間**: 特定の時間帯（業務/非業務/メンテナンス時間など）にイベントが動作するようにタグを設定してください。タグを設定しない場合、イベントが有効な状態で24時間常時動作します。

    1. **+ 追加**をクリックしてください。
    2. **イベント動作時間**で**+新しいタグを作成**をクリックしてください。
    3. **イベント動作時間タグの作成**で**タグ名、曜日、時間、色**を選択し、[適用] ボタンをクリックしてください。
    4. 作成したタグはタグリストで確認でき、チェックボックスを選択すると適用されます。
    5. 変更または削除するタグの✎アイコンをクリックして、**イベント動作時間タグの編集**でタグを編集または削除できます。

    > タグを削除すると、そのタグが適用されているすべてのユーザーのタグが削除されます。ただし、イベントルールで使用中のタグは削除できません。

- **イベント受信タグ**: イベントに受信タグを設定すると、そのタグを持つプロジェクトメンバーに通知が送信されます。受信タグを設定しない場合、プロジェクト内のすべてのユーザーに通知が送信されます。

    1. **+ タグ追加**または+をクリックしてください。
    2. **イベント受信タグ**ウィンドウ下部の**+ 新しいタグを作成**をクリックしてください。
    3. **タグ作成**ウィンドウでタグ名を入力し、色を選択してから、[タグ作成] ボタンをクリックしてタグを作成してください。
    4. 作成されたタグはタグリストの✎アイコンをクリックして編集または削除してください。
    5. **イベント受信タグ**ウィンドウの**タグリスト**から目的のタグを選択すると適用されます。

    > イベント設定時にイベント受信タグを選択して、そのタグを持つプロジェクトメンバーとサードパーティプラグインに通知を送信できます。**アラート通知 > 通知設定**でプロジェクトメンバーとサードパーティプラグインにそれぞれタグを指定できます。

シミュレーション

通知を保存する前に、過去のデータでシミュレーションして通知発生予測件数を確認できます。

1. 通知設定を完了後、下部の時間範囲を選択してください。
2. [シミュレーション] ボタンをクリックしてください。
3. チャートに時間帯別データと通知発生回数が表示されます。

   > シミュレーション時間範囲を調整して、さまざまな時間帯の通知発生パターンを確認できます。直近1〜2分のデータはインデックスビルド遅延により照会されない場合があります。

4. **イベントルール追加**ウィンドウで設定が完了したら、[保存] ボタンをクリックしてください。

ログエクスプローラーから通知を作成

ログエクスプローラーから直接通知を作成できます。

1. **ログ** > **ログ検索**メニューに移動してください。
2. カテゴリと検索クエリを設定してください。
3. 画面右上の [イベント追加] ボタンをクリックしてください。
4. Drawerで通知設定フォームが開き、現在のカテゴリと検索クエリが自動的に入力されます。
5. 追加設定を完了し、[保存] ボタンをクリックしてください。
6. 保存完了後、[移動] ボタンをクリックすると通知一覧に移動します。

イベントの編集/削除

1. **アラート通知** > **イベント設定**の**複合ログ**タブに移動してください。
2. ログイベントリストで編集または削除するイベントの アイコンをクリックしてください。
3. **イベント設定**画面でオプションを変更し、[保存] ボタンをクリックしてください。

   a. 選択したイベントを削除するには、**イベント設定**画面の右上の [削除] ボタンをクリックしてください。

イベントの共有

メトリクスイベント設定を JSON ファイルとして保存し、他のユーザーと設定を共有したり、他のユーザーの設定を取り込んだりすることができます。

- JSON ファイル名: event-rules-`YYYYMMDD`.json

エクスポート

1. 画面右上の [**JSON**] ボタンをクリックします。
2. JSON 編集画面が表示されたら、[エクスポート] ボタンをクリックします。
    - イベントを検索した状態でエクスポート機能を使用すると、検索結果のみが JSON ファイルとしてダウンロードされます。
3. JSON ファイルがダウンロードされたら、共有したい他のユーザーに渡してください。

インポート

1. 画面右上の [ ] ボタンをクリックします。
2. **エクスポート** 機能でダウンロードした JSON ファイルを選択します。
3. JSON 編集画面が表示されたら、[一覧に追加] または [上書き] を選択します。

> 同じ種類の製品間での利用を推奨します。異なる製品のプロジェクトからイベント設定を取り込むことは可能ですが、正常に動作しない場合があります。

JSON 形式で編集する

1. 画面右上の [**JSON**] ボタンをクリックします。
2. 編集画面が表示されたら、JSON 形式に従って内容を修正してください。
3. 修正が完了したら、画面下部の [保存] ボタンをクリックします。

修正した内容が JSON 形式に合っていない場合、画面下部にエラーメッセージが表示され、保存できません。表示されるエラーメッセージは、形式エラーの内容によって異なる場合があります。

JSON error

通知設定

ホーム > プロジェクト選択 > **アラート通知** > **通知設定**

ユーザー別アラート受信設定

ユーザー別アラート受信設定 における他メンバーの情報は、**メンバー管理** 権限を持つメンバーのみが確認できます。それ以外のメンバーには、必須情報がマスキングされて表示されます。

- イベント通知の一括受信設定およびアクセス設定のためのモバイル端末管理は、**アカウント情報** で行うことができます。
- メンバー権限の詳細については、こちらのドキュメント を参照してください。

通知手段の設定

メール、**SMS**、**WhatsApp**、**モバイル** による通知を提供します。

ユーザー別アラート受信設定 セクションで、受信したい通知手段のチェックボックスを選択すると警告アラートを受信できます。通知手段のチェックボックスを解除すると、その手段ではアラートは送信されません。

プロジェクトの最高管理者を除き、すべてのユーザーは自分自身の受信設定のみ変更できます。

メール通知

メール通知は、会員登録時に入力したメールアドレス宛に送信されます。

SMS通知

SMSで通知を受信できます。

SMS通知機能は、有料プロジェクトでのみ利用できます。

1. 画面右上の **プロフィールアイコン** をクリックし、**アカウント管理** に移動します。
2. **アカウント情報** の **ユーザー電話番号** セクションで、[一般携帯電話] ボタンをクリックします。
3. 認証番号を受信する **電話番号** を入力します。
    - SMS通知を受信できる電話番号は、**韓国の携帯電話番号のみ**対応しています。
4. [保存] ボタンをクリックします。

> **登録した電話番号の変更および削除**
>
> - 電話番号を変更する場合は、[変更] ボタンをクリックしてください。SMS認証が必要です。
> - 電話番号を削除する場合は、[削除] ボタンをクリックしてください。削除後は、SMSによる通知を受信できなくなります。

WhatsApp通知

WhatsAppを通じて通知を受信できます。

> WhatsApp通知機能は、有料プロジェクトでのみ利用できます。

1. 画面右上の **プロフィールアイコン** をクリックし、**アカウント管理** に移動します。
2. **アカウント情報** の **ユーザー電話番号** セクションで、[WhatsApp] ボタンをクリックします。
3. 認証番号を受信する **電話番号** を入力します。
4. [認証番号送信] ボタンをクリックします。
5. WhatsAppアプリケーションに送信された6桁の認証番号を入力します。
    - 10分以内に認証番号を受信できなかった場合は、[認証番号再送信] ボタンをクリックしてください。
6. [認証] ボタンをクリックします。

モバイル通知

モバイル通知を受信するには、モバイル端末にWhaTapアプリケーションをインストールし、ログインする必要があります。

- WhaTapアプリケーションをダウンロード – **Android**
- WhaTapアプリケーションをダウンロード – **iOS**

繰り返し通知（エスカレーション）設定

アラート発生後も状況が解消されない場合、初回アラート発生時刻を基準に通知を繰り返す間隔を設定できます。この設定は **Critical** レベルのアラートにのみ適用されます。

例えば、初回アラート発生から即時（0分）、1時間後、1日後に通知を繰り返す場合は、**繰り返し通知（エスカレーション）** カラムに `0,1H,1D` と入力します。

1. **アラート通知** > **通知設定** に移動し、**ユーザー別アラート受信設定** セクションを開きます。
2. **繰り返し通知（エスカレーション）** 項目で設定します。
   - **M**: 分、**H**: 時間、**D**: 日。単位を省略した場合は分単位として扱われます。
   - 数値、または 数値＋単位（**M**, **H**, **D**）の形式で入力してください。入力形式が正しくない場合はメッセージが表示されます。
3. [保存] ボタンをクリックしてください。保存しない場合、設定は反映されません。

イベント受信タグ

イベント設定時に **イベント受信タグ** を選択すると、そのタグを持つプロジェクトメンバーや 3rd-party プラグインに通知を送信できます。イベント受信タグを設定しない場合、**すべてのプロジェクトメンバー** に警告アラートが送信されます。

イベント受信タグの追加・編集・削除

> イベントに **適用中のイベント受信タグ** は削除できません。

1. **アラート通知** > **通知設定** に移動します。
2. **ユーザー別アラート受信設定** セクションのユーザー一覧で、**+ タグ追加** または + をクリックします。
3. **イベント受信タグ** 画面で **+ 新しいタグ作成** をクリックします。
4. **タグ作成** 画面でタグ名を入力し、色を選択して [タグ作成] ボタンをクリックします。

5. 作成されたタグは、タグ一覧の をクリックして編集または削除できます。
6. **イベント受信タグ** 画面の **タグ一覧** から使用したいタグを選択すると適用されます。

イベント受信タグを解除する

1. **アラート通知** > **通知設定** に移動し、**ユーザー別アラート受信設定** セクションを開きます。
2. イベント受信タグを解除したいユーザーの **イベント受信タグ** 項目にある + をクリックします。
3. **イベント受信タグ** 画面で、解除したいタグの **X** をクリックすると、そのタグが解除されます。

受信タグ未設定のアラートを受信する

イベント設定時にイベント受信タグを選択すると、そのタグを持つプロジェクトメンバーや 3rd-party プラグインに通知を送信できます。

- イベント受信タグが設定されていないアラートを受信するには、**受信タグ未設定のアラートを受信** オプションを有効にしてください。
- イベント受信タグが設定されているアラートのみ受信したい場合は、このオプションを無効にしてください。

> すべての警告アラートを受信しない場合は、このオプションを無効にし、**イベント受信タグ** が一つも選択されていない状態にしてください。

イベント受信モードの選択

ユーザー別アラート受信設定セクションで、メンバーまたはサードパーティチャネルごとにイベント通知の受信範囲を次の3つのモードから選択できます。受信モードは、各メンバーまたはチャネル行の**イベント受信タグ**列で選択します。

Mode	Description
すべての通知を受信	イベント受信タグに関係なく、すべての警告通知を受信
選択したタグのみ通知を受信	選択したイベント受信タグを持つ警告通知のみ受信。タグを最低1つ以上選択
すべての通知を受信しない	いかなるイベント通知も受信しない。選択したタグは削除されずそのまま保存

- **選択したタグのみ通知を受信**モードでは、タグ選択領域で通知を受信するタグを1つ以上選択してください。
- イベント受信タグが指定されていない警告通知まで受信するには、タグ選択領域で**未分類**仮想タグを選択してください。**未分類**タグは通常のタグと同様に選択または解除でき、タグ自体を編集・削除することはできません。

> ⓘ **すべての通知を受信しない**モードでは、該当メンバーまたはチャネルがイベント通知をまったく受信しません。このとき選択したタグは削除されず保存されるため、再び**選択したタグのみ通知を受信**モードに切り替えると、以前に選択したタグ設定をそのまま使用できます。

通知受信言語の選択

ユーザーが受信するアラート通知の言語を選択できます。

1. **アラート通知** > **通知設定** に移動します。
2. **ユーザー別アラート受信設定** セクションのユーザー一覧で、言語を設定したいユーザーの一番左にある ▶ ボタンをクリックします。
3. **通知受信言語** を選択します。

曜日・時間別通知設定

ユーザーごとに、曜日および時間帯別の通知受信可否を設定できます。

1. **アラート通知** > **通知設定** に移動します。
2. **ユーザー別アラート受信設定** セクションのユーザー一覧で、曜日・時間別通知を設定したいユーザーの一番左にある ▶ ボタンをクリックします。
3. 警告アラートを受信したい曜日を選択するか、時間帯を入力します。
    - 通知手段ごとに設定できます。

サードパーティプラグイン

外部アプリケーションを通じて警告アラートを受信できます。

- **対応アプリケーション**: Slack、Telegram、Teams、Teams Workflows、Jandi、AlertNow、Webhook、Webhook

JSON、Opsgenie、PagerDuty、LINE、ilert

> WhaTap のサポート範囲に含まれない社内メッセンジャーは、標準の **Webhook** または **Webhook JSON** を使用して連携できます。

表 | サードパーティプラグイン一覧

項目	説明
プラグイン名	プラグインの名称
認証キー	認証キー
認証値	認証値
受信レベル	警告、全体、危険のいずれか
繰り返し通知（エスカレーション）	問題が解消されない場合、初回通知時刻から再通知する間隔 - Critical レベルのアラートにのみ適用
Smart メッセージ（AI）	Telegram プラグイン追加時に AI 通知機能を提供 - 対応言語：韓国語、英語、日本語
イベント受信タグ	プロジェクト内の特定メンバーまたはチームに対する通知受信設定
削除	サードパーティプラグインの削除

サードパーティプラグインの追加

1. **アラート通知** > **通知設定** に移動します。
2. **サードパーティプラグイン** セクションで **追加** ボタンをクリックします。
3. **サードパーティプラグイン追加** 画面で、使用するサービスのタブを選択します。
4. 選択したサービスの案内（サードパーティプラグイン通知の追加方法）に従って設定します。
5. **受信テスト** ボタンをクリックして設定を確認します。

6. すべて完了したら **追加** または **登録** ボタンをクリックします。

サードパーティプラグイン通知の追加方法

Slack

サードパーティプラグイン追加 画面の **Slack** タブで、[**Slackに登録**] ボタンをクリックすると、Slack のチームおよびチャンネルに **WhaTapNotiBot** を登録できます。

Telegram

1. **Telegram**で **@WhaTapApmNotiBot** を検索してください。

   Telegram

2. ボットを選択し、[開始] ボタンをクリックします。
3. チャット画面に `/license` コマンドと、通知を受信したいプロジェクトのアクセスキーを入力してください。

   Telegram

使用可能なコマンド一覧

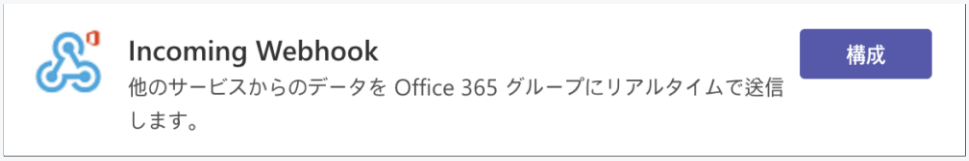
- `/help` : コマンド一覧を表示
- `/license` : 通知を受信するプロジェクトのアクセスキーを入力
- `/remove` : 通知を受信しないプロジェクトコードを入力
- `/list` : 現在通知を受信しているプロジェクトコードを表示

Teams

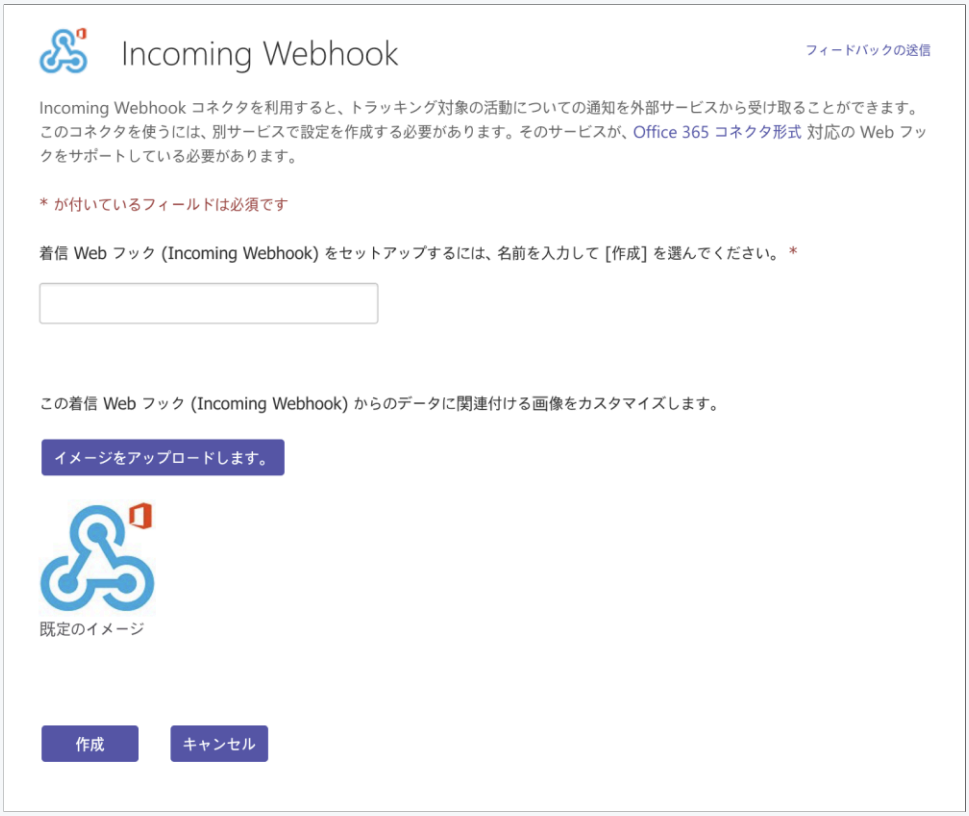
1. 通知を受信したいチャンネルの **コネクタ** に移動します。



2. **Webhook** を追加し、[構成] ボタンをクリックします。



3. Webhook 名を入力し、[作成] ボタンをクリックします。



4. 生成された **Webhook URL** をコピーします。

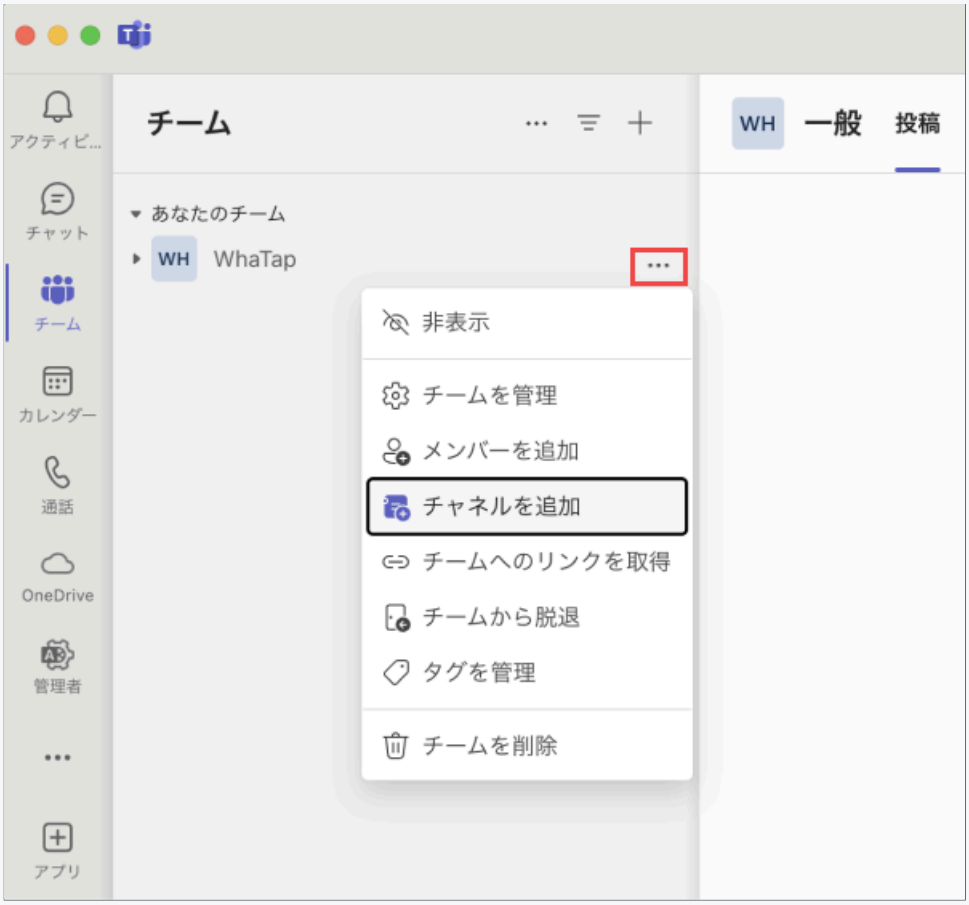


5. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Teams** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
6. [**登録**] ボタンをクリックします。

Teams Workflows

ワークフローを使用するには、**Microsoft Teams** にアクセス可能なアカウントと **ビジネスプランのサブスクリプション**が必要です。

1. **チーム** メニューの **その他の操作 (...)** > **チャネルを追加** をクリックします。

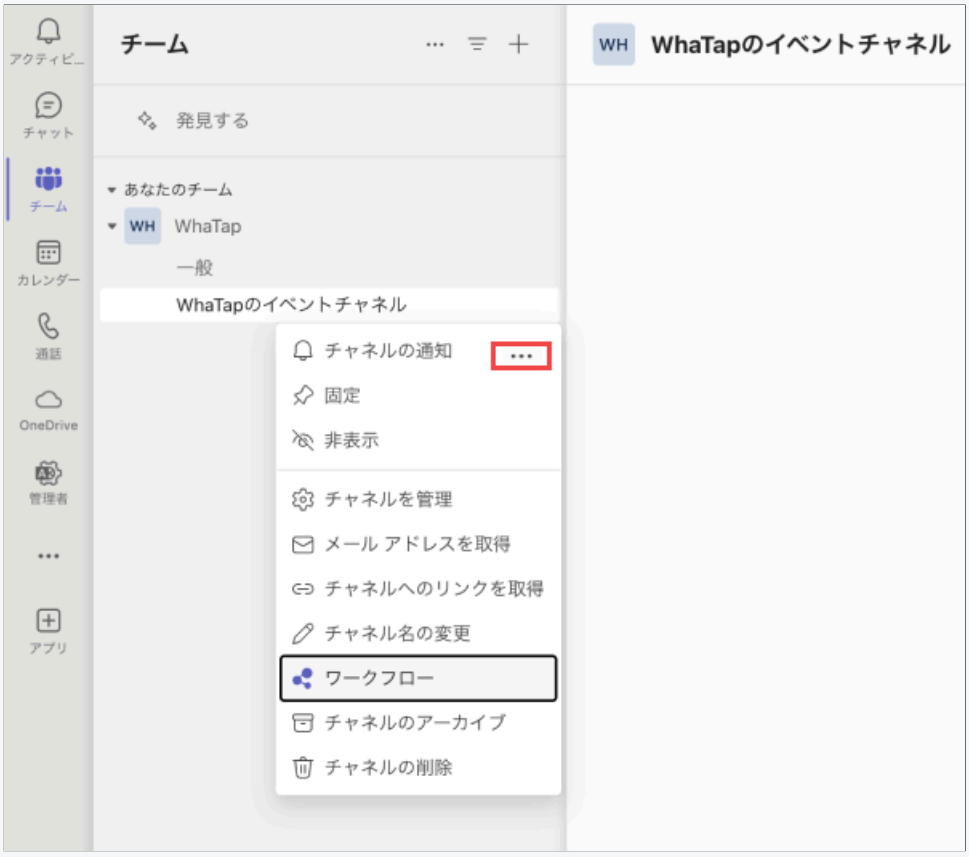


2. **チャネルの作成** 画面で **チャネル名** と **説明** を入力します。**チャネルの種類** は **標準** を選択し、[作成] ボタンをクリックします。

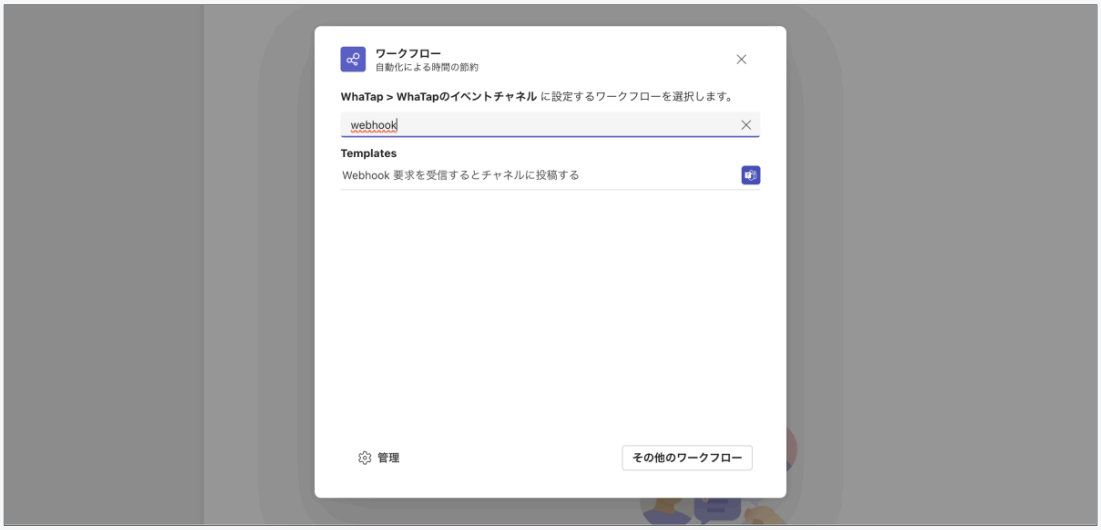


> チャネルの種類を **共有** にするとワークフローは使用できません。**プライベート** を選択した場合、ワークフローは追加できますが通知は受信できません。

3. 作成したチャネルの **その他の操作 (...)** > **ワークフロー** をクリックします。



4. **ワークフロー** ダイアログの一覧から **Webhook 要求を受信したらチャネルに投稿** を選択し、[ワークフローを追加] ボタンをクリックします。



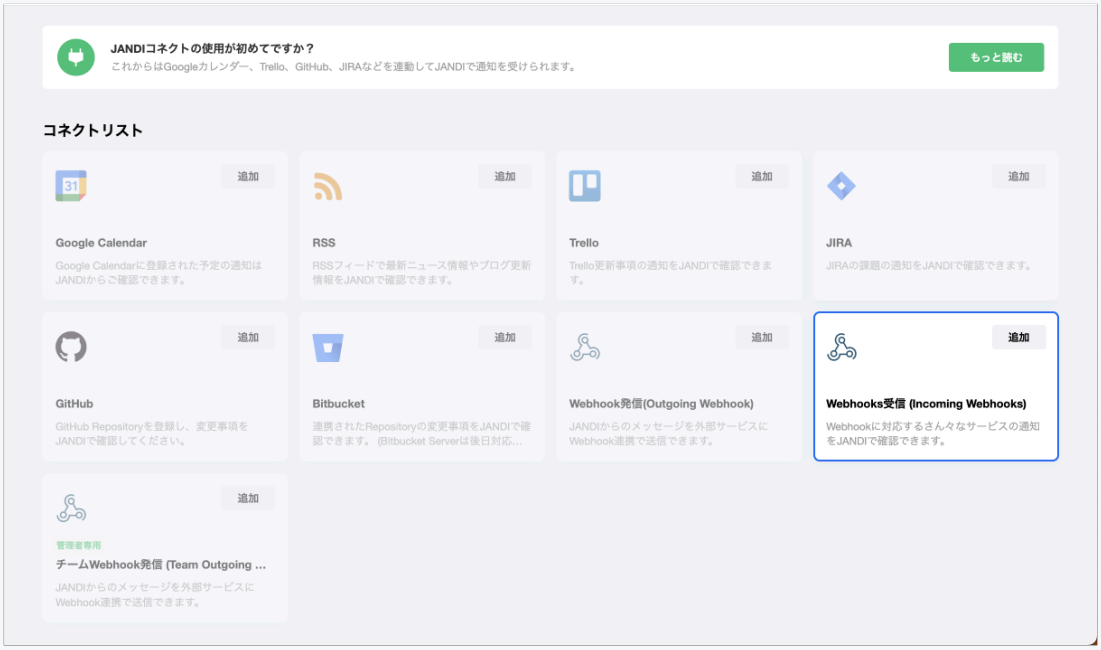
5. **名前** と **接続** の状態を確認し、[次へ] ボタンをクリックします。
6. **詳細** を確認後、[ワークフローを追加] ボタンをクリックします。
7. ワークフローが追加されました。生成された URL をコピーします。



8. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Teams Workflows** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
9. [**登録**] ボタンをクリックします。

Jandi

1. 通知を受信する **トピック** または **チャット** で、**Jandi Connect 設定** に移動します。
2. **コネクト一覧** から **Webhook 受信（Incoming Webhook）** を選択し、[連携する] ボタンをクリックします。



3. [Webhook 受信を追加] ボタンをクリックします。
4. **Webhook URL 登録** 項目に表示される Webhook URL をコピーします。



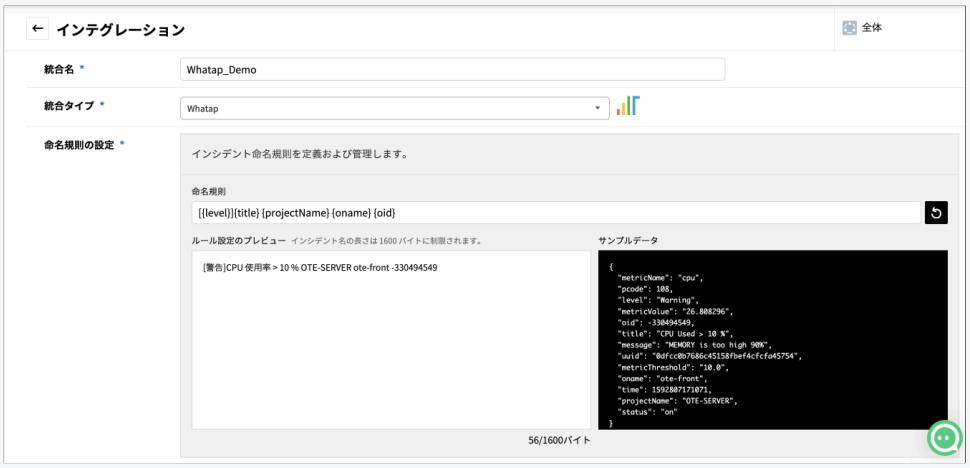
5. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Jandi** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
6. [**登録**] ボタンをクリックします。

AlertNow

1. **AlertNow** サービスにアクセスし、**設定** > **Integration** メニューに移動します。



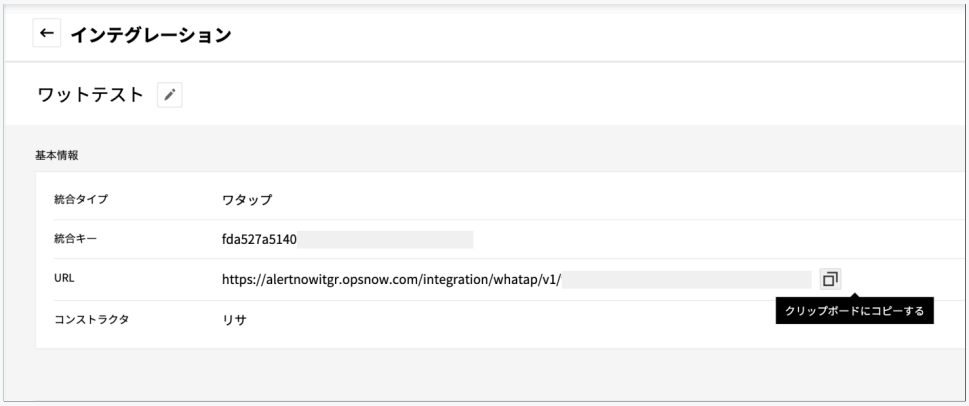
2. **Integration** 画面で [Integration 作成] ボタンをクリックします。
3. Integration タイプの中から **WhaTap** を選択します。
4. **Integration 作成** 画面で、**名前** および **命名規則** を設定します。



5. [確認] ボタンをクリックすると、Integration が作成されます。
6. 作成された Integration の一覧を確認し、該当する Integration を選択します。



7. 詳細画面で、URL 右側のコピーアイコンをクリックして URL をコピーします。



8. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**AlertNow** タブ下部の **Channel Name** と **Webhook URL** に値を入力します。
9. [**登録**] ボタンをクリックします。

Webhook

1. 通知を受信する Webhook を登録します。登録した URL に対して、通知の `title` と `text` が **POST** メソッドで送信されます。

```
{
  "title":"Alert Title",
  "text": "Sample warning message"
}
```

2. **Webhook Name** と **Webhook URL** の項目を入力します。
3. [**登録**] ボタンをクリックします。

Webhook JSON

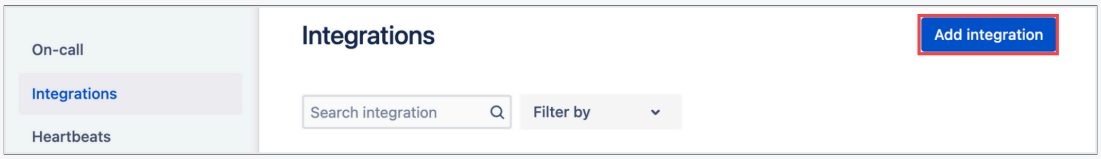
1. 通知を受信する Webhook を登録します。登録した URL に対して、通知データが **JSON 形式**で **POST** メソッドにより送信されます。JSON に含めることができる置換変数については、**使用可能な変数**を参照してください。
   - `${something}` は WhaTap が提供する置換値を表します。対応する全一覧と意味については、**使用可能な変数**を参照してください。
   - イベント状態（ `status` ）は、**イベント発生**（ `on` ）と **イベント解消**（ `off` ）に区別されます。
   - `level` は `None` 、 `Info` 、 `Warning` 、 `Critical` に区別されます。

```
{
  "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}"
}
```

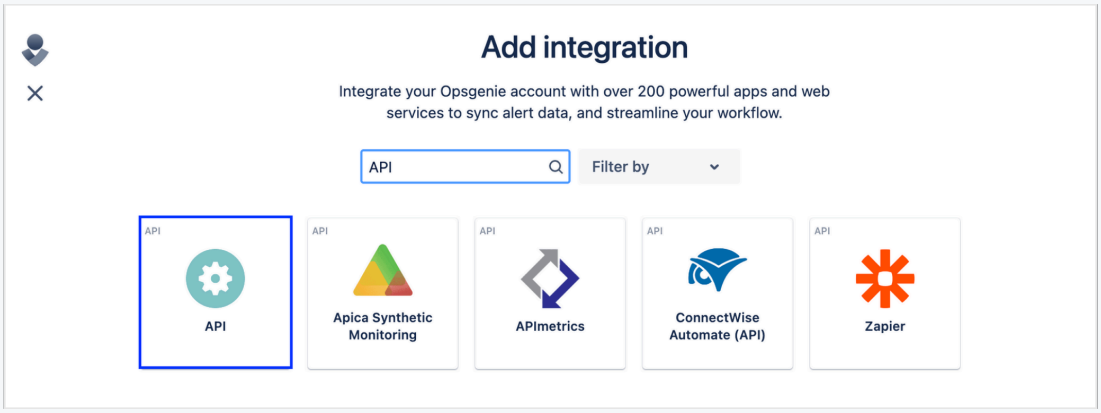
2. **Webhook Name** と **Webhook URL** の項目を入力してください。
3. ヘッダー値を追加するには、**ユーザーヘッダー値の追加** の **+ 追加** をクリックし、key と value を入力してください。
4. [**登録**] ボタンをクリックしてください。

Opsgenie

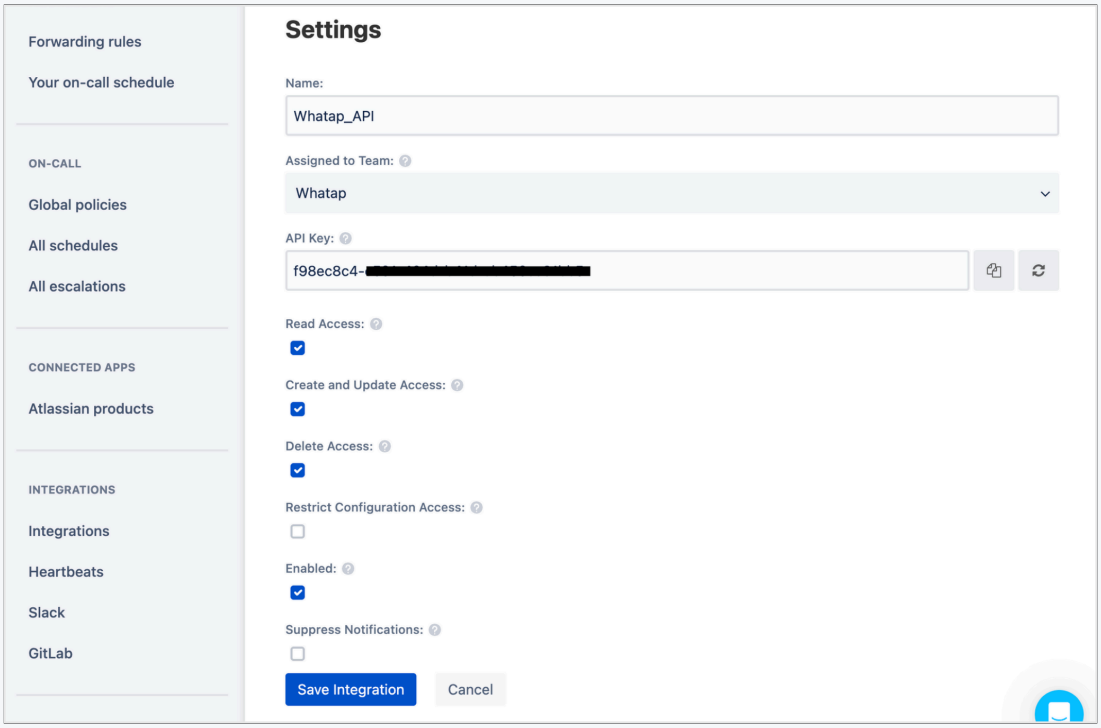
1. **Teams** メニューで、アラートを受信するチームを選択してください。
2. **Integration** メニューに移動し、[Add Integration] ボタンをクリックしてください。



3. インテグレーション一覧で **API** を検索し、選択してください。



4. API 名を設定し、チームを割り当て、各権限を設定した後、[Save Integration] ボタンをクリックしてください。



5. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Opsgenie** タブ下部の **Name** と **API Key** を入力してください。
6. [**登録**] ボタンをクリックしてください。

PagerDuty

1. PagerDuty アカウントを登録し、ログインしてください。
2. PagerDuty で **Services** > **Service Directory** メニューに移動し、[+ New Service] ボタンをクリックしてください。

   PagerDuty
3. サービス情報（**Name**）を入力し、[Next] ボタンをクリックしてください。

   PagerDuty
4. **Escalation Policy** を設定し、[Next] ボタンをクリックしてください。

PagerDuty

5. **Alert Grouping** と **Transient Alerts** を設定し、[Next] ボタンをクリックしてください。

   PagerDuty

6. **Integrations** で **Event API V2** を選択し、[Create Service] ボタンをクリックしてください。

   PagerDuty

7. 作成されたサービス画面で **Integration Name** と **Integration Key** の値をコピーしてください。

   PagerDuty

8. WhaTap モニタリングサービスの **サードパーティプラグインの追加** 画面に戻り、**Integration Name** と **Integration Key** を入力してください。
9. [登録] ボタンをクリックしてください。

> メッセージを受信できない、またはプラグインが登録されない場合は、**Integration Name** または **Integration Key** の値が正しいか確認してください。

LINE

1. LINE アプリをインストールしてください。
2. 以下の QR コードをスキャンして、**WhaTapLabs** 公式チャンネルを友だち追加してください。

   LINE 3rd パーティプラグイン追加

3. 次のコマンドを使用して、プロジェクトを LINE と連携し、通知を設定してください。

   コマンド

```
/help # 使用可能なコマンド一覧を表示
/list # 連携されているプロジェクト一覧を表示
/license [アクセスキー] # 新しいプロジェクト連携を追加
```

```
/remove [プロジェクトコード] # プロジェクト連携を解除
/language [ko|ja|en] # 通知言語を設定（韓国語／日本語／英語）
```

- プロジェクトのアクセスキーは **管理 > プロジェクト管理** で確認できます。
- LINE 内で `/language` コマンドを使用した場合、その設定は **該当する LINE チャットルーム内のみ** に適用されます。他の 3rd パーティプラグインやプロジェクトの通知言語は変更されません。

使用例

```
/license XXXXXXXXXX # プロジェクトを連携
/language ko # 言語を韓国語に設定
/list # 連携済みプロジェクトを確認
```

- **プロジェクトアクセスキー**: WhaTap プロジェクト内の **管理 > プロジェクト管理** で確認できます。
- **言語設定**: `/language` コマンドは LINE チャットルーム内でのみ有効で、他の通知チャネルの言語は変更されません。

ilert

1. ilert コンソールの **Alert sources** 画面右上にある [+ Create alert source] をクリックします。
2. **Integration type** ステップ下部の **Filter integrations** 検索欄に `WhaTap` を入力して検索します。
3. 検索結果から **WhaTap** タイルを選択し、[Next] ボタンをクリックします。
4. アラートソース名を入力し、チームを指定した後、[Next] ボタンをクリックします。
   - エスカレーションポリシーは、新規作成または既存のものを選択できます。
5. アラートのグループ化方式を選択し、[Continue setup] ボタンをクリックします。
   - 最初に選択した **Do not group alerts** は後から変更可能です。
6. 次の画面で **アラートテンプレートや優先度などの追加設定** を行うことができます。ここでは [

Finish setup] ボタンをクリックして設定を完了します。

7. 設定が完了すると、完了画面で **API キー** または **Webhook URL** が生成されます。
8. **Webhook JSON** タブで生成された **Webhook URL** をコピーし、下部の **Webhook URL** 入力欄に貼り付けます。
9. [登録] ボタンをクリックします。

大量アラート発生の防止

短時間に大量のアラートが発生した場合、指定した時間だけアラート通知を一時的に停止します。たとえば、5 分間に 20 件のイベントが発生すると、その後 5 分間は警告アラートが送信されません。設定した **停止時間** が経過すると、大量アラート防止機能は自動的に解除されます。

1. 次の画面に移動してください。**アラート通知** > **通知設定** > **大量アラート通知発生防止** セクション
2. トグルボタンをクリックして **有効化** または **無効化** します。
   - **有効**: 大量アラートを検知した場合、一時的に通知を停止
   - **無効**: アラート通知を再開
3. **検知時間, 検知回数, 停止時間** を設定してください。
   - **検知時間** 内に **検知回数** 以上のイベントが発生すると、**停止時間** の間、警告アラートが停止されます。

大量アラートによってブロックされた場合、ブロック案内ボックスが表示されます。大量アラート防止機能をすぐに解除するには、ボックス内の **大量アラート発生中止** ボタンをクリックしてください。

イベント通知フォーマット

共通アラートテンプレート

共通テンプレートは、ほとんどのケースで適用できます。さまざまな環境において、同一のフォーマットでアラートを受信できます。

- プロダクトタイプ: アプリケーション、データベース、Kubernetes
- イベントタイプ: アプリケーションアラート、データベースアラート、メトリクスアラート
- イベントチャネル: SMS、Mobile、3rd-party プラグイン、プラグイン

イベントタイトル

Event title format

```
[Level][Platform][ProjectName][ApplicationName][EventTitle]
```

Event title example

```
[Info][JAVA][アプリケーションプロジェクト][TC-0-1-8081][CRITICAL_HIGH_MEMORY]
```

ApplicationName が設定されていない場合、イベントタイトルから省略されます。

- Platform は以下のいずれかで表示されます。
  - 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

イベントメッセージ

イベントメッセージに含めることができる情報です。`optional` が `false` の場合、常にメッセージに含まれます。`optional` が `true` の場合、該当データが確認できるときのみ表示されます。

イベントメッセージ例

```
Project Name : アプリケーションプロジェクト
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
```

表 | イベントメッセージ構成要素

En	Ko	対応するアラートタイプ	説明
Project Name	プロジェクト名	全て	-
Project Code	プロジェクトコード	全て	-
Agent Name	エージェント名	全て（optional）	oname
Message	イベントメッセージ	全て	-
Alert Type	イベント種別	全て	下記の AlertType 表を参照
Event Time	イベント発生時間	全て	`2022-04-13 10:40:49 +0900` の `+0900` は GMT オフセット
Event Off Time	イベント解除時間	全て（optional）	`2022-04-13 10:40:49 +0900` の `+0900` は GMT オフセット
Metric Name	メトリクス名	全て（optional）	イベント条件判定に使用されるメトリクス名
Metric Value	メトリクス値	全て（optional）	メトリクス値が閾値を超えた場合に含まれる
Metric Threshold	メトリクス閾値	全て（optional）	メトリクス条件の閾値

En	Ko	対応するアラートタイプ	説明
Stateful	復旧イベント通知	全て（optional）	復旧通知が有効な場合 `true`、それ以外は `false`
Event Rule	イベント発生条件	メトリクスアラート	-
Target Filter	イベント対象選択	メトリクスアラート	特定対象から収集されたメトリクスのみ評価
Repeat Count	イベント繰り返し回数	メトリクスアラート	繰り返し時間内に指定回数条件を満たすと発生
Repeat Duration	イベント繰り返し時間	メトリクスアラート	繰り返し回数を評価する時間
Receiver	受信者	メトリクスアラート	-
Query	MXQL クエリ	複合メトリクスアラート	-
Rule	イベント発生条件	複合メトリクスアラート	-
Query Period	クエリ期間	複合メトリクスアラート	-
Query Interval	クエリ間隔	複合メトリクスアラート	-
Silent Time	サイレント時間	複合メトリクスアラート	-
Query	URL	Exception アラート	例外を発生させたリクエストの URL
TXID	トランザクション ID	Exception アラート	-

En	Ko	対応するアラートタイプ	説明
Class	エラークラス名	Exception アラート	-
Log Message	ログメッセージ	サーバー－ファイルログアラート	-
Log File	ログファイルパス	サーバー－ファイルログアラート	-
IP	IP	サーバーアラート全体	-
CPU	CPU	サーバーアラート全体	イベント発生時のスナップショット
CPU_load1	CPU_load1	サーバーアラート全体	イベント発生時のスナップショット
CPU_loadPerCore	CPU_loadPerCore	サーバーアラート全体	イベント発生時のスナップショット
Memory	メモリ	サーバーアラート全体	イベント発生時のスナップショット
Swap	スワップ	サーバーアラート全体	イベント発生時のスナップショット
Disk Name | Used Percent | Free Size | IO Percent	ディスクパフォーマンス	サーバーアラート全体	イベント発生時のスナップショット
Name | Bps | Pps	トラフィックパフォーマンス	サーバーアラート全体	イベント発生時のスナップショット
Message | Time | Name	処理履歴メッセージ	サーバーアラート全体	-

このセクションでは、該当するイベントで提供可能な情報を最大限に表示します。

- **AlertType** は以下のいずれかの値で表示されます。

AlertType	説明
APPLICATION_CPU	アプリケーション CPU アラート
APPLICATION_MEMORY	アプリケーション メモリアラート
APPLICATION_DISK	アプリケーション ディスクアラート
APPLICATION_ACTIVE_TRANSACTION	アプリケーション アクティブTXアラート
APPLICATION_ERROR_TRANSACTION	アプリケーション エラートランザクションアラート
APPLICATION_SLOW_TRANSACTION	アプリケーション トランザクション応答時間アラート
METRICS	メトリクスアラート
COMPOSITE_METRICS	複合メトリクスアラート
ANOMALY	異常値アラート
LOG_REALTIME	リアルタイムログアラート
COMPOSITE_LOG	複合ログアラート
SERVER_REBOOT	サーバー 再起動アラート
SERVER_NO_DATA	サーバー データ未受信アラート
SERVER_PORT	サーバー ポートアラート
SERVER_NETWORK_IOPS	サーバー ネットワーク IOPS アラート
SERVER_NETWORK_BPS	サーバー ネットワーク BPS アラート

AlertType	説明
SERVER_DISK_IO	サーバー ディスク I/O アラート
SERVER_DISK_QUOTA	サーバー ディスク使用量アラート
SERVER_DISK_INODE	サーバー inode アラート
SERVER_CPU	サーバー CPU アラート
SERVER_MEMORY	サーバー メモリアラート
SERVER_CPU_STEAL	サーバー CPU steal アラート
SERVER_MEMORY_SWAP	サーバー スワップメモリアラート
SERVER_LOG_FILE	サーバー ログファイルアラート
SERVER_WINDOW_EVENT	サーバー Windows イベントアラート
SERVER_OFF	サーバー アラート OFF 通知
SERVER_ACKNOWLEDGE	サーバー 処理確認通知
SERVER_PROCESS_COUNT	サーバー プロセス数アラート
SERVER_PROCESS_CPU	サーバー プロセス CPU アラート
SERVER_PROCESS_MEMORY	サーバー プロセス メモリアラート
SERVER_PROCESS_OFF	サーバー プロセスアラート OFF 通知
AGENT_ACTIVE	エージェント 有効化アラート
AGENT_INACTIVE	エージェント 無効化アラート

AlertType	説明
AGENT_REACTIVATED	エージェント 再有効化アラート
URL	URL アラート
TOO_MANY_EVENT	イベント多発アラート
CLOUD_WATCH	CloudWatch アラート
EXCEPTION	例外アラート

イベント履歴

ホーム > プロジェクト選択 > **アラート通知** > **イベント履歴**

イベント履歴 では、プロジェクトで発生したさまざまなイベントを確認し、詳細情報を分析できます。Elasticsearch DSL クエリをサポートしており、指定した条件に一致するイベントを迅速かつ正確に検索できます。

イベント履歴の確認

検索結果で **イベントタイトルをクリック**すると、そのイベントが発生した時間帯の詳細データを分析できる画面に移動します。これにより、イベント発生前後の状況を総合的に把握できます。

1. **アラート通知** > **イベント履歴** に移動してください。
2. **イベント履歴** 画面の **時間選択** で、イベント履歴を確認する期間を設定してください。
3. フィルター入力欄にクエリを入力し、[虫眼鏡] ボタンをクリックしてください。
    - クエリを入力しない場合、すべてのイベントが表示されます。

カラム設定

表示するカラムや非表示にするカラムを設定できます。

1. **イベント履歴** 画面の左上にある [カラム設定] ボタンをクリックしてください。
2. **カラム設定** 画面で、表示または非表示にするカラムを選択してください。
    - **表示項目** でカラムの順序を変更できます。
    - **カラム名検索** 欄で目的のカラムを検索できます。

カラム項目	説明
イベント名	イベントのタイトル
メッセージ	イベントのメッセージまたはスナップショット情報

カラム項目	説明
イベント状態	イベントの現在の状態
エージェント名	イベントが発生した対象エージェントの名前（Oname） - 特定のエージェントに紐づかないイベントの場合は空欄で表示されます
エージェントグループ	イベントが発生した対象エージェントの種類（OkindName） - 特定のエージェントに紐づかないイベントの場合は空欄で表示されます
エージェントノード	イベントが発生した対象エージェントのサーバー（OnodeName） - 特定のエージェントに紐づかないイベントの場合は空欄で表示されます
イベント発生時刻	イベントが発生した時刻
イベント解消時刻	イベントが解消された時刻 - 解消状態を持たないイベントの場合は `-` と表示されます

3. 設定が完了したら、[確認] ボタンをクリックしてカラム設定を反映してください。

CSV ダウンロード

検索結果を CSV ファイル形式でダウンロードできます。

1. **アラート通知** > **イベント履歴** で [CSV] ボタンをクリックしてください。
2. 現在のフィルター条件に一致するイベント一覧が CSV ファイルとして保存されます。

進行中のイベントのみ表示

現在進行中の状態のイベントのみを表示できます。

1. **アラート通知** > **イベント履歴** で [進行中のイベントのみ表示] ボタンをクリックしてください。
2. フィルタークエリに次の条件が自動的に追加され、検索が実行されます。

```
Stateful: true and Status : "ON"
```

イベント検索

- **正確な検索**: 必要なイベントだけを取得できるよう、条件を正確に指定
- **複合条件**: 複数のフィールドを同時に検索し、複雑な条件も一度に処理
- **柔軟なパターン**: ワイルドカードや部分一致など、さまざまな検索方式をサポート
- **高速パフォーマンス**: インデックス化されたデータを活用した高速検索

検索可能なフィールド

フィールド名および値は大文字と小文字を区別しません。

- **フィールド名タグ**: フィールドがサポートされるイベントタイプ
- **フィールドタイプ**: フィールドのデータ型
  - `#`: Number
  - `T`: String
  - : Boolean
  - : Date

イベント情報フィールド

フィールド名	タイプ	説明	例
Title	String	イベントタイトル	"Database Connection Error"
OffTitle	String	イベント解消タイトル	"RECOVERED: Database Connection Error"
Message	String	イベントメッセージ	"Connection timeout occurred"
OffMessage	String	イベント解消メッセージ	"RECOVERED: Connection timeout occurred"

フィールド名	タイプ	説明	例
Level	String	現在のイベントレベル	`Critical`, `Warning`, `Info`
OriginLevel	String	元のイベントレベル	`Critical`, `Warning`, `Info`
Status	String	イベント状態	`ON`, `OFF`, `CANCEL`, `ACKNOWLEDGE`, `MAINTENANCE`, `DISABLED`
ActCount	Number	発生回数	2
MetricName	String	指標名	"memory"
MetricValue	String	指標値	"85.5"
OffValue	String	解消時の値	"72.8"
MetricThreshold	String	閾値	"80"
alertType	String	イベント種別	"BASIC", "METRICS", "TRANSACTION" など
alertId	String	イベントルールの一意識別子	"zf3uojer0fv4v7"

イベントタイプ別対応フィールド

> ⓘ 特定のイベントタイプでのみサポートされるフィールドです。該当しないイベントタイプの場合、値が存在しないことがあります。

フィールド名	タイプ	イベント種別	説明	例
eventRule	String	基本、メト	イベント発生	"memory ≥ 80"

フィールド名	タイプ	イベント種別	説明	例
		リクス	ルール	
field	String	リアルタイムログ	ログ検索キー	"content"
keyword	String	リアルタイムログ	ログ検索値	"Error"
logCategory	String	リアルタイムログ	ログカテゴリ	"AppLog"
logContent	String	リアルタイムログ	ログ内容	"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."

対象フィールド

フィールド名	タイプ	説明	例
Oid	Number	エージェントの一意識別子	-98765432
Oname	String	エージェント名	"web-server-01"
Okind	Number	エージェント種別の一意識別子	867318026

フィールド名	タイプ	説明	例
OkindName	String	エージェント種別名	"web-server"
Onode	Number	エージェントノードの一意識別子	334634079
OnodeName	String	エージェントノード名	"production-node-1"

イベント固有値フィールド

フィールド名	タイプ	説明	例
Id	Number	イベントの一意識別子	5768121
UUID	String	イベント固有識別子	"550e8400-e29b-41d4-a716"

状態／フラグフィールド

フィールド名	タイプ	説明	例
Stateful	Boolean	状態ベースイベントかどうか	true, false
Disabled	Boolean	無効化されたイベントかどうか	true, false
Escalation	Boolean	エスカレーション有無	true, false
SystemEvent	Boolean	システムイベントかどうか	true, false

検索タイプ別クエリ構文

> ⓘ 検索の基本構文については、WhaTap ログ検索クエリ構文 ドキュメントを参照してください。

1. キーワード検索

「Database Connection」という文字列を含むイベント

```
title: "Database Connection"
```

2. 複数値検索

指定した複数の値のうち、いずれかを含むイベントを検索します。

```
title: Database Connection
```

現在のイベントレベルが Info または Warning のイベント

```
level: info warning
```

3. パターン検索

ワイルドカードを使用してパターン検索ができます。

名前パターンで特定のエージェントで発生したイベントを検索

```
oname: web-*-prod
```

タイトルが Connection で終わるイベントを検索

```
title: *Connection
```

4. 複合条件検索

複数の条件を組み合わせて、より正確に検索できます。

レベルが Critical で、タイトルに Database を含むイベント

```
level: critical and title: Database
```

特定のエージェントにおける Warning または Critical のイベント

```
oname: web-server-01 and level: Warning Critical
```

5. OR 条件検索

複数の条件のうち、**いずれか一つでも満たす**イベントを検索します。

タイトルまたはメッセージに特定のキーワードを含むイベント

```
title: Connection or message: Connection
```

6. 除外条件

特定の条件を除外して検索できます。

Info レベルを除いたすべてのイベント

```
not level: info
```

リアルタイムアラート

リアルタイムアラート 機能は、**イベント設定** で設定したイベントが発生した際に、ポップアップ形式で表示されます。イベント発生と同時に通知を受け取ることで、問題を迅速に把握し、対応することができます。また、すでに発生した通知や見逃した通知の履歴を確認することで、リアルタイム対応が難しかった状況を把握し、解決に役立てることができます。

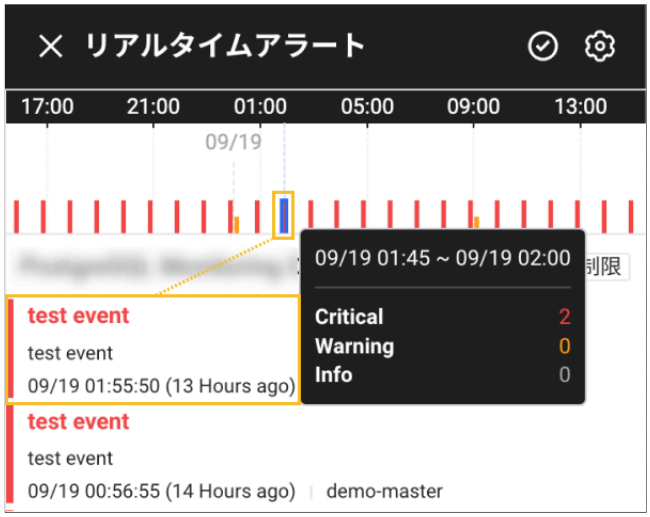
リアルタイムアラートを確認する

リアルタイムアラートは、画面右上にポップアップとして表示されます。通知が発生した時刻、タイトル、メッセージを確認できます。

- 個別の通知を閉じるには、メッセージ右上の × ボタンをクリックしてください。
- 画面に表示されているすべてのポップアップ通知を閉じるには、**すべての通知を閉じる** をクリックしてください。

通知履歴を確認する

1. プロジェクトを選択して表示される画面の最上部にある 🔔 をクリックしてください。
2. **リアルタイムアラート** ウィンドウが表示されます。
    - 現在のプロジェクトについて、直近24時間以内の通知を最大100件まで確認できます。
    - 個別のイベント項目を選択すると、関連するメニューに移動します。
    - **リアルタイムアラート** ウィンドウ内の時間別チャートで任意の時間帯を選択すると、該当するイベント項目に移動します。



- すべての通知履歴を確認するには、**全体の通知履歴** ボタンをクリックしてください。**アラート通知** > **イベント履歴** メニューに移動します。

> ⓘ **リアルタイムアラート** の一覧が空の場合、**全体の通知履歴** ボタンが表示されます。**受信タグ未設定の通知を受け取る** オプションを無効にしてください。

イベントを一括で解消する

リアルタイムアラート の一覧から、進行中のすべてのイベントを一括で解消できます。
イベント解消が完了するまでに、約30秒〜1分程度かかる場合があります。

- **イベント一括処理** 機能を使用すると、現在進行中のイベントに対する解消通知が大量に発生する可能性があります。
- **イベント一括処理** 機能を使用した後でも、イベントルールが有効であり、データが引き続きしきい値を超えている場合、イベントは再度発生することがあります。
- イベント解消通知は、**アラート通知** > **イベント設定** メニューでイベントを編集し、**イベントが解消されたときに追加通知** または **解消通知** オプションを無効にすることで停止できます。

使用方法:

1. プロジェクトを選択して表示される画面上部の をクリックします。
2. **リアルタイムアラート** ウィンドウが表示されたら、右上の をクリックします。
3. リアルタイムアラートリストの進行中イベントを一括で解消します。 というメッセージが表示されたら、**解消** ボタンをクリックします。

リアルタイムアラートを設定する

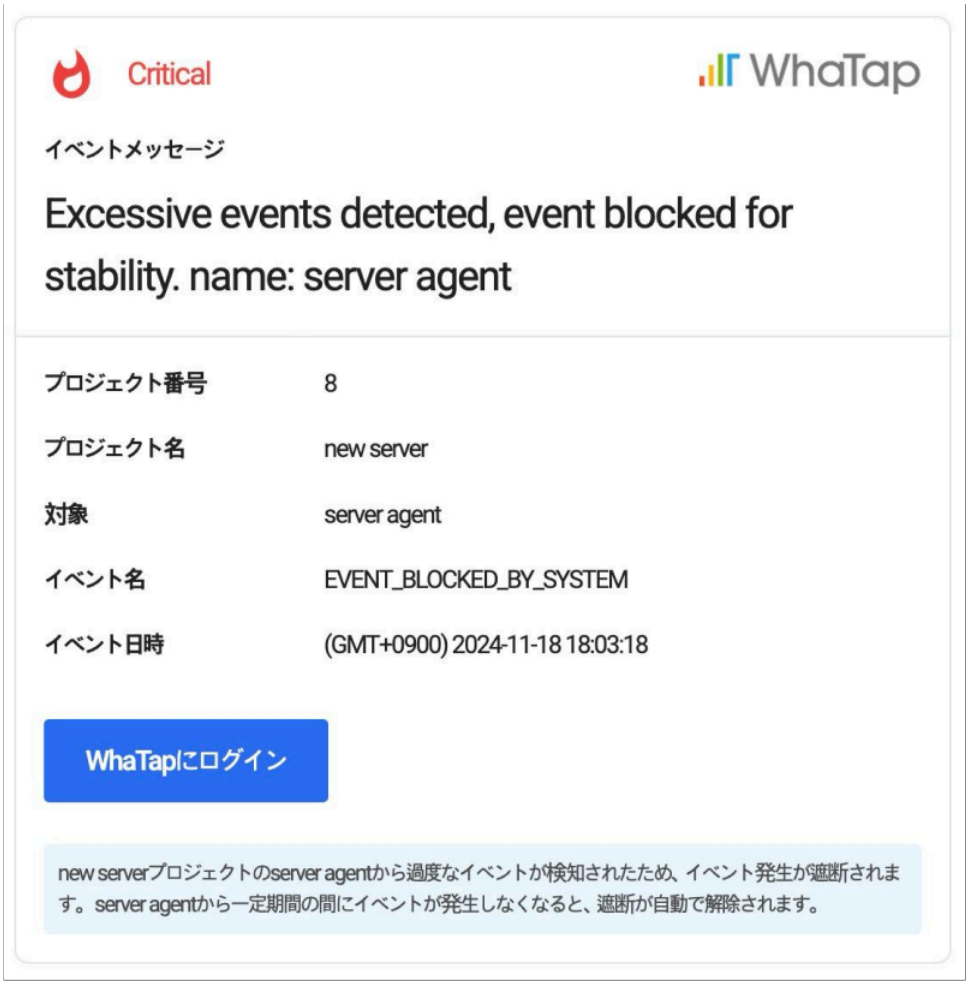
1. プロジェクトを選択して表示される画面上部の をクリックします。
2. **リアルタイムアラート** ウィンドウが表示されたら、右上の をクリックします。
3. リアルタイムアラート設定画面で各項目を設定します。
   - **ポップアップ通知**: **リアルタイムアラート** ウィンドウが閉じている状態でも、ポップアップ形式で通知を表示します。
   - **解消済みイベントを表示**: **リアルタイムアラート** 一覧に解消済みイベントも含めて表示します。
   - **通知音の試聴**: ブラウザで再生される通知音を事前に確認できます。Critical または Warning の横にある をクリックしてください。音が再生されない場合は、ブラウザの設定をご確認ください。

システムイベント

WhaTap モニタリングは、プラットフォーム全体で自動的に検知・生成される共通イベントを提供します。本ドキュメントでは、システムによって発生するイベントの種類、発生条件、および対応方法について説明します。

EVENT_BLOCKED_BY_SYSTEM

プロジェクト内で短時間に過剰な数のイベントが発生した場合、WhaTap システムがこれを検知し、イベントの発生を一時的にブロックします。イベントがブロックされると、システムは次のメッセージをユーザーに通知します。このイベントが発生した場合は、イベント設定内容を確認し、必要に応じて調整してください。



ブロック基準

イベントは次の基準に基づいてブロックされます。

1. **エージェント単位のブロック**

   1つのエージェントで **1分以内に10件** のイベントが発生した場合、そのエージェントから発生する以降のイベントはブロックされます。

2. **複数エージェントによる全体ブロック**

   条件 **1** によりイベントがブロックされたエージェントが **1分以内に3件以上** 発生した場合、プロジェクト全体のイベントがブロックされます。

3. **プロジェクト単位のブロック**

   1つのプロジェクトで **1分以内に100件** のイベントが発生した場合、プロジェクト全体のイベントがブロックされます。

ブロック解除方法

システムによってイベントがブロックされた場合、対象のエージェントまたはプロジェクトで **1分間** イベントが発生しない状態を維持する必要があります。

- **アラート通知 > イベント設定** メニューから設定されているすべてのイベントを無効化してください。**イベント設定を見直し、過剰なイベントが発生しないように閾値を調整する必要があります。**
- イベントを設定する前に、**シミュレーション** 機能を活用してイベント発生回数を事前に予測できます。

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.