データフローコントローラ

DXシリーズ

SpeeDBee Synapse

ユーザーズマニュアル

形DX100-0010

CPUユニット



SBSB-540A

おことわり

- 本マニュアルの内容の一部または全部を無断で複写、複製、転載することを禁じます。
- 本マニュアルの内容に関しては、改良のため予告なしに仕様などを変更することがあります。
  あらかじめご了承ください。
- 本マニュアルの内容に関しては、万全を期しておりますが、万一ご不審な点や誤りなど、お気づきのことがありましたら、お手数ですが、弊社支店または営業所までご連絡ください。
  その際、巻末記載のMan.No.（マニュアルNo.）も併せてお知らせください。

商標

- Microsoft、Windows、Excel、Visual Basic、Microsoft Edge は、米国Microsoft Corporationの米国およびその他の国における登録商標または商標です。
- ODVA、CIP、CompoNet、DeviceNet、EtherNet/IP はODVA の商標です。
- SpeeDBee Synapseは株式会社ソルティスターの商標です。
- GrafanaはGrafana Labsの商標です。

その他、本文中に掲載している会社名および製品名は、それぞれ各社の商標または登録商標です。

著作権

- スクリーンショットはマイクロソフトの許諾を得て使用しています。
- 本製品にはサードパーティ製のソフトウェアが組み込まれています。このソフトウェアに関連するライセンスと著作権については、https://www.fa.omron.co.jp/product/tool/dx-info/index.html をご覧ください。

Table of Contents

1 目次

2 はじめに

3 SpeeDBee Synapseの概要

4 導入フロー

- 4.1 導入フロー
- 4.2 インストール（Linux）
- 4.3 インストール（Windows）
- 4.4 初期設定
- 4.5 コンポーネントの設定
- 4.6 収集サービスの実行

5 各種機能説明

- 5.1 各種機能説明
- 5.2 エンジン機能の全体像
- 5.3 フローエディタ
- 5.4 コレクタ
- 5.5 シリアライザ
- 5.6 エミッタ
- 5.7 ロジック（イベント）
- 5.8 ロジック（リアルタイム分析/データ操作）
- 5.9 アクション
- 5.10 開発用コンポーネント
- 5.11 システムコンポーネント
- 5.12 コンポーネント共通の設定項目
- 5.13 ステータスタイムライン
- 5.14 各種ログ
- 5.15 DBクエリ（SQL）
- 5.16 演算式・条件式
- 5.17 設定情報の管理(プロジェクト)

6 応用的な利用方法

6.1 応用的な利用方法
6.2 カスタムコンポーネントの作成・利用
6.2.1 カスタムコンポーネントの作成・利用
6.2.2 カスタムコンポーネントの概要
6.2.3 Pythonカスタムコンポーネント作成
6.2.4 C言語カスタムコンポーネント作成
6.2.5 カスタムUI作成
6.2.6 カスタムコンポーネント開発環境(SCCDE)
6.3 制御プログラムとの連携（WebAPI）
6.4 Grafanaとの連携
6.5 グローバル変数
7 トラブルシュート
8 セキュア利用ガイド
9 後方互換に関して
10 実行ライセンス
11 i-Trigger（拡張パッケージ）
11.1 i-Trigger（拡張パッケージ）
11.2 概要
11.3 使用できるカメラについて
11.4 インストール
11.5 動画再生用ビューワ
11.6 関連コンポーネント
11.7 チュートリアル
11.8 i-Triggerの記録仕様
12 リリースノート
13 商標
14 製造元への問い合わせ
15 付録
15.1 付録
15.2 エラーコード一覧
15.3 カスタムコンポーネントPython-APIリファレンス
15.4 カスタムコンポーネントC-APIリファレンス
15.5 カスタムUI JS-APIリファレンス

15.6 利用しているオープンソース
15.7 索引
15.8 DBクエリサービスWeb-API仕様

1 SpeeDBee Synapse ドキュメント

- はじめに
- SpeeDBee Synapseの概要
- 導入フロー
- 各種機能説明
- 応用的な利用方法
- トラブルシュート
- セキュア利用ガイド
- 後方互換に関して
- 実行ライセンス
- i-Trigger（拡張パッケージ）
- リリースノート
- 商標
- 製造元への問い合わせ
- 付録

2 はじめに

本マニュアルは、「SpeeDBee Synapse（略称：SBS)」の操作方法について記載したものです。 SpeeDBee Synapseを活用することで、製造現場設備のデータ収集、他システム連携が可能となります。 もちろん、製造現場以外でもIoTシステムを構築することができます。

2.1 対象読者

本マニュアルは、次の方を対象に記述しています。

- IoTシステム、FAシステムを設計される方
- IoTシステム、FAシステムを管理される方
- IoTシステム、FAシステムを実際に操作、運用される方

また、プログラミング言語を用いて、独自の機能を「カスタムコンポーネント」として実装することができます。その場合は、プログラミングに関する知識が必要となります。 詳細についてはマニュアル内で解説します。

2.2 ご承諾事項

SpeeDBee Synapseのインストールされているハードウェア本体のマニュアル、もしくはSpeeDBee Synapseのインストーラー実行時に表示される内容をご確認ください。

2.3 安全上の注意

下記の機能について、対向機器のレジスタや変数、ステータスなどを変更することが可能なため、変更結果により設備の挙動へ影響を与える可能性があります。
これらの機能を使用する際は、設備影響など十分に考慮および検証したうえで、使用してください。

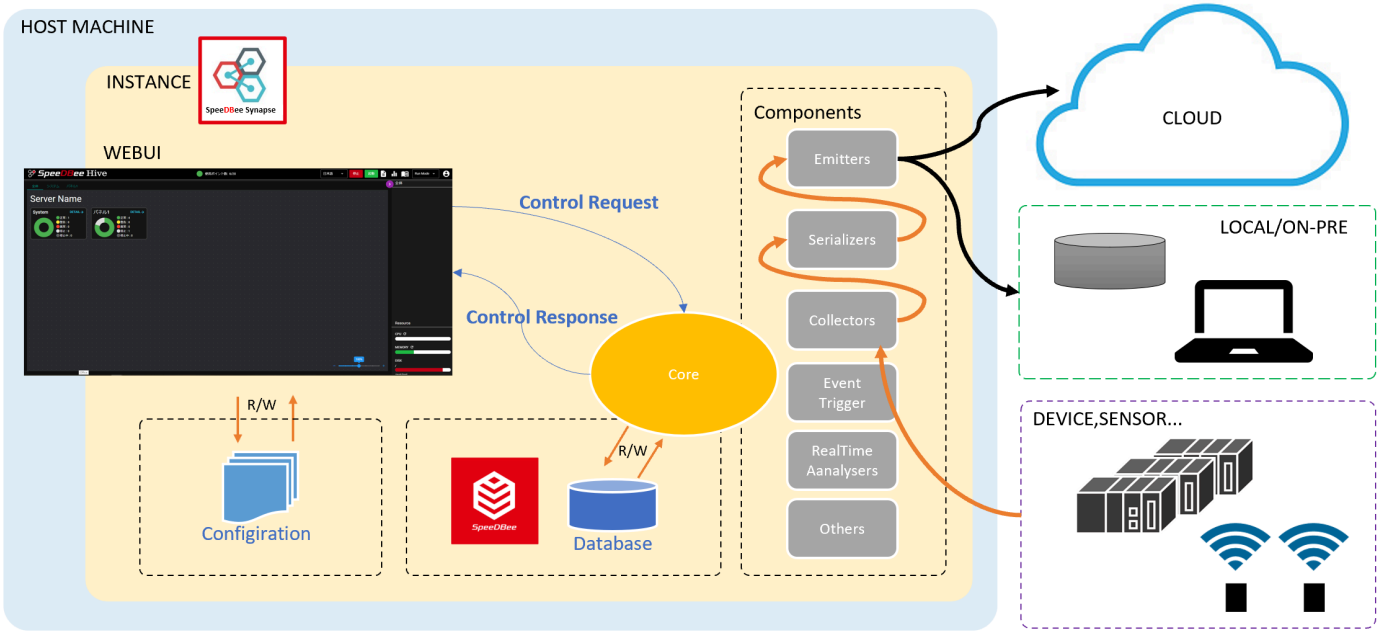
・PLC書込み
・Modbus書込み
・EtherNet/IP書込み

3 SpeeDBee Synapseの概要

本章では、SpeeDBee Synapseの特徴、構成、運用までの流れについて説明します。

3.1 SpeeDBee Synapseの特徴

SpeeDBee Synapseは、FAやIoTを構成する様々な機器（PLC等）よりデータを収集できます。 収集したデータをSpeeDBee Synapseの各種機能を利用する事により、変換や分析、そして、他システムとのデータ連携が可能です。これによりデータ収集からデータ活用までを実現するシステムを構築できます。



3.2 SpeeDBee Synapseの構成

SpeeDBee Synapseは、大きく4つの機能により構成されます。

機能名	説明
コア	収集エンジンの全体機能を制御する中心部です。
コンポーネント	データの収集、加工、分析、送信などを実現するモジュール群です。 コンポーネント間を接続することで様々な機能が実現できます。
時系列データベース	収集したデータを時間に沿って保存するデータベースです。
WEBUI	各種機能の設定、データの可視化をするユーザーインターフェースです。

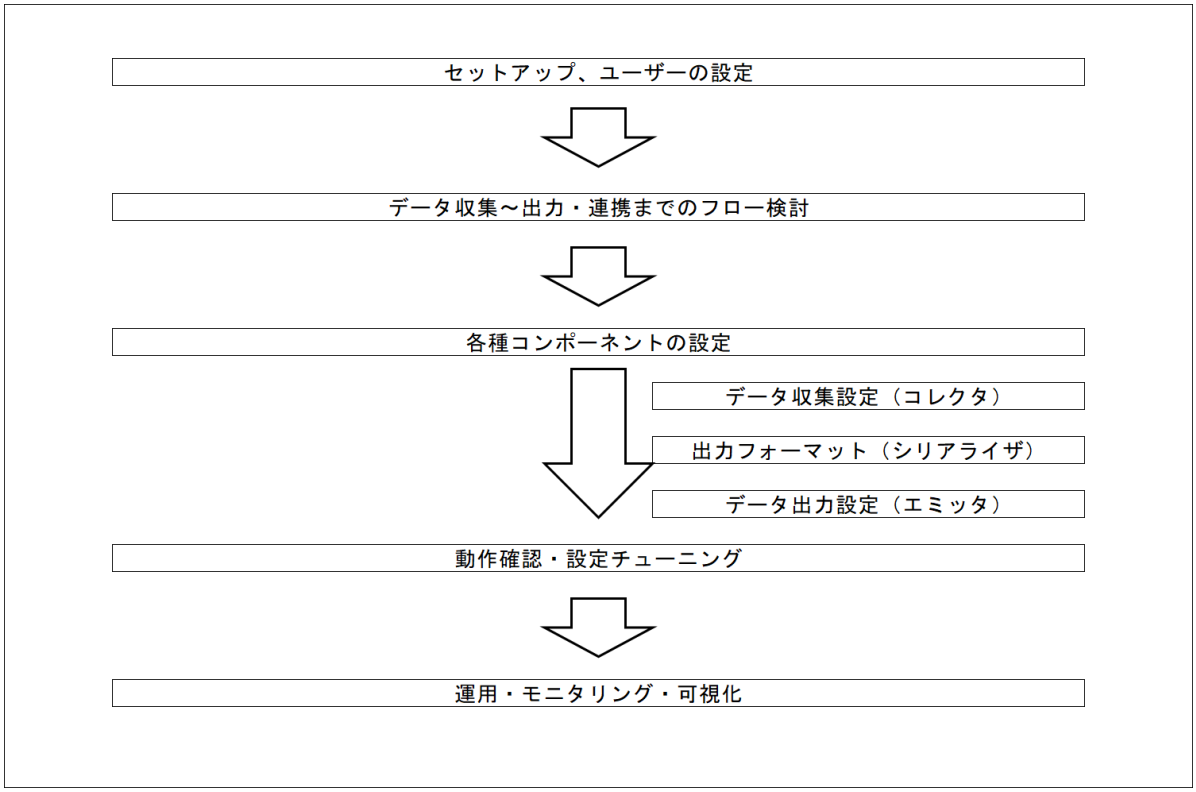
コンポーネントは７つのカテゴリで分類されます。 各カテゴリに属するコンポーネントを下表に示します。

カテゴリ	名称	説明
コレクタ	-	データ収集機能を提供します。データ収集後の各種機能を利用するための起点となります。
	EtherNet/IP	EtherNet/IP対応機器からデータ収集します。
	PLC	PLCのレジスタ/変数情報からデータ収集します。
	Modbus TCP	Modbus TCP対応機器からデータ収集します。
	MQTT	MQTTプロトコル（Subscribe）によるデータ収集を行います。
	リソース	稼働マシン内のシステムリソース（CPU使用率、メモリ消費など）を収集します。
	演算コレクタ	収集データを使った演算の結果を新しい別データとして登録します。
エミッタ	-	データ出力またはデータ送信機能を提供します。外部システムとの連携に活用できます。
	Cloud（Azure用）	Azure IoT Hub向けにデータ送信を行います。
	MQTT	MQTTプロトコル（Publish）によるデータ送信を行います。
	FTP/FTPS/SFTP	FTP、FTPSまたはSFTPによるデータ転送を行います。
	File	収集したデータをファイルとして、ストレージへ保存します。
	Cloud（AWS用）	AWS IoT Core向けにデータ送信を行います。
シリアライザ	-	データを指定フォーマットへ加工します。
	CSV	指定データをCSV形式へ加工します。
	JSON	指定データをJSON形式へ加工します。
アクション	-	収集したデータを使った任意のアクションを提供します。

カテゴリ	名称	説明
	シェルコマンド	収集したデータを使った任意のシェルコマンドを実行します。
	メール送信	収集したデータを使ってメール送信を行います。
	Modbus書き込み	Modbus TCP対応機器の指定アドレスへ値を書き込みます。
	PLC書き込み	PLCの指定アドレスへ値を書き込みます。
	EtherNet/IP書き込み	EtherNet/IP対応機器へ値を書き込みます。
ロジック	-	リアルタイム分析や収集データの判定処理を提供します。
	イベントデータ	収集したデータを条件判定し、条件合致した場合のみ、新たに別データとして登録します。
	イベントトリガ	収集したデータを条件判定し、条件合致している区間の情報提供を行います。
	基本統計	収集データを用いて統計処理した結果を別データとして登録します。
	移動平均	収集データの移動平均値をデータとして登録します。
	FFT	収集データのFFTした結果をデータとして登録します。（周波数、強度）
システム	-	システムに関連する機能を提供します。
	DBクエリサービス	SpeeDBee Synapse内で集めたデータへの問い合わせが可能です。Grafana連携でも使用します。
	コアインターナル	SpeeDBee Synapse内の内部ステータスを管理するコンポーネントです。
	エラーマネージャー	SpeeDBee Synapse内で発生した全エラーを検知できるコンポーネントです。
カスタム	-	ユーザーがPythonまたはC言語で設計したプログラムで構成されたコンポーネントです。標準のコンポーネントでは出来ない様々な機能を追加できます。

3.3 運用までの流れ

SpeeDBee Synapseの運用までの基本的な流れは、次の通りです。



①セットアップ・ユーザーの設定

本マニュアルを参考に、SpeeDBee Synapseのセットアップを行います。 また、利用者の権限に応じたユーザーを設定します。

②データ収集～出力・連携までのフロー検討

システムの目的に応じて、実際に、何のデータを収集するか、使用する機器の選定・検討を行います。 また、集めたデータを最終的に、どのような形式で出力して活用するかを設計します。 利用するコレクタ、シリアライザ、エミッタなど対応機能を確認します。 この段階で、サポートしていないプロトコルなどが必要な場合は、カスタムコンポーネントの開発も視野に検討します。

③各種コンポーネントの設定

検討したフローに従って、各種コンポーネントの設定を行います。 基本的な流れとしては、データ収集、加工、出力の順で設定していきます。

- データ収集設定（コレクタ）

データ収集の設定を行います。収集元の機器やその設定が正しい事を確認します。 その後、実際にコンポーネントの設定を行います。

- 出力フォーマットの設定（シリアライザ）

データの出力形式の設定を行います。CSVやJSONのいずれかを設定します。 また、出力ファイルをどのような単位で分割するかなどもここで設定します。

- データ出力設定（エミッタ）

出力したデータを最終的にどのように処理するかを設定します。 例えば、Fileエミッタでは、出力ファイルをストレージへ保存するため、その保存先の設定などを行います。 外部へのデータ送信を行う場合は、外部の接続に必要な情報を確認して設定を行います。

④動作確認・設定チューニング

設定した各コンポーネントが正しく動作しているか確認します。正常にデータ収集できている事、指定した形式でのデータ出力および 送信ができている事を確認します。意図していない動作などあれば設定を見直して、設定変更、再確認を行います。

⑤運用・モニタリング・可視化

実際にシステムを運用します。データ活用エンジンの提供する機能を使い、データのモニタリングやグラフの可視化ができます。 運用中に問題が生じた場合は、再検討を行い設定チューニングを再実施して下さい。

この他に、目的に応じて、リアルタイム分析やイベント連携などの設定を行う事で、SpeeDBee Synapseを活用したシステムが構築できます。

４導入フロー

4.1 導入フロー

本章では、SpeeDBee Synapseを導入するための手順を説明します。

導入フローとして、次の流れでシステムを構築します。

1. インストール（Linux）
2. インストール（Windows）
3. 初期設定
4. コンポーネントの設定
5. 収集サービスの実行

ここで説明するシステムは、非常にシンプルなものですが、SpeeDBee Synapseの基本を理解するために一読することを推奨します。

4.2 インストール（Linux）

ここでは、SpeeDBee Synapse（Linux）のインストール/アンインストールの方法を説明します。

4.2.1 実行環境

本パッケージの動作環境は、Ubuntu20(AMD64)/Ubuntu22(AMD64、ARM64)です。
上記以外の環境では、当該環境のマニュアルをご確認ください。

> Data Flow Controller版はSynapseが初期状態でインストールされているため、この章の内容を行う必要はありません。

4.2.2 インストールおよびアップデート

下記の手順で、デバイスにSpeeDBee Synapseをインストールすることができます。アップデート時も同様の手順となります。

ファイルを配置

1. SpeeDBee Synapseのファイルを、デバイスの/tmpディレクトリへ配置します。下記は、Windows PowerShellからscpコマンドで配置する例です。

デバイスのOSがUbuntu20(amd64)の場合

```
PS >scp speedbeesynapse_ubuntu20_x.x.x-r0_amd64.deb (アカウント名)@(デバイスのIPアドレス):/tmp
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(amd64)の場合

```
PS >scp speedbeesynapse_ubuntu22_x.x.x-r0_amd64.deb (アカウント名)@(デバイスのIPアドレス):/tmp
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(arm64)の場合

```
PS >scp speedbeesynapse_ubuntu22_x.x.x-r0_arm64.deb (アカウント名)@(デバイスのIPアドレス):/tmp
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがRaspberry Pi OS(bookworm 64-bit)の場合

```
PS >scp speedbeesynapse_raspi_bookworm_x.x.x-r0_arm64.deb (アカウント名)@(デバイスのIPアドレス):/tmp
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがdebian12(bookworm arm64)の場合

```
PS >scp speedbeesynapse_debian12_x.x.x-r0_arm64.deb (アカウント名)@(デバイスのIPアドレス):/tmp
```

※x.x.xの部分には、バージョン番号が入ります。

インストール及び起動

1. sshを使用してデバイスへログインします。下記はWindows PowerShellからsshコマンドでログインする例です。

```
PS >ssh (アカウント名)@(デバイスのIPアドレス)
```

2. 下記のコマンドを実行し、SpeeDBee Synapseをインストールします。

デバイスのOSがUbuntu20(amd64)の場合

```
$ sudo apt install /tmp/speedbeesynapse_ubuntu20_x.x.x-r0_amd64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(amd64)の場合

```
$ sudo apt install /tmp/speedbeesynapse_ubuntu22_x.x.x-r0_amd64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(arm64)の場合

```
$ sudo apt install /tmp/speedbeesynapse_ubuntu22_x.x.x-r0_arm64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがRaspberry Pi OS(bookworm 64-bit)の場合

```
$ sudo apt install /tmp/speedbeesynapse_raspi_bookworm_x.x.x-r0_arm64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがdebian12(bookworm arm64)の場合

```
$ sudo apt install /tmp/speedbeesynapse_debian12_x.x.x-r0_arm64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

プロキシ経由でのインストール

社内にプロキシサーバ（フォワードプロキシ）を配置している環境でのインストールは、プロキシへの接続情報を明示的に設定する必要があります。 下記コマンドのように環境変数 `http_proxy`, `https_proxy` を指定してください。

```
$ sudo http_proxy=<プロキシサーバーのurl> https_proxy=<プロキシサーバーのurl> apt install ...
```

※Basic認証のユーザー名・パスワードを指定する場合は、以下のようにurlに含めてください。

```
http://<ユーザー名>:<パスワード>@<ホスト>:<ポート>
```

3. 下記のコマンドを実行し、SpeeDBee Synapseを起動します。

```
$ sudo systemctl start speedbeesynapse
```

ブラウザからアクセス

1. ブラウザから下記のURLにアクセスすることで、SpeeDBee Synapseの画面を開くことができます。
   - `http://{ホスト名orデバイスのIPアドレス}:8120`

     例） `http://192.168.1.3:8120`

     > **SpeeDBee Synapseにアクセスするpcについて**
     >
     > インストールしたデバイスと同一のネットワークに属するPCから、ブラウザ（Google Chrome/Microsoft Edge）でSpeeDBee SynapseのURLにアクセスしてください。
     > ホスト名でアクセスする場合、PCがホスト名でアクセスできる環境設定が必要です。
     > ※一部のiPad端末（Safari使用）でも動作確認を行っており、基本的な操作が可能です。

     > **アップデート後の操作について**
     >
     > SpeeDBee Synapseのアップデートを画面を表示したまま実施し、そのまま操作を継続すると正常に動作しない可能性があります。 アップデート後に設定を行う場合は、画面の再読み込み（リロード）を行ってください。 アップデート後に、マニュアルへ記載されている新しい画面項目が表示されないなどの症状がある場合は、ブラウザのキャッシュをすべて削除してください。

2. http経由のため、初回アクセス時には画面上にセキュリティに関する警告が表示される場合があります。

> **https経由でのアクセス**
>
> https経由でSpeeDBee Synapseにアクセスすることも可能です。詳細は「HTTP/HTTPS通信」をご覧ください。

4.2.3 SpeeDBee Synapseの制御

SpeeDBee Synapseをインストールしたデバイスにsshを使用してログインし、下記の各コマンドを実行することで、SpeeDBee Synapseを制御することができます。

- 起動

```
$ sudo systemctl start speedbeesynapse
```

- 再起動

```
$ sudo systemctl restart speedbeesynapse
```

- 停止

```
$ sudo systemctl stop speedbeesynapse
```

- 自動起動の有効化

  この設定を行うことで、デバイスを再起動した場合に自動でSpeeDBee Synapseが起動するようになります。

```
$ sudo systemctl enable speedbeesynapse
```

- 自動起動の無効化

  この設定を行うと、デバイスを再起動した際にSpeeDBee Synapseは自動では起動しません。

```
$ sudo systemctl disable speedbeesynapse
```

4.2.4 アンインストール

下記の手順で、デバイスからSpeeDBee Synapseをアンインストールすることができます。

1. sshを使用してデバイスへログインします。下記はWindows PowerShellからsshコマンドでログインする例です。

```
PS >ssh (アカウント名)@(デバイスのIPアドレス)
```

2. 下記のコマンドを実行し、SpeeDBee Synapseをアンインストールします。
   - ログや設定ファイル類を残す場合

```
$ sudo apt remove speedbeesynapse
```

   - ログや設定ファイル類もすべて削除する場合

```
$ sudo apt purge speedbeesynapse
```

再度SpeeDBee Synapseをインストールする場合の手順は、「インストール」に記載の手順と同じです。

アンインストール時にログや設定ファイル類を残していた場合は、再インストール後もその設定内容が引き継がれます。

4.3 インストール（Windows）

ここでは、SpeeDBee Synapse（Windows）のインストール/アンインストールの方法を説明します。操作内容は、Windows 11を使って記載しています。OSバージョンにより使用するWindowsアプリケーションの起動方法が異なる場合があります。OSバージョンに合わせて、操作してください。

4.3.1 実行環境

本パッケージの動作環境は、次のとおりです。

- OS：Windows 10（64bit） version 22H2 以降 、Windows 11（64bit） version 23H2 以降
- アーキテクチャ：amd64（x86_64）

上記以外の環境では、当該環境のマニュアルをご確認ください。

4.3.2 インストールおよびアップデート

下記の手順で、PCにSpeeDBee Synapseをインストールすることができます。アップデート時も同様の手順となります。

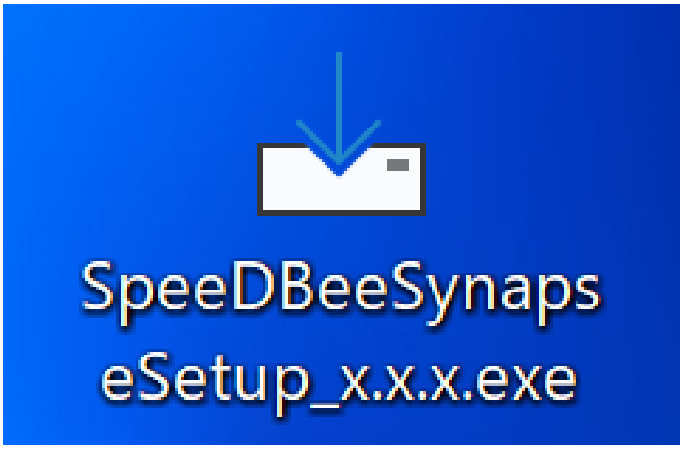
インストール準備

下記のインストーラを使用します。事前にインストール先のWindows PCのデスクトップにファイルを配置してください。

- SpeeDBeeSynapseSetup_x.x.x.exe （x.x.xの部分はバージョンにより異なります）

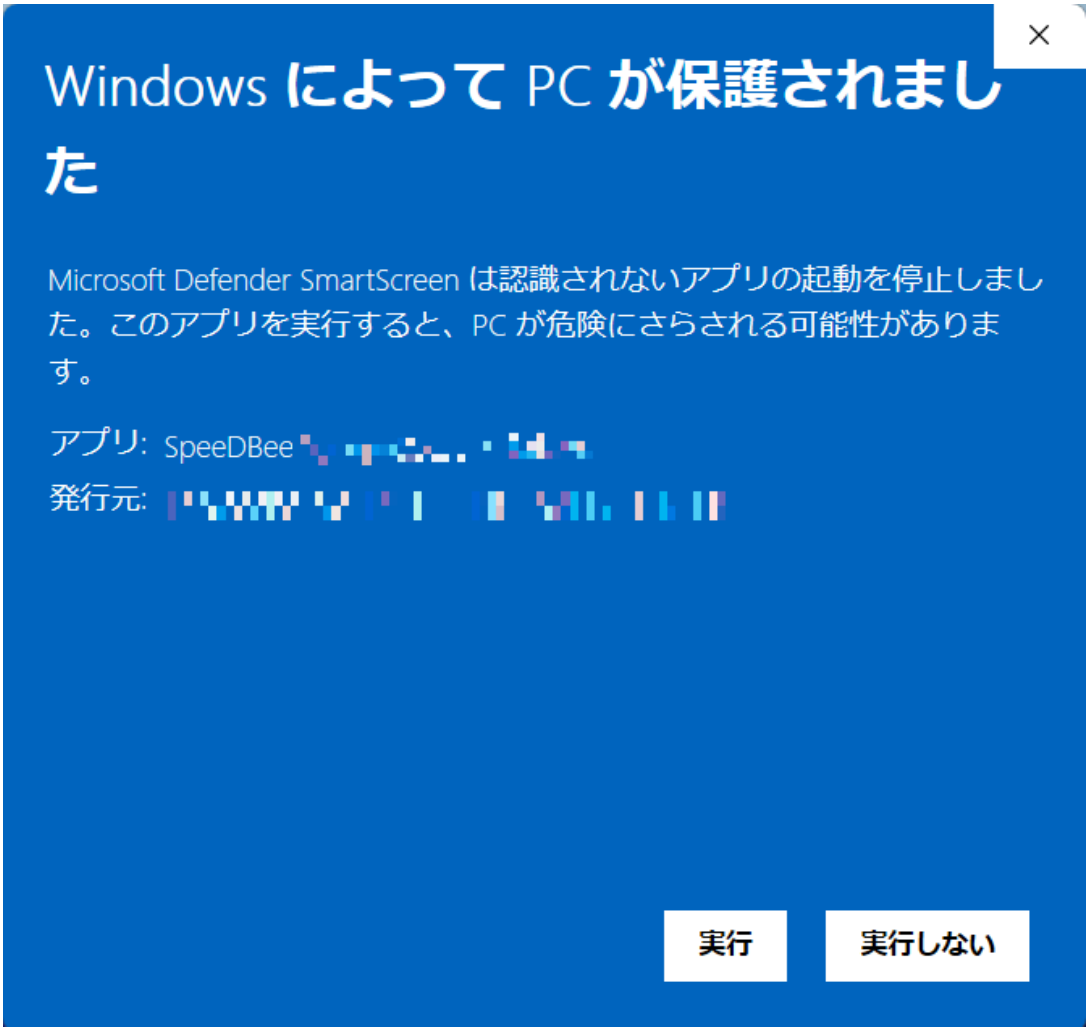
インストール及び起動

1. SpeeDBee Synapseのインストーラを実行します。

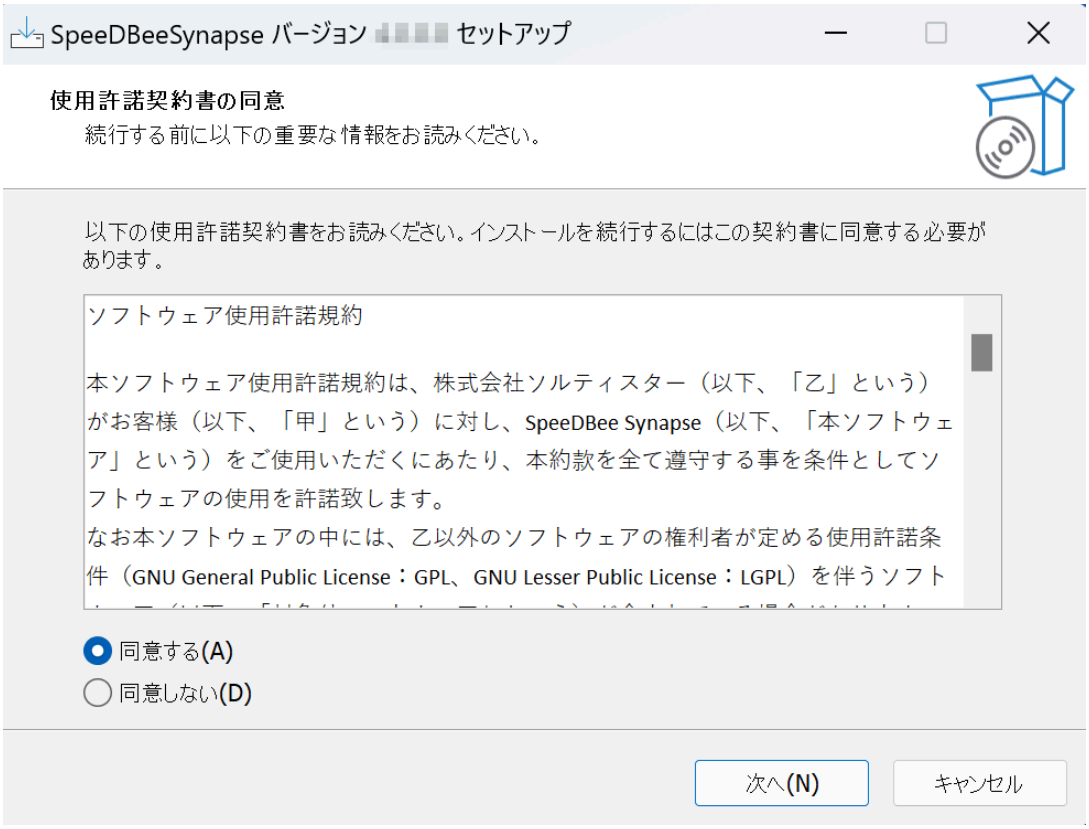


    - インストーラの実行には、Windowsの管理者権限が必要です

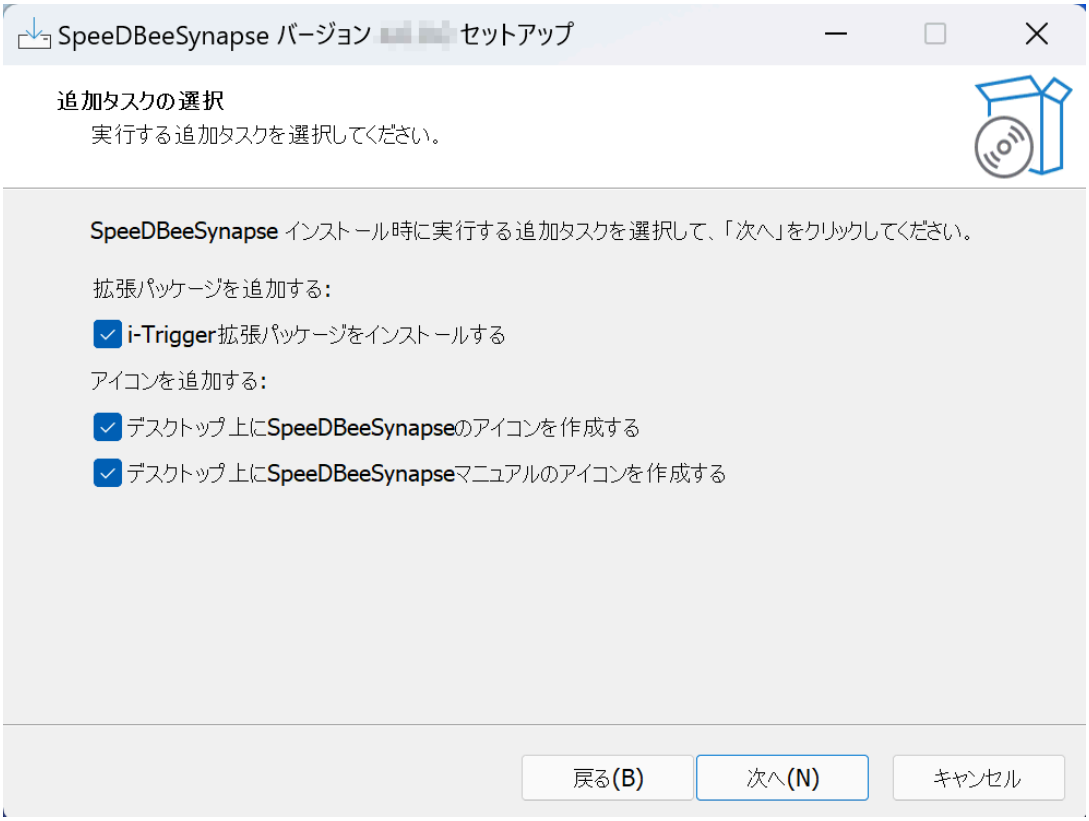
2. Microsoft Defender が有効な場合に、警告が表示されることがあります。発行元が「OMRON Corporation」であることを確認して、「実行」をクリックしてください。



3. 「ソフトウェア使用許諾規約」を確認して、「同意する」、「次へ」をクリックします。

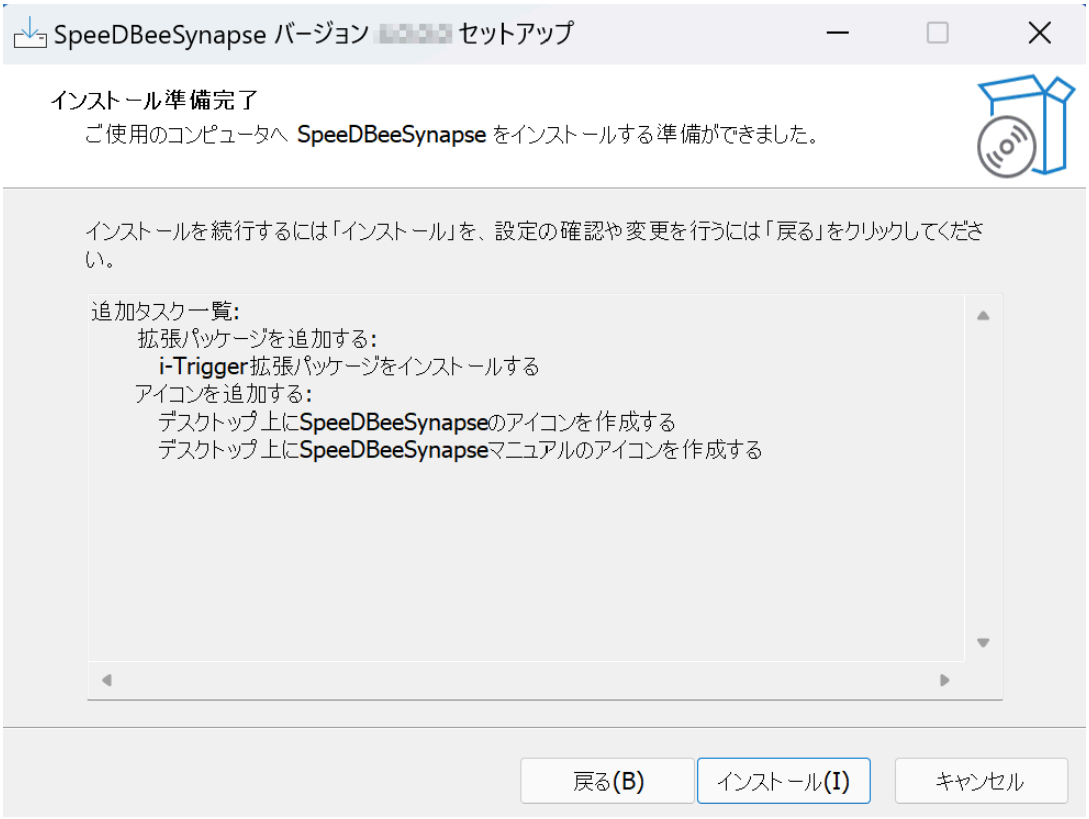


4. 追加タスクの選択が表示されます。「次へ」をクリックします。

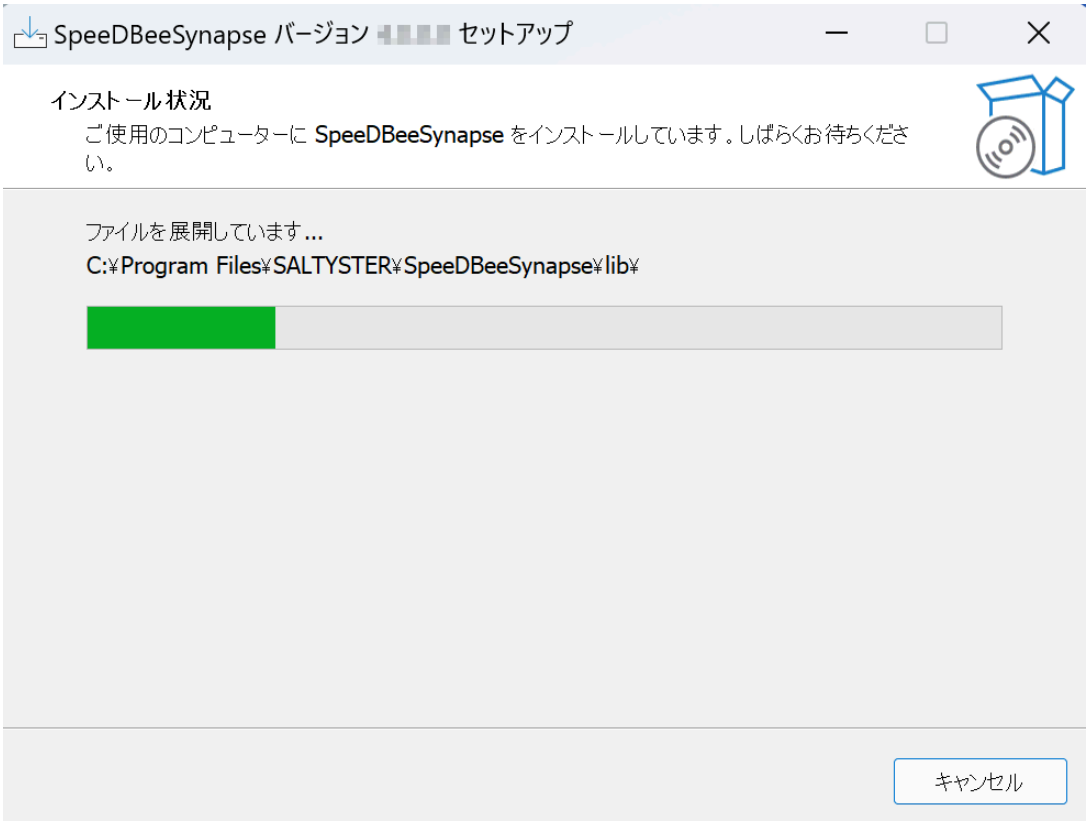


- i-Trigger（ドラレコ）機能が不要であればチェックを外してください
- 不要なアイコンがあればチェックを外してください

5. 表示内容を確認して、「インストール」を実行します。



6. インストールが進行します。



7. インストールが終了したら、「完了」をクリックします。



インストール後にPC再起動を求められることがあります。その際は再起動を行ってください。

ブラウザからアクセス

1. ブラウザから下記のURLにアクセスすることで、SpeeDBee Synapseの画面を開くことができます。
   - `http://{ホスト名orデバイスのIPアドレス}:8120`

     例） `http://192.168.1.3:8120`

> **SpeeDBee Synapseにアクセスするpcについて**
>
> インストールしたPC、または対象のPCと同一のネットワークに属するPCから、ブラウザ（Google Chrome/Microsoft Edge）でSpeeDBee SynapseのURLにアクセスしてください。
> ホスト名でアクセスする場合、PCがホスト名でアクセスできる環境設定が必要です。
> ※一部のiPad端末（Safari使用）でも動作確認を行っており、基本的な操作が可能です。

> **アップデート後の操作について**
>
> SpeeDBee Synapseのアップデートを画面を表示したまま実施し、そのまま操作を継続すると正常に動作しない可能性があります。 アップデート後に設定を行う場合は、画面の再読み込み（リロード）を行ってください。 アップデート後に、マニュアルへ記載されている新しい画面項目が表示されないなどの症状がある場合は、ブラウザのキャッシュをすべて削除してください。

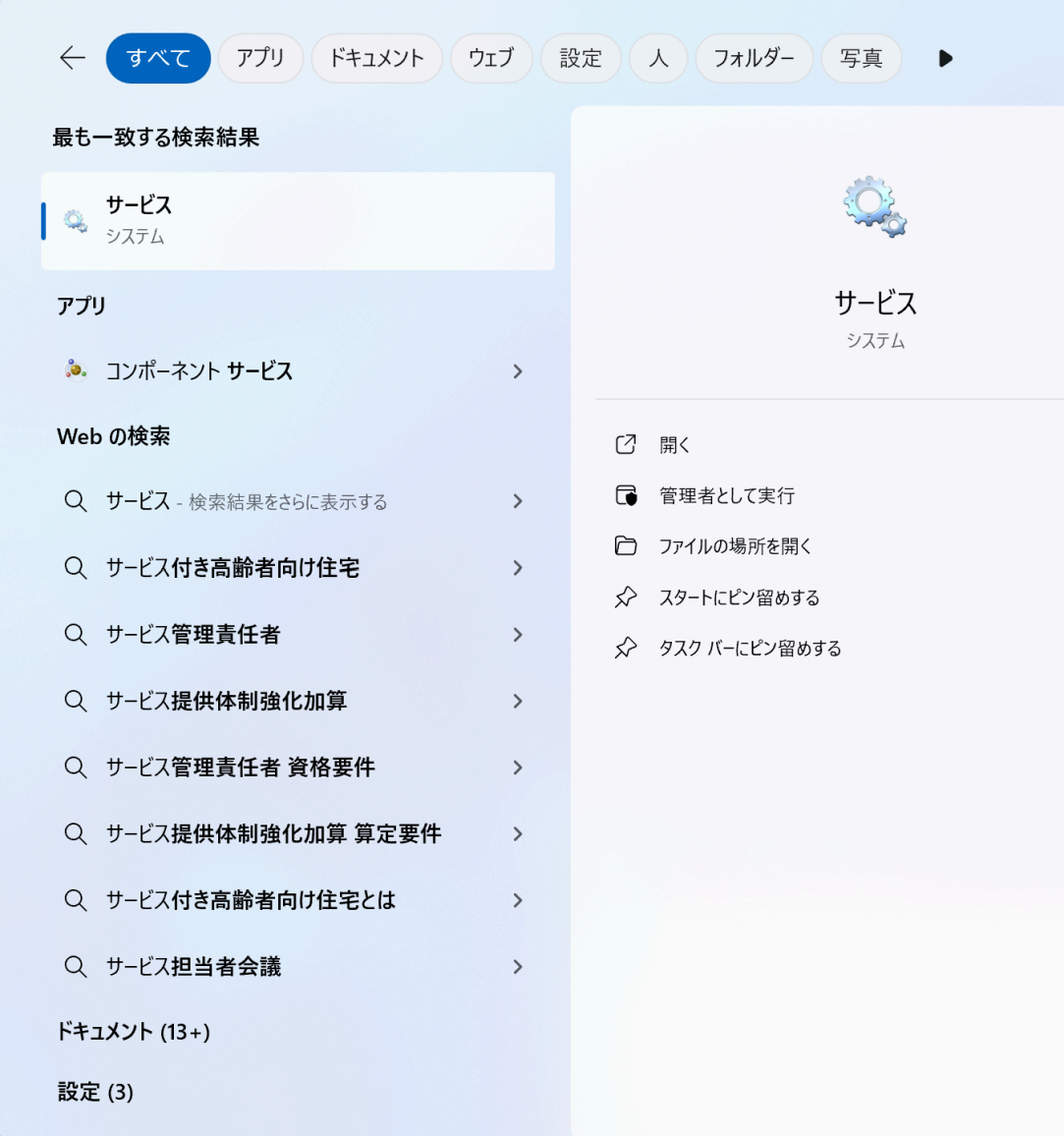
2. http経由のため、初回アクセス時には画面上にセキュリティに関する警告が表示される場合があります。

> **https経由でのアクセス**
>
> https経由でSpeeDBee Synapseにアクセスすることも可能です。詳細は「HTTP/HTTPS通信」をご覧ください。

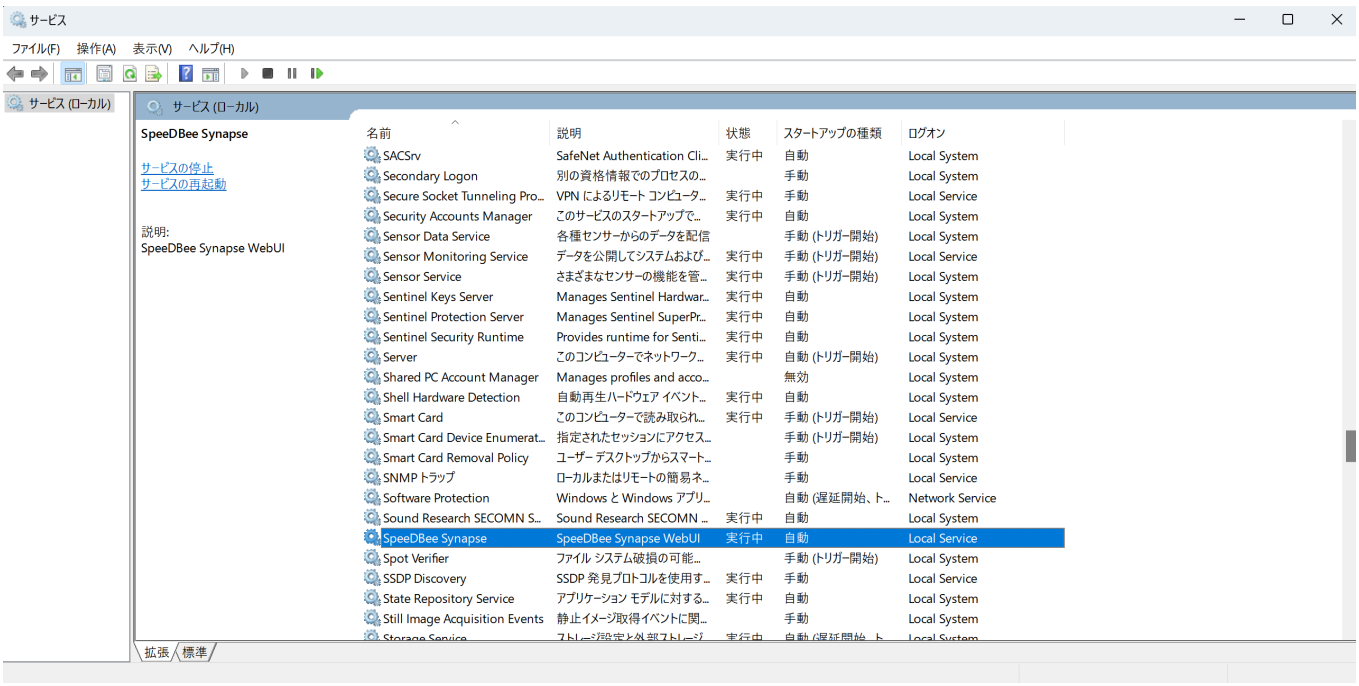
4.3.3 SpeeDBee Synapseの制御

SpeeDBee Synapseは、インストールするとWindowsのサービスとして登録されます。そのため、各制御はWindowsのサービスから操作します。

1. デスクトップ下部の検索欄へ「サービス」と入力して、検索結果からサービスを起動します。



2. 「SpeeDBee Synapse」を選択して、各種制御を行ないます。

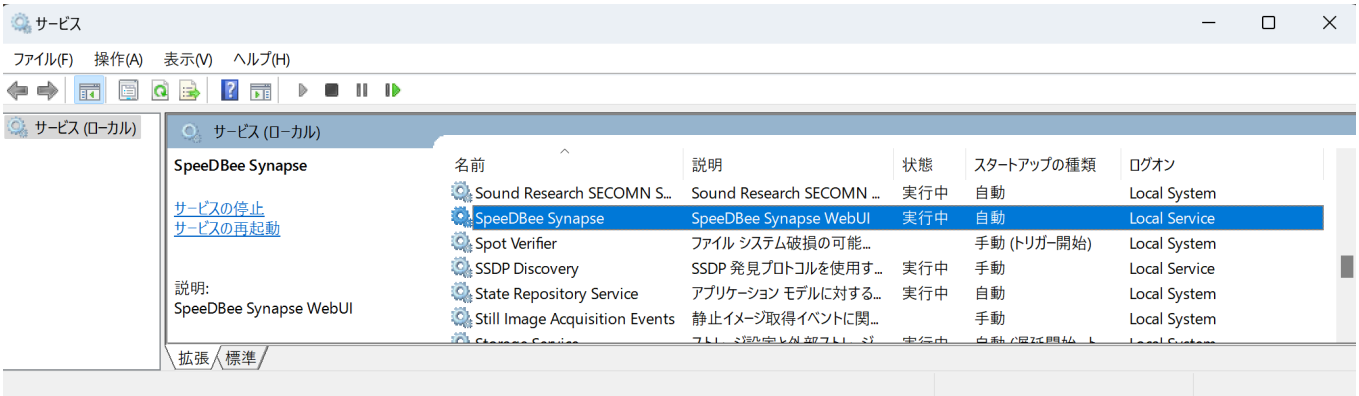


4.3.3.1 起動

1. SpeeDBee Synapseを選択して、「サービスの開始」をクリックします。

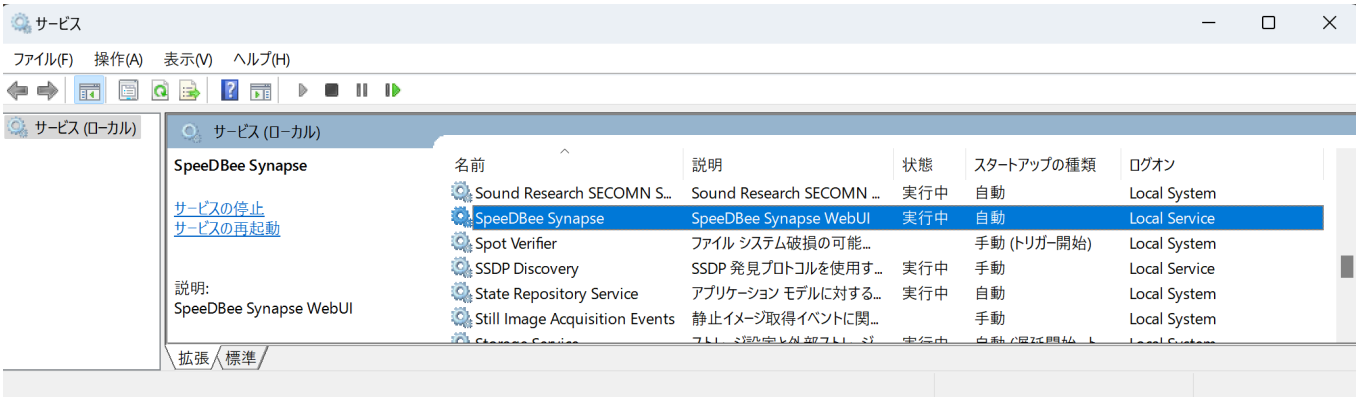


2. サービスが起動します。正常に起動していれば、状態が「実行中」になります。

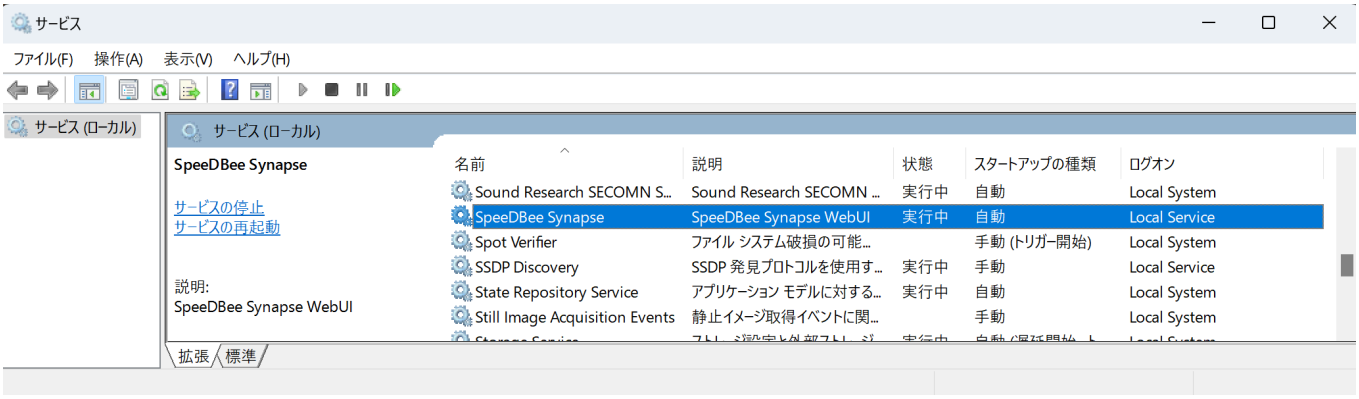


4.3.3.2 再起動

1. SpeeDBee Synapseを選択して、「サービスの再起動」をクリックします。

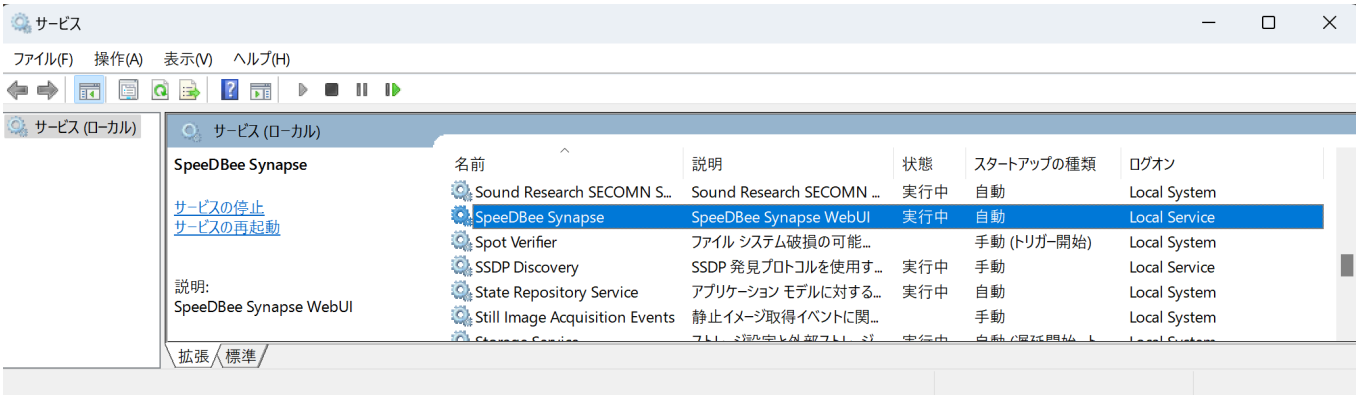


2. サービスが再起動します。



4.3.3.3 停止

1. SpeeDBee Synapseを選択して、「サービスの停止」をクリックします。



2. サービスが停止します。



4.3.3.4 自動起動の有効化

1. SpeeDBee Synapseを選択して、右クリックのメニューから「プロパティ」をクリックします。
2. 「スタートアップの種類」を「自動」に設定して、「OK」をクリックします。



3. この設定によりPC起動時、SpeeDBee Synapseのサービスが自動起動するようになります。

4.3.3.5 自動起動の無効化

1. SpeeDBee Synapseを選択して、右クリックのメニューから「プロパティ」をクリックします。

2. 「スタートアップの種類」を「無効」に設定して、「OK」をクリックします。

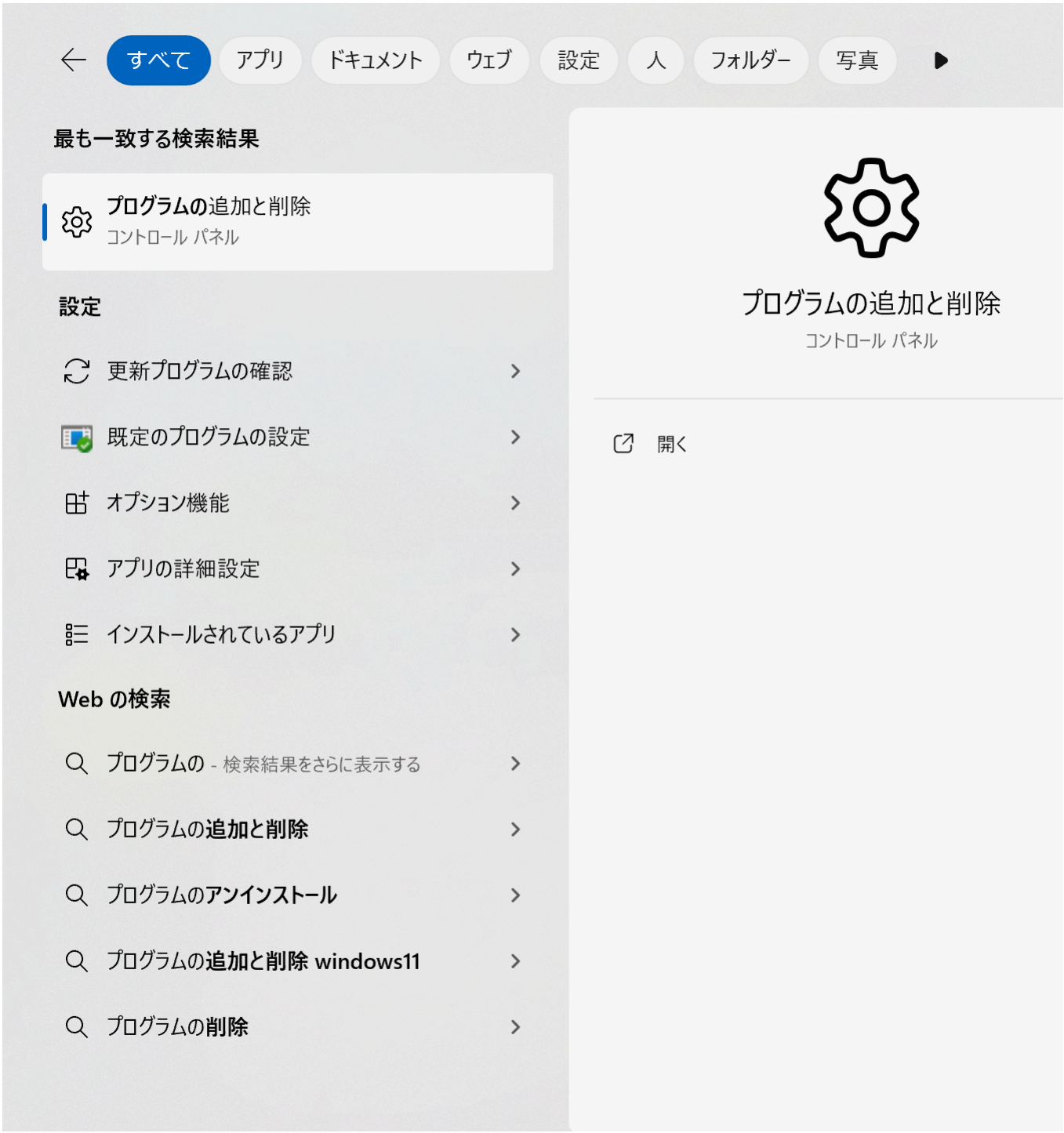


3. この設定を適用すると、PC起動時、SpeeDBee Synapseのサービスは起動しません。起動する場合は、手動で起動操作を行なう必要があります。

4.3.4 アンインストール

下記の手順で、デバイスからSpeeDBee Synapseをアンインストールすることができます。

1. デスクトップ下部の検索欄へ「プログラムの追加と削除」と入力して、検索結果から「プログラムの追加と削除」を起動します。



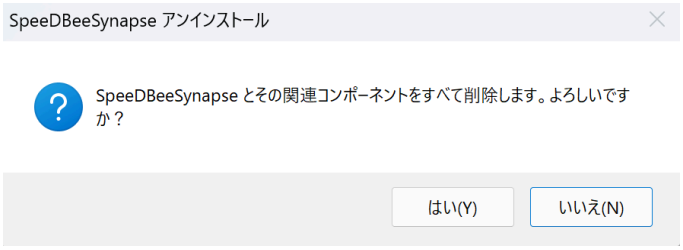
2. インストールされているアプリの検索欄へ「SpeeDBee Synapse」と入力して、検索結果からSpeeDBee Synapseのメニューから「アンインストール」をクリックします。



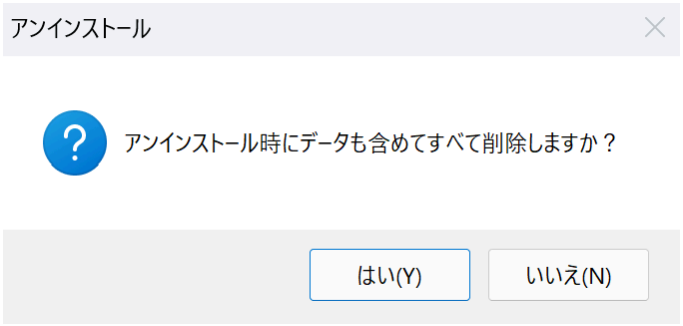


- アンインストールの実行には、Windowsの管理者権限が必要です

3. 確認メッセージが表示されたら、「はい」をクリックします。



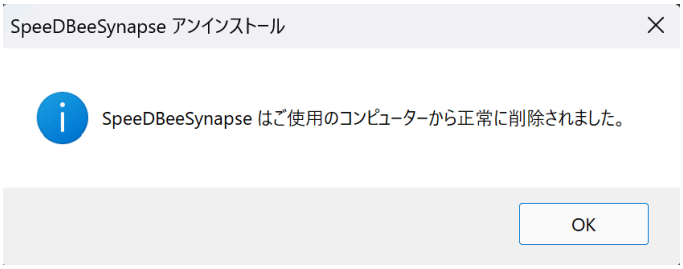
4. 続けて、次のメッセージ内容を確認します。データ（ProgramData内に保存される情報）をすべて削除する場合は、「はい」をクリックします。データを残す場合は、「いいえ」をクリックしてください。



5. アンインストール処理が実行されます。



6. 完了後に、「OK」をクリックします。これでSpeeDBee Synapseのアンインストールは完了です。



4.4 初期設定

ここでは初期設定（ライセンスの登録、adminユーザーのパスワード変更）の方法と、ユーザーを追加する方法を説明します。

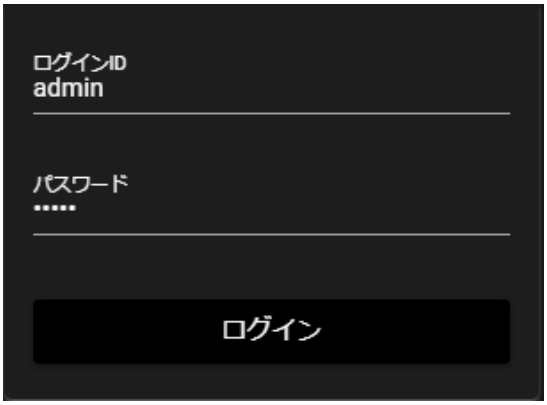
4.4.1 管理者パスワードの変更

初期設定として、ライセンスを登録し、adminユーザーのパスワードをデフォルトの設定値から変更します。 adminは、SpeeDBee Synapseにデフォルトで登録されているユーザーです。 adminユーザーのデフォルトの設定値は、以下の通りです。

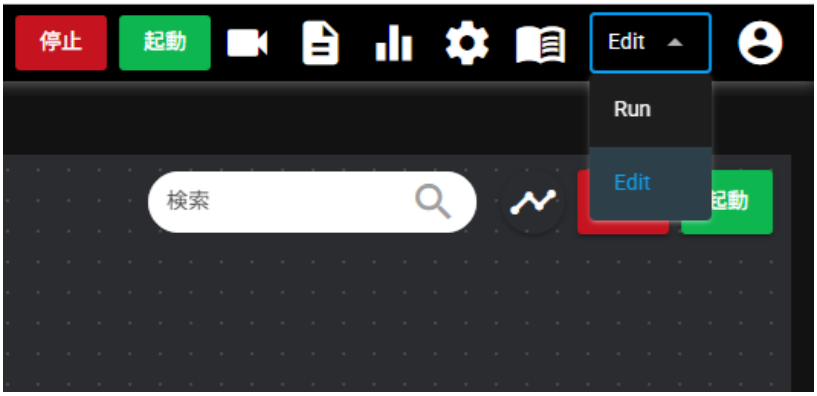
項目	説明
ログインID	admin
ユーザー名	admin
権限	システム管理者
パスワード	admin
Eメールアドレス	admin@example.com

Data Flow Controller版、OMRON-SBS for Windows版では初期状態でライセンス登録済みとなっています。その場合、手順7からご確認ください。

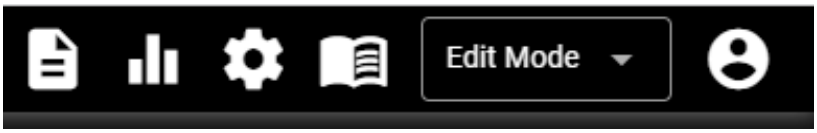
1. ログイン画面で、ログインIDとパスワードに「admin」と入力して「ログイン」を押下します。



2. ヘッダのモード選択で"Edit Mode"を選択し、Editモードの画面を開きます。



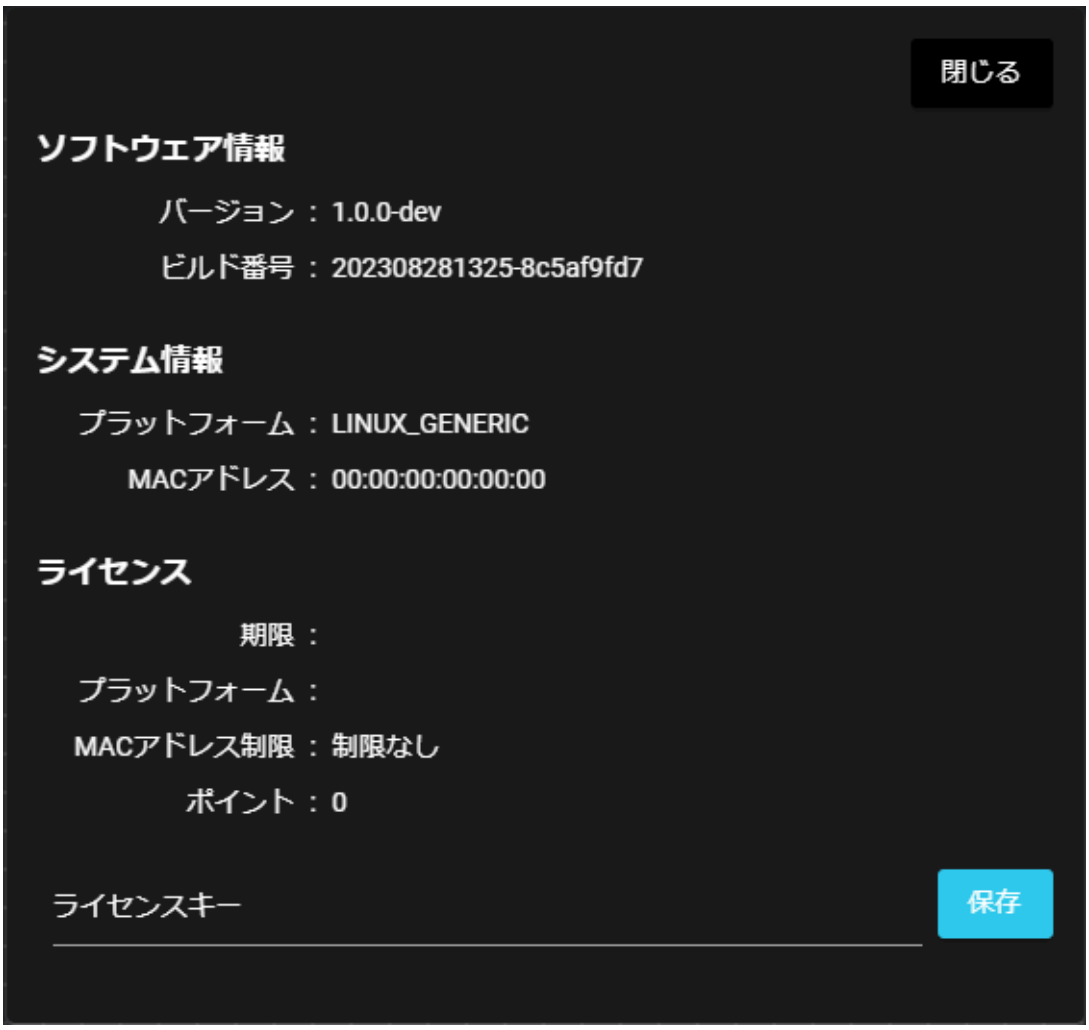
3. ヘッダの設定メニューアイコンを押下します。



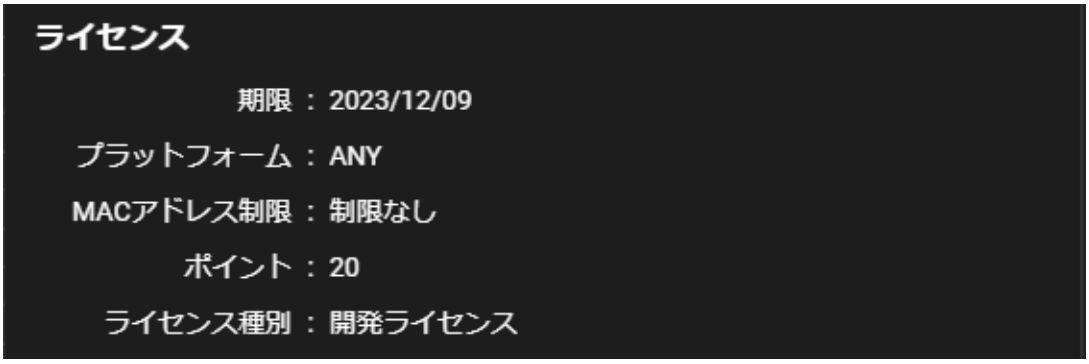
4. 「ソフトウェア情報」を選択します。



5. ライセンスキー入力欄にライセンスキーを入力し、「保存」を押下します。



6. 正しいライセンスキーが入力されると、入力されたライセンスの情報が表示されます。



7. 次に、ユーザーアイコンを押下して「ユーザー管理」を選択し、ユーザー管理ダイアログを開きます。



8. "admin"ユーザーの編集アイコンを押下します。



9. 「パスワード」と「パスワード(確認)」に、"admin"とは異なる任意のパスワードを入力して、「保存」を押下します。



4.4.2 ユーザーの追加

次の手順で、ユーザーを追加することができます。

1. ユーザー管理ダイアログで、「追加」ボタンを押下します。



2. 以下の項目を入力します。権限は「管理者」を選択します。入力が完了したら、「保存」ボタンを押下します。



項目	説明
無効にする	チェックを付けると、ユーザーが無効になり、ログインができなくなります。
ログインID	ログイン時に入力するIDです。
ユーザー名	ユーザーの名前です。※入力を省略する事が可能です。
権限	下表を参照
パスワード	ログイン時に入力するパスワードです。
Eメールアドレス	ユーザーに紐づけるEメールアドレスです。※入力を省略する事が可能です。

ユーザーの権限は4種類あります。

権限	説明
システム管理者	すべての操作が可能です。初期ユーザー"admin"に付与されており、1人以上必須です。
管理者	すべての操作が可能です。システム管理者の追加・更新・削除は不可です。
オペレーター	Run Modeの閲覧、およびコンポーネントの起動・停止が可能です。
閲覧者	Run Modeの閲覧が可能です。

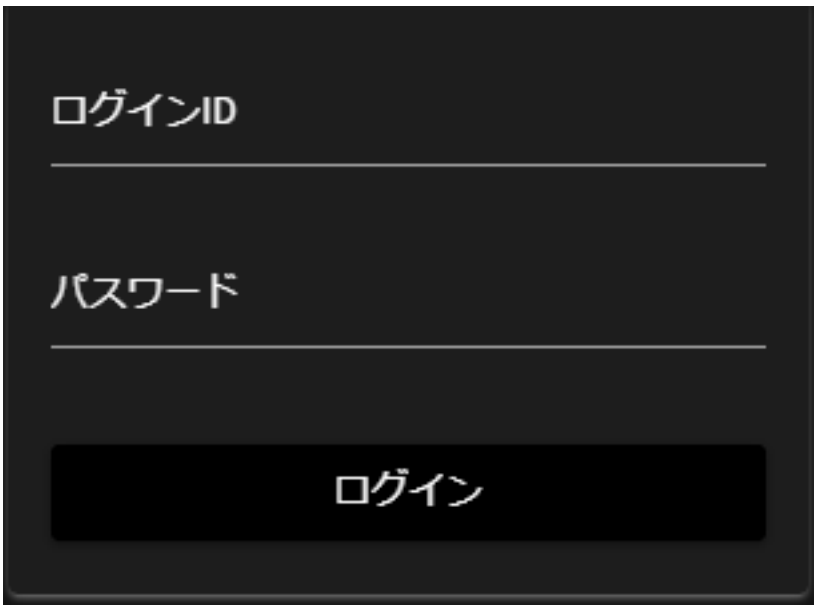
3. ユーザー管理ダイアログの「閉じる」を押下します。



4. ユーザーアイコンを押下し、「ログアウト」を選択します。



5. ログイン画面で、新しく追加したユーザーのログインIDとパスワードを入力して、「ログイン」を押下します。
   この操作は、該当するユーザー本人が操作します。



6. ユーザーアイコンを押下し、「プロフィール」を選択します。

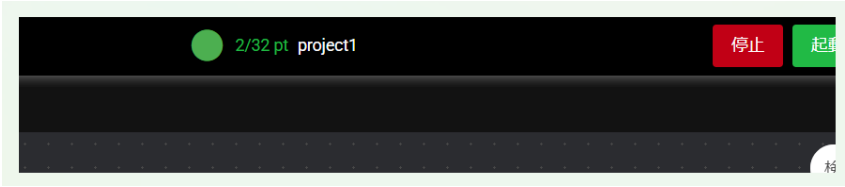


7. 現在ログインしているユーザーの情報が表示され、上で追加したユーザーでログインしている事がわかります。 設定値を変更して「保存」を押下すると、現在ログインしているユーザーの設定を変更できます。



4.4.3 プロジェクト名の設定

SpeeDBee Synapseは、設定したデータフローをプロジェクトという単位で管理します。
設定したデータフローを判別しやすくするためにプロジェクト名を設定してください。
（例：「ラインA生産情報可視化」、「設備連携システム」など任意で設定）
初期はproject1という名前で設定されており、プロジェクト名はヘッダに表示されます。



設定方法は、以下を参考にしてください。

設定情報の管理(プロジェクト)

4.5 コンポーネントの設定

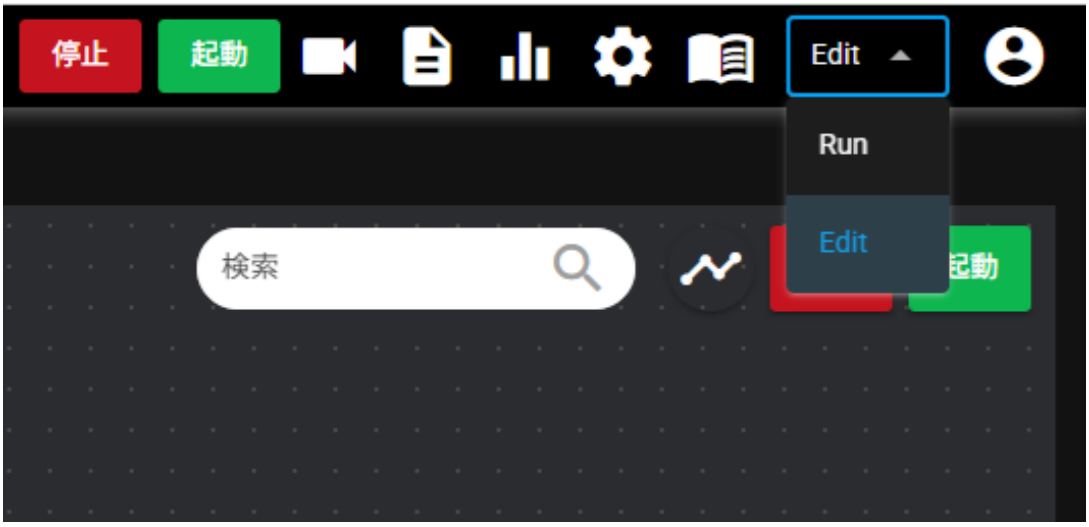
ここでは、収集したデータをファイルに保存するまでの設定方法を説明します。

4.5.1 リソースコレクタの設定

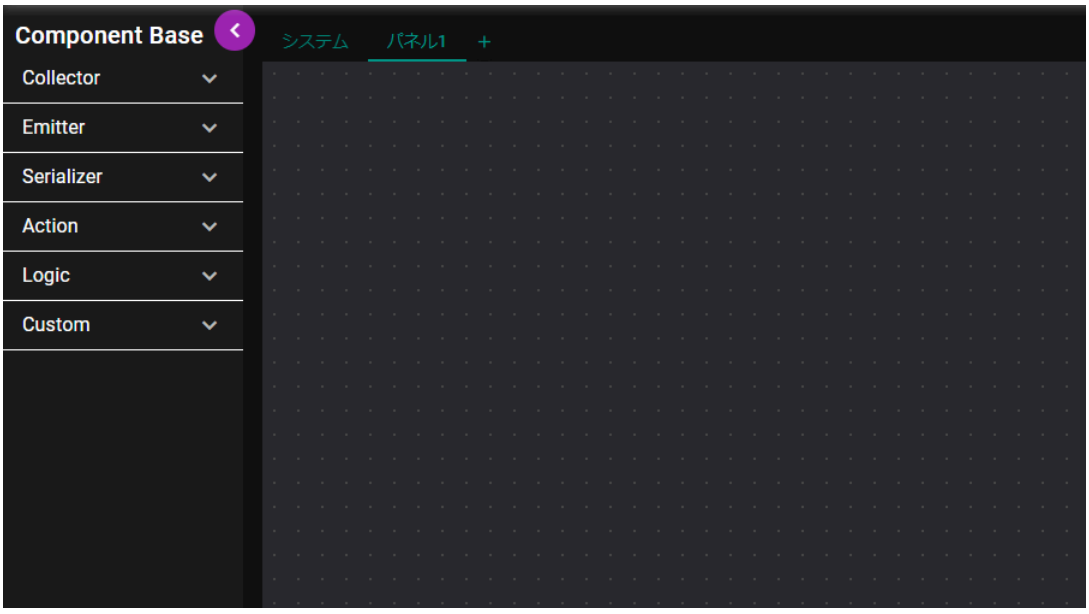
稼働マシンのCPU使用率を収集するために、"リソースコレクタ"を設定する方法を説明します。 "リソースコレクタ"は、稼働マシンのシステムリソース（CPU使用率、メモリ消費量など）を収集するコンポーネントです。

次の手順で、リソースコレクタを設定することができます。

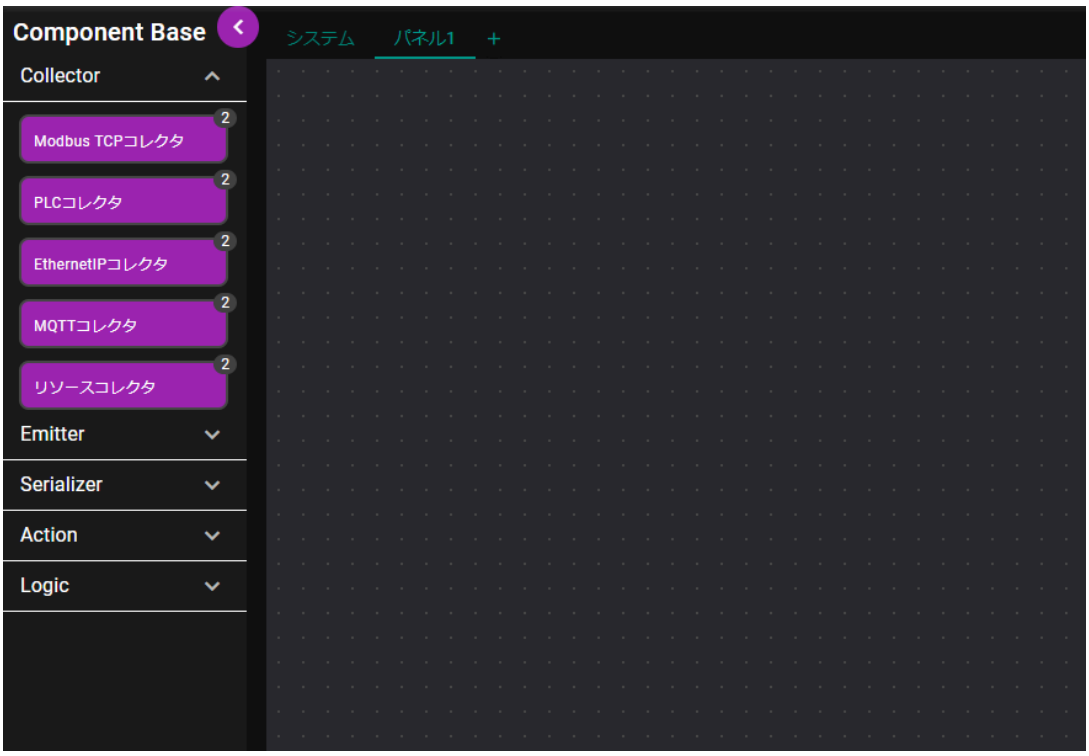
1. ヘッダのモード選択で"Edit mode"を選択し、Editモードの画面を開きます。



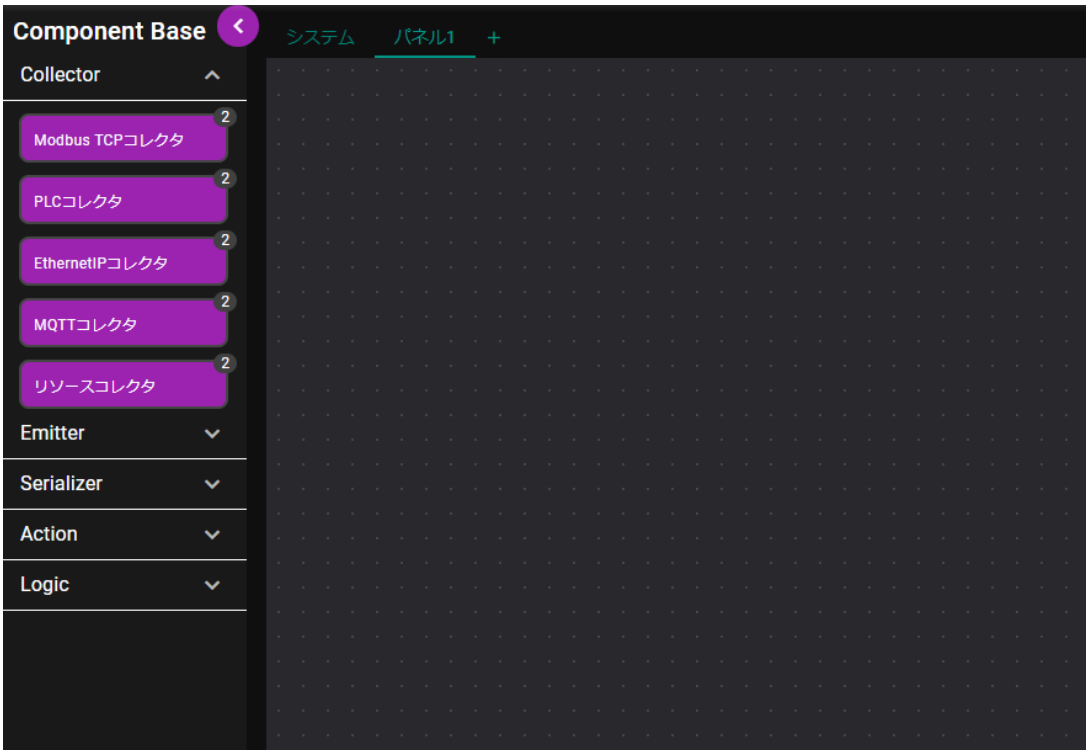
2. パネル1を表示します。



3. 左側のメニューで「Collector」のインジケーターを押下し、コレクター覧を表示します。



4. 「リソースコレクタ」をドラッグし、パネル上にドロップします。



5. 設定ダイアログが自動で開きます。"名称"を任意で入力し、"CPU使用率"にチェックを付けて「保存」ボタンを押下します。
   名称例：システムリソース



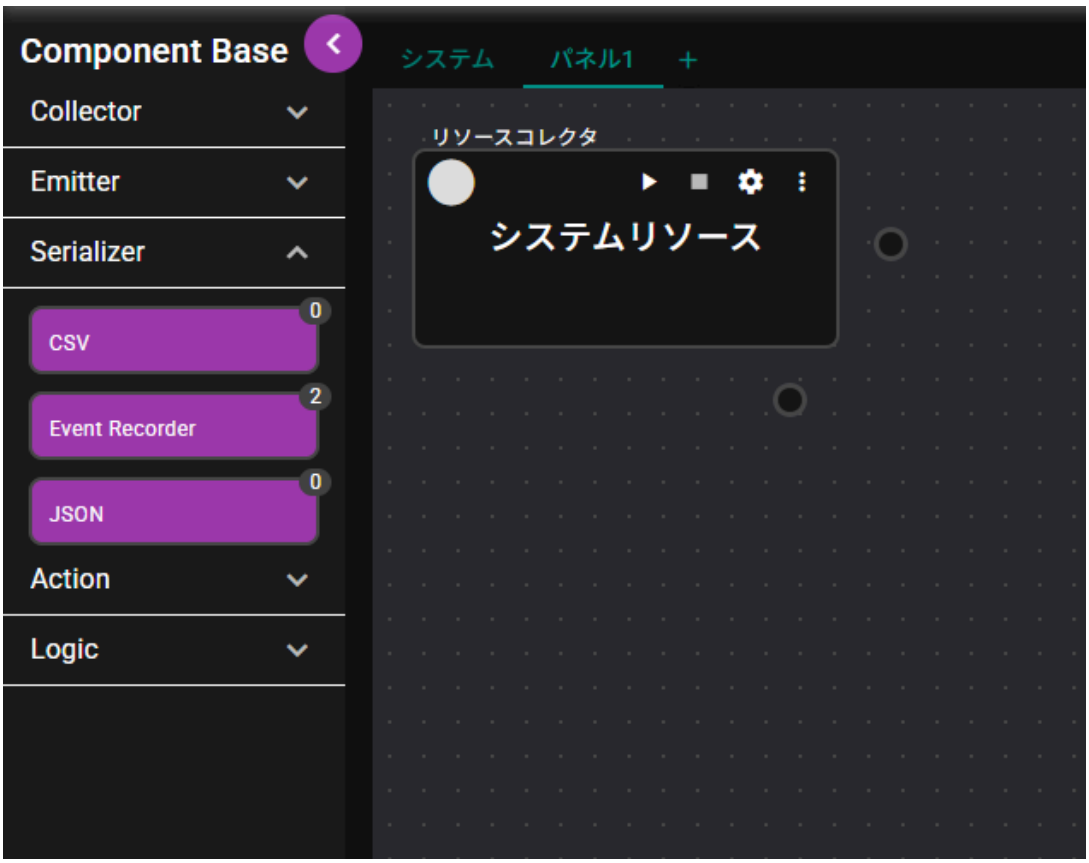
4.5.2 CSVシリアライザの設定

リソースコレクタで収集したCPU使用率を、CSV形式でファイル出力するために、"CSVシリアライザ"を設定する方法を説明します。"CSVシリアライザ"は、任意のデータを受け取り、CSV形式に加工して再出力するコンポーネントです。

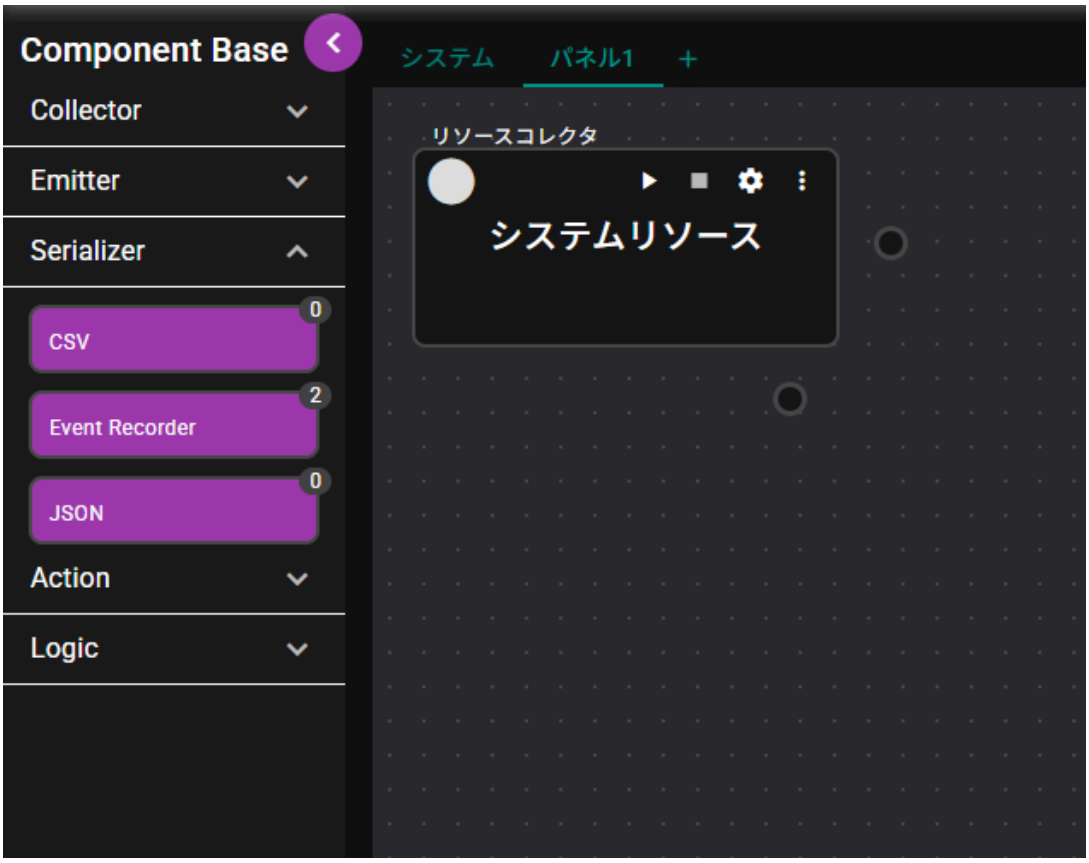
コンポーネント間のデータの流れは、フローリンクで定義します。 フローリンクとは、2つのコンポーネントをつなぎ、繋いだコンポーネント間にデータを流すものです。

次の手順で、CSVシリアライザを設定することができます。

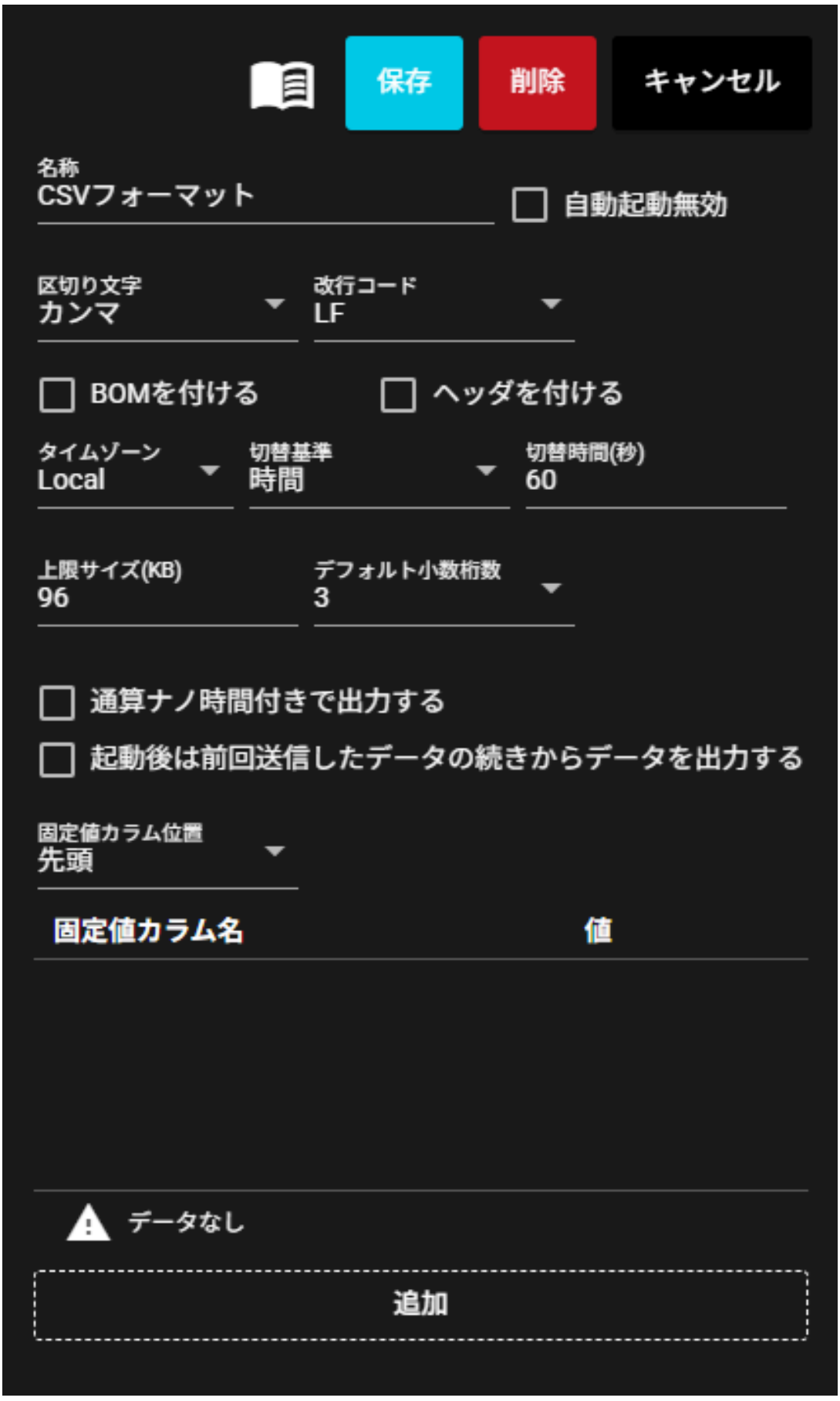
1. Editモードの画面で、左側のメニューで「Serializer」のインジケーターを押下し、シリアライザー一覧を表示させます。



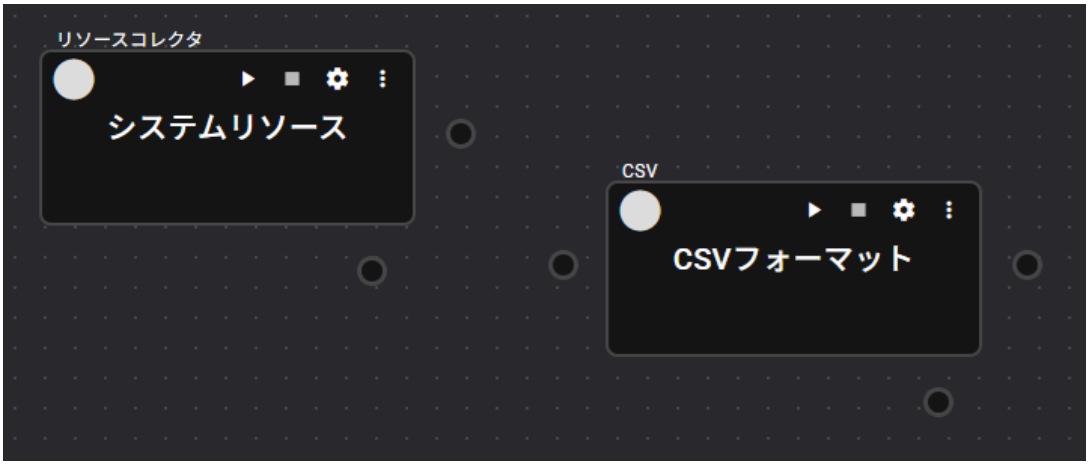
2. 「CSVシリアライザ」をドラッグし、パネル上にドロップします。



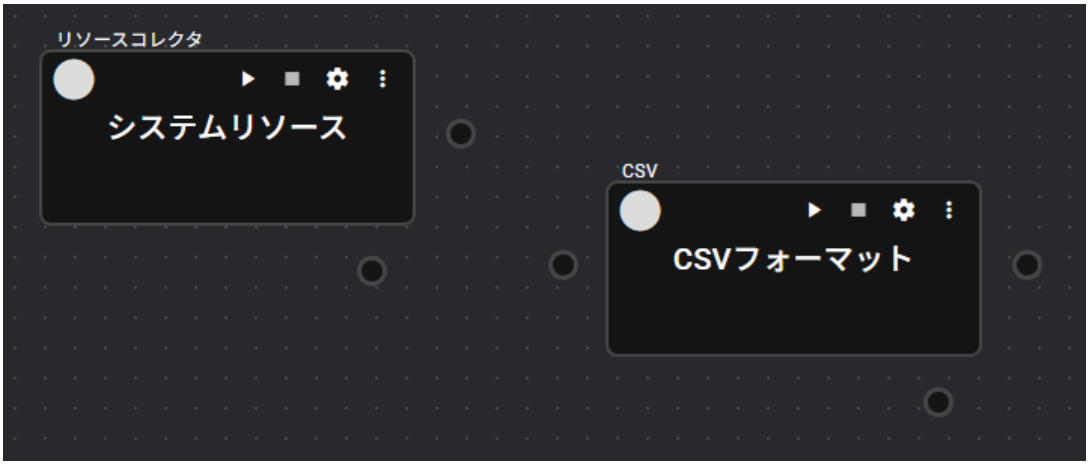
3. 設定ダイアログが自動で開きます。"名称"を任意で入力し、「保存」ボタンを押下します。
   名称例：CSVフォーマット



4. リソースコレクタの出力ポートをドラッグします。



5. CSVシリアライザの入力ポートでマウスのボタンを離します。



6. コンポーネントの間にフローリンクが追加されます。

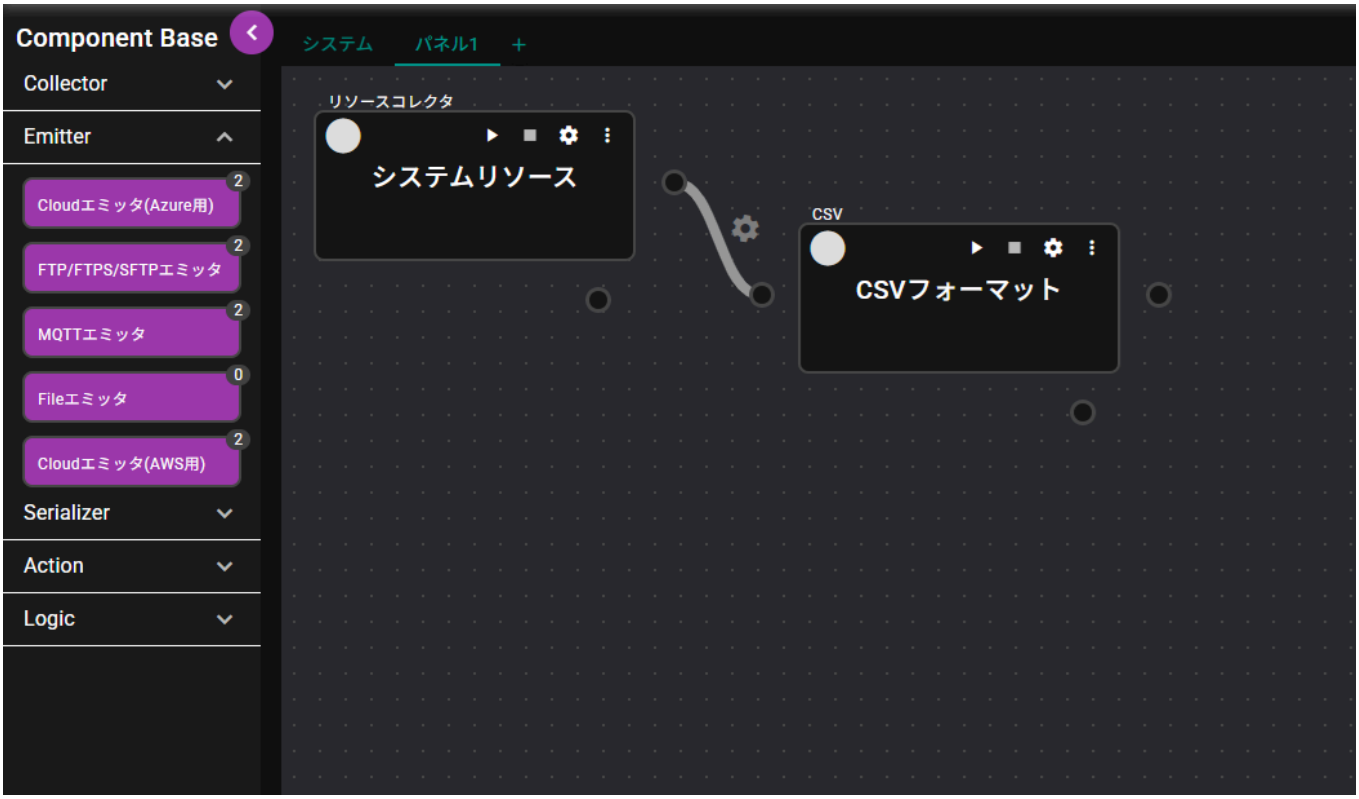


4.5.3 Fileエミッタの設定

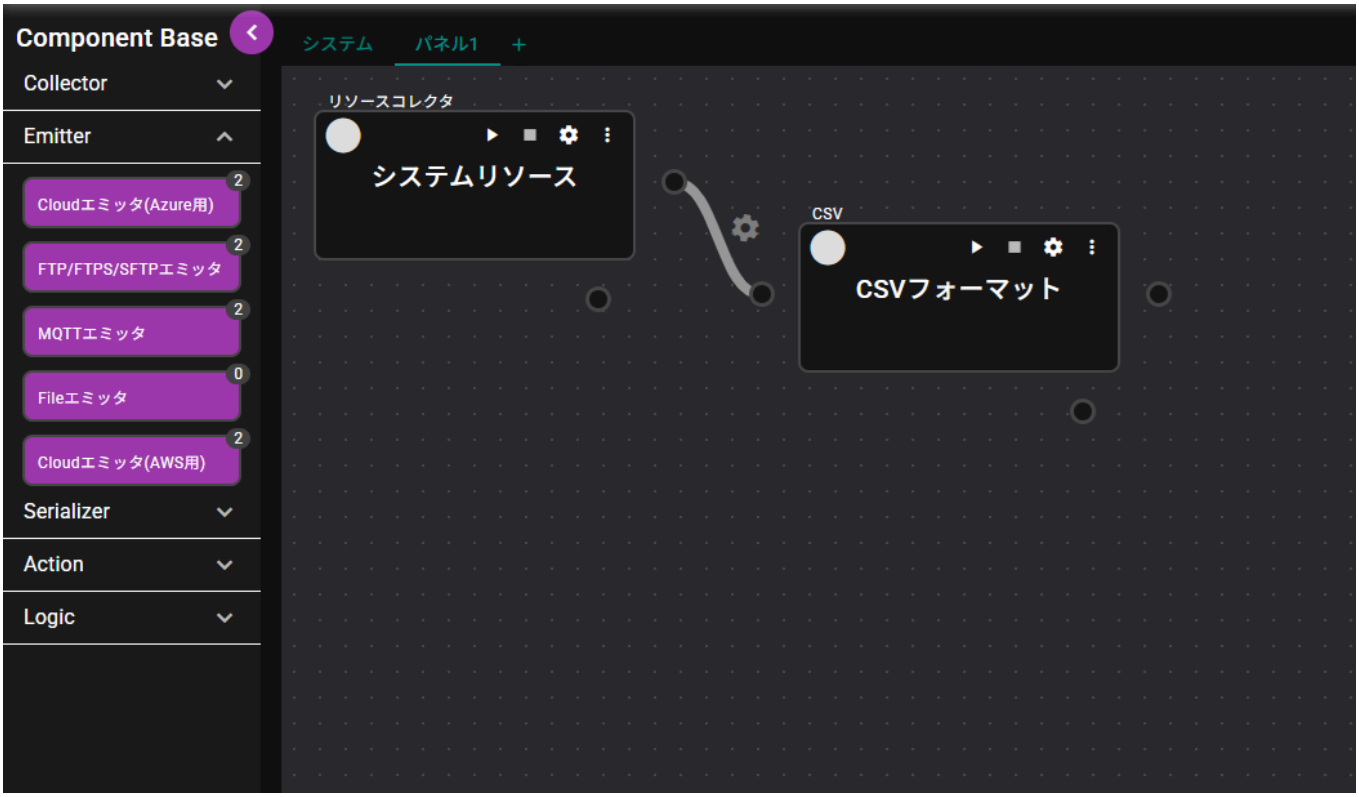
続けて"Fileエミッタ"を設定する方法を説明します。"Fileエミッタ"は、シリアライザによって加工されたデータを受け取り、ローカル環境にファイル出力するコンポーネントです。

次の手順で、Fileエミッタを設定することができます。

1. Editモードの画面で、左側のメニューで「Emitter」のインジケーターを押下し、エミッター覧を表示させます。



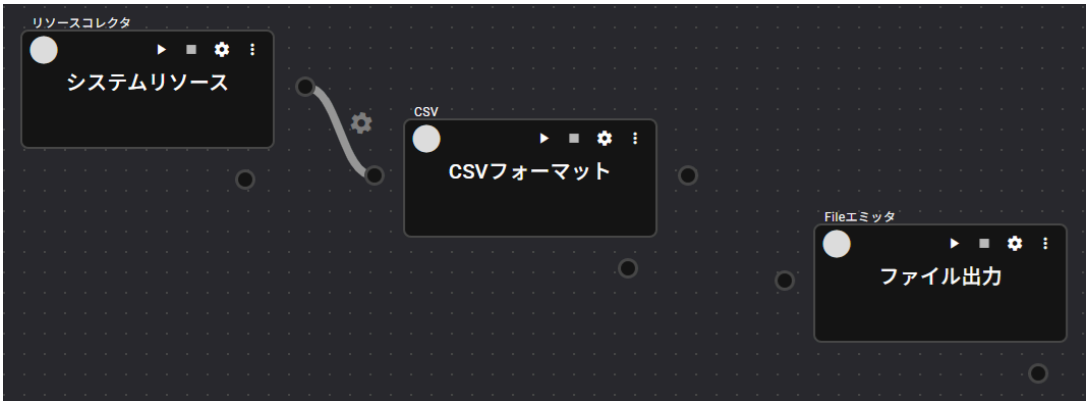
2. 「Fileエミッタ」をドラッグし、パネル上にドロップします。



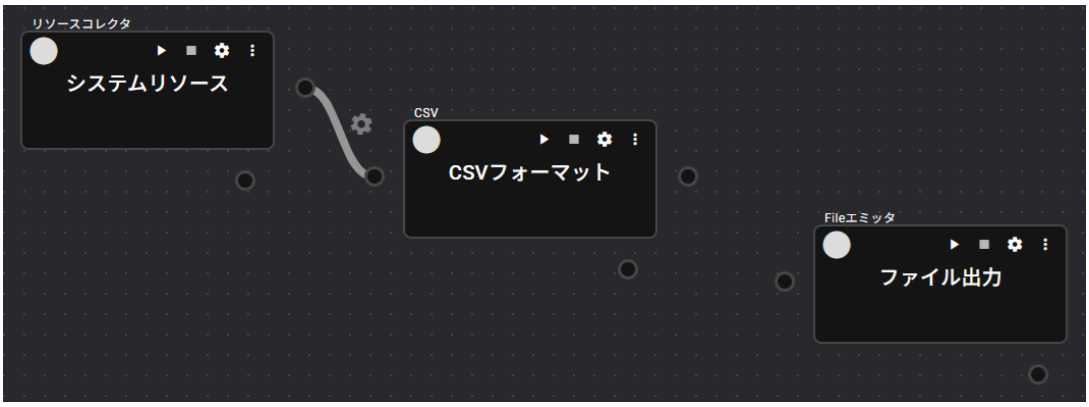
3. 設定ダイアログが自動で開きます。"名称"と"区分"を任意で入力し、「保存」ボタンを押下します。
   名称例：ファイル出力
   区分例：file1
   ※"区分"は半角のみ入力可能です。



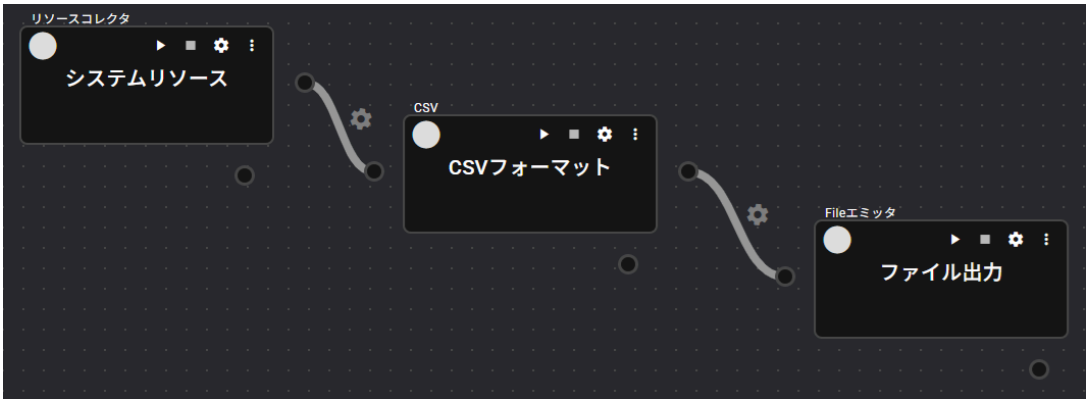
4. CSVシリアライザの出力ポートをドラッグします。



5. Fileエミッタの入力ポートでマウスのボタンを離します。



6. コンポーネントの間にフローリンクが追加されます。


リソースコレクタ
システムリソース
CSV
CSVフォーマット
Fileエミッタ
ファイル出力

4.6 収集サービスの実行

ここではコンポーネントを起動して、収集サービスを開始する方法を説明します。

また、収集サービスを開始した後、前項までに設定したコレクタ・シリアライザ・エミッタが正常に稼働している事を確認する方法と、設定をチューニングする方法を説明します。

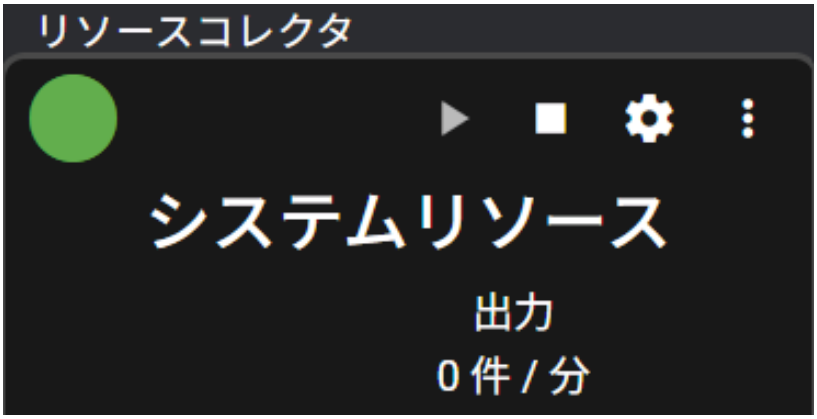
4.6.1 収集サービスの開始/停止

コンポーネントを起動する事で、収集サービスを開始する事ができます。 コンポーネントの起動は、コンポーネント単位、パネル全体、SpeeDBee Synapse全体の3つの範囲で行う事ができます。

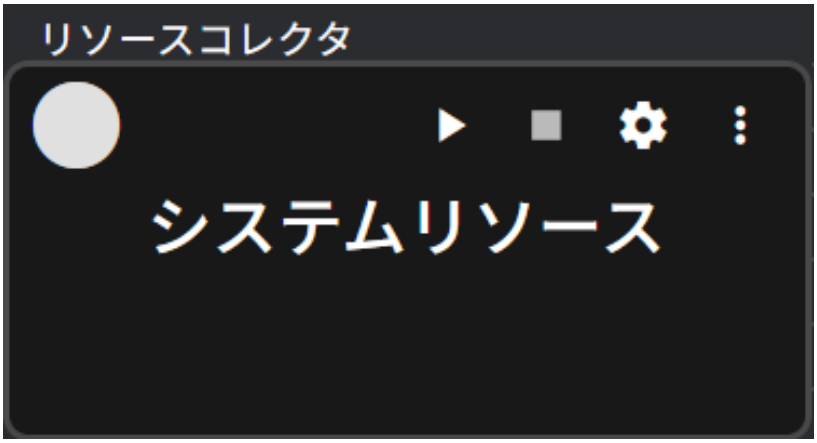
4.6.1.1 コンポーネントの起動/停止

次の手順で、一つのコンポーネントを起動/停止することができます。

1. コンポーネントの起動アイコンを押下すると、コンポーネントを起動する事ができます。 コンポーネントが起動すると、ステータスランプが緑になります。



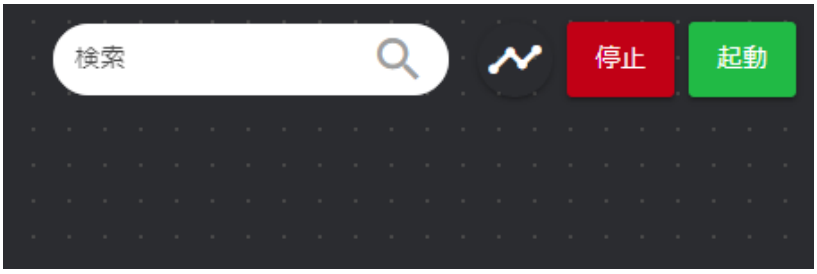
2. コンポーネントの停止アイコンを押下すると、コンポーネントを停止する事ができます。 コンポーネントが停止すると、ステータスランプが灰色になります。



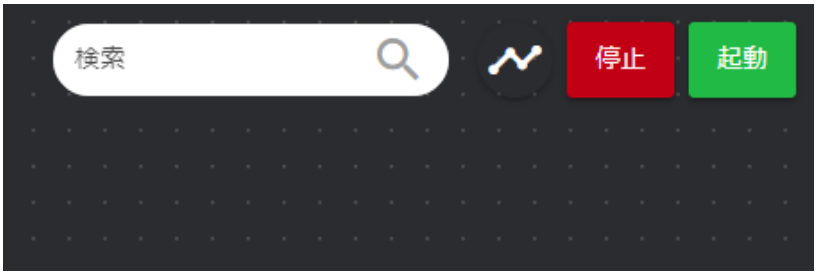
4.6.1.2 パネル上の全コンポーネントの起動/停止

次の手順で、一つのパネル上のコンポーネントを起動/停止することができます。

1. パネルの「起動」ボタンを押下すると、パネル上の「自動起動無効」がOFFのコンポーネントを全て起動する事ができます。



2. パネルの「停止」ボタンを押下すると、パネル上の全てのコンポーネントを停止する事ができます。



4.6.1.3 SpeeDBee Synapse上の全コンポーネントの起動/停止

次の手順で、SpeeDBee Synapse上のコンポーネントを起動/停止することができます。

1. ヘッダの「起動」ボタンを押下すると、SpeeDBee Synapse上の「自動起動無効」がOFFのコンポーネントを全て起動する事ができます。



2. ヘッダの「停止」ボタンを押下すると、SpeeDBee Synapse上の全てのコンポーネントを停止する事ができます。



4.6.2 動作確認

ここでは、リソースコレクタが収集したデータの確認方法、Fileエミッタが出力したファイルをダウンロードする方法について説明します。

リソースコレクタの動作確認

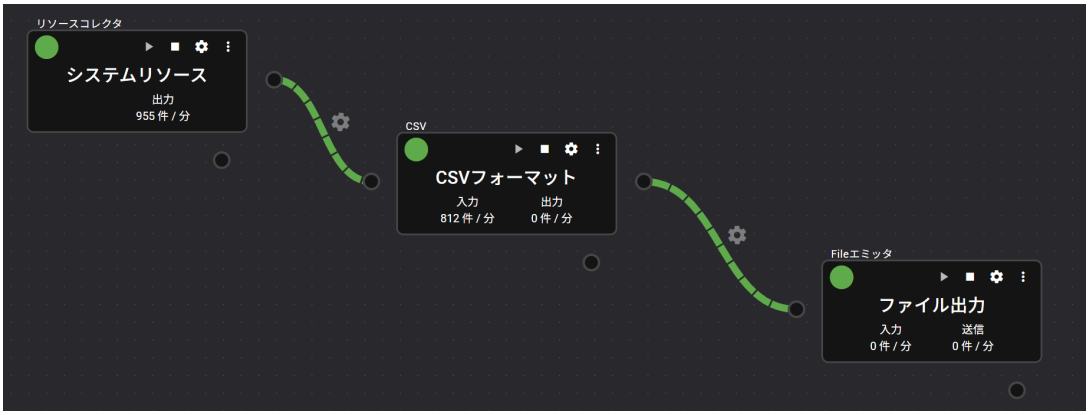
1. ヘッダのモード選択で"Run"を選択し、Runモードの画面を開きます。



2. パネル1を表示します。



3. リソースコレクタの出力ポートをクリックします。

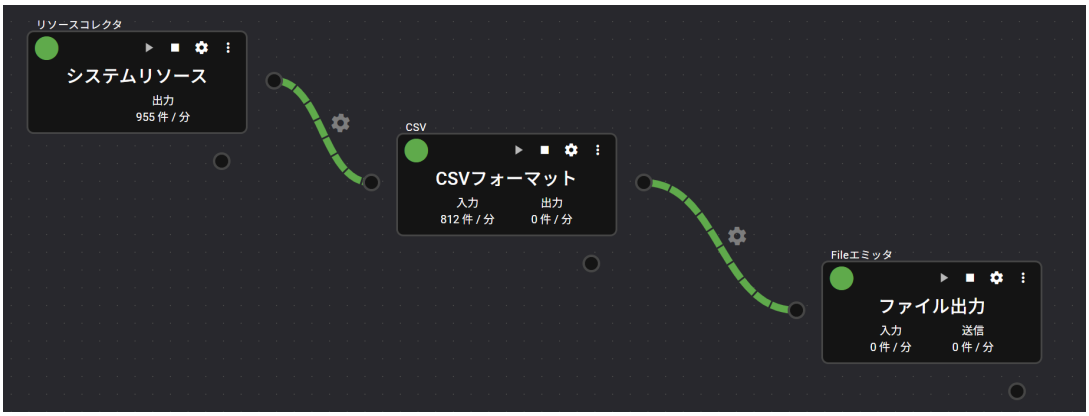


4. 画面下部のインジケータをクリックするとデータモニタが表示され、リソースコレクタが収集しているデータが確認できます。



Fileエミッタの動作確認

1. Fileエミッタの設定アイコンを押下します。



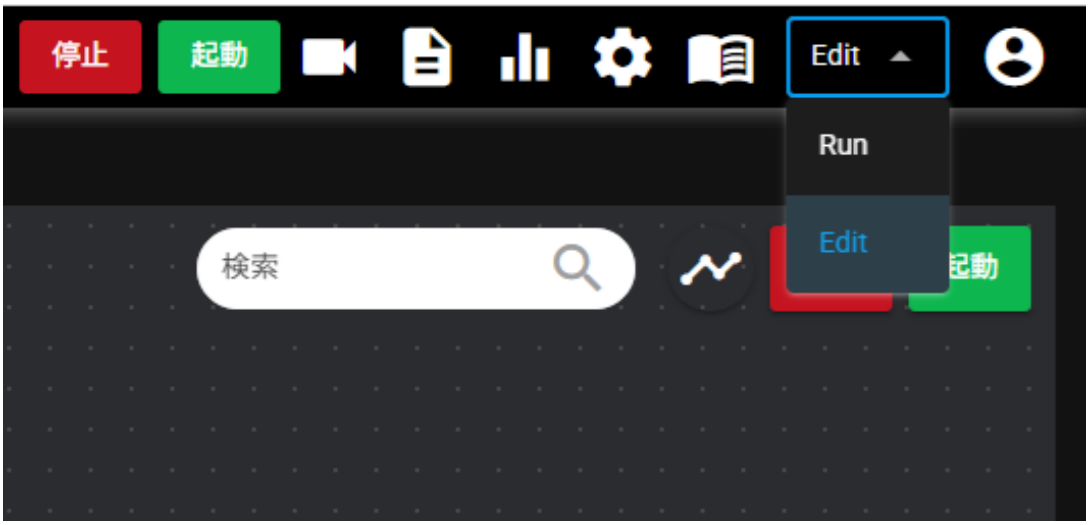
2. 時間範囲を入力し、「ダウンロード」を押下します。 指定した時間範囲の間にFileエミッタが出力したファイルが、zip形式でダウンロードされます。



4.6.3 設定チューニング

ここでは、コンポーネントの設定を編集する方法、コンポーネントを削除する方法について説明します。

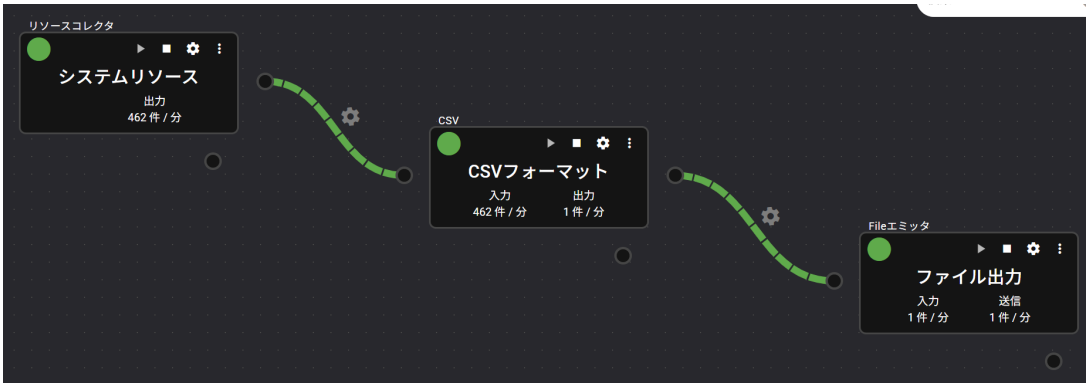
1. ヘッダのモード選択で"Edit mode"を選択し、Editモードの画面を開きます。



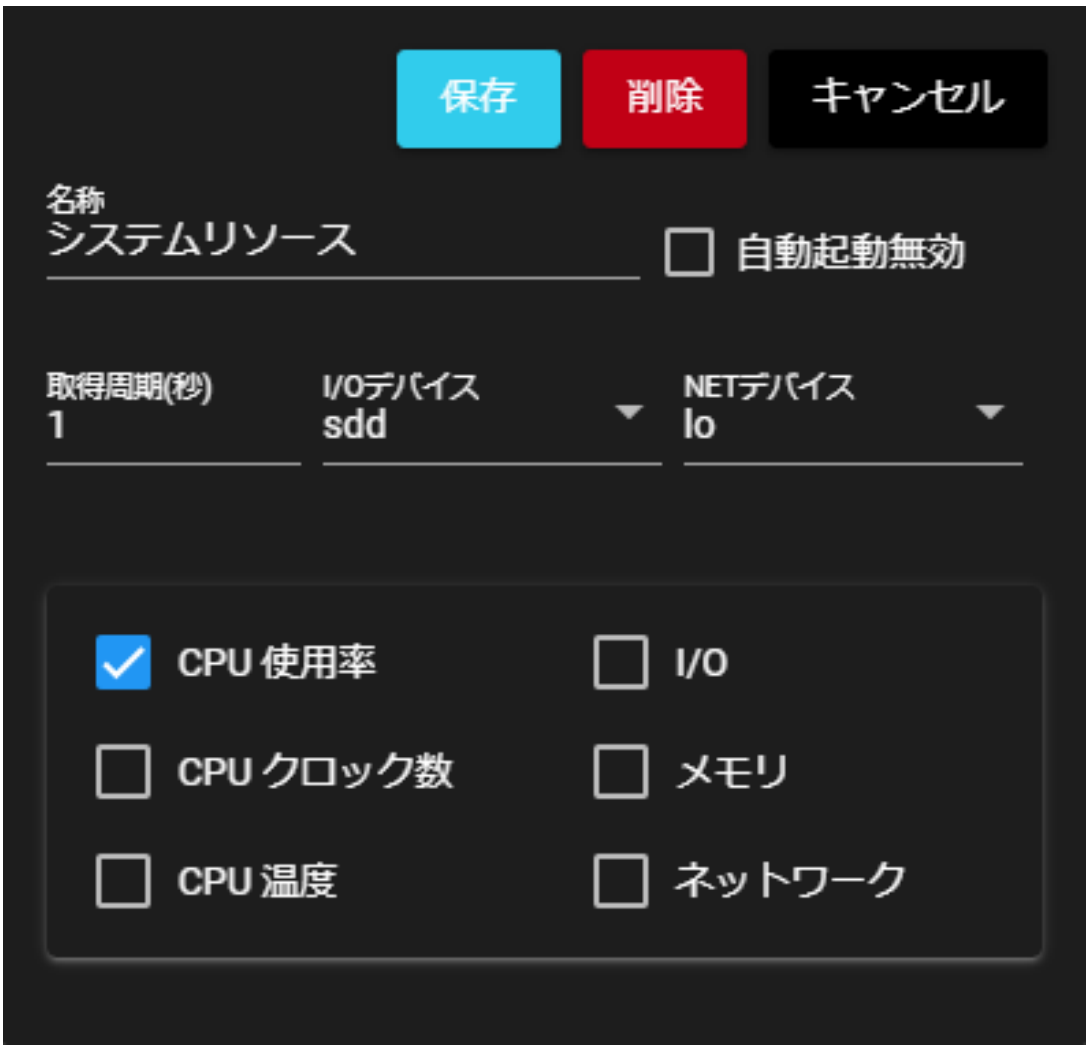
2. パネル1を開きます。



3. 設定を変更するコンポーネントの設定アイコンを押下します。



4. 設定ダイアログにて設定を変更し、「保存」を押下します。ここではリソースコレクタの画像を使用しています。例えば、メモリ使用量などを収集する場合、チェックをONにします。 また、コンポーネントを削除する場合は、「削除」を押下し、確認ダイアログで「はい」を選択します。



5. 設定を変更したコンポーネントを停止し、再度起動する事で、設定を収集サービスに反映させる事ができます。 コンポーネントの起動/停止については「収集サービスの開始/停止」をご覧ください。

4.6.4 収集サービスの終了

SpeeDBee Synapse上の全コンポーネントを停止することで、収集サービスは待機状態となります。 手順については、「SpeeDBee Synapse上の全コンポーネントの起動/停止」をご覧ください。

5 各種機能説明

5.1 各種機能説明

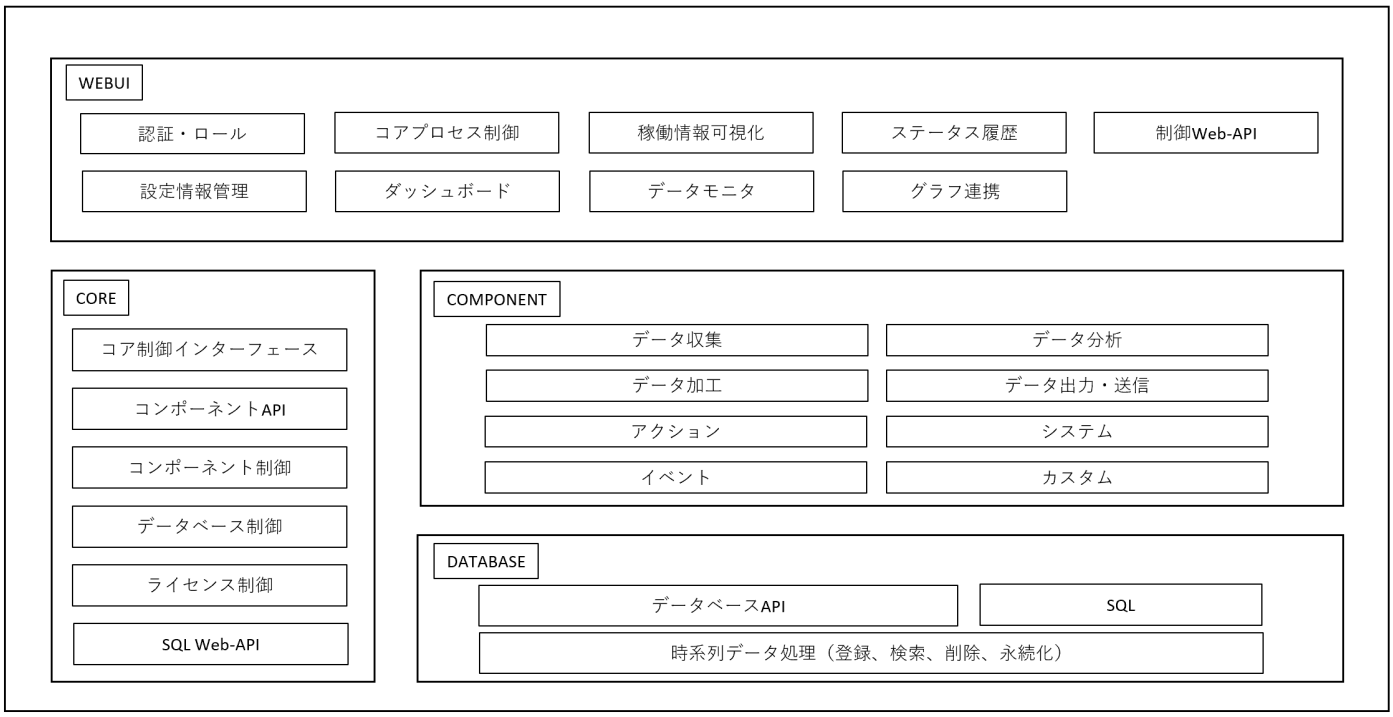
本章では、各種機能について説明します。

記載内容は次のとおりです。

1. エンジン機能の全体像
2. フローエディタ
3. コレクタ
4. シリアライザ
5. エミッタ
6. ロジック（イベント）
7. ロジック（リアルタイム分析/データ操作）
8. アクション
9. 開発用コンポーネント
10. システムコンポーネント
11. コンポーネント共通の設定項目
12. ステータスタイムライン
13. 各種ログ
14. DBクエリ（SQL）
15. 演算式・条件式
16. 設定情報の管理(プロジェクト)

5.2 エンジン機能の全体像

ここでは、エンジン機能の全体像について説明します。 この節は、内部の処理について、詳しく知りたい場合にご参照ください。 次の図は、SpeeDBee Synapseの構成となります。



5.2.1 機能一覧

ここでは、構成図に示した内容について解説します。

5.2.1.1 コア

コアは、SpeeDBee Synapseの中心で、各機能の制御を行っている部分です。 コアは次の機能を提供します。

項目	説明
コア制御インターフェース	SpeeDBee Synapseの各機能を制御します。WEBUIからの要求を受け付けて処理します。ユーザーが直接使用することはありません。
コンポーネントAPI	コンポーネントが使用するAPIを提供します。コンポーネント開発者は、これらのAPIを利用してプログラミングを行う必要があります。
コンポーネント制御	コンポーネント(データの収集、加工、分析、送信などを実現するモジュール群)の処理を制御します。コアが内部で自動的に処理します。

項目	説明
データベース制御	SpeeDBee Synapseに搭載されている時系列データベースを制御します。コアが内部で自動制御します。
ライセンス制御	ライセンスに関する制御を行います。正しいライセンスのもとSpeeDBee Synapseが利用できるよう制御します。
SQL Web-API	SpeeDBee Synapseに搭載されている時系列データベースに対し、SQLを実行できるWeb-APIを提供します。 詳細は「DBクエリ（SQL）」、「DBクエリサービス」をご覧ください。

5.2.1.2 コンポーネント

コンポーネントは、SpeeDBee Synapseにおいて、任意の処理を行うモジュール群です。 コンポーネントを使用することで、以下のことが可能です。

項目	説明
データ収集	特定の対象からデータを収集することができます。 詳細は「コレクタ」 をご覧ください。
データ分析	収集したデータを分析することができます。 詳細は「ロジック（リアルタイム分析/データ操作）」をご覧ください。
イベント	収集したデータに応じたイベントを処理します。 詳細は「ロジック（イベント）」をご覧ください。
データ加工	データを特定のフォーマットに加工することができます。 詳細は「シリアライザ」をご覧ください。
データ出力・送信	データを特定の対象に出力・送信することができます。 詳細は「エミッタ」をご覧ください。
アクション	任意のアクションを実行します。例えば、OSコマンドの実行やメール送信などが可能です。詳細は「アクション」をご覧ください。
システム	システムに関連する機能を提供します。初期状態で配置されており、ユーザーが新しいインスタンスを追加することはできません。詳細は「システムコンポーネント」をご覧ください。

項目	説明
カスタム	ユーザー自身でコンポーネントを開発することができます。 詳細は「カスタムコンポーネントの作成・利用」の「概要」をご覧ください。

5.2.1.3 WEBUI

WEBUIは、ユーザーが操作するSpeeDBee Synapseの画面を提供します。 WEBUIは、次の機能を提供します。

項目	説明
認証・ロール	SpeeDBee Synapseを正当なユーザーが利用できるよう、認証機能を提供します。また、ユーザーの権限に合わせたロール（ロール）を割り当てることができます。 詳細は、「ユーザー管理」をご覧ください。
コアプロセス制御	コアプロセスの制御を行います。ユーザーの操作およびコアプロセスの異常時に、プロセス制御を行います。
ダッシュボード	データ収集フローを視覚的に表示し、インタラクティブにカスタマイズできるインターフェースです。詳細は、「パネル」をご覧ください。
設定情報管理	SpeeDBee Synapseの設定情報を管理することができます。ユーザーの入力した情報は、設定ファイルとして管理されます。
稼働情報可視化	SpeeDBee Synapseおよびコンポーネントの稼働状態を表示します。 詳細は、「ステータス一覧」をご覧ください。
ステータス履歴	コンポーネントの稼働状態の履歴を表示します。 詳細は、「ステータスタイムライン」をご覧ください。
制御Web-API	SpeeDBee Synapseを制御するWeb-APIを提供します。 詳細は、「制御プログラムとの連携」をご覧ください。
データモニタ	収集したデータをモニタで閲覧することができます。 詳細は、「データモニタ」をご覧ください。
グラフ連携	収集したデータをもとに、グラフを作成することができます。 詳細は、「grafana連携」をご覧ください。

5.2.1.4 データベース

SpeeDBee Synapseには、時系列データベースが搭載されています。 時系列データベースは、次の機能を提供します。

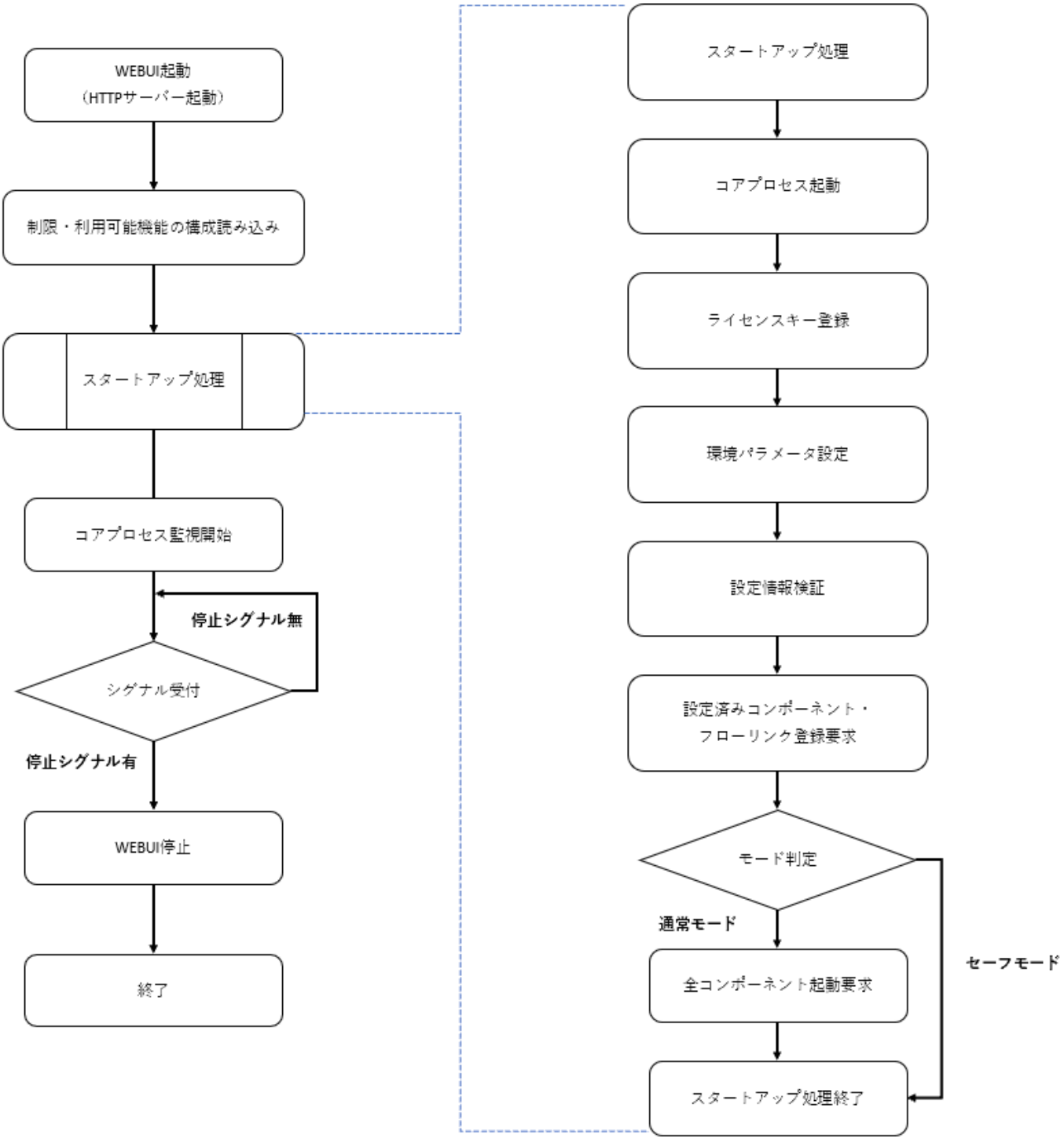
項目	説明
データベースAPI	時系列データベースを操作するAPIを提供します。SpeeDBee Synapseにおいて、ユーザーがこのAPIを意識する必要はありません。
SQL	時系列データベースに対して、SQLを実行することができます。 詳細は「DBクエリ（SQL）」をご覧ください。
時系列データ処理 (登録、検索、削除、永続化)	収集したデータを登録、検索、削除、永続化することができます。コアによって、これらは制御されます。

5.2.2 動作仕様

本項では、SpeeDBee Synapseの動作仕様について説明します。

5.2.2.1 基本動作

SpeeDBee Synapseの基本動作仕様は次の通りです。



上記フロー図で行われる処理の詳細を下表に示します。

- 基本動作

処理	説明
WEBUI起動	SpeeDBee SynapseのWEBUIを起動します。内部的にはHTTPサーバーが動作します。

処理	説明
制限・利用可能機能の構成読み込み	構成情報を読み込みます。有効な機能や各種制限値に関する情報が対象です。 基本的には、ユーザーが準備・設定などを行うものではありません。
スタートアップ処理	コアプロセスの起動およびSpeeDBee Synapseを使用可能にするための処理を行います。 詳細は後述します。
コアプロセス監視開始	コアプロセスの監視処理を起動します。コアプロセスの状態監視を行い、異常発生時には、コアプロセスの再起動を行います。異常が続く場合は、コアプロセスを起動せずにセーフモードへ遷移します。
シグナル受付	OSからのシグナルを受け付けます。停止シグナルを受信すると、終了処理へ移行します。
WEBUI停止	WEBUIを停止します。コアプロセス監視などの終了処理や各処理の後始末を行います。

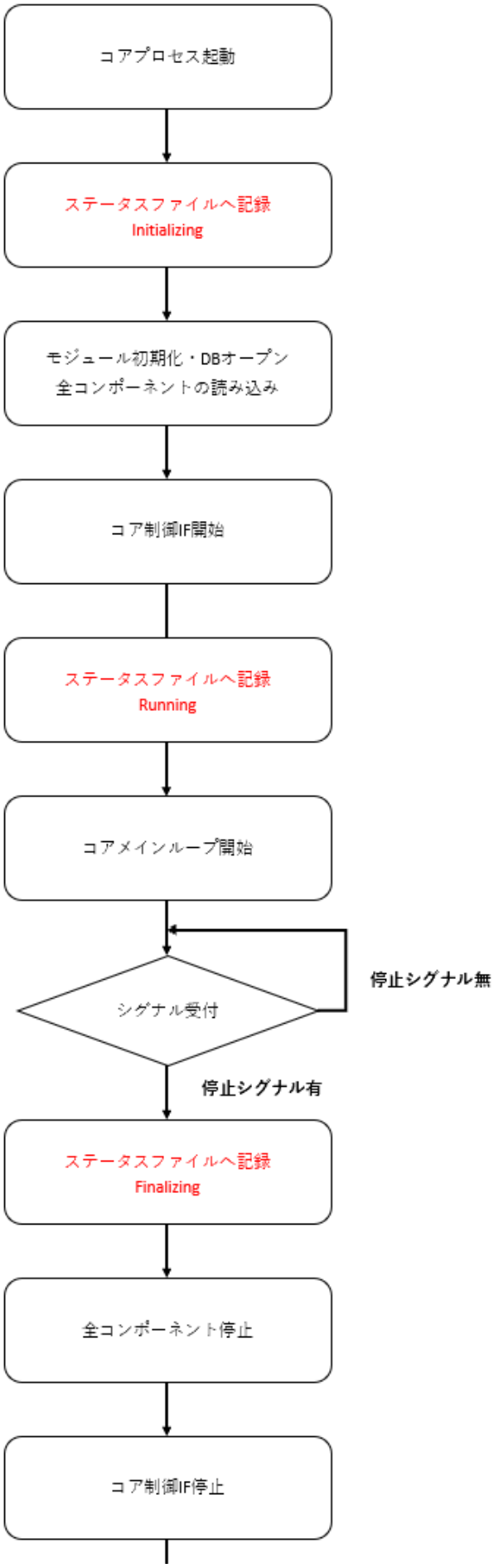
- スタートアップ処理（詳細）

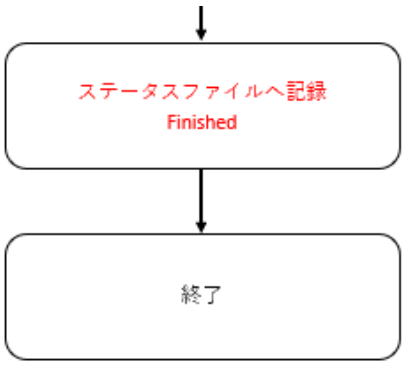
処理	説明
コアプロセス起動	SpeeDBee Synapseのデータエンジンとなるコアプロセス（バックグランドプロセス）を起動します。詳細は後述します。
ライセンスキー登録	SpeeDBee Synapseのライセンスキー登録します。WEBUIで、設定されたライセンスキーが適用され、使える機能などが制御されます。
環境パラメータ設定	WEBUIで設定されたデータベースのパラメータをコアプロセスへ適用します。
設定情報検証	WEBUIで設定されているコンポーネント等の情報を検証します。設定の矛盾がないか確認して整合性を保持します。
設定済みコンポーネント・フローリンク登録要求	ユーザーの設定したコンポーネントおよびフローリンクをコアプロセスへ適用します。
モード判定	モード判定を行います。モードには次の種類があります。 ・通常モード （スタートアップ処理内でコンポーネントの起動までを行う通常のモードです） ・セーフモード （異常時に全コンポーネントが起動できない場合のモードです）

処理	説明
全コンポーネント起動要求	登録済みの全コンポーネント（自動起動無効のものを除く）を起動します。これによりデータ収集・加工・出力・送信等の処理が実行される状態となります。セーフモードではこの処理はスキップされます。

コアプロセス起動の処理詳細は次の通りです。

- コアプロセス起動（詳細）


コアプロセス起動
ステータスファイルへ記録
Initializing
モジュール初期化・DBオープン
全コンポーネントの読み込み
コア制御IF開始
ステータスファイルへ記録
Running
コアメインループ開始
シグナル受付
停止シグナル無
停止シグナル有
ステータスファイルへ記録
Finalizing
全コンポーネント停止
コア制御IF停止

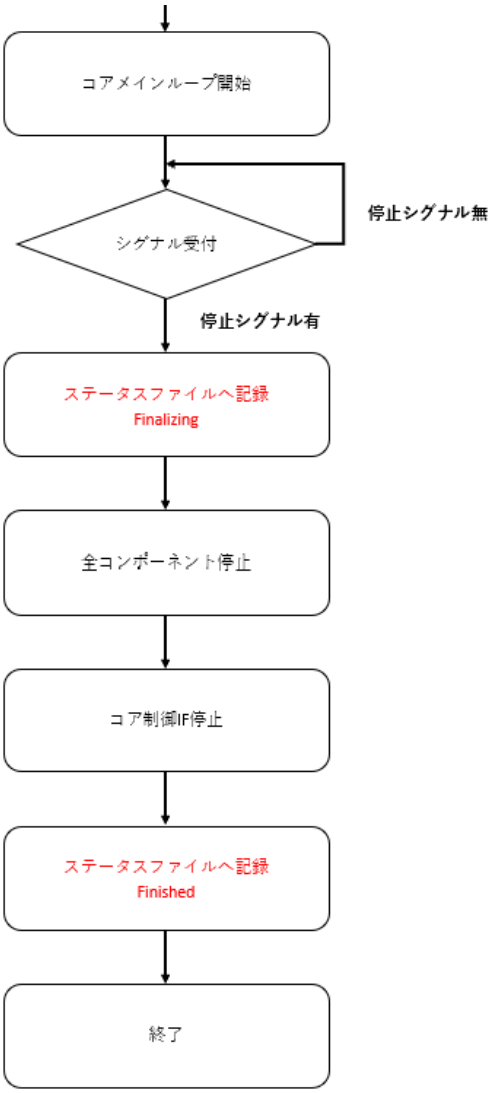


処理	説明
ステータスファイルへ記録	コアの起動・停止状態記録用のステータスファイルへ記録を書き込みます。 この段階では、ステータスは「Initializing」となります。
モジュール初期化・DBオープン 全コンポーネントの読み込み	コアモジュールの初期化を行います。初期内で、データベースのオープン、全コンポーネントの読み込みを行います。 この段階で、問題のあるコンポーネントは読み込みされません。配置はされているが、初期化が実施できないものが読み込み対象外となります。
コア制御IF開始	コアプロセスを制御するためのインターフェースを有効にします。WEBUIからの要求を受信できる状態となります。
ステータスファイルへ記録	コアの起動・停止状態記録用のステータスファイルへ記録を書き込みます。この段階では、ステータスは「Running」となります。このステータスが記録されたことで正常に起動したことが判別できます。
コアメインループ開始	コアのメインループが開始されます。コア制御IFへの要求に応じて、各種処理を行います。
シグナル受付	OSからのシグナルを受け付けます。停止シグナルを受信すると、終了処理へ移行します。
ステータスファイルへ記録	コアの起動・停止状態記録用のステータスファイルへ記録を書き込みます。 この段階では、ステータスは「Finalizing」となります。このステータスが記録されたことで終了処理が行われたことが判別できます。ただし、終了処理が完全に終了した後に次のステータスが記録されるので、この状態のままステータスが残っている場合は異常と判断されます。
全コンポーネント停止	起動中の全コンポーネントを停止、破棄を行います。
コア制御IF停止	コア制御IFを停止します。外部からの要求は受付できなくなります。

処理	説明
ステータスファイルへ記録	コアの起動・停止状態記録用のステータスファイルへ記録を書き込みます。 この段階では、ステータスは「Finished」となります。このステータスが記録されたことで終了処理が正常に完了したことが判別できます。次回コアプロセス起動時には、このステータスを参照して、前回起動時の異常診断を行います。「Finished」以外のステータスが記録されたことを検知すると、コアインターナルコンポーネントは異常状態をステータス登録します。異常があった場合の情報は、ステータスタイムラインで参照できます。

5.2.2.2 正常停止時

正常停止時の動作について解説します。 下図は、コアプロセス起動（詳細）の抜粋となります。赤枠部分が重要な処理となります。



正常な停止処理が実行されることで、ステータスファイルへ正しい記録が行われます。 ステータスファイルへ「Finished」が記録されている状態が正しい停止が行われた事を示しています。 詳細は、コアプロセス起動（詳細）を参照して下さい。

5.2.2.3 異常停止時(電断など)

異常時停止時の動作について解説します。 異常時とは、SpeeDBee Synapse稼働中に電断によりホストマシンが停止したなどを指します。

- コアプロセスの異常停止診断

正常停止時の説明へ記載したように、稼働途中でSpeeDBee Synapseが停止した場合は、ステータスファイルへ正しい記録が行われません。次回コアプロセス起動時に、ステータスファイルの診断を行い、正常停止が確認できない場合は、コアインターナルのエラーステータスとして状態が登録されます。エラー情報については、ステータスタイムラインで参照できます。

- データベース診断

データベース書き込み中に、SpeeDBee Synapseが異常停止を起こした場合、タイミングによっては データベース内のデータ欠損が起こる可能性があります。データ欠損とは、メモリ上に、保持していたデータをストレージへ書き出している際中に、電断などにより正常に書き込みが終了せず一部のデータが失わることを指します。この診断も、次回コアプロセス起動時に内部で診断されます。診断結果で、異常が見つかればコアインターナルのエラーステータスとして状態が登録されます。エラー情報については、ステータスタイムラインで参照できます。

5.2.2.4 データ収集に関連する動作仕様について

SpeeDBee Synapseでは、コンポーネントのデータ収集を起点に処理を行います。ただし、設定や操作のタイミングによって注意すべき動作仕様があるため、以下に記載します。

コンポーネントの設定変更が適用されるタイミングについて

設定を変更した場合、それが即座に適用されるものと、一度コンポーネントを停止した後、再起動後に適用されるものがあります。また、適用タイミングが特殊な設定があります。

1.コンポーネント再起動後に、新しい設定が適用されるもの

- コンポーネントそれぞれの固有設定

コンポーネント固有の設定項目は、再起動後に反映されます。
例えば、PLCコレクタなどでアドレスを追加して、設定を保存しても、そのままでは設定は適用されません。一度、PLCコレクタを停止して、再び起動することで新しい設定が反映された状態で動作します。

2.設定変更後、即座に新しい設定が適用されるもの

- フローリンクのフィルタ
- フローリンクの連結/削除
- 入力ポート設定

3.設定変更後、特殊なタイミングで新しい設定が適用されるもの

- 出力ポート

出力ポートに関しては、設定変更後にカラムが作成されることで、新しい設定が適用されます。
これは起動時に、カラムを作成するコンポーネントであれば再起動されるまで適用されませんが、正規表現などで設定をしている場合かつ特殊なタイミングでカラムを生成するコンポーネントの場合は、途中で反映される可能性があります。
例えば、MQTTコレクタなどはデータを受信して、始めてカラムを生成する事があります。この場合に、出力ポートの設定を正規表現などで指定していると、新しい設定が適用される可能性があります。
※確実に設定を適用したい場合は、該当コンポーネントを再起動して下さい。

フローリンク連結/削除時のデータの流れについて

データ収集を行うコンポーネントを起動後、別のコンポーネントへフローリンクを連結しても連結する前のデータは収集されません。
また、フローリンクを途中で削除すると、そのタイミングで連結先に該当データが流れなくなり、該当コンポーネントの処理するデータが変化します。これは設定を変更しなくても、流れるデータが変化するため動作しているコンポーネントの挙動が変化します。

例えば、PLCコレクタとModbus TCPコレクタをJSONシリアライザへフローリンクをそれぞれ連結しているとします。この時、PLCコレクタとJSONシリアライザのフローリンクを削除すると、JSONシリアライザが参照するデータが、途中でModbus TCPコレクタのデータのみとなります。

データの永続化について

データ収集するコンポーネントが取得したデータは、デフォルトではDBへ永続化しません。DBへ永続化すると、GrafanaやDBクエリサービスを使用して、永続化されている期間の過去データが参照できます。永続化していない場合は、現在メモリ上に保持しているデータのみが参照可能となります。

これら過去データ参照をするデータを永続化したい場合は、「出力ポート」の「永続化の設定」の設定を行って下さい。

データ参照時の遅延時間について

入力ポートには、「遅延時間」という設定があります。
これはデータ参照時に、取得するデータの時間範囲を現在時刻に対して何秒遅らせて行うか設定するものです。
データ収集とデータ参照が非同期に並列で処理されるため、データ収集が完全に完了していないデータを取りこぼすのを防ぐ設定です。
データ収集に時間がかかるコンポーネントのデータを扱っている場合は、この遅延時間を調整してください。
コンポーネントによっては、データ収集の時間保証を行い、この遅延時間より短い間隔でデータ参照できる場合もあります。これは、SpeeDBee Synapse内部で自動で行われます。

遅延時間の設定を行うには、「入力ポート」を参照して下さい。

コンポーネント停止時のデータについて

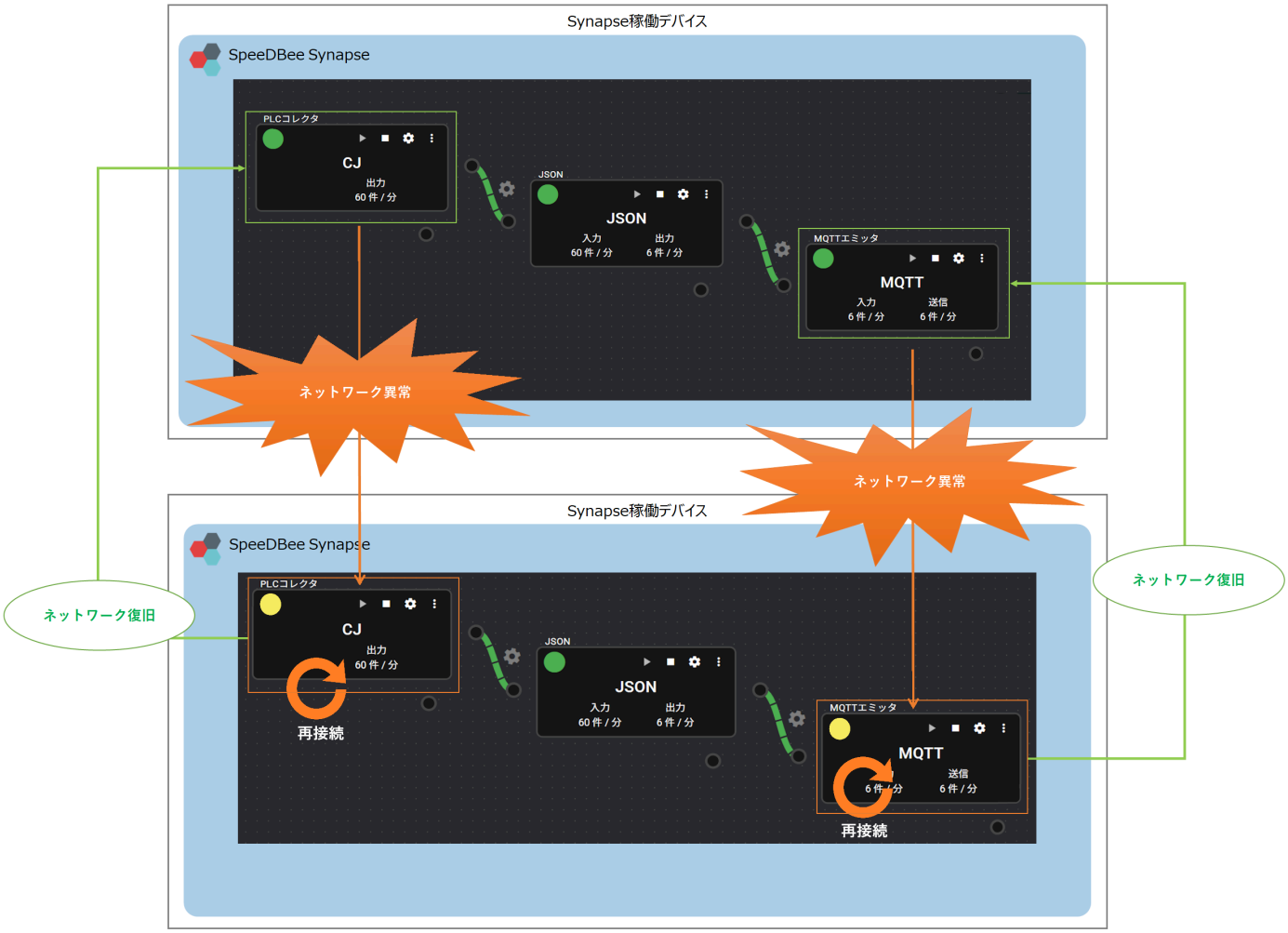
コンポーネントを停止すると、その操作直近のデータは完全保証されません。

例えば任意のコレクタからシリアライザへフローリンクを連結している場合に、シリアライザの停止を行うと、シリアライザはなるべく早く停止するよう動作します。従って、停止中となったシリアライザは、データの参照を実行しないため、データは保証されません。
処理したいデータが流れている間、不用意にコンポーネントを停止しないで下さい。

5.2.2.5 ネットワーク異常、復旧時の動作仕様について

SpeeDBee Synapseのネットワーク異常時、そして、ネットワーク異常からの復旧時に関する動作仕様について説明します。

基本的には、通信を使用するコンポーネントはネットワーク異常が発生した場合、通信の再接続を行います。



1. データ収集を行うコンポーネント
   通信を使用したコレクタ系のコンポーネントは、データ収集中にネットワーク異常が発生した場合は、通信の再接続を試行します。 通信に異常がある間は、ネットワーク異常状態となりステータスランプが警告状態となります。 ネットワーク異常が発生している間のデータは収集できません。稼働マシンのネットワークが復旧し、再接続に成功すると、通常のデータ収集を再開します。
   該当するコンポーネントは、次のとおりです。
   - PLCコレクタ
   - Modbus TCPコレクタ
   - EtherNet/IPコレクタ
   - VIDEOコレクタ
   - MQTTコレクタ

2. 収集したデータを送信するコンポーネント
   通信を使用したアクション系コンポーネント、エミッタ系コンポーネントは、動作中にネットワーク異常が発生した場合は、通信の再接続を試行します。 通信に異常がある間は、ネットワーク異常状態となりステータスランプが警告状態となります。

   アクション系コンポーネント
   ネットワーク異常が発生している間のデータは送信できません。
   該当するコンポーネントは、次のとおりです。

   - PLC書き込み
   - Modbus書き込み
   - EtherNet/IP書き込み
   - メール送信

   エミッタ系コンポーネント
   ネットワーク異常が発生している間のデータは送信できません。 ただし、リトライモードの設定により送信できていなかったデータの再送を行う設定も可能です。
   該当するコンポーネントは、次のとおりです。

   - Cloudエミッタ(Azure)
   - Cloudエミッタ(AWS)
   - FTP/FTPS/SFTPエミッタ
   - MQTTエミッタ

FTP/FTPS/SFTPエミッタは、常に未送信のデータは常に再送を試みます。 その他エミッタのリトライモードについては、該当のコンポーネントの機能説明を参照してください。
なお、ユーザの作成したカスタムコンポーネントやシェルコマンドで通信処理を行う場合の動作は、実装次第となります。
ネットワーク異常時に、通信の再接続を行う必要がある場合は、それぞれの実装時に、再接続する処理を実装する必要があります。

5.3 フローエディタ

ここでは、フローエディタについて説明します。フローエディタは、SpeeDBee Synapseの操作画面です。

5.3.1 モードについて

フローエディタには2種類のモードが存在します。

モード	説明
Edit Mode	データ収集フローの設定などを行います。権限がシステム管理者・管理者のユーザーのみ利用できます。
Run Mode	データ収集フローの運用を行います。すべての権限のユーザーが利用できます。

5.3.2 画面項目

ここでは、フローエディタの画面項目について説明します。

5.3.2.1 Edit Mode

Edit Modeのフローエディタの画面項目は次の通りです。



Edit Modeの画面は、 ① ヘッダ、 ② コンポーネントベース、 ③ パネル、 ④ データモニタ で構成されています。

① ヘッダ



番号	項目	説明
①	使用ポイント数	登録されたライセンスのポイント数と、使用されたポイント数が表示されます。
②	現在のプロジェクト	現在選択中のプロジェクトが表示されます。 ※詳細は「設定情報の管理(プロジェクト)」をご覧ください。
③	停止ボタン	クリックすると、SpeeDBee Synapseに登録されたすべてのコンポーネントが停止します。
④	起動ボタン	クリックすると、SpeeDBee Synapseに登録されている、「自動起動無効」がOFFのコンポーネントがすべて起動します。
⑤	ビューワーアイコン	クリックすると、ビューワー画面に遷移します。 ※拡張パッケージ「i-Trigger」を適用した場合のみ表示されます。詳細は「i-Trigger（拡張パッケージ）」をご覧ください。

番号	項目	説明
⑥	ログアイコン	クリックすると、SpeeDBee Synapseの操作ログ画面が表示されます。 ※詳細は「各種ログ」をご覧ください。
⑦	グラフアイコン	クリックすると、Grafanaの画面が表示されます。 ※詳細は「Grafanaとの連携」をご覧ください。
⑧	設定メニューアイコン	クリックすると、設定メニューが表示されます。 ※詳細は「設定メニュー」をご覧ください。
⑨	マニュアル	クリックすると、マニュアル画面が表示されます。
⑩	モード選択	フローエディタの表示モードを、次の中から選択できます。 ・Edit Mode ・Run Mode ※権限がシステム管理者・管理者のユーザーでログインした場合のみ表示されます。
⑪	ユーザー管理アイコン	クリックすると、ユーザー管理のメニューが表示されます。 ※詳細は「ユーザー管理」をご覧ください。

② コンポーネントベース



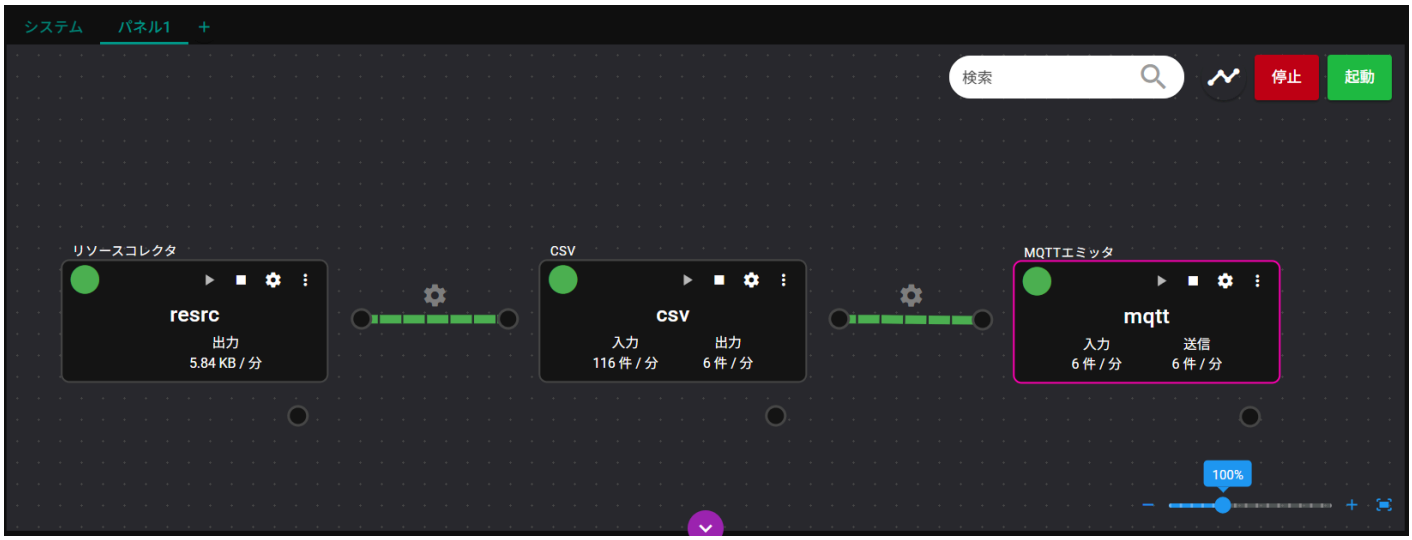
番号	項目	説明
①	開閉アイコン	クリックすることで、コンポーネントベースの表示・非表示を切り替えることができます。
②	インジケーター	クリックすることで、各項目の表示・非表示を切り替えることができます。
③	コンポーネントベース	パネルに登録することができるコンポーネントです。中央に、ベース名が表示されます。右上に、登録時に消費されるポイント数が表示されます。 また、マウスポインタを合わせるとコンポーネントの説明、入出力ポートの有無がさらに表示されます。

- コンポーネントベースには次の種類があります。

種類	リンク
Collector	詳細は「コレクタ」をご覧ください。

種類	リンク
Serializer	詳細は「シリアライザ」をご覧ください。
Emitter	詳細は「エミッタ」をご覧ください。
Logic	詳細は「ロジック（イベント）」、「ロジック（リアルタイム分析/データ操作）」をご覧ください。
Action	詳細は「アクション」をご覧ください。
Custom	詳細は「カスタムコンポーネントの作成・利用」の「概要」をご覧ください。 ※カスタムコンポーネントが追加されている場合のみ表示されます。

③ パネル



番号	項目	説明
①	タブ	表示するパネルを切り替えるための見出しです。
②	パネル追加アイコン	クリックすると、新しいパネルを追加することができます。
③	コンポーネント検索ボックス	コンポーネントの位置を検索できるボックスです。
④	ステータスタイムライン	クリックすると、コンポーネントのステータスを時系列順に並べた画面が表示されます。 ※詳細は「ステータスタイムライン」をご覧ください。

番号	項目	説明
⑤	停止ボタン	クリックすると、パネル上のすべてのコンポーネントが停止します。
⑥	起動ボタン	クリックすると、パネル上の、「自動起動無効」がOFFのコンポーネントがすべて起動します。
⑦	コンポーネント	パネルに登録されているコンポーネントです。
⑧	フローリンク	パネルに登録されているフローリンクです。 ※フローリンクは、2つのコンポーネントをつなぎ、コンポーネント間にデータを流すオブジェクトです。詳細は「フローリンク」をご覧ください。
⑨	表示倍率	パネルの表示倍率を変更できます。

- 詳細は「パネル」をご覧ください。
- "システム"という名前のパネルは、システムコンポーネント専用のパネルです。

③-① コンポーネント

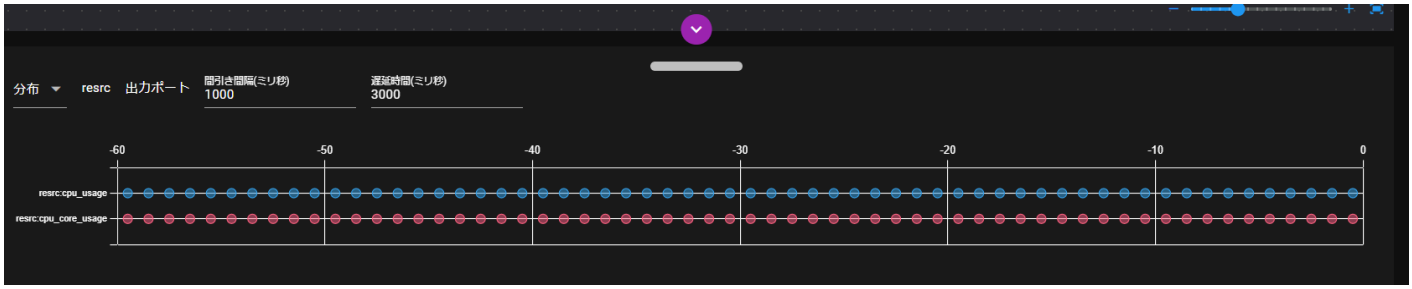


番号	項目	説明
①	入力ポート	データの入力を受け付けるポートです。 ※1 クリックすると、ポートに流れているデータがデータモニタに表示されます。詳細は「データモニタ」をご覧ください。 ※2 右クリックすると、設定を行うことができます。詳細は「入力ポート」をご覧ください。
②	コンポーネントベース名	コンポーネントのベース名が表示されます。

番号	項目	説明
③	ステータスランプ	コンポーネントの稼働状況に応じた色が表示されます。 ※詳細は「ステータス一覧」をご覧ください。
④	起動アイコン	クリックすると、コンポーネントが起動します。
⑤	停止アイコン	クリックすると、コンポーネントが停止します。
⑥	設定アイコン	クリックすると、コンポーネントの設定内容を表示します。
⑦	メニューアイコン	クリックすると、コンポーネントの各種ポート設定、ログ、ステータスタイムライン等のメニューを表示します。
⑧	コンポーネント名	コンポーネント名が表示されます。
⑨	データ量モニタ	コンポーネントが扱っているデータ量が表示されます。
⑩	出力ポート	データを出力するポートです。 ※1 クリックすると、ポートに流れているデータがデータモニタに表示されます。詳細は「データモニタ」をご覧ください。 ※2 右クリックすると、設定を行うことができます。詳細は「出力ポート」をご覧ください。
⑪	システムポート	システムに関連するデータを出力するポートです。 ※1 クリックすると、ポートに流れているデータがデータモニタに表示されます。詳細は「データモニタ」をご覧ください。 ※2 右クリックすると、設定を行うことができます。詳細は「システムポート」をご覧ください。

- 詳細は「コンポーネント」をご覧ください。

④ データモニタ

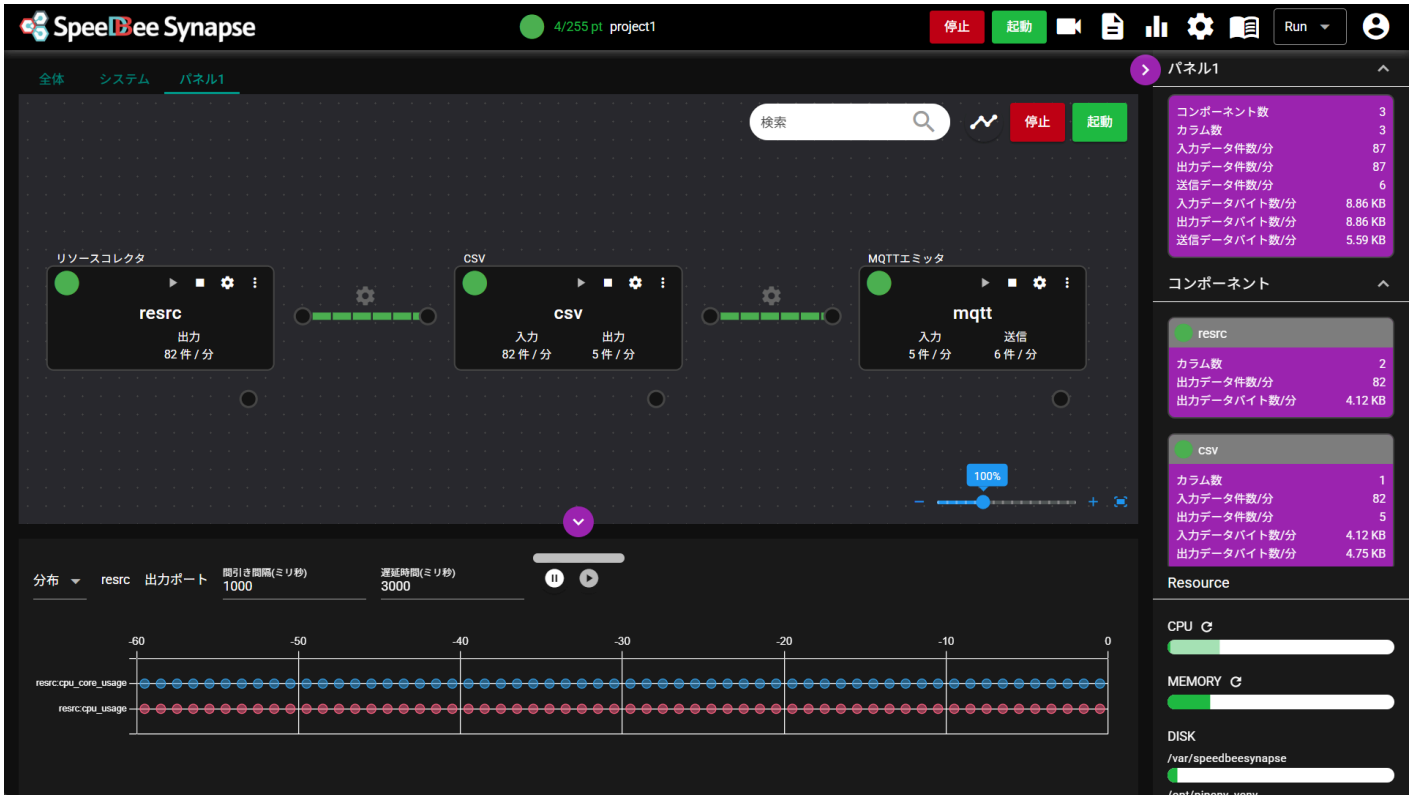


番号	項目	説明
①	開閉アイコン	クリックすることで、データモニタの表示・非表示を切り替えることができます。
②	表示サイズ調整グリップ	ドラッグしてマウスを上下に動かすことで、データモニタの表示サイズを調整することができます。
③	表示種類選択フォーム	データモニタの表示方法を選択できます。
④	選択ポート	選択したポートが表示されます。
⑤	間引き間隔	表示するデータの間引き間隔を設定できます。
⑥	遅延時間	表示するデータの遅延時間を設定できます。
⑦	モニタ表示エリア	選択した種類のモニタが表示されます。

- 詳細は「データモニタ」をご覧ください。

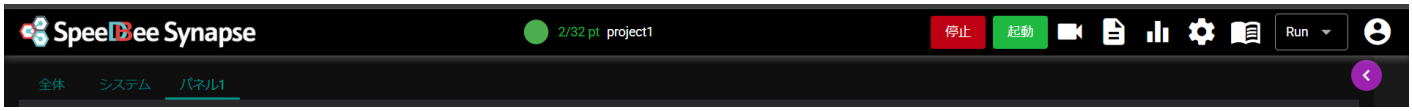
5.3.2.2 Run Mode

Run Modeのフローエディタの画面項目は次の通りです。



Run Modeの画面は、① ヘッダ、② パネル、③ パネルステータス、④ データモニタ、⑤ システムリソースで構成されています。

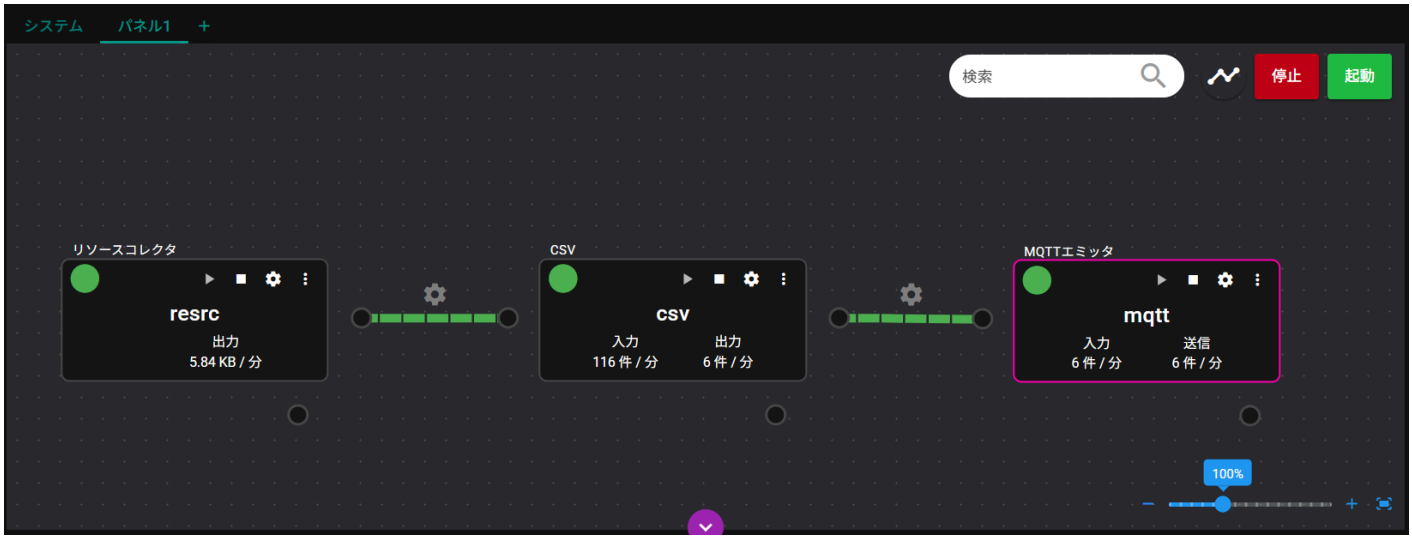
① ヘッダ



番号	項目	説明
①	使用ポイント数	登録されたライセンスのポイント数と、使用されたポイント数が表示されます。
②	現在のプロジェクト	現在選択中のプロジェクトが表示されます。 ※詳細は「設定情報の管理(プロジェクト)」をご覧ください。
③	停止ボタン	クリックすると、SpeeDBee Synapseに登録されたすべてのコンポーネントが停止します。
④	起動ボタン	クリックすると、SpeeDBee Synapseに登録されている、「自動起動無効」がOFFのコンポーネントがすべて起動します。

番号	項目	説明
⑤	ビューワーアイコン	クリックすると、ビューワー画面に遷移します。 ※拡張パッケージ「i-Trigger」を適用した場合のみ表示されます。詳細は「i-Trigger（拡張パッケージ）」をご覧ください。
⑥	ログアイコン	クリックすると、SpeeDBee Synapseの操作ログ画面が表示されます。 ※詳細は「各種ログ」をご覧ください。
⑦	グラフアイコン	クリックすると、Grafanaの画面が表示されます。 ※詳細は「Grafanaとの連携」をご覧ください。
⑧	設定メニューアイコン	クリックすると、設定メニューが表示されます。 ※選択できるメニューは言語設定のみです。詳細は「設定メニュー」をご覧ください。
⑨	マニュアル	クリックすると、マニュアル画面が表示されます。
⑩	モード選択	フローエディタの表示モードを、次の中から選択できます。 ・Edit Mode ・Run Mode ※権限がシステム管理者・管理者のユーザーでログインした場合のみ表示されます。
⑪	ユーザー管理アイコン	クリックすると、ユーザー管理のメニューが表示されます。 ※詳細は「ユーザー管理」をご覧ください。

② パネル



番号	項目	説明
①	タブ	表示するパネルを切り替えるための見出しです。
②	コンポーネント検索ボックス	コンポーネントの位置を検索できるボックスです。
③	ステータスタイムライン	クリックすると、コンポーネントのステータスを時系列順に並べた画面が表示されます。 ※詳細は「ステータスタイムライン」をご覧ください。
④	停止ボタン	クリックすると、パネル上のすべてのコンポーネントが停止します。
⑥	起動ボタン	クリックすると、パネル上の、「自動起動無効」がOFFのコンポーネントがすべて起動します。
⑥	コンポーネント	パネルに登録されているコンポーネントです。 ※コンポーネントは、データの収集、加工、分析、送信などを実現するモジュール群です。
⑦	フローリンク	パネルに登録されているフローリンクです。
⑧	表示倍率	パネルの表示倍率を変更できます。

- 詳細は「パネル」をご覧ください。
- "システム"という名前のパネルは、システムコンポーネント専用のパネルです。

②-① コンポーネント



番号	項目	説明
①	入力ポート	データの入力を受け付けるポートです。 ※クリックすると、ポートに流れているデータがデータモニタに表示されます。詳細は「データモニタ」をご覧ください。
②	コンポーネントベース名	コンポーネントのベース名が表示されます。
③	ステータスランプ	コンポーネントの稼働状況に応じた色が表示されます。 ※詳細は「ステータス一覧」をご覧ください。
④	起動アイコン	クリックすると、コンポーネントが起動します。
⑤	停止アイコン	クリックすると、コンポーネントが停止します。
⑥	設定アイコン	クリックすると、コンポーネントの設定内容を表示します。
⑦	メニューアイコン	クリックすると、コンポーネントのログ、ステータスタイムライン等のメニューが表示されます。
⑧	コンポーネント名	コンポーネント名が表示されます。
⑨	データ量モニタ	コンポーネントが扱っているデータ量が表示されます。
⑩	出力ポート	データを出力するポートです。 ※クリックすると、ポートに流れているデータがデータモニタに表示されます。詳細は「データモニタ」をご覧ください。
⑪	システムポート	システムに関連するデータを出力するポートです。 ※クリックすると、ポートに流れているデータがデータモニタに表示されます。詳細は「データモニタ」をご覧ください。

- 詳細は「コンポーネント」をご覧ください。

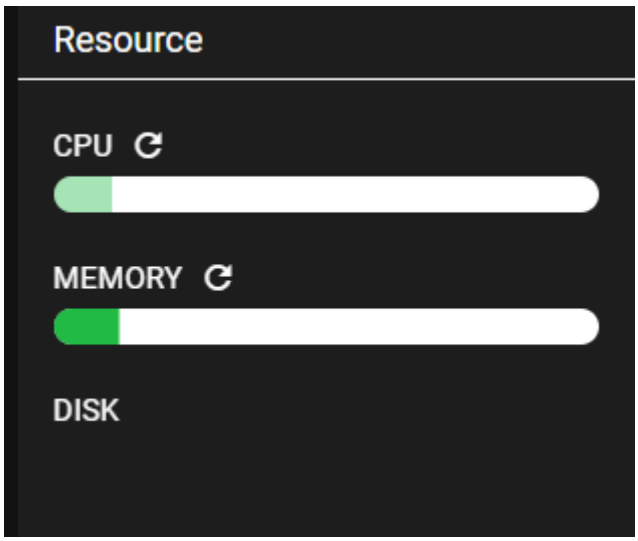
③ パネルステータス



番号	項目	説明
①	開閉アイコン	クリックすることで、パネルステータスの表示・非表示を切り替えることができます。
②	インジケーター	クリックすることで、各項目の表示・非表示を切り替えることができます。
③	パネルステータス	パネルのステータスが表示されます。
④	コンポーネントステータス	コンポーネントのステータスが表示されます。

- 詳細は「パネルステータス」をご覧ください。

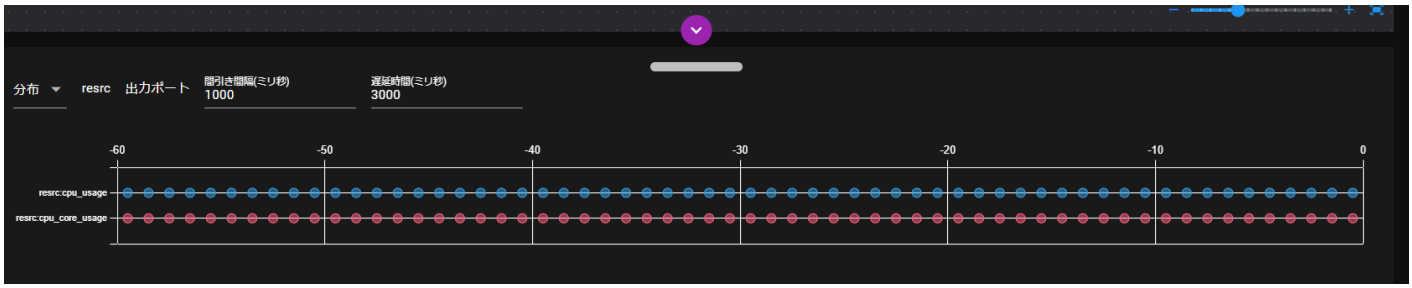
④ システムリソース



番号	項目	説明
①	リフレッシュアイコン	クリックすると、表示内容が更新されます。
②	ゲージ	リソースの割合が表示されます。

- 詳細は「システムリソース」をご覧ください。

⑤ データモニタ



番号	項目	説明
①	開閉アイコン	クリックすることで、データモニタの表示・非表示を切り替えることができます。
②	表示サイズ調整グリップ	ドラッグしてマウスを上下に動かすことで、データモニタの表示サイズを調整することができます。
③	表示種類選択フォーム	データモニタの表示方法を選択できます。
④	選択ポート	選択したポートが表示されます。

番号	項目	説明
⑤	間引き間隔	表示するデータの間引き間隔を設定できます。
⑥	遅延時間	表示するデータの遅延時間を設定できます。
⑦	モニタ表示エリア	選択した種類のモニタが表示されます。

- 詳細は「データモニタ」をご覧ください。

5.3.3 画面操作

ここでは、フローエディタの操作方法について説明します。

5.3.3.1 設定メニュー

設定メニューの操作について説明します。

> 設定メニューの操作はEdit Modeでのみ行うことができます。

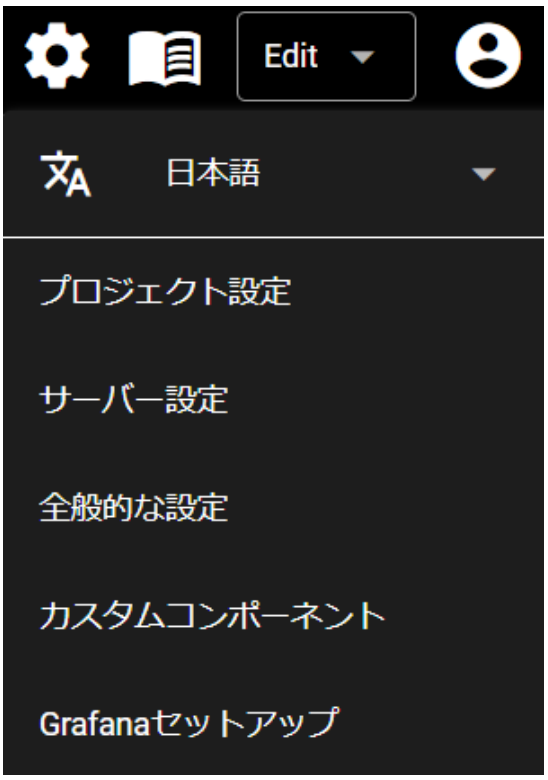
5.3.3.1.1 言語設定

次の手順で、言語の切り替えを行うことができます。

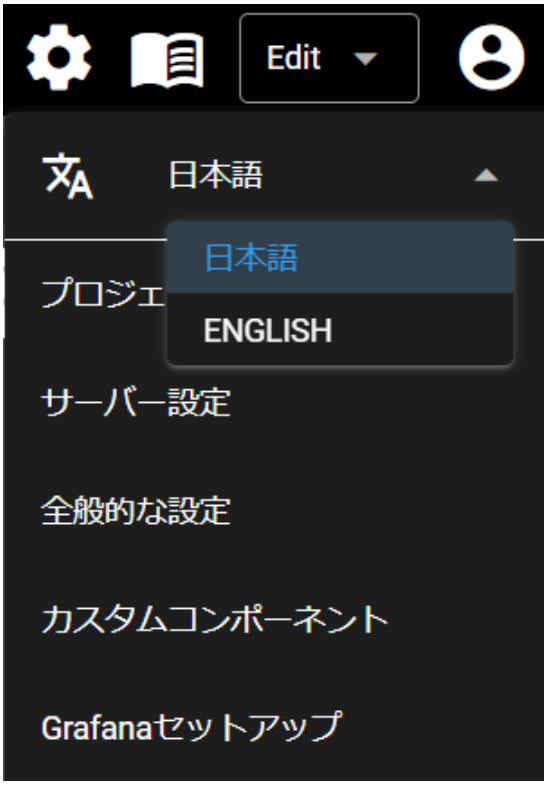
1. 設定メニューアイコンを押下します。



2. 言語選択のプルダウンメニューを選択します。



3. 言語を選択します。



4. 選択した言語がSpeeDBee Synapseに適用されます。

5.3.3.1.2 プロジェクト設定

プロジェクトの設定を行うことができます。詳細は「設定情報の管理(プロジェクト)」をご覧ください。

5.3.3.1.3 サーバー設定

次の手順で、サーバー設定の確認および更新、データベースの初期化を行うことができます。

1. 設定メニューアイコンを押下します。



2. 「サーバー設定」を選択します。



3. 次の項目のうち、任意のものを入力し「保存」を押下すると、設定が更新されます。



項目		説明
サーバー	サーバーインスタンス名	SpeeDBee Synapseに付与する名称。 Run Modeの全体パネルに表示される。（複数台のSpeeDBee Synapse を利用している場合、それぞれに名称を設定することで、識別に利用可能）
	説明	SpeeDBee Synapseの用途などの説明
データベース	ストレージパス	データベースの保存先ディレクトリ ※一部の環境では、この設定を変更/編集できません。
	ファイル型ストレージ	ファイル型データベースの保存先ディレクトリ ※一部の環境では、この設定を変更/編集できません。

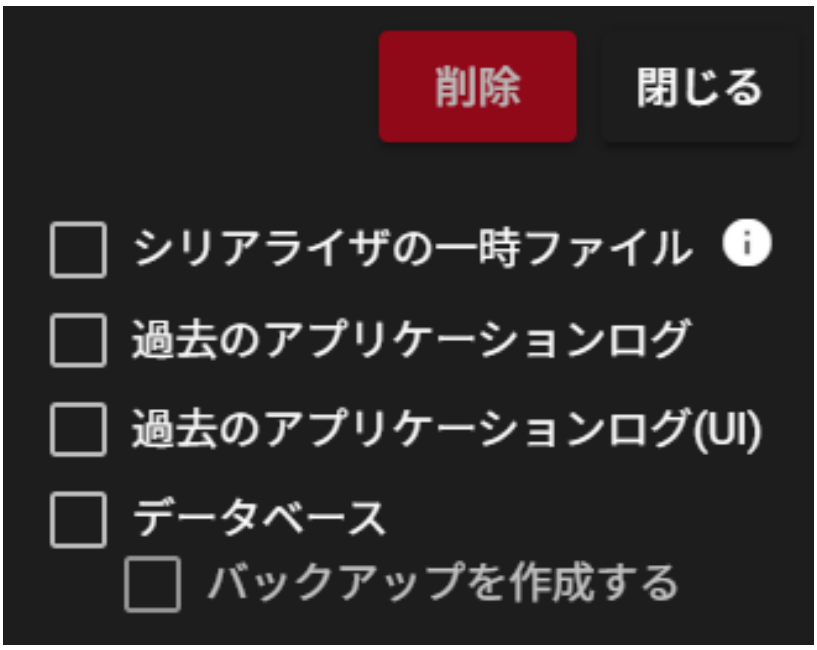
	パス	
	DBログパス	データベースのログの保存先ディレクトリ ※一部の環境では、この設定を変更/編集できません。
	ストレージ上限サイズ	ストレージにデータを保存する上限サイズ。 ストレージ上のデータがこの上限サイズを超える場合、古いデータから削除される。未指定の場合は無制限。
	グループ単位で保存期間を指定	ONの場合、グループごとのストレージにデータを保持する期間を設定可能。 保存期間を過ぎたデータは削除される。ただし、保存期間内であっても、ストレージ上限サイズを超える場合は古いデータから削除される。
	メモリ保存期間(秒)	メモリ上にデータを保持する時間。
コア API	ポート	SpeeDBee Synapseが、WEB画面と内部サーバー間でやり取りするために使用されるポート番号。 ※一部の環境では、この設定を変更/編集できません。

データ削除

1. 「データ削除」を押下します。



2. 削除するデータを選択し、「削除」を押下します。



項目	説明
シリアライザの一時ファイル	シリアライザコンポーネントが出力した一時ファイル ※削除する場合はシステムが再起動されます。
アプリケーションログ	コア、コンポーネント、データベースの運用ログ
アプリケーションログ(UI)	WEBUIの運用ログ
データベース	SpeeDBee Synapseの時系列データベース ※削除する場合はシステムが再起動されます。
バックアップを作成する	データベースを削除する前にバックアップを作成する ※データベースにチェックを付けた場合のみ選択できます。

3. 「はい」を押下します。



4. 選択したデータが削除されます。

コア起動・停止

1. 「コア停止」を押下すると、SpeeDBee Synapseのコア機能を停止します。



2. 「コア起動」を押下すると、SpeeDBee Synapseのコア機能を起動します。



5.3.3.1.4 全般的な設定

次の手順で、全般的な設定の確認および更新を行うことができます。

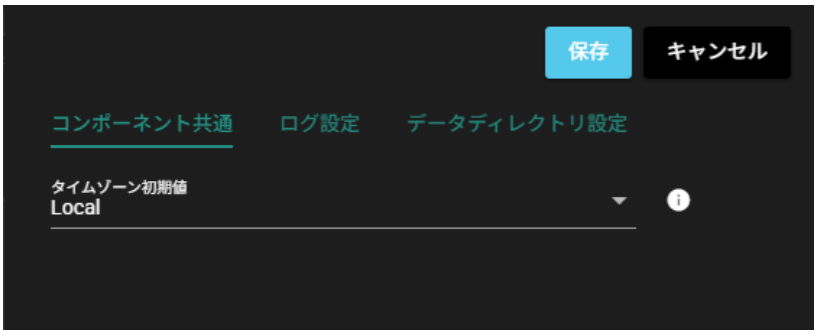
1. 設定メニューアイコンを押下します。



2. 「全般的な設定」を選択します。



3. タブを切り替えて任意の項目を編集し、「保存」を押下すると設定が更新されます。



項目	説明
コンポーネント共通	コンポーネントの共通の初期値を設定します。
ログ設定	アプリケーションログの設定を行います。

データディレクトリ設定	データディレクトリの設定を行います。

5.3.3.1.5 コンポーネント共通

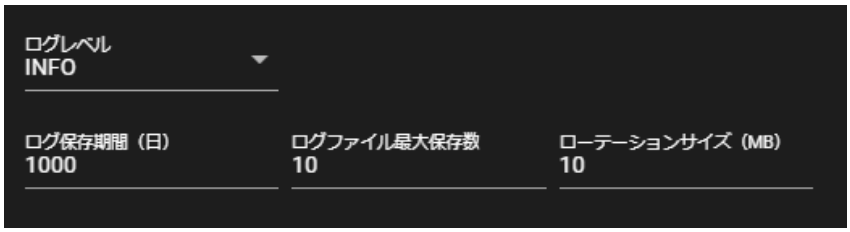
コンポーネントの共通する入力項目の初期値を確認、編集します。



項目	説明
タイムゾーン初期値	CSV, JSONシリアライザ、Fileエミッタで設定できるタイムゾーンの初期値

5.3.3.1.6 ログ設定

アプリケーションログの設定を確認、編集します。



項目	説明
ログレベル	ログ出力のレベル
ログ保存期間（日）	ログの保存期間日数
ログファイル最大保存数	ログファイルの最大保存数
ローテーションサイズ（MB）	ログローテーションのサイズ（MB）

5.3.3.1.7 データディレクトリ設定

データディレクトリの設定を確認、編集します。



項目	説明
データディレクトリ	VIDEOコレクタ、Fileエミッタ、Event Recorderで設定できる保存先、出力先のベースディレクトリ。 ※一部の環境では、この設定を変更/編集できません。

※データディレクトリを変更した場合、以下のコンポーネントで次回起動時から変更が反映され、新しいディレクトリが使用されます。

- VIDEOコレクタ
- Fileエミッタ
- Event Recorder

5.3.3.1.8 カスタム(Python)

Pythonカスタムコンポーネントの設定を行うことができます。詳細は「Pythonカスタムコンポーネント作成」をご覧ください。

5.3.3.1.9 カスタム(C)

Cカスタムコンポーネントの設定を行うことができます。詳細は「C言語カスタムコンポーネント作成」をご覧ください。

5.3.3.1.10 Grafanaセットアップ

Grafanaの設定を行うことができます。詳細は「Grafanaとの連携」をご覧ください。

5.3.3.1.11 サポート用ファイルダウンロード

次の手順で、SpeeDBee Synapseのサポート担当に渡すファイルをダウンロードできます。

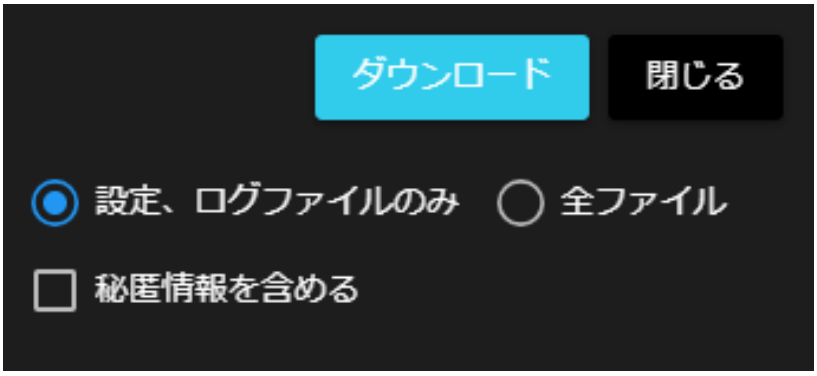
1. 設定メニューアイコンを押下します。



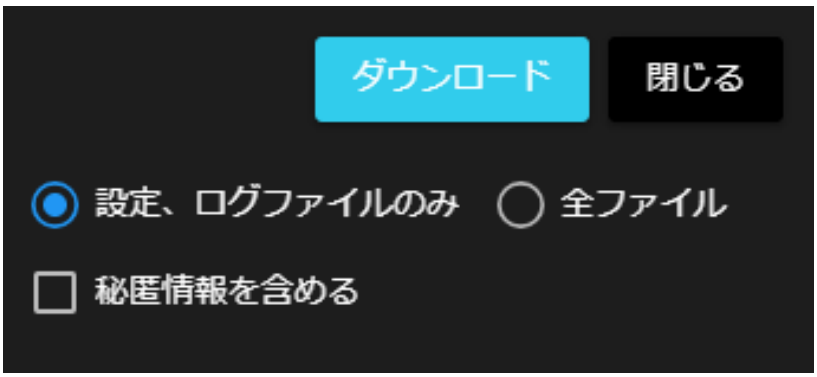
2. 「サポート用ファイルダウンロード」を選択します。



3. ダウンロードするファイルを選択します。
   ※基本的には「設定、ログファイルのみ」を選択してください。秘匿情報は、サポート担当より指示がない限り含める必要はありません。



4. 「ダウンロード」を押下します。



5. サポート用ファイルがダウンロードされます。

5.3.3.1.12 アクセストークン発行

次の手順で、アクセストークンを発行することができます。 発行したアクセストークンは、「DBクエリサービス」、「Grafanaとの連携」、「制御プログラムとの連携」で使用することができます。

1. 設定メニューアイコンを押下します。



2. 「アクセストークン発行」を選択します。



3. 次の項目を入力し、「発行」を押下します。



項目	説明
トークン名	トークンの識別名
有効期限	トークンの有効期限

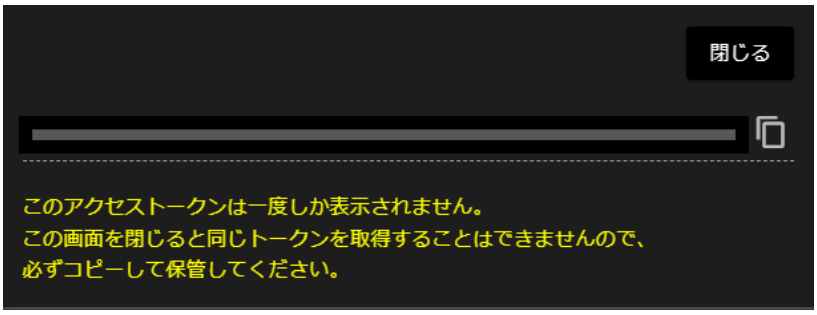
アクセストークンの有効期限について

アクセストークンの有効期限を長くした場合、トークンが漏洩した際のセキュリティリスクが高くなります。 反対に、有効期限を短くすると、セキュリティは強化されますが、頻繁にアクセストークンを更新する必要があります。 あらかじめ定めた運用方針により、定期的にアクセストークンを更新することもご検討ください。

4. 発行されたアクセストークンをコピーし、「閉じる」を押下します。

アクセストークンの保管について

アクセストークンは一度しか表示されません。 この画面を閉じると同じトークンを取得することはできないので、必ずコピーして保管してください。



5. 発行したアクセストークンは、一覧に追加されます。



5.3.3.1.13 HTTP/HTTPS通信

次の手順で、ユーザーのデバイスとSpeeDBee SynapseのHTTP通信についての設定を行います。

1. 設定メニューアイコンを押下します。



2. 「HTTP/HTTPS通信」を選択します。



3. 以下の項目を入力し、「反映して再起動」を押下します。



項目	説明
バインドアドレス	SpeeDBee Synapseがリクエストを受け付けるアドレス ※一部の環境では、この設定を変更/編集できません。
ポート	SpeeDBee Synapseがリクエストを受け付けるポート ※一部の環境では、この設定を変更/編集できません。
暗号化通信	暗号化通信(HTTPS)を利用する場合ON ※一部の環境では、この設定を変更/編集できません。
SSL/TLS証明書	暗号化のために使用するデジタル証明書 ※証明書は、CA（認証局）から取得する必要があります。
秘密鍵	SSL/TLS証明書に紐づけられた秘密鍵 ※秘密鍵は、CA（認証局）から取得する必要があります。

4. 設定反映のため、システム全体の再起動が行われます。再起動後、設定内容に合わせたURLで、SpeeDBee Synapseにアクセスすることができます。

暗号化通信を行う場合

```
https://<ホスト名orデバイスのIPアドレス>:<SpeeDBee Synapseがリクエストを受け付けるポート>/
```

暗号化通信を行わない場合

```
http://<ホスト名orデバイスのIPアドレス>:<SpeeDBee Synapseがリクエストを受け付けるポート>/
```

5.3.3.1.14 プロキシ設定

ここでは、プロキシ設定の操作について説明します。 ここで設定したプロキシ情報は、各コンポーネントの設定から参照することで、 該当コンポーネントの通信をプロキシ経由に切り替えることができます。

プロキシ対応コンポーネント一覧

- MQTTコレクタ
- MQTTエミッタ
- Cloudエミッタ(AWS用)
- Cloudエミッタ(Azure用)

また、既定のプロキシを設定することで、特定の操作における通信がプロキシ経由で行われるようになります。

対象となる操作一覧

- Pythonカスタムコンポーネントの外部ライブラリ追加

追加

次の手順で、プロキシ設定の追加を行うことができます。

1. 設定メニューアイコンを押下します。



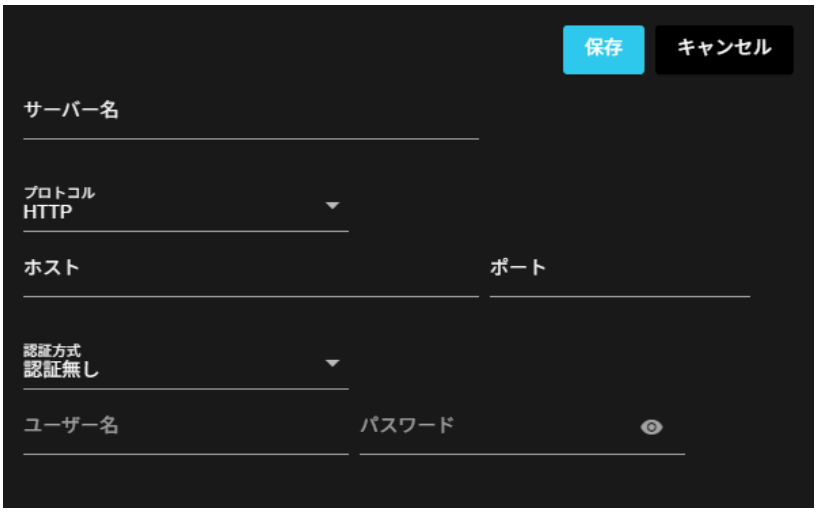
2. 「プロキシ設定」を選択します。


Edit
日本語
プロジェクト設定
サーバー設定
全般的な設定
カスタムコンポーネント
Grafanaセットアップ
サポート用ファイルダウンロード
アクセストークン発行
HTTP/HTTPS通信
プロキシ設定
実行ライセンス設定ダウンロード
ソフトウェア情報

3. 「追加」を押下します。



4. 以下の項目を入力し、「保存」を押下します。



項目	説明
サーバー名	任意のサーバー識別名。 ※プロキシ経由の通信を設定しているコンポーネントはサーバー名を参照して設定を読みこんでいるため、変更するとコンポーネントの再設定が必要となります。
プロトコル	使用するプロキシの種類。（HTTP）
ホスト	プロキシサーバーのホスト名またはIPアドレス。
ポート	プロキシサーバーのポート番号。
認証方式	プロキシサーバーの認証方式。（認証無し、Basic認証）
ユーザー名	Basic認証を使用する場合のユーザー名。
パスワード	Basic認証を使用する場合のパスワード。

5. 保存した設定は、一覧に追加されます。



編集

次の手順で、プロキシ設定の編集を行うことができます。

1. 設定メニューアイコンを押下します。



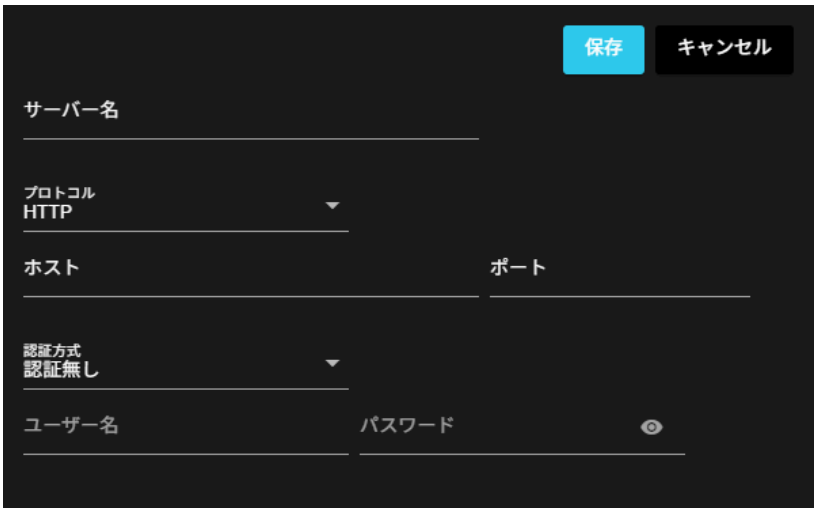
2. 「プロキシ設定」を選択します。



3. 編集アイコンを押下します。



4. 以下の項目を入力し、「保存」を押下します。



項目	説明
サーバー名	任意のサーバー識別名。 ※プロキシ経由の通信を設定しているコンポーネントはサーバー名を参照して設定を読みこんでいるため、変更するとコンポーネントの再設定が必要となります。
プロトコル	使用するプロキシの種類。（HTTP）
ホスト	プロキシサーバーのホスト名またはIPアドレス。
ポート	プロキシサーバーのポート番号。
認証方式	プロキシサーバーの認証方式。（認証無し、Basic認証）
ユーザー名	Basic認証を使用する場合のユーザー名。
パスワード	Basic認証を使用する場合のパスワード。

既定のプロキシ

次の手順で、既定のプロキシを設定することができます。

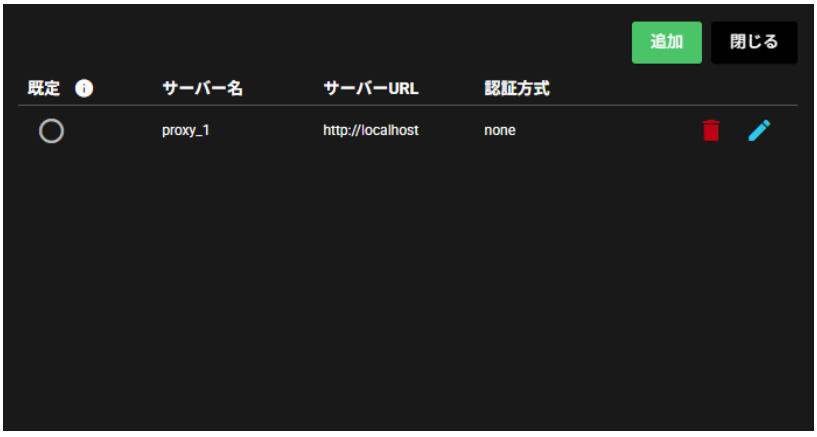
1. 設定メニューアイコンを押下します。



2. 「プロキシ設定」を選択します。



3. 既定として設定したいプロキシにチェックを入れます。



※既定として設定済みのプロキシを押下すると、既定のプロキシを未設定の状態にすることができます。



削除

次の手順で、プロキシ設定の削除を行うことができます。

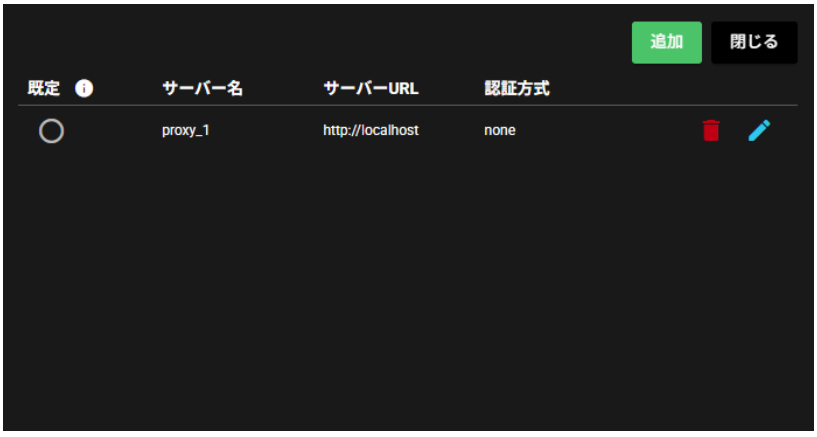
1. 設定メニューアイコンを押下します。



2. 「プロキシ設定」を選択します。



3. 削除アイコンを押下します。



※削除したプロキシ設定がコンポーネントから参照されていた場合、コンポーネントの再設定が必要となります。

5.3.3.1.15 実行ライセンス設定ダウンロード

実行ライセンス環境用に、設定をダウンロードすることができます。詳細は「実行ライセンス」をご覧ください。

5.3.3.1.16 ソフトウェア情報

次の手順で、ソフトウェア情報およびライセンス情報の確認・ライセンスの登録を行うことができます。

1. 設定メニューアイコンを押下します。



2. 「ソフトウェア情報」を選択します。



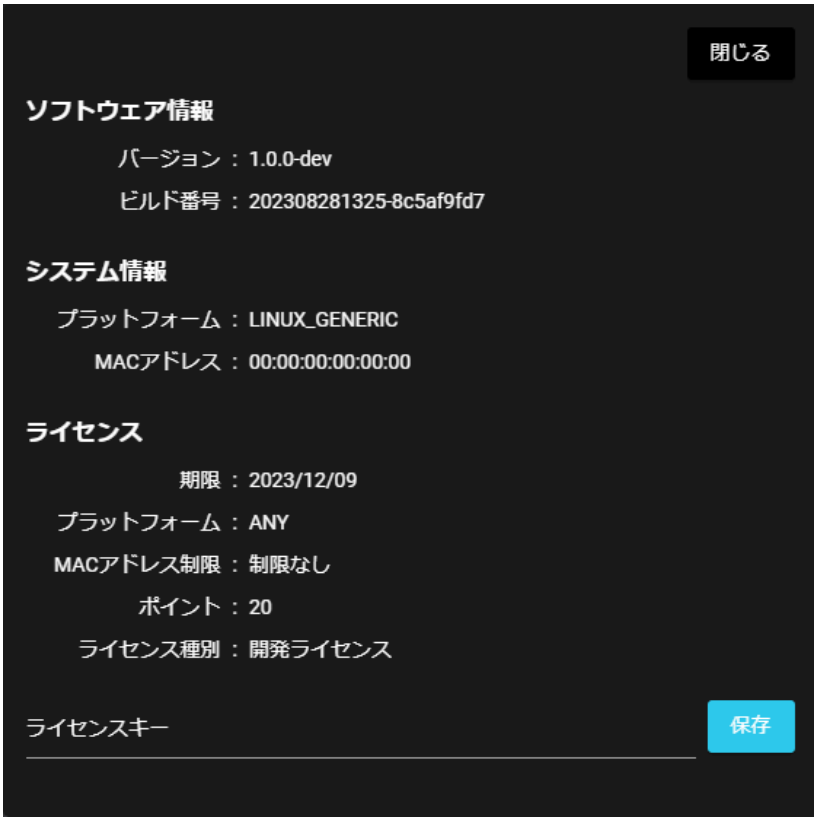
3. 次の項目が表示されます。



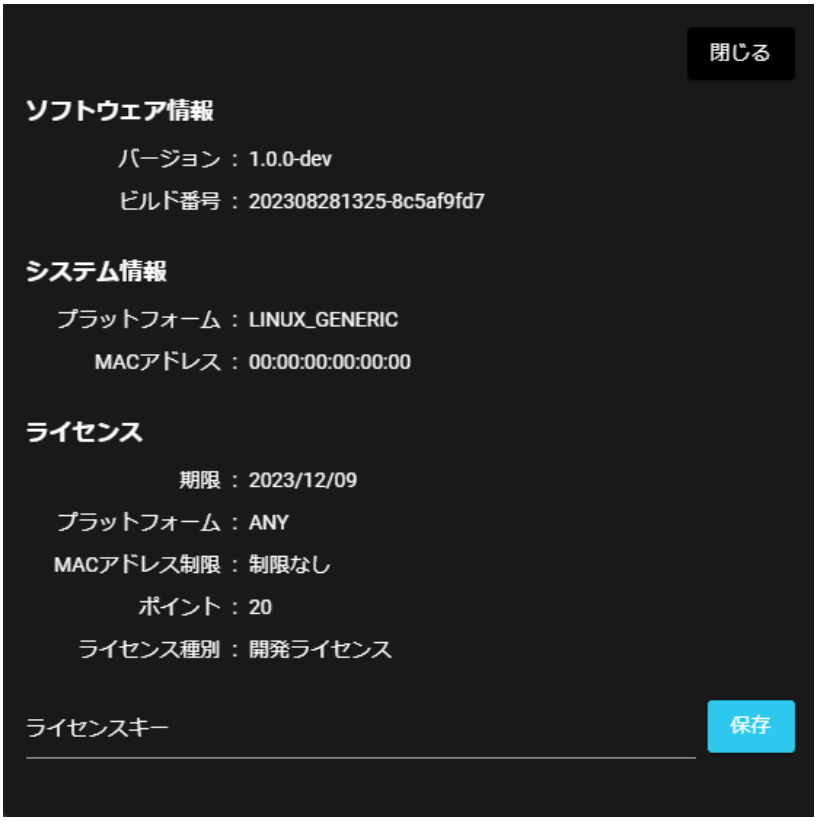
項目		説明
ソフトウェア情報	バージョン	SpeeDBee Synapseのバージョン
	ビルド番号	SpeeDBee Synapseのビルド番号
システム情報	プラットフォーム	システムのプラットフォーム
	MACアドレス	システムのMACアドレス
ライセンス	期限	登録されたライセンスの有効期限
	プラットフォーム	登録されたライセンスがサポートするプラットフォーム
	MACアドレス制限	登録されたライセンスがサポートするMACアドレス
	ポイント	登録されたライセンスに付与されているポイント
	ライセンス種別	登録されたライセンスの種別

ライセンスの登録

1. ライセンスキー入力欄に、ライセンスキーを入力します。



2. 「保存」を押下します。



3. ライセンスが登録されます。登録したライセンスの情報が、画面に反映されます。



5.3.3.1.17 ポイントについて

SpeeDBee Synapseのライセンスには、各機能を利用するためのポイントが付与されています。 SpeeDBee Synapseの機能を利用すると、その機能ごとのポイント数が消費されます。

Data Flow Controller版、OMRON-SBS for Windows版の注意事項

Data Flow Controller版、OMRON-SBS for Windows版はポイントを無制限に利用できます。
しかし、無制限に動作を保証するものではありません。ご利用のハードウェアスペック、動作させるアプリケーション、関連するシステムなどの影響を受けます。



ポイントが0になっているものは、ポイントを消費しません。
ポイントが2になっているものは、設定を保存し稼働させると2P消費します。
また同じ機能を同時に利用した場合の消費量も2Pのままです。
例えば、MQTTエミッタを２つ動作させても消費されるのは2Pとなります。



同時に利用したい機能分のポイントが不足する場合、コンポーネントを動作することができません。



同じ機能を同時に使用する場合の注意事項

同じ機能を同時に利用する場合、ポイント消費数はかわりませんが、無制限に動作を保証するものではありません。
ご利用のハードウェアスペック、動作させるアプリケーション、関連するシステムなどの影響を受けます。

5.3.3.2 ユーザー管理

ここでは、ユーザー管理の操作について説明します。

5.3.3.2.1 追加

次の手順で、ユーザーを追加することができます。

1. ユーザー管理アイコンを押下します。



2. 「ユーザー管理」を選択します。



3. 「追加」を押下します。



4. 次の項目を入力し、「保存」を押下します。



項目	説明
無効にする	チェックを付けると、ユーザーが無効になり、ログインができなくなります。
ログインID	ログイン時に入力するID
ユーザー名	ユーザーに付与する名前※省略可
権限	システム管理者：すべての操作が可能です。初期ユーザー"admin"に付与されており、1人以上必須です。 管理者：すべての操作が可能です。システム管理者の追加・更新・削除は不可です。 オペレーター：Run Modeの閲覧、およびコンポーネントの起動・停止が可能です。 閲覧者：Run Modeの閲覧が可能です。
パスワード	ログイン時に入力するパスワード
Eメールアドレス	ユーザーに紐づけるEメールアドレス※省略可
閲覧可能パネル	ユーザーが閲覧可能なパネル※権限がオペレーター・閲覧者の場合のみ設定します。

5. ユーザーが追加されます。

5.3.3.2.2 編集

次の手順で、ユーザー情報を編集することができます。

1. ユーザー管理アイコンを押下します。



2. 「ユーザー管理」を選択します。



3. 編集するユーザーの、編集アイコンを押下します。



4. 次の項目を編集し、「保存」を押下します。



項目	説明
無効にする	チェックを付けると、ユーザーが無効になり、ログインができなくなります。
ログインID	ログイン時に入力するID
ユーザー名	ユーザーに付与する名前※省略可
権限	システム管理者：すべての操作が可能です。初期ユーザー"admin"に付与されており、1人以上必須です。 管理者：すべての操作が可能です。システム管理者の追加・更新・削除は不可です。 オペレーター：Run Modeの閲覧、およびコンポーネントの起動・停止が可能です。 閲覧者：Run Modeの閲覧が可能です。
パスワード	ログイン時に入力するパスワード
Eメールアドレス	ユーザーに紐づけるEメールアドレス※省略可
閲覧可能パネル	ユーザーが閲覧可能なパネル※権限がオペレーター・閲覧者の場合のみ設定します。

5. 編集した内容が保存されます。

5.3.3.2.3 削除

次の手順で、ユーザーを削除することができます。

1. ユーザー管理アイコンを押下します。



2. 「ユーザー管理」を選択します。



3. 削除するユーザーの、編集アイコンを押下します。



4. 「削除」を押下します。



5. ユーザーが削除されます。

> **システム管理者について**
>
> システム管理者のユーザーは1人以上必須です。 システム管理者が1人の場合は削除できません。

5.3.3.2.4 プロフィール

次の手順で、現在ログインしているユーザーの情報を確認することができます。

1. ユーザー管理アイコンを押下します。



2. 「プロフィール」を選択します。



3. 現在ログインしているユーザーについて、次の項目が表示されます。



項目	説明
ログインID	ログイン時に入力するID
ユーザー名	ユーザーに付与する名前※省略可
パスワード	ログイン時に入力するパスワード
Eメールアドレス	ユーザーに紐づけるEメールアドレス※省略可

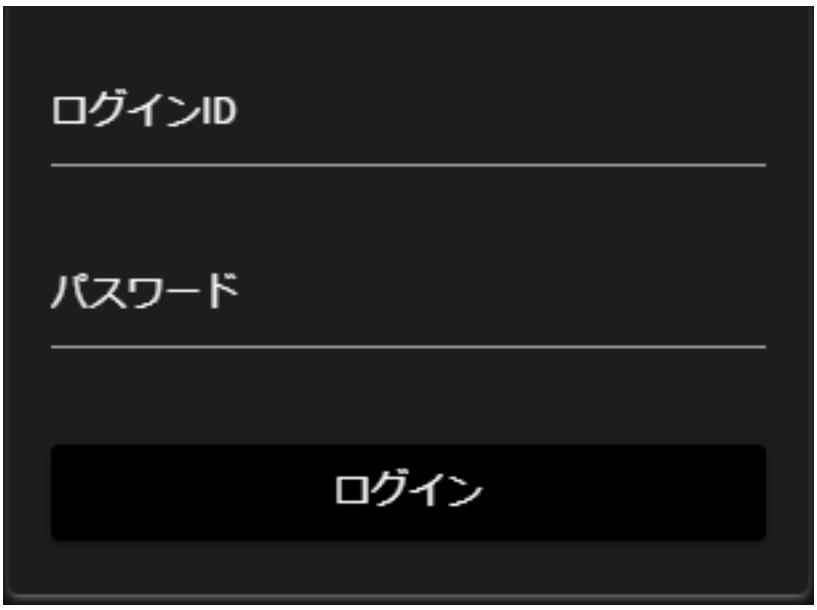
4. 項目を編集して「保存」を押下すると、編集した内容が保存されます。



5.3.3.2.5 ログイン

次の手順で、SpeeDBee Synapseにログインすることができます。

1. 「http://{ SpeeDBee Synapseが稼働している環境のIPアドレス }:8120」にアクセスします。
2. ユーザーの「ログインID」と「パスワード」を入力して、「ログイン」を押下します。



3. SpeeDBee Synapseにログインします。

5.3.3.2.6 ログアウト

次の手順で、SpeeDBee Synapseからログアウトすることができます。

1. ユーザー管理アイコンを押下します。



2. 「ログアウト」を選択します。



3. SpeeDBee Synapseからログアウトします。

5.3.3.3 パネル

ここでは、パネルの操作について説明します。

パネルの閲覧権限について

オペレーター・閲覧者ユーザーの場合は、ユーザー管理の設定にて、パネルごとに閲覧の権限を設定することができます。複数のパネルで別のデータ収集～出力・連携までのフローを作成し、オペレーター・閲覧者に任意のパネルのみを表示することが可能です。

5.3.3.3.1 追加

次の手順で、パネルを追加することができます。

1. パネルのタブメニューにある、＋アイコンを押下します。



2. パネルが追加されます。



この操作はEdit Modeでのみ行うことができます。

5.3.3.3.2 名前の変更

次の手順で、パネル名を変更することができます。

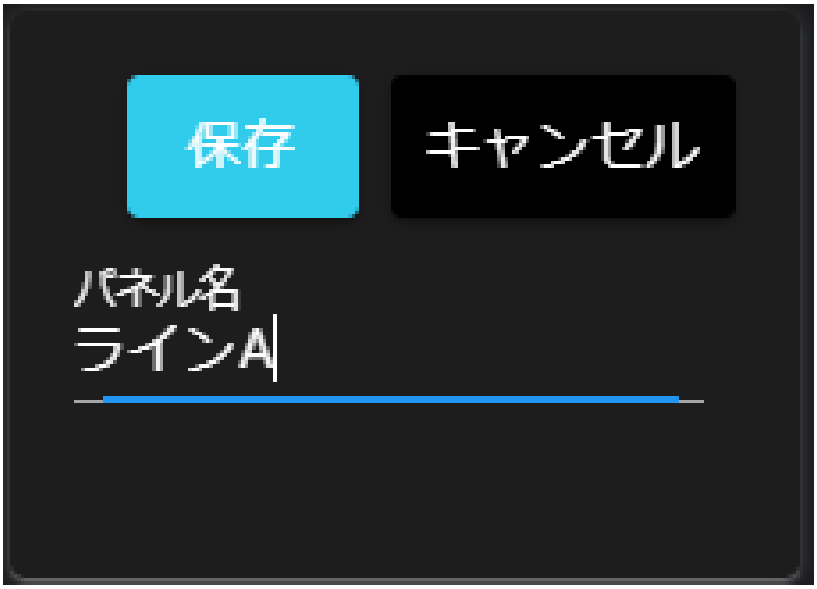
1. パネルタブを右クリックします。



2. 「パネル設定」を選択します。



3. パネル名を入力し、「保存」を押下します。



4. パネル名が、入力したものに変更されます。



この操作はEdit Modeでのみ行うことができます。

5.3.3.3.3 並び替え

次のいずれかの手順で、パネルを並び替えることができます。

- パネルタブのコンテキストメニューから並び替え

  a. パネルタブを右クリックします。

  b. 「右に移動」または「左に移動」を選択します。

  c. パネルが並び変わります。

- パネルタブのドラッグ&ドロップによる並び替え

  a. パネルタブをドラッグします。

  b. 移動したい位置にドロップします。



c. パネルが並び変わります。



この操作はEdit Modeでのみ行うことができます。

5.3.3.3.4 削除

次の手順で、パネルを削除することができます。

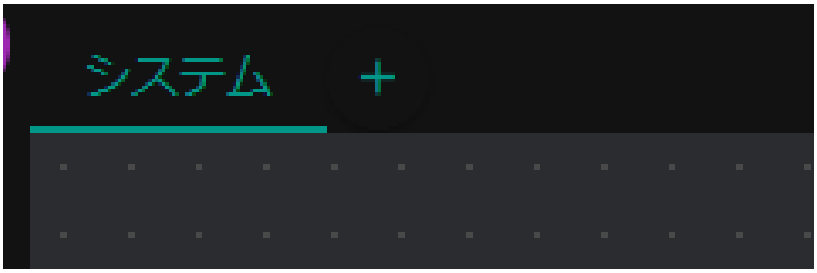
1. パネルタブを右クリックします。



2. 「削除」を選択します。



3. パネルが削除されます。



この操作はEdit Modeでのみ行うことができます。

5.3.3.3.5 複製

次の手順で、同じ設定内容のパネルを複製することができます。

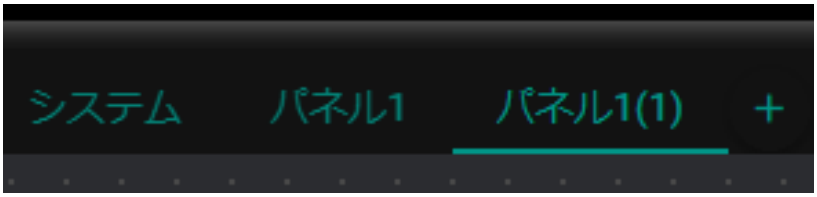
1. パネルタブを右クリックします。



2. 「複製」を選択します。



3. 複製元と同じ設定内容のパネルが複製されます。

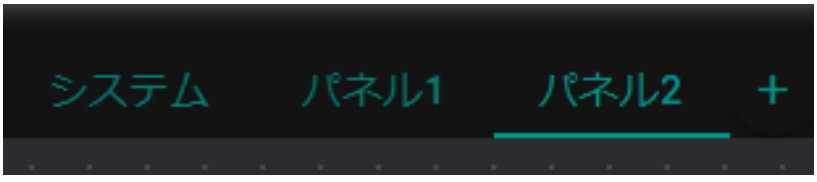


この操作はEdit Modeでのみ行うことができます。

5.3.3.3.6 表示

次の手順で、パネルを表示することができます。

1. 表示するパネルのタブをクリックします。

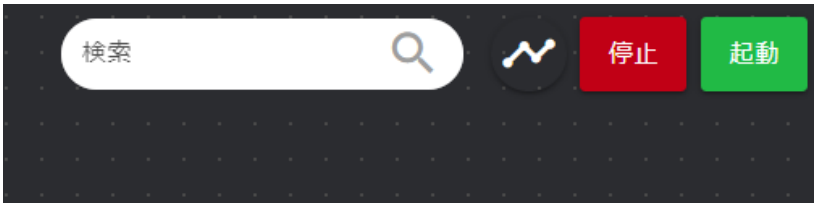


2. タブをクリックしたパネルが表示されます。

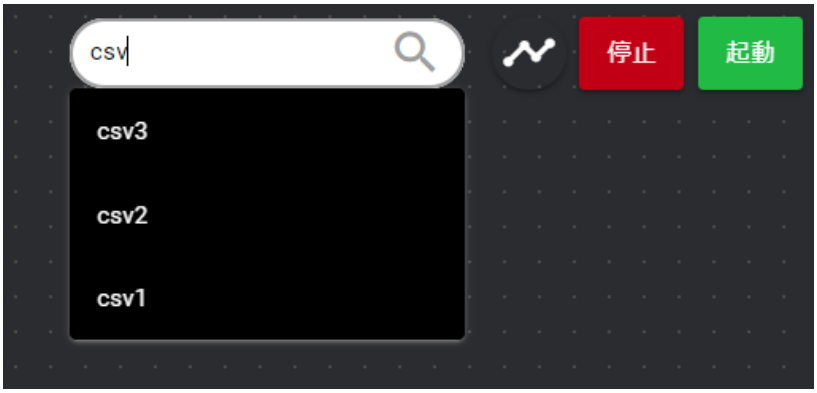
5.3.3.3.7 コンポーネント検索

次の手順で、パネルに登録されているコンポーネントの位置を検索することができます。

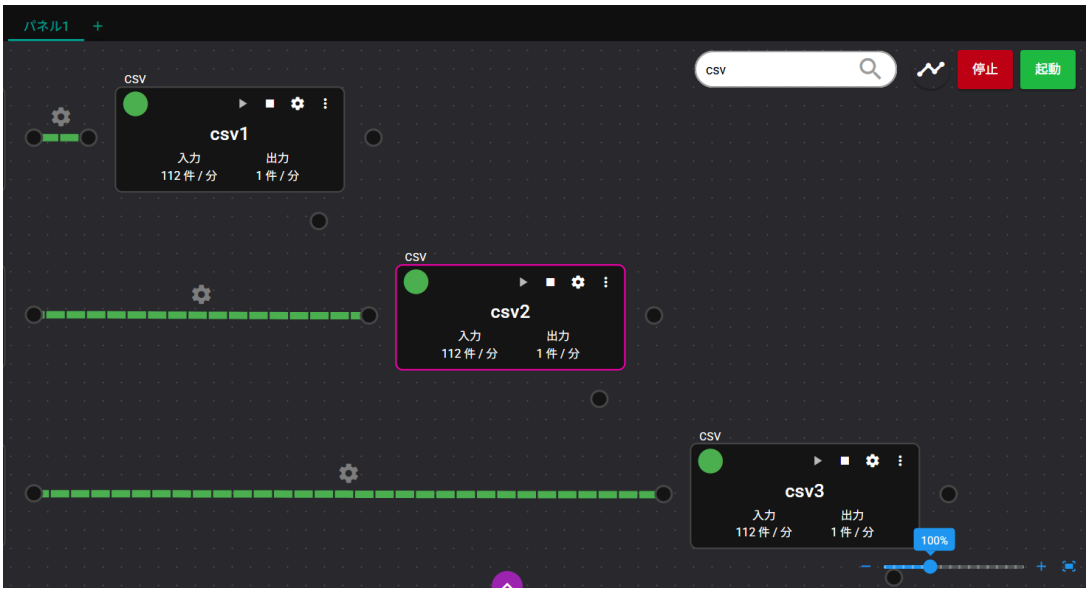
1. パネルの右上の検索ボックスに、コンポーネント名を入力します。



2. 候補の中から選択します。



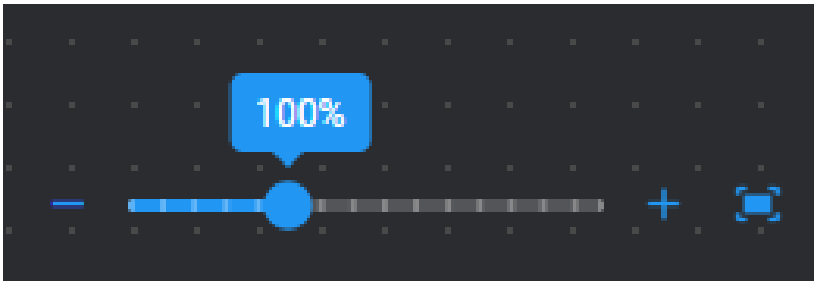
3. 選択したコンポーネントが、パネルの中央に表示されます。



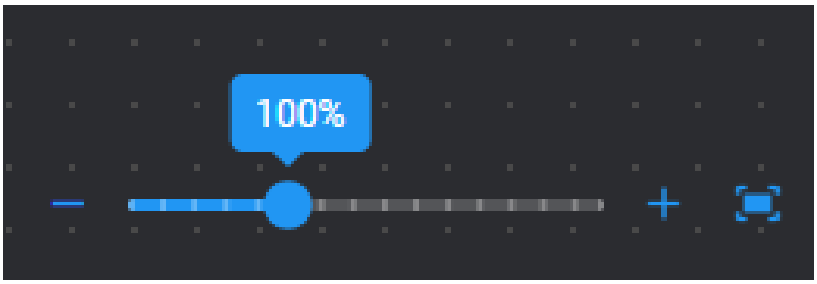
5.3.3.3.8 表示倍率の調整

次の手順で、パネルの表示倍率を調整することができます。

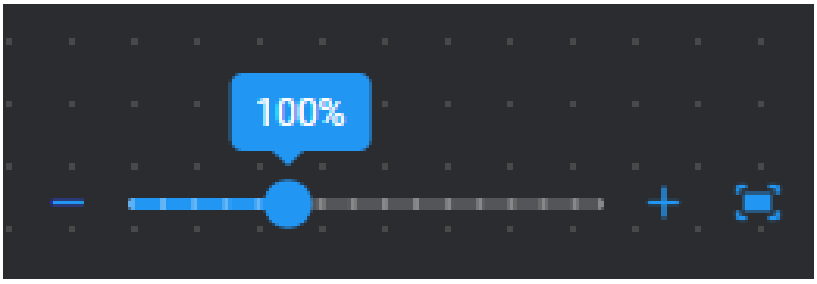
1. パネルの右下に、パネルの現在の表示倍率が表示されます。



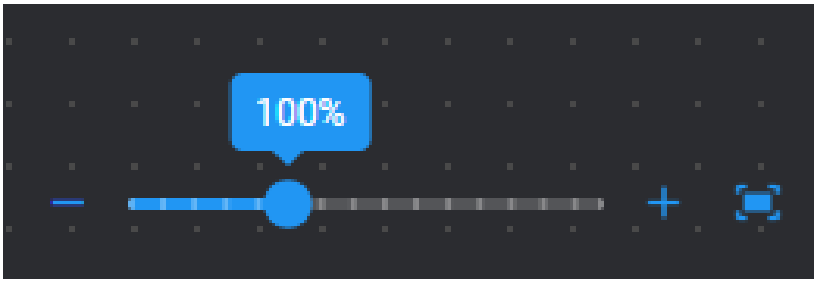
2. スライドバーを+方向に動かすと表示を拡大し、-方向に動かすと表示を縮小します。



3. +を押下すると表示倍率を5%拡大し、-を押下すると表示倍率を5%縮小します。



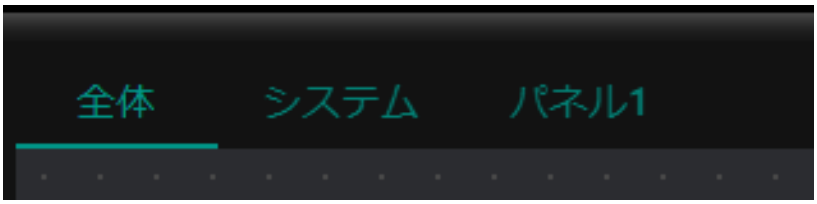
4. 全体表示ボタンを押下するとパネルに配置されたコンポーネント全体を画面に表示します。



5.3.3.3.9 全体パネル

次の手順で、全体パネルを表示することができます。全体パネルは、各パネルの情報を表示するパネルです。

1. パネルタブで「全体」を選択します。



2. サーバーインスタンス名・パネルの一覧・各パネルのコンポーネントの稼働ステータスが表示されます。



3. 「DETAIL」を押下すると、各パネルへ遷移します。



この操作はRun Modeでのみ行うことができます。

5.3.3.3.10 パネルステータス

パネル毎に、次の情報が表示されます。



項目	説明
コンポーネント数	パネル上の全てのコンポーネントの数
カラム数	パネル上の全てのコンポーネントのカラムの数
入力データ件数/分	パネル上の全てのコンポーネントの、1分毎のデータ入力件数の合計
出力データ件数/分	パネル上の全てのコンポーネントの、1分毎のデータ出力件数の合計
受信データ件数/分	パネル上の全てのコンポーネントの、1分毎のデータ受信件数の合計
送信データ件数/分	パネル上の全てのコンポーネントの、1分毎のデータ送信件数の合計

項目	説明
入力データバイト数/分	パネル上の全てのコンポーネントの、1分毎のデータ入力バイト数の合計
出力データバイト数/分	パネル上の全てのコンポーネントの、1分毎のデータ出力バイト数の合計
受信データバイト数/分	パネル上の全てのコンポーネントの、1分毎のデータ受信バイト数の合計
送信データバイト数/分	パネル上の全てのコンポーネントの、1分毎のデータ送信バイト数の合計

パネル上にある、稼働中のコンポーネントについて、次の情報が表示されます。



項目	説明
カラム数	コンポーネントが出力するデータの項目数
入力データ件数/分	1分毎のデータ入力件数
出力データ件数/分	1分毎のデータ出力件数
受信データ件数/分	1分毎のデータ受信件数
送信データ件数/分	1分毎のデータ送信件数
入力データバイト数/分	1分毎のデータ入力バイト数
出力データバイト数/分	1分毎のデータ出力バイト数
受信データバイト数/分	1分毎のデータ受信バイト数
送信データバイト数/分	1分毎のデータ送信バイト数

この操作はRun Modeでのみ行うことができます。

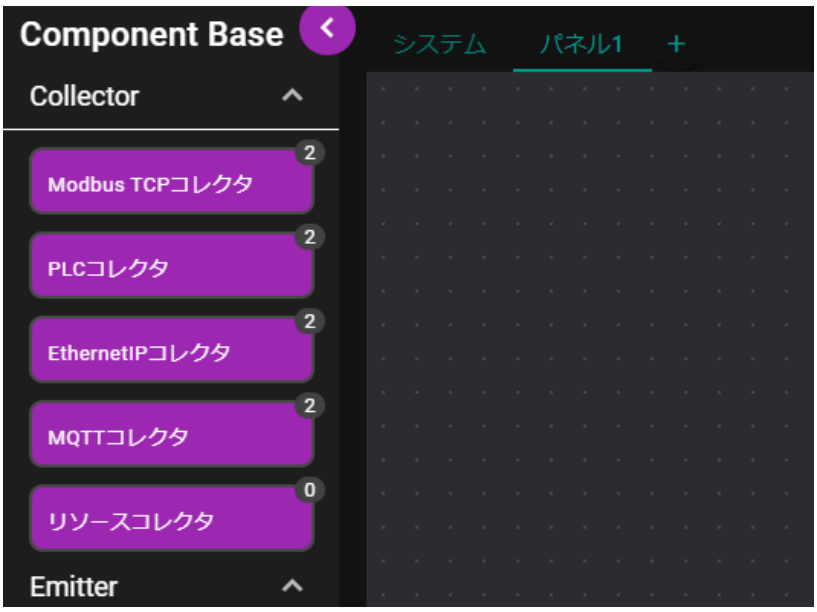
5.3.3.4 コンポーネント

ここでは、コンポーネントの操作について説明します。

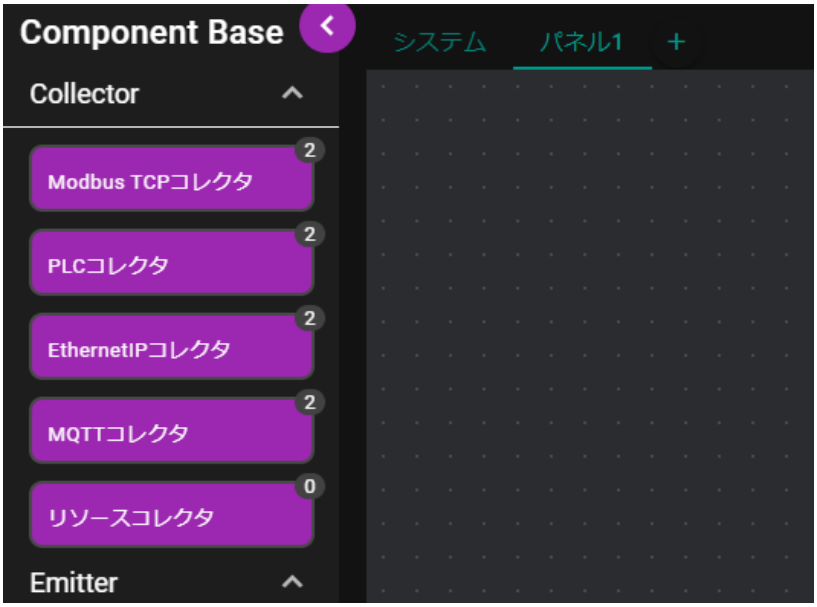
5.3.3.4.1 追加

次の手順で、コンポーネントを追加することができます。

1. 左メニューからコンポーネントを選び、アイコンをドラッグします。



2. パネルの上にドロップします。



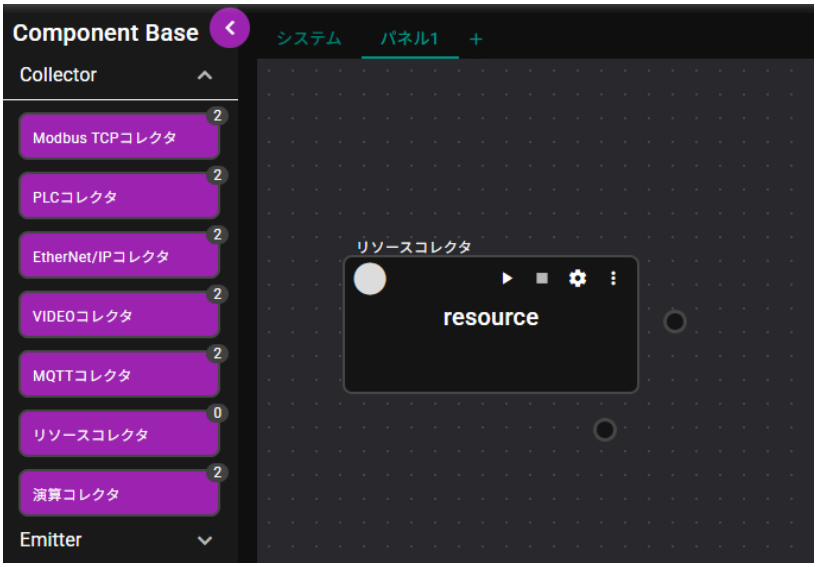
3. 一部のコンポーネントのみ、入力ポートに接続するコンポーネントの選択ダイアログが表示されます。接続するコンポーネントを選択して「選択」を押下します。



4. 設定ダイアログが開くので、設定項目を入力して「保存」を押下します。



5. パネルにコンポーネントが追加されます。

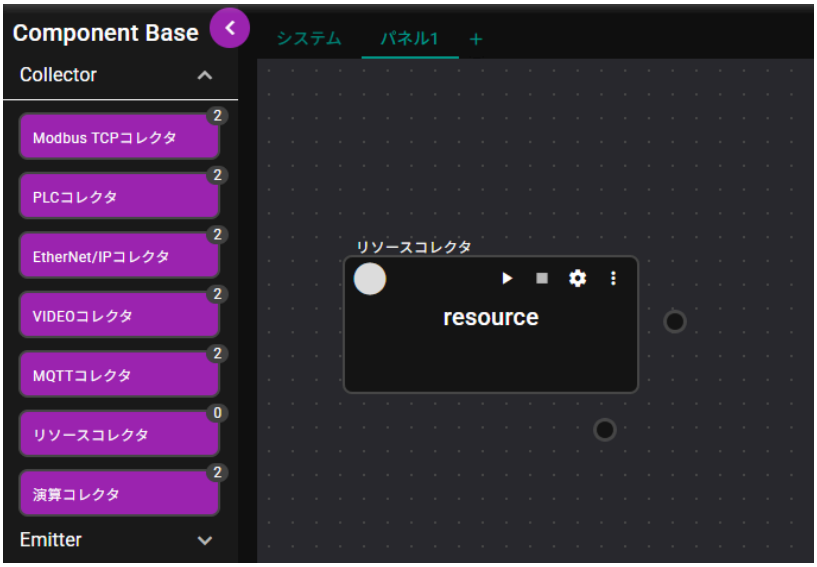


この操作はEdit Modeでのみ行うことができます。

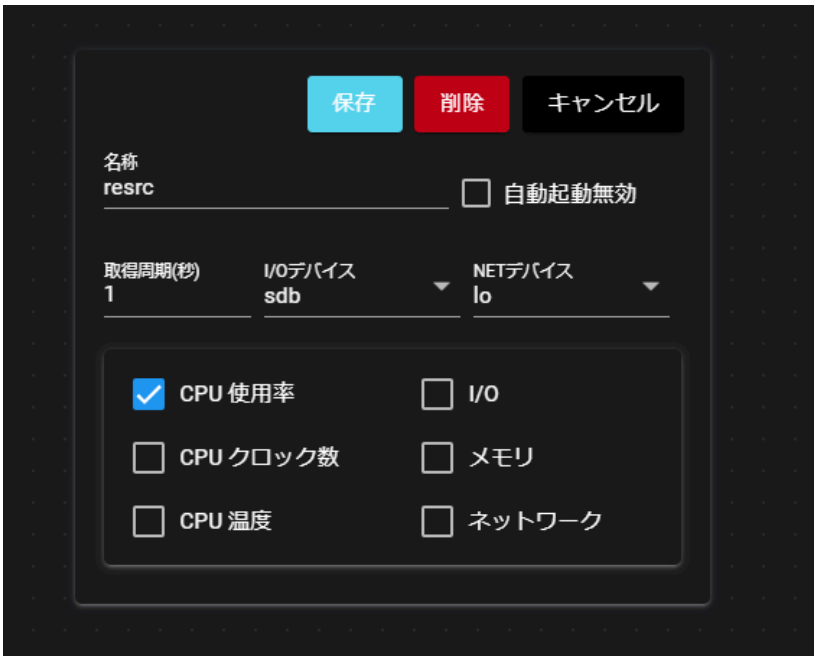
5.3.3.4.2 編集

次の手順で、コンポーネントの設定を編集することができます。

1. コンポーネントの設定アイコンを押下します。



2. 設定ダイアログが開くので、設定項目を編集して「保存」を押下します。



3. 編集した設定が保存されます。

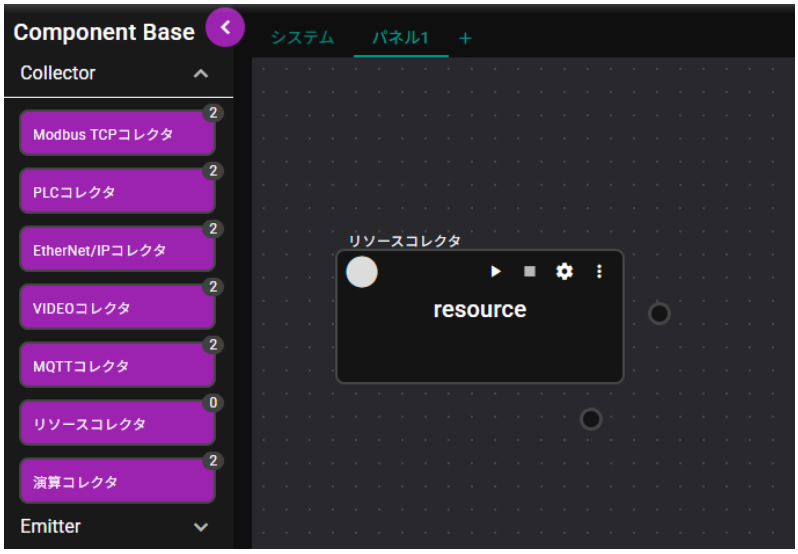
この操作はEdit Modeでのみ行うことができます。

5.3.3.4.3 削除

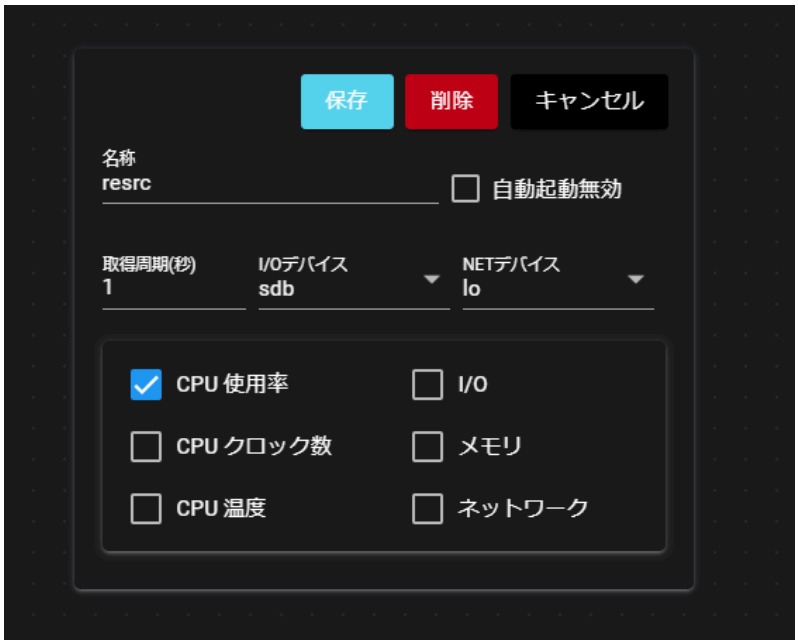
次のいずれかの手順で、コンポーネントを削除することができます。

- コンポーネントの設定アイコンからコンポーネントを削除する方法です。

    a. コンポーネントの設定アイコンを押下します。



b. 設定ダイアログが開くので、「削除」を押下します。



c. コンポーネントが削除されます。

- コンポーネントのコンテキストメニュー上からコンポーネントを削除する方法です。

  a. コンポーネントを右クリックし、「削除」を選択します。



b. コンポーネントが削除されます。

この操作はEdit Modeでのみ行うことができます。

5.3.3.4.4 コピー&ペースト

次の手順で、同じ設定内容のコンポーネントを複製することができます。

1. 次のいずれかの方法で、コンポーネントをコピーします。
    - コンポーネントを右クリックし、「コピー」を選択します。



- コンポーネントをクリックして選択した状態で、キーボードの「Ctrl」キーを押しながら「C」キーを押します。



2. 次のいずれかの方法で、コンポーネントをペーストします。

- パネルを右クリックし、「ペースト」を選択します。



- キーボードの「Ctrl」キーを押しながら「V」キーを押します。

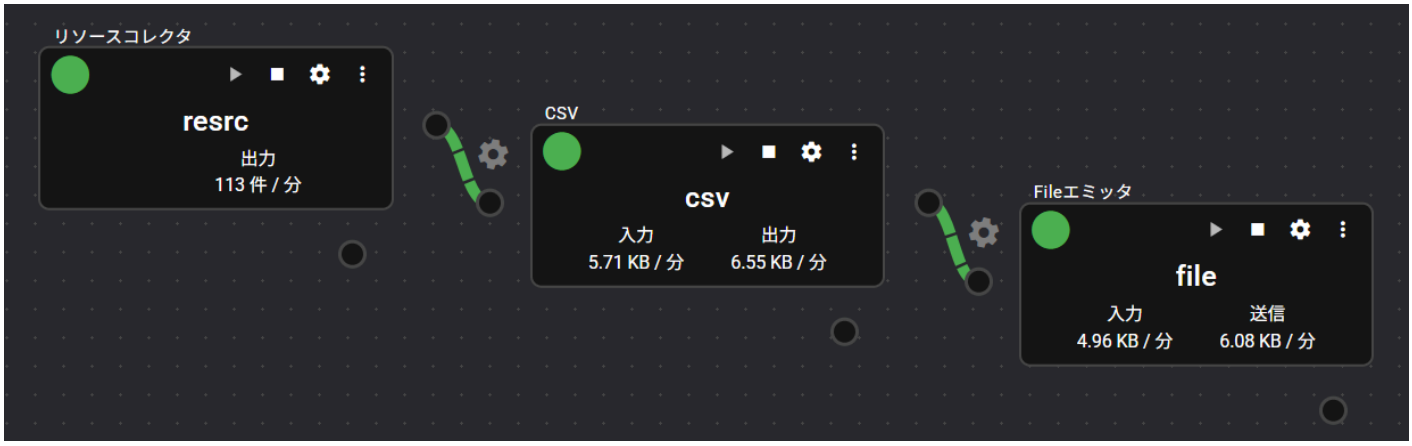
3. コピーしたものと同じ設定内容をもつコンポーネントが作成されます。



この操作はEdit Modeでのみ行うことができます。

5.3.3.4.5 複数選択

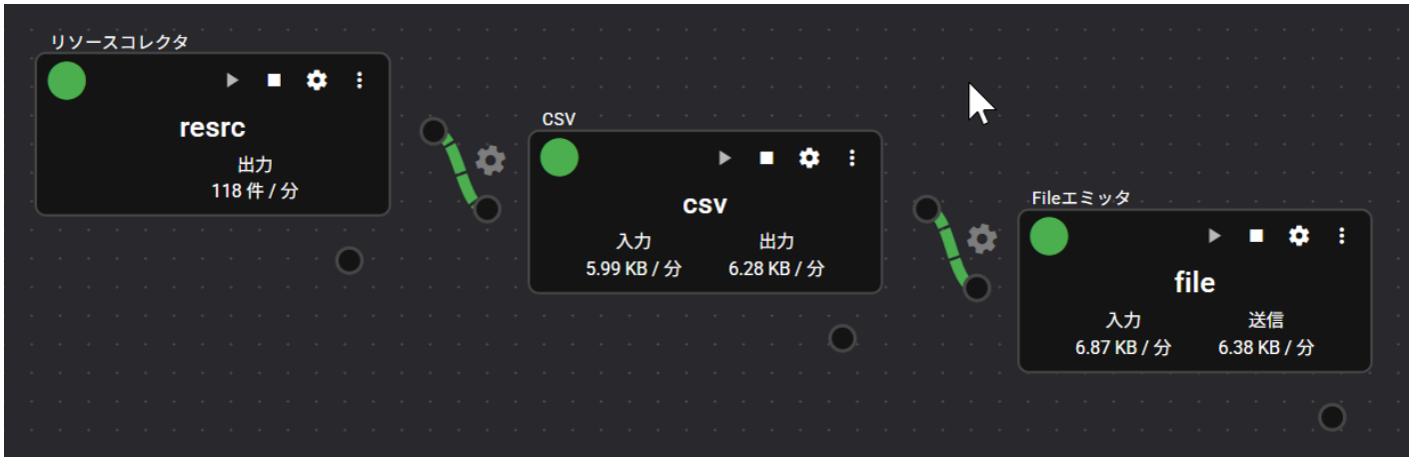
キーボードの「Ctrl」キーを押しながらそれぞれのコンポーネントをクリックすると、コンポーネントを複数選択することができます。



複数選択後、コンテキストメニューやキーボード操作から選択したコンポーネントの「コピー&ペースト」や「削除」を行うことができます。



複数選択を解除するには、パネル内のコンポーネント以外の場所をクリックします。



この操作はEdit Modeでのみ行うことができます。

5.3.3.4.6 設定内容の確認

次の手順で、コンポーネントの設定内容を確認することができます。

1. コンポーネントの設定アイコンを押下します。



2. ダイアログが開き、コンポーネントの設定項目と、その設定値が表示されます。



この操作はRun Modeでのみ行うことができます。

5.3.3.4.7 稼働状況

次の手順で、コンポーネントの稼働状況を確認することができます。

- ステータス

  ステータスランプに、コンポーネントの稼働ステータスが表示されます。稼働ステータスの詳細は、「ステータス一覧」をご覧ください。



- データ量

  コンポーネントの以下のデータ量が表示されます。

  - 1分毎の入力・出力データ
  - 1分毎の受信・送信データ

クリックすると、データの件数・バイト数のどちらを表示するか、切り替えることができます。



5.3.3.4.8 起動/停止

次の手順で、コンポーネントを起動/停止することができます。

(全コンポーネント)

1. ヘッダの「起動」ボタンを押下します。



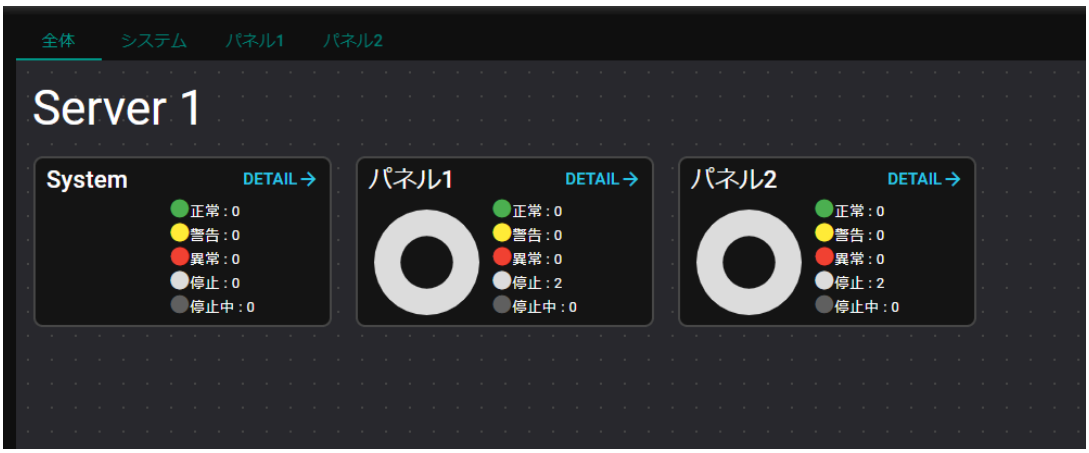
2. SpeeDBee Synapseに登録されている、全てのデータ収集フローの稼働を開始します。



3. ヘッダの「停止」ボタンを押下します。

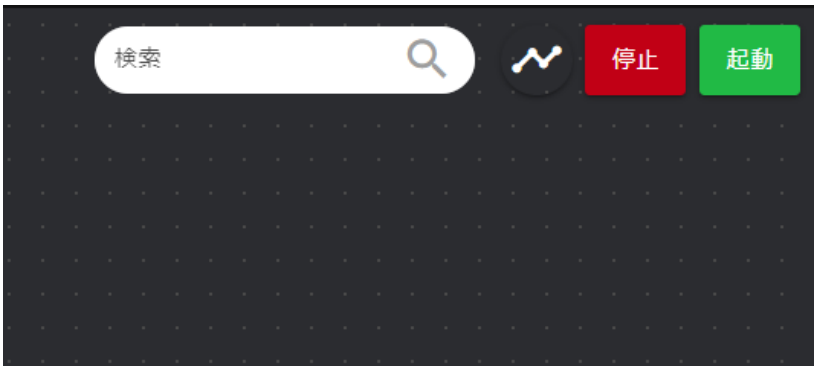


4. SpeeDBee Synapseに登録されている、全てのデータ収集フローの稼働を停止します。

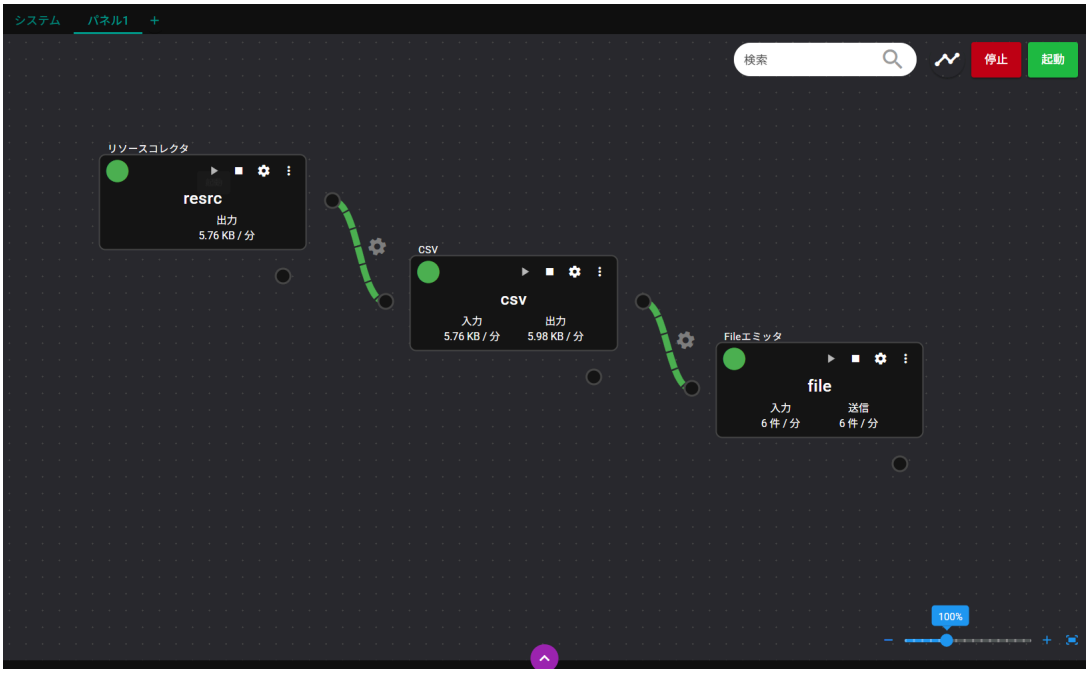


(パネル単位)

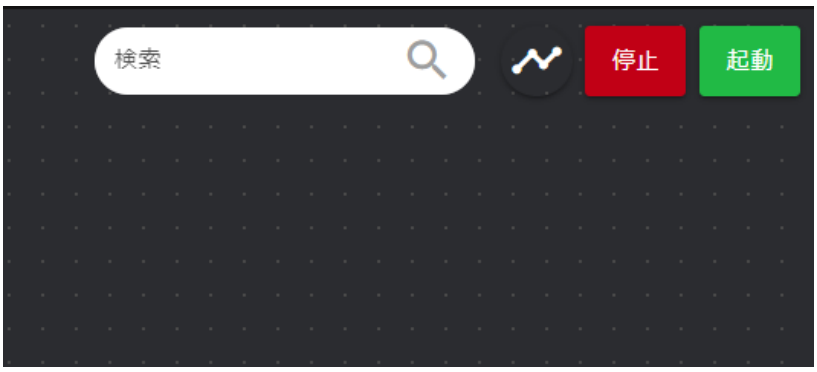
1. パネルの「起動」ボタンを押下します。



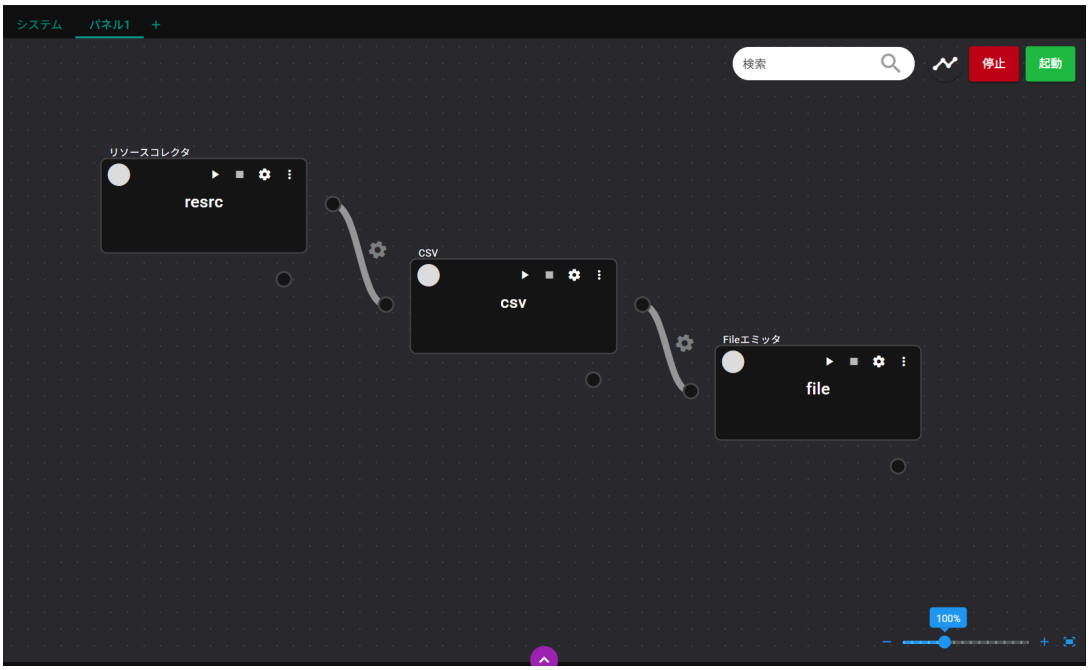
2. パネルに登録されているデータ収集フローの稼働を開始します。



3. パネルの「停止」ボタンを押下します。



4. パネルに登録されているデータ収集フローの稼働を停止します。



(コンポーネント単体)

1. コンポーネントの起動アイコンを押下します。



2. コンポーネントが起動し、処理を開始します。



3. コンポーネントの停止アイコンを押下します。



4. コンポーネントが停止し、処理を終了します。



権限が閲覧者のユーザーは、コンポーネントの起動/停止を行うことはできません。

5.3.3.4.9 操作対象コンポーネントのマニュアル表示

操作対象のコンポーネント表示時に、対象画面のマニュアルを表示できます。

1. コンポーネント表示時にマニュアルアイコンを押下します。



2. 操作対象のコンポーネントのマニュアルが別ウィンドウで開きます。



5.3.3.5 フローリンク

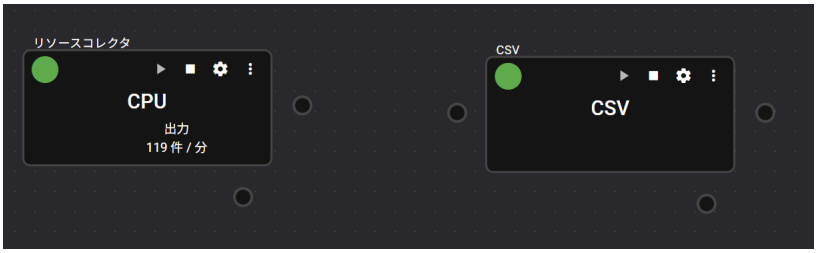
ここでは、フローリンクの操作について説明します。

> この操作はEdit Modeでのみ行うことができます。

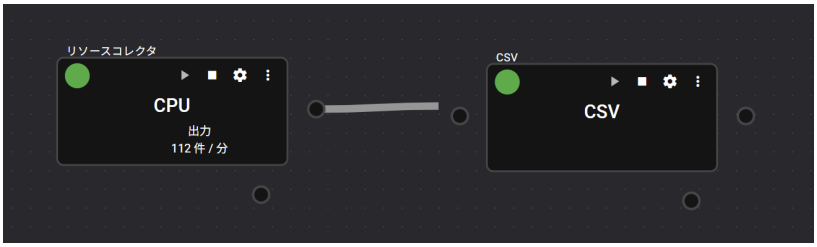
5.3.3.5.1 追加

次の手順で、フローリンクを追加することができます。

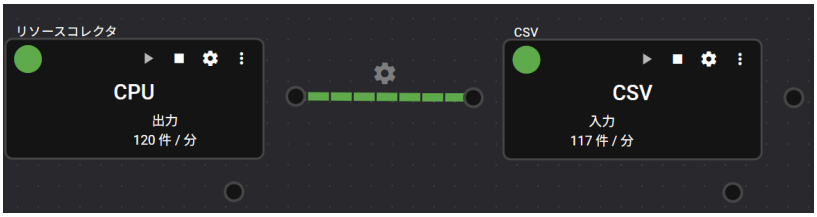
1. コンポーネントの出力ポートまたはシステムポートをドラッグします。



2. 他のコンポーネントの入力ポートでマウスのボタンを離します。



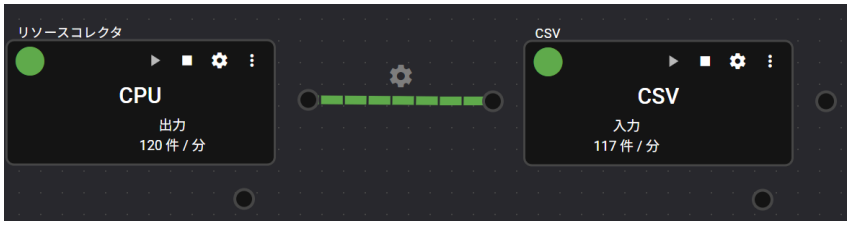
3. コンポーネントの間にフローリンクが追加されます。



5.3.3.5.2 フィルタ

フローリンクアイコンを左クリック、またはフローリンクを右クリックするとフィルタを設定することができます。

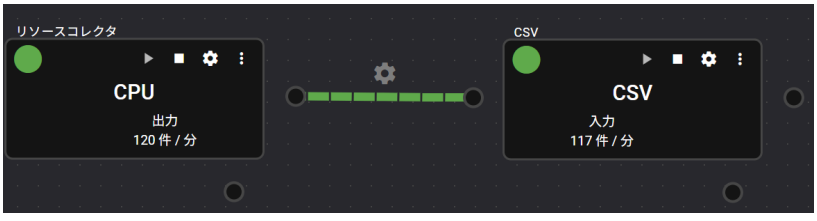
フローリンク上を流れるデータを制限する設定を行うことができます。 詳細は「フィルタ」をご覧ください。



5.3.3.5.3 削除

次の手順で、フローリンクを削除することができます。

1. フローリンクを右クリックします。またはアイコンを左クリックします。



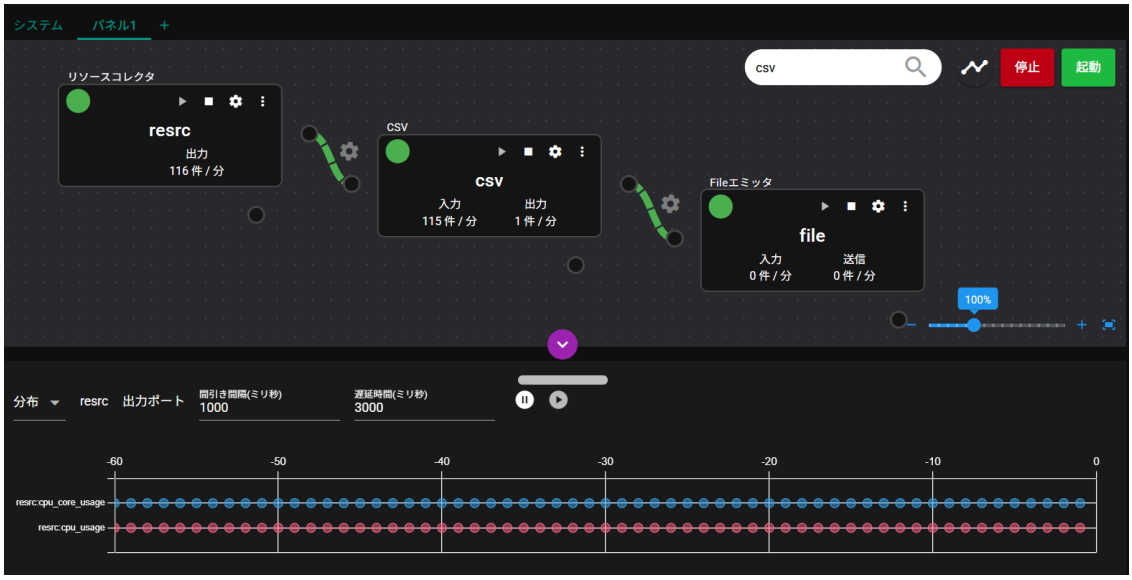
2. 「フローリンク削除」を選択します。



3. フローリンクが削除されます。

5.3.3.6 データモニタ

コンポーネントのポートをクリックすると、ポートに流れているデータを、データモニタで確認することができます。

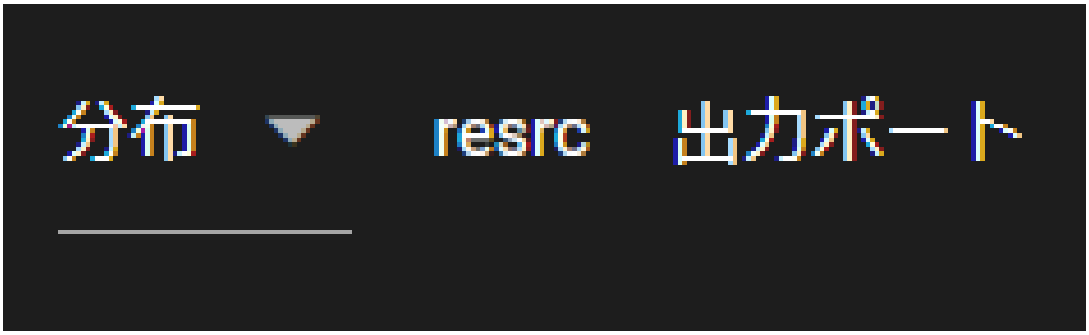


5.3.3.6.1 設定の変更

種類の変更

次の手順で、データモニタの種類を変更することができます。

1. データモニタの表示種類選択フォームをクリックします。



2. 種類を選択します。



3. 選択した種類のデータモニタが表示されます。

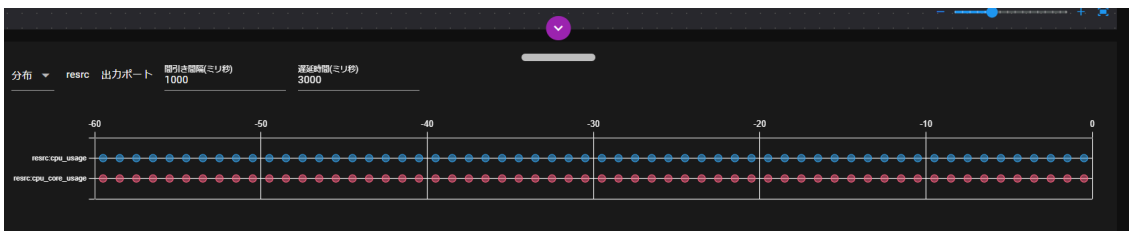
間引き間隔の設定

データモニタは、一定間隔ごとにデータを間引いて表示しています。
「間引き間隔(ミリ秒)」に間引きの間隔を入力すると、データモニタに反映されます。



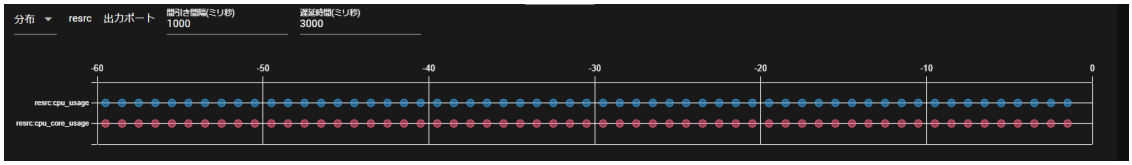
遅延時間の設定

データモニタには、現在時刻から「遅延時間(ミリ秒)」を引いた時刻のデータが、最新値として表示されます。
「遅延時間(ミリ秒)」に値を入力すると、データモニタに反映されます。



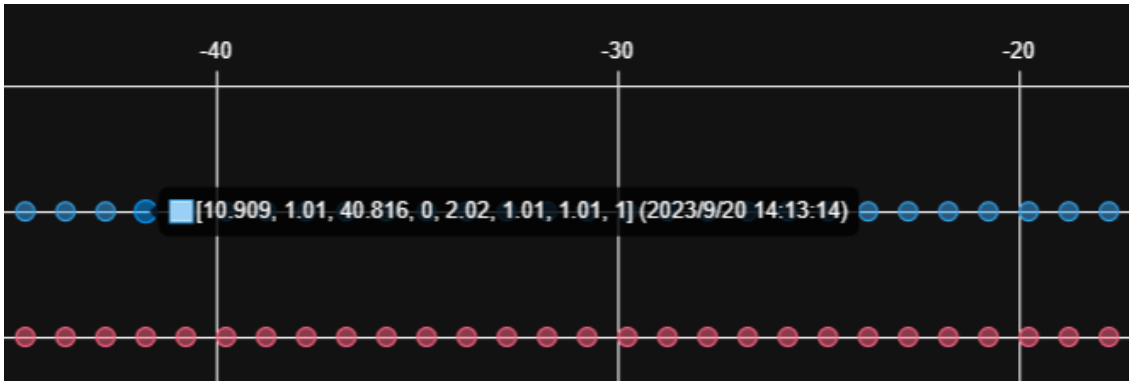
5.3.3.6.2 分布

1分の間にポートに流れたデータが、グラフに表示されます。 グラフには以下の情報が表示されます。



項目	説明
横軸	時刻。0は現在時刻から「遅延時間(ミリ秒)」を引いた時刻で、-xはその時刻からx秒前の時刻。
縦軸	データの項目名
点	データ

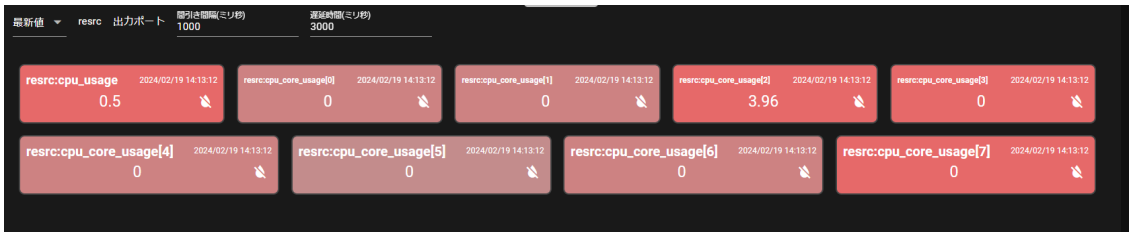
点にカーソルを合わせると、データの内容・データが流れた時刻が表示されます。



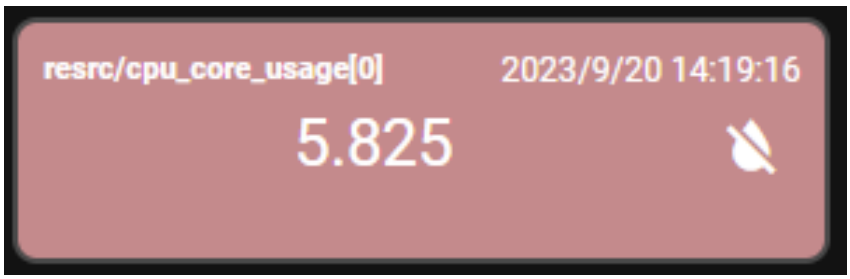
5.3.3.6.3 最新値

ポート上の全てのデータ項目において、以下の情報が表示されます。

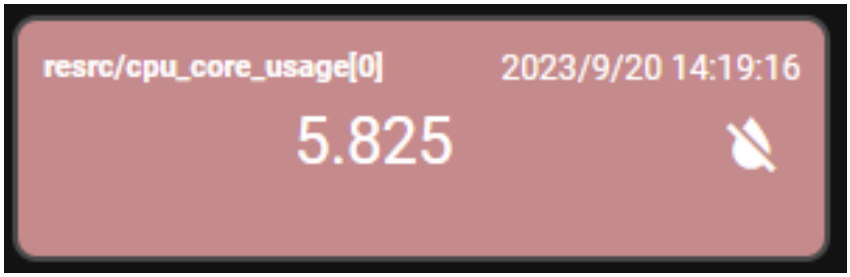
- 最新値
- 最新値が流れた時間



データの値が変化したものは、赤色で表示されます。



クリアボタンを押下すると、赤色に変化したデータ項目が、元の色に戻ります。





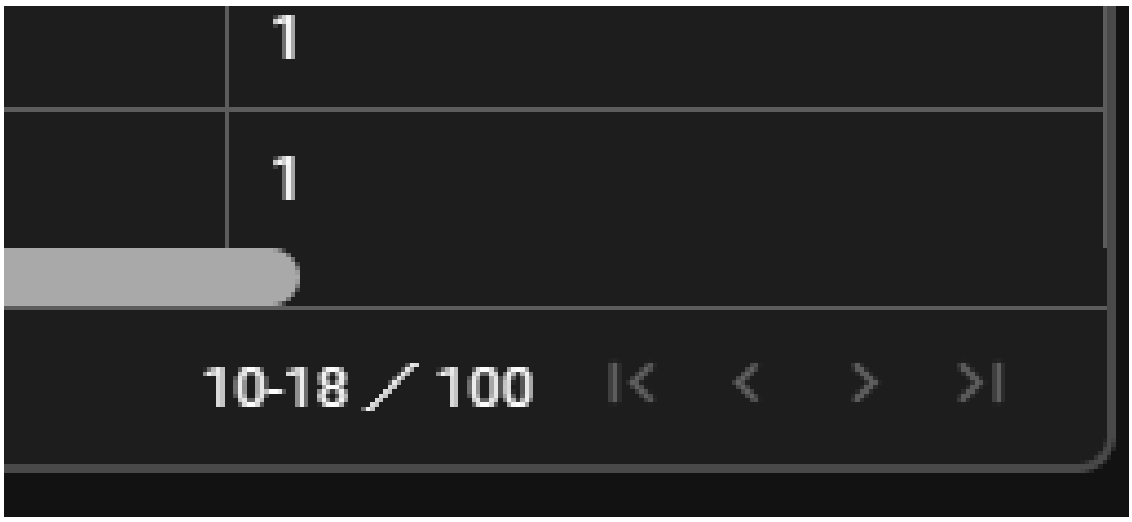
5.3.3.6.4 テーブル

ポートに流れたデータが、テーブル形式で表示されます。テーブルには以下の情報が表示されます。 1ページ毎に表示するテーブルの行数は、データモニタの高さによって自動で調整されます。

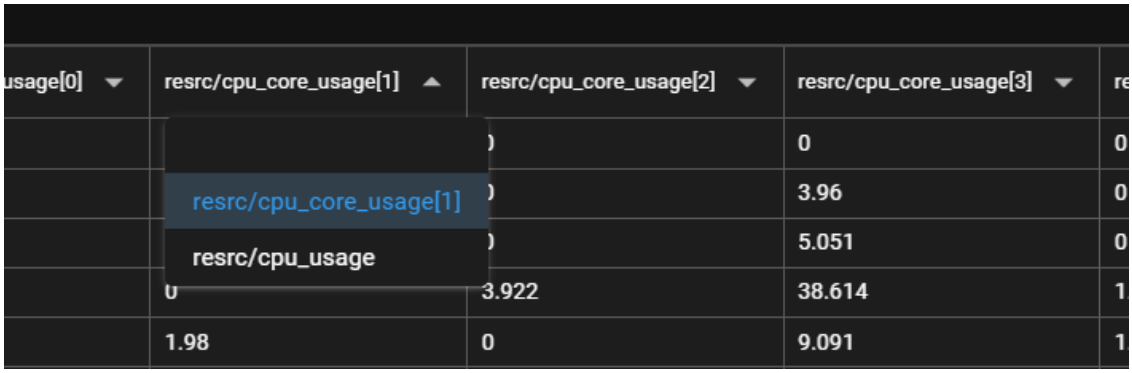
項目	説明
列	データの項目名。列数は最大で8。
行	データが流れた時刻。行数は最大で100。
内容	データ



テーブルの行数が1ページに収まらない場合、ページを切り替えることができます。



ポート上のデータ項目数が8を超える場合、テーブルの列の見出しを選択すると、その列に表示するデータ項目を変更することができます。 空白を選択すると、その列は非表示になります。



5.3.3.6.5 ファイル

データがファイル形式の場合、以下のファイルの情報が表示されます。

- ファイルサイズ...ファイルのサイズ。
- メディアタイプ...ファイルのメディアタイプ。
- タイムスタンプ...ファイルの内容に入っているデータの時刻範囲
- ファイルの内容



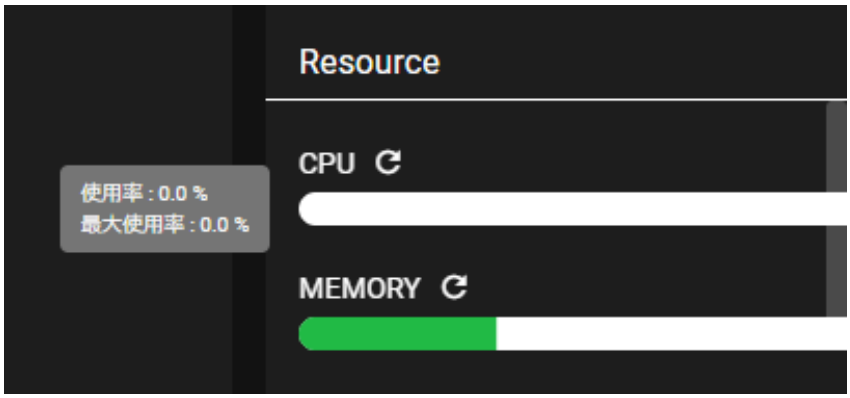
5.3.3.7 システムリソース

ここでは、システムリソースについて説明します。

この操作はRun Modeでのみ行うことができます。

5.3.3.7.1 CPU

ゲージにカーソルを合わせると、以下の情報が表示されます。



項目	説明
使用率	SpeeDBee Synapseの現在のCPU使用率
最大使用率	SpeeDBee SynapseのCPU使用率の最大値

ゲージの濃い色の部分が現在のCPU使用率、薄い色の部分が動作中に達した最大CPU使用率を表します。



ゲージの色は、CPU使用率のパーセンテージに合わせて、次の表のように変化します。

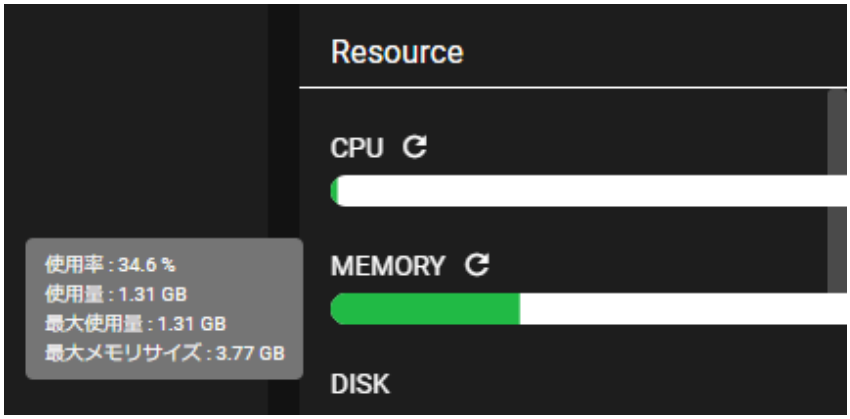
パーセンテージ	色
0~40	緑
40.1~70	黄
70.1~100	赤

リフレッシュアイコンを押下すると、最大使用率がリセットされます。



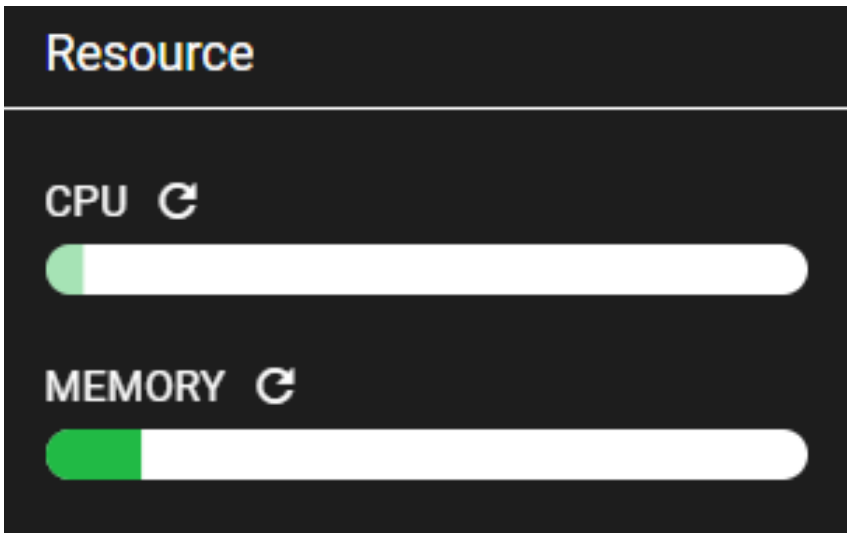
5.3.3.7.2 MEMORY

ゲージにカーソルを合わせると、以下の情報が表示されます。



項目	説明
使用率	SpeeDBee Synapseの現在のメモリ使用率
使用量	SpeeDBee Synapseの現在のメモリ使用量
最大使用量	SpeeDBee Synapseのメモリ使用率の最大値
最大メモリサイズ	システム全体のメモリサイズ

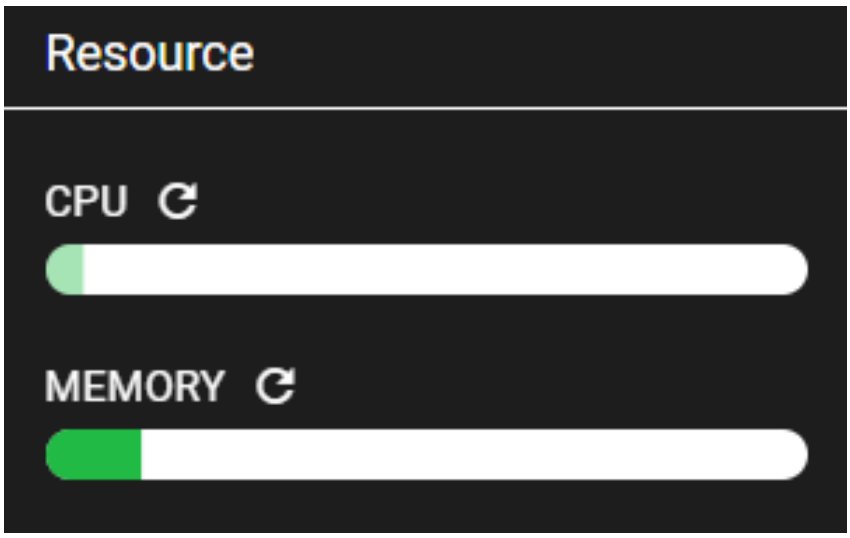
ゲージの色付きの部分が、メモリ使用率を表します。



ゲージの色は、メモリ使用率のパーセンテージに合わせて、次の表のように変化します。

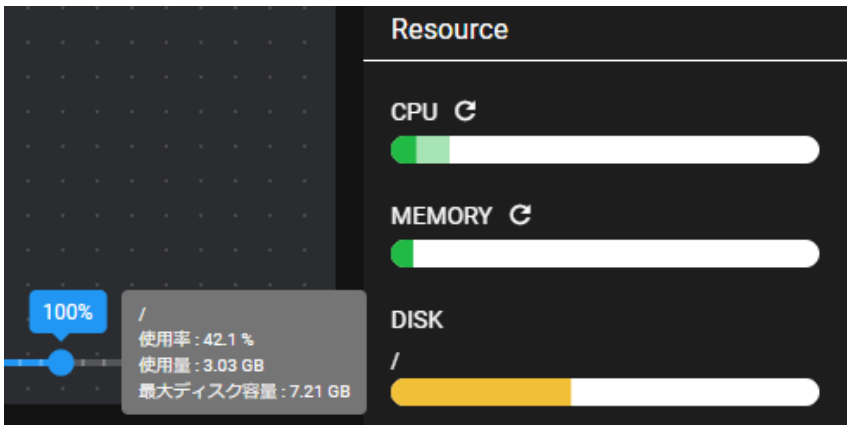
パーセンテージ	色
0~40	緑
40.1~70	黄
70.1~100	赤

リフレッシュアイコンを押下すると、最大使用量がリセットされます。



5.3.3.7.3 DISK

ゲージにカーソルを合わせると、以下の情報が表示されます。



項目	説明
使用率	最大ディスク容量に対する、SpeeDBee Synapseの現在のディスク使用の割合
使用量	SpeeDBee Synapseの現在のディスク使用量
最大ディスク容量	SpeeDBee Synapseが使用できるディスクの最大容量
ディスクフル予想時間	1分間のディスク使用量の増加をもとに算出された、ディスク使用量がディスク最大容量に達するまでの予想時間 ※ディスク使用量の増加がない場合は表示されない

ゲージの色付きの部分が、ディスク使用率を表します。



ゲージの色は、ディスク使用率のパーセンテージに合わせて、次の表のように変化します。

パーセンテージ	色
0~40	緑
40.1~70	黄
70.1~100	赤

5.3.3.8 シミュレーションモード

シミュレーションモードについて説明します。

5.3.3.8.1 シミュレーションモードの概要

シミュレーションモードは、CSVファイルの内容をもとにカラムを生成し、ダミーデータを登録する機能です。 本機能により、実際の対向機（PLCなど）がない環境でも設定や動作検証が可能になります。 シミュレーションモードに対応しているコンポーネントは以下の通りです。

- Modbus TCPコレクタ
- PLCコレクタ
- EtherNet/IPコレクタ
- MQTTコレクタ

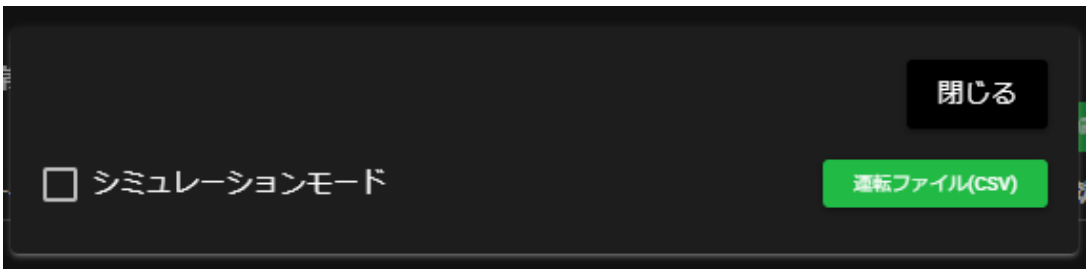
5.3.3.8.2 シミュレーションモードの設定・切替

ここでは、PLCコレクタを用いて説明します。

1. コンポーネント設定画面の左上のシミュレーション設定ボタンをクリックします。

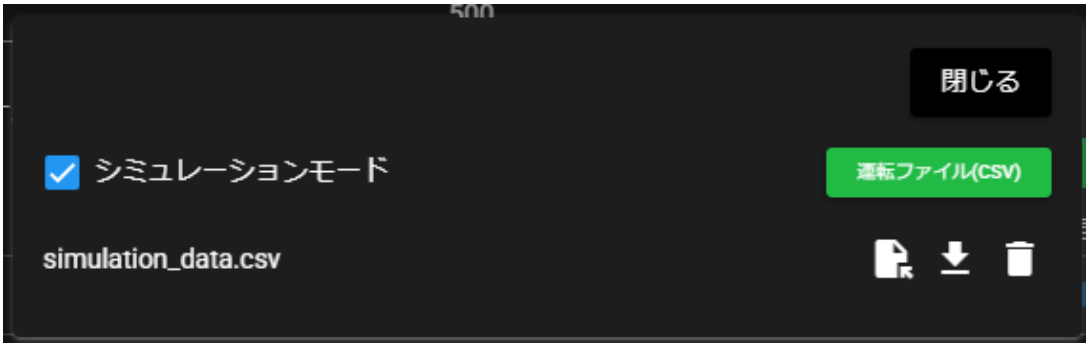


2. シミュレーション設定画面が開くので、設定項目を入力し「閉じる」をクリックします。



項目	説明
シミュレーションモード	シミュレーションモードの有効・無効
運転ファイル(csv)	シミュレーションモードで使用するカラムやデータを定義したCSV形式のファイル。詳細は、運転ファイルのフォーマットを参照してください。

運転ファイルは、一度アップロードすると、削除するまではその設定が保持されます。



3. コンポーネント設定画面に戻るので、コンポーネントの設定項目を入力し「保存」をクリックします。 コンポーネント設定画面で「保存」をクリックするまでは、シミュレーション設定の変更は反映されません。
4. 起動ボタンをクリックして起動します。 シミュレーションモードONの状態で起動している場合、コンポーネントのステータスランプが青になります。



5.3.3.8.3 運転ファイルのフォーマット

運転ファイルは、CSV形式となっています。 ヘッダ部（1行目）とデータ部（2行目以降）があり、ヘッダ部でカラム定義、データ部でタイムスタンプと登録するデータを定義します。

example.csv

```
timestamp_ms,data1:bool,data2:int32,data3:string,data4:binary
        1000,      true,          1,       test1,    YWl1ZW8=
        2000,     false,          2,       test2,    YWl1ZW8=
        3000,      true,          3,       test3,    YWl1ZW8=
```

ヘッダ部

```
timestamp_ms,data1:bool,data2:int32,data3:string,data4:binary
```

- ヘッダ部の1列目はtimestamp_ms固定、2列目以降にカラム定義を入力します。
- カラム定義は、`カラム名:データ型`のように、`:`区切りでカラム名とデータ型を指定します。
- 配列の場合は、`カラム名:データ型[配列数]`のように、`[]`で配列数を囲って指定します。

データ部

```
        1000,      true,          1,       test1,    YWl1ZW8=
        2000,     false,          2,       test2,    YWl1ZW8=
        3000,      true,          3,       test3,    YWl1ZW8=
```

- データ部の1列目はtimestamp_msの値、2列目以降に実際に登録するデータの値を入力します。
- 起動してからtimestamp_msの値経過後の時間でデータを登録します。上記のCSVの例だと起動から1000ミリ秒、2000ミリ秒、3000ミリ秒後の時間でデータを登録します。
- timestamp_msが前回値よりも小さい（時刻が戻る）場合、その行のデータは登録されません。
- ファイルの終端に達すると、先頭のデータに戻り、再び先頭からデータが登録されます。
- データの値が空白の場合、データは登録されません。
- 配列データは、スペース区切りで指定します。

指定可能なデータ型	備考
bool	true, false, TRUE, FALSEで指定。
int8	int8の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
int16	int16の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
int32	int32の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
int64	int64の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
uint8	uint8の範囲で指定。プレフィックスに0xを付けることでHex指定可能。

指定可能なデータ型	備考
uint16	uint16の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
uint32	uint32の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
uint64	uint64の範囲で指定。プレフィックスに0xを付けることでHex指定可能。
float	floatの範囲で指定。プレフィックスに0xを付けることでHex指定可能。
double	doubleの範囲で指定。プレフィックスに0xを付けることでHex指定可能。
string	CSVにおいて特殊文字(`,` `\n` など)を指定する際は `"` で囲う。 `"` のエスケープは `"` 。
binary	Base64形式で指定。

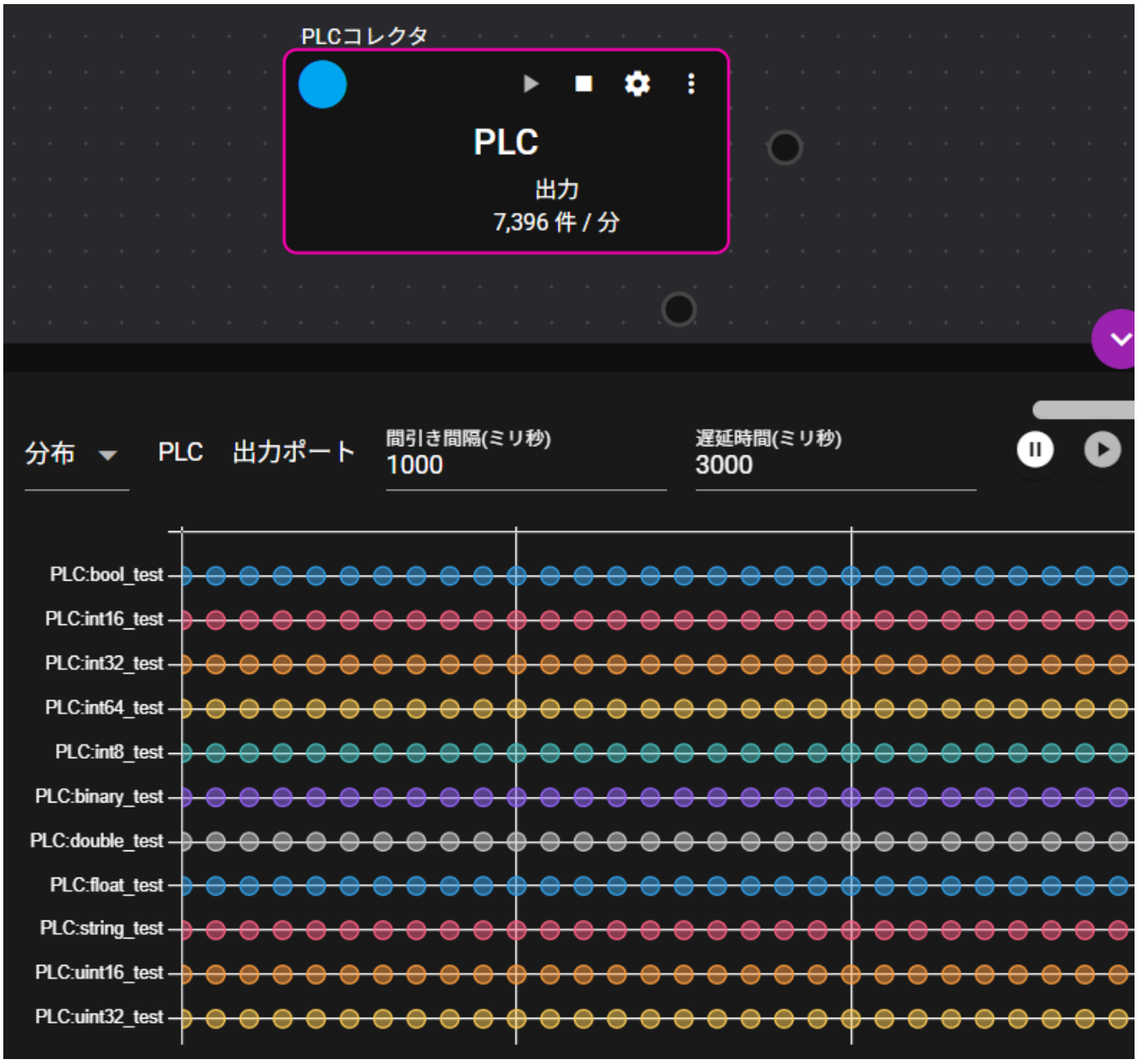
運転ファイルのサンプルは以下の通りです。

- 全データ型サンプル
- 配列データサンプル

5.3.3.8.4 シミュレーションモードの動作

シミュレーションモードで動作中は、ダミーデータの登録のみを行います。 実際の機器との通信によるデータ収集は行われません。 また、シミュレーションモード動作中のコンポーネントステータスについて、 ステータスランプが青色で表示されますが、状態自体は通常の `Running` と同じです。

- 登録されたダミーデータは、通常のコンポーネントと同様にデータモニタリングで確認することができます。



- フローリンクを繋いで別のコンポーネントへデータを伝達することも可能です。



シミュレーションモードは、開発中のデータフローの検証には問題ありませんが、 本番環境で運用する際は、OFFにするのを忘れないようご注意ください。

5.3.3.9 タブレット端末での操作方法

SpeeDBee Synapseは、タブレット端末での操作に対応しています。 検証済み端末は、次のとおりです。

機種名/モデル	OSバージョン	ブラウザ
iPad(A16)	iOS 18.5	Safari 605.1.15
HP Spectre x360 Convertible 14-ea0xxx	Windows 11 24H2	Google Chrome 136.0.7103.114

タブレット端末での動作は、PCのマウス操作と一部異なります。以下に主な違いをまとめますので、ご参照ください。

機能・操作	PCでの操作方法	タブレットでの操作方法
コンテキストメニュー	右クリック	対象要素をロングタップ(長押し)します
ドラッグ&ドロップ	マウスでのドラッグ	指で対象をタップし、そのまま移動することでドラッグ操作を行います
ホバー操作	マウスオーバーで表示	タッチデバイスではホバー操作は行えませんが、一部の操作はタップやロングタップで代用されています。

5.4 コレクタ

ここでは、各コレクタコンポーネントについて説明します。

5.4.1 リソースコレクタ

稼働マシンのシステムリソースを収集するコンポーネントです。
本アプリが稼働しているの機器のCPU使用率、メモリ消費などのリソースの使用状況を収集して登録します。



5.4.1.1 基本情報

リソースコレクタの基本情報は、次の通りです。

項目	説明
ベース名	リソースコレクタ
説明	稼働マシンのシステムリソースを収集します。
消費ポイント	0
ポート	入力：0 出力：1 収集した稼働マシンのシステムリソースを出力します。出力するデータの詳細は、「出力データ項目」をご覧ください。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.4.1.2 設定項目

リソースコレクタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
取得周期(秒)	コレクタがデータを取得する周期を入力
I/Oデバイス	データを収集するI/Oデバイスを選択
NETデバイス	データを収集するネットワークデバイスを選択

項目	説明
CPU使用率	稼働マシンのCPU使用率を収集する場合ON
CPUクロック数	稼働マシンのCPUクロック数を収集する場合ON ※Windowsでは使用できません ※Linuxでも使用できないことがあります
CPU温度	CPU温度を収集する場合ON ※Windowsでは使用できません ※Linuxでも使用できないことがあります
I/O	「I/Oデバイス」で選択したデバイスのI/Oの状態を収集する場合ON ※Windows版において、I/Oの収集を有効にすると、取得周期1秒では、処理が追い付かないことがあります（遅延する場合、周期を5秒以上に設定して調整してください）
メモリ	稼働マシンのメモリ使用量を収集する場合ON
ネットワーク	「NETデバイス」で選択したデバイスのネットワークの状態を収集する場合ON

5.4.1.3 出力データ項目

リソースコレクタは、次のデータを出力します。

項目	カラム名	データ型	説明
CPU使用率	cpu_usage	double	CPU使用率を収集する。単位は %、範囲は0.000～100.000
CPUコア毎の使用率	cpu_core_usage	double[n]	CPUコア毎の使用率を収集する。単位は %、範囲は0.000～100.000
CPU実行周波数	cpu_core_freq	int32[n]	CPUコア毎の実行周波数を収集する。単位は kHz
CPU温度	cpu_temp	double	CPU温度を収集する。単位は℃
IO情報収集デバイス名	I_O	string	IO情報収集デバイス名(物理のみ)

項目	カラム名	データ型	説明
平均読込みリクエスト処理数	kb_r_s	double	平均読込みリクエスト処理数を収集する。単位はKB
平均書込みリクエスト処理数	kb_w_s	double	平均書込みリクエスト処理数を収集する。単位はKB
平均読込・書込み回数	IOPS	double	平均読込・書込み回数を収集する。単位はKB
ディスク空き容量(MB)	storage_free_space_mb	double	指定されたディスクの空き容量を収集する。単位はMB。
ディスク使用率(%)	storage_usage	double	指定されたディスクの使用率を収集。単位は%、範囲は0.0~100.0
メモリ全容量	mem_total	int32	メモリ全容量を収集する。単位はKB
メモリ使用量	mem_used	int32	メモリ使用量を収集する。単位はKB
メモリ使用率	mem_usage	double	メモリ使用率を収集する。単位は%、範囲は0.0~100.0
SWAP全容量	swap_total	int32	SWAP全容量を収集する。単位はKB
SWAP使用量	swap_used	int32	SWAP使用量を収集する。単位はKB
SWAP使用率	swap_usage	double	SWAP使用率を収集する。単位は%、範囲は0.0~100.0
合計受信バイト数	rx_bytes1	uint64	受信バイト数を収集する。単位はbytes
合計受信パケット数	rx_pckt1	uint64	受信パケット数を収集する。
合計送信バイト数	tx_bytes1	uint64	送信バイト数を収集する。単位はbytes
合計送信パケット数	tx_pckt1	uint64	送信パケット数を収集する。

5.4.2 PLCコレクタ

PLCのレジスタ/変数情報からデータ収集を行うコンポーネントです。
オムロンや三菱など多様なメーカーのPLCと接続して、レジスタ/変数情報からデータを収集、登録します。
一部のPLCでは、ラダー編集ツールで出力したファイルをインポートしてレジスタ情報を設定できます。



5.4.2.1 基本情報

PLCコレクタの基本情報は、次の通りです。

項目	説明
ベース名	PLCコレクタ
説明	PLCのレジスタ/変数情報からデータを収集します。
消費ポイント	0
ポート	入力：0 出力：1 「設定項目」に従って、PLCから収集したデータを出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	・PLCに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。 ・アドレスを自動で設定することができます。詳細は「アドレス自動設定」をご覧ください。

5.4.2.2 設定項目

PLCコレクタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
メーカ	接続するPLCのメーカを選択 ※選択できるメーカは次の通りです。 ・三菱電機 ・ジェイテクト ・オムロン ・キーエンス ・パナソニック

項目	説明
機種	接続するPLCの機種を選択 ※選択できる機種はメーカによって異なります。
ホスト	接続するPLCのホスト名またはIPアドレスを入力
ポート	接続するPLCのポートを入力
接続タイムアウト(ミリ秒)	接続時のタイムアウト時間を入力
読込タイムアウト(ミリ秒)	読込時のタイムアウト時間を入力
常時接続する	PLC接続を問い合わせの都度、切断しない場合ON
一括取得	一度に複数のワードレジスタの読み込みを行う場合ON
一括ブロック取得	一度に複数のビットレジスタ・ワードレジスタの読み込みを行う場合ON
厳密なタイムスタンプ	収集したデータにより厳密なタイムスタンプを付与する場合ON

- PLCコレクタの出力データは、次の設定項目によって定義されます。

項目	説明
データ名	データの名前を入力
アドレス	PLCのレジスタアドレスを入力
プログラム	レジスタのプログラム番号を選択 ※メーカがジェイテクトの場合のみ選択可能
データ型	取得するレジスタ値のデータ型を選択
連続領域数	入力したアドレスを先頭として連続するレジスタの中から、値取得の対象とする範囲を入力
収集間隔(ミリ秒)	レジスタ値を取得する間隔を入力
有効	レジスタ値を取得する場合ON

5.4.2.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したPLCコレクタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.4.2.4 アドレス自動設定

一部のPLCでは、ラダー編集ツールで出力したファイルを読み込むことで、アドレスを自動で一括設定する事が出来ます。

1. ラダー編集ツールで、作成したラダーをファイルとして出力します。出力するファイルの形式は、PLCのメーカ・機種によって異なります。ファイルのサンプルは次の通りです。
   - 三菱電機 MELSEC-Qシリーズ(QnA互換3Eフレーム)、MELSEC iQ-Rシリーズ(QnA互換3Eフレーム)

     サンプルファイルのダウンロード
   - オムロン SYSMAC CS/CJ/CP/NSJシリーズ(FINS)

     サンプルファイルのダウンロード
2. 「アドレス自動設定」を押下します。



3. ラダー編集ツールで出力したファイルを選択します。
4. ファイルをもとに、アドレスが自動で設定されます。

5.4.3 Modbus TCPコレクタ

Modbus TCP対応機器からデータ収集を行うコンポーネントです。
本コンポーネントがModbusのマスターとなり、Modbusのスレーブ機器からデータを収集、登録します。



5.4.3.1 基本情報

Modbus TCPコレクタの基本情報は、次の通りです。

項目	説明
ベース名	Modbus TCPコレクタ
説明	Modbus TCP対応機器からデータを収集します。
消費ポイント	2
ポート	入力：0 出力：1 「設定項目」に従って、Modbus TCP対応機器から収集したデータを出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	Modbus TCP対応機器に対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.4.3.2 設定項目

Modbus TCPコレクタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	接続するModbus機器のホスト名またはIPアドレスを入力
ポート	接続するPLCのポートを入力
エンディアン	バイトオーダを選択
接続タイムアウト(ミリ秒)	接続時のタイムアウト時間を入力
読込タイムアウト(ミリ秒)	読込時のタイムアウト時間を入力
スレーブID	接続先のスレーブIDを入力
一括取得	一度に複数のワードレジスタの読み込みを行う場合ON
厳密なタイムスタンプ	収集したデータにより厳密なタイムスタンプを付与する場合ON

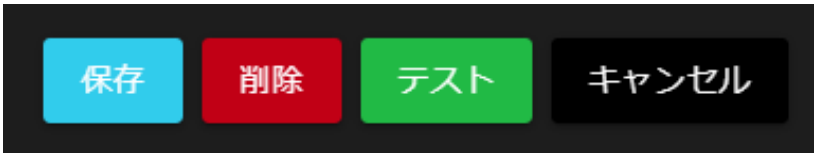
- Modbus TCPコレクタの出力データは、次の設定項目によって定義されます。

項目	説明
データ名	データの名前を入力
アドレス	レジスタアドレスを入力 ※Modbusの仕様上、アドレス範囲は通常「1~65536」となっていますが、SpeeDBee Synapseで設定する際のアドレス範囲は「0~65535」のように-1したアドレスを設定する必要があります。
データ型	取得するレジスタ値のデータ型を選択
連続領域数	入力したアドレスを先頭として連続するレジスタの中から、値取得の対象とする範囲を入力
種類	レジスタ値の種類を選択
エンディアン	レジスタ単位でのバイトオーダを選択
収集間隔(ミリ秒)	レジスタ値を取得する間隔を入力
有効	レジスタ値を取得する場合ON

5.4.3.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したModbus TCPコレクタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.4.4 EtherNet/IPコレクタ

EtherNet/IP対応機器からデータ収集を行うコンポーネントです。
本コンポーネントがオリジネーターとなり、Ethernet/IP対応機器からデータを収集、登録します。



5.4.4.1 基本情報

EtherNet/IPコレクタの基本情報は、次の通りです。

項目	説明
ベース名	EtherNet/IPコレクタ
説明	EtherNet/IP対応機器からデータを収集します。
消費ポイント	2
ポート	入力：0 出力：1 「設定項目」に従って、EtherNet/IP対応機器から収集したデータを出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	変数を自動で設定することができます。詳細は「変数自動設定」をご覧ください。

5.4.4.2 設定項目

EtherNet/IPコレクタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	EtherNet/IP対応機器のホスト名またはIPアドレスを入力
ポート	EtherNet/IP対応機器のポートを入力
接続タイムアウト(ミリ秒)	接続時のタイムアウト時間を入力
取得間隔(ミリ秒)	データを取得する間隔を入力
厳密なタイムスタンプ	収集したデータにより厳密なタイムスタンプを付与する場合ON

- EtherNet/IPコレクタの出力データは、次の設定項目によって定義されます。

項目	説明
設定方法	取得するデータの設定方法を選択
変数名	タグ名を入力 ※設定方法が変数名の場合のみ入力可能です。

項目	説明
構造体	複数のメンバを登録する場合ON ※設定方法が変数名の場合のみ設定可能です。
クラスID	CIPオブジェクトのクラスIDを入力 ※設定方法がCIPObjectの場合のみ入力可能です。
インスタンスID	CIPオブジェクトのインスタンスIDを入力 ※設定方法がCIPObjectの場合のみ入力可能です。
アトリビュートID	CIPオブジェクトのアトリビュートIDを入力 ※設定方法がCIPObjectの場合のみ入力可能です。
カラム名	メンバの名前
データ型	メンバのデータ型
配列数	メンバの配列数

5.4.4.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したEtherNet/IPコレクタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.4.4.4 変数自動設定

Sysmac Studioがエクスポートするグローバル変数情報などを、指定のフォーマットのテキストファイルに変換して読み込むことで、変数を自動設定することができます。

1. Sysmac Studioがエクスポートするグローバル変数情報などを、次の列情報を持つテキストファイルに変換します。
   - HOST
   - NAME
   - DATATYPE
   - ADDRESS
   - COMMENT
   - TAGLINK
   - RW
   - POU

   例

```
HOST    NAME    DATATYPE    ADDRESS COMMENT TAGLINK RW  POU
        b_test  BOOL        test1           TRUE    RW
        i_test  UINT        test2           TRUE    RW
        s_test  STRING(256) test3           TRUE    RW
```

2. 「変数自動設定」を押下します。



3. 用意したテキストファイルを選択します。
4. ファイルをもとに、変数が自動で設定されます。

5.4.5 MQTTコレクタ

MQTTプロトコルを用いてJSONデータの収集を行うコンポーネントです。
本コンポーネントはMQTTで特定のトピック名をサブスクライブします。そこから受信したデータをJSONデータとして解析し、その内容を収集データとして登録します。
受信できるJSONデータのフォーマットは、JSONシリアライザの出力と同じ形式です。
通常のMQTTブローカーとの接続のほか、AWS IoT Coreとも接続可能です。



5.4.5.1 基本情報

MQTTコレクタの基本情報は、次の通りです。

項目	説明
ベース名	MQTTコレクタ
説明	MQTTプロトコル（Subscribe）によるデータ収集を行います。
消費ポイント	2
ポート	入力：0 出力：1 「設定項目」によって指定されたMQTTブローカーから受け取ったデータを出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	MQTTブローカーに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.4.5.2 設定項目

MQTTコレクタの設定項目は、次の通りです。


保存
キャンセル
名称
自動起動無効
ホスト
ポート
1883
MQTT QoS
1
匿名アクセス
ユーザー名
パスワード
暗号化
CA証明書選択
プロキシを利用
プロキシサーバー
トピック
クライアントID（省略可能）
カスタム認証を使用してAWS Iot Coreからデータを受け取る
NULLの値を無視する
キープアライブ(秒)
20
タイムアウト(ミリ秒)
5000
タイムスタンプ誤差許容範囲(秒)
0
動的カラム生成
データ名
データ型
データ数
データなし
追加

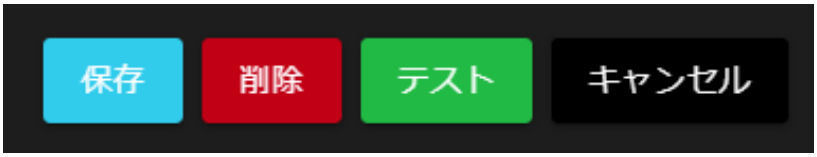
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	MQTTブローカーのホスト名またはIPアドレスを入力
ポート	MQTTブローカーのポートを入力
MQTT Qos	MQTTのQuality of Serviceを選択
匿名アクセス	ユーザー認証を行わない場合ON
ユーザー名	認証ユーザー名を入力 ※匿名アクセスが無効の場合のみ入力が可能です。
パスワード	認証パスワードを入力 ※匿名アクセスが無効の場合のみ入力が可能です。
暗号化	暗号化を行う場合ON
CA証明書	暗号化通信に使用するCA証明書をアップロード ※暗号化が有効の場合のみアップロードが可能です。
プロキシを利用	MQTTブローカーとの通信をプロキシサーバーを経由して行う場合ON
プロキシサーバー	プロキシサーバーを選択 ※事前にプロキシ設定で使用するサーバーを設定しておく必要があります。
トピック	収集対象のトピックを入力
クライアントID	MQTTクライアントID ※接続先仕様によっては指定する必要があります。省略時は、ランダムな文字列が使用されます。 ※複数のコンポーネントで、同一MQTTブローカーへ接続する場合、クライアントIDは被らないように設定してください。通信が正しく行われなくなります。

項目	説明
カスタム認証を使用してAWS Iot Coreからデータを受け取る	AWS Iot Coreのカスタム認証の機能を使用して、AWS IoT Coreからデータを受け取る場合ON ※AWS Iot Coreでカスタム認証の設定を有効にしておく必要があります。くわしくは、こちら を参照してください。
NULLの値を無視する	受信したデータ内の値がNULLだった場合の挙動へ影響します。チェックONの場合は、NULLは無視されエラーになりません。チェックOFFの場合は、NULLの値を受信すると、エラー状態へ遷移します。
キープアライブ(秒)	MQTTキープアライブ間隔を入力
タイムアウト(ミリ秒)	MQTTブローカー接続時のタイムアウト秒数を入力
タイムスタンプ誤差許容範囲(秒)	タイムスタンプ誤差の許容範囲を入力
動的カラム生成	MQTTブローカーから受け取ったデータから、動的にカラムを作成する場合ON ※1 MQTTブローカーからデータを受け取って初めてカラムが生成されます。 ※2 動的に生成されたカラムはコンポーネントが停止するとなくなります。データを使いたい場合は固定カラムとして登録してください。
データ名	固定カラムの名前を入力
データ型	固定カラムの型を選択
データ数	固定カラムの配列数を入力

5.4.5.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したMQTTコレクタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



MQTTコレクタのテスト接続に関する注意点

テスト接続は、次の場合のみ操作可能です。
・コンポーネントが停止している場合
・コンポーネントのステータスがエラーの場合
※ステータスについての詳細は「ステータス一覧」をご覧ください。

5.4.5.4 受信データのフォーマット

MQTTコレクタの受信データのフォーマットは、JSONシリアライザと互換性があります。JSONシリアライザの出力データ項目を確認してください。

mosquittoを使ったデータ送信例

MQTTコレクタでデータを受信するには、JSONの"records"にデータが格納される設定のため、キーは"records"を指定する必要があります。
例えばmosquitto_pubを使用してtopicにデータをPublishする場合、以下のコマンド操作を行います。

```
$ mosquitto_pub -d -h 192.168.1.10 -p 1883 -t "topic" -m '{"records": [{"data1":1,
"data2":"Hello,World"}, {"data3":[111,222,333], "data4":3.14}]}'
```

jsonの第一階層に、"records"がないとデータは受信できません。また、"records"以外は、無視します。

5.4.5.5 クラウドサービスからのデータ受信

MQTTコレクタは、次のクラウドサービスからのデータを受信可能です。それぞれ次の補足事項を確認してください。

AWS IoT Core

AWS IoT Coreへ送信されたデータをMQTTコレクタで受信するには、
「カスタム認証を使用してAWS Iot Coreからデータを受け取る」の設定が必須となります。
また、クラウド側での設定も必要となります。
詳細についてはAWS公式ページの「カスタム認証と認可」を参照してください。
これらの情報は、2024年6月時点のものとなっています。

Azure IoT Hub

Azure IoT Hubのデータを受信するには、IoT Hubの仕様に即した設定が必要となります。また、MQTTコレクタで受信できるのは、「cloud-to-device向けに送信されたデータ」となります。
他のデバイス側からIoT Hub向けに送信されたデータは受信できません。
cloud-to-deviceから複数のデバイス向けにブロードキャストすることはできません。
これらの情報は、2024年6月時点のものとなっています。詳しくは、次の公式ページを確認してください。

- MQTT プロトコルを使用した IoT ハブとの通信
- IoT ハブから cloud-to-device メッセージを送信する
- IoT Hub を使用したクラウドからデバイスへのメッセージの送信 (Python)
- Shared Access Signature を使用して IoT Hub へのアクセスを制御する(SASトークン)

要約した設定内容は以下です。必ず上記の公式ページを確認して設定してください。

- IoT Hubは、Standardレベルである必要があります。Basic、Freeでは使用できません。
- ホスト：{IoT Hubの名称}.azure-devices.net
- ポート：8883
- ユーザー名：{IoT Hubの名称}.azure-devices.net/{デバイスId}/?api-version=2021-04-12
- パスワード：SASトークン
- トピック：devices/{デバイスId}/messages/events/"
- クライアントID：{デバイスId}
- 暗号化：有効
- CA証明書：DigiCert ルート証明書（DigiCert Global Root G2）

Azure Event GridのMQTTブローカーには現在対応していません。

5.4.5.6 各種エミッタからのデータ受信

JSONシリアライザで生成したデータを、各種エミッタでデータ送信して、別のMQTTコレクタで受信することが可能です。

送信側	受信側	可否
MQTTエミッタ	MQTTコレクタ	○ ※同一のMQTTブローカーへの接続設定が必要です。
Cloudエミッタ(AWS用)	MQTTコレクタ	○ ※MQTTコレクタはカスタム認証を設定する必要があります。
Cloudエミッタ(Azure用)	MQTTコレクタ	× ※デバイス向けのデータをMQTTコレクタでは受信できません。

他エミッタからデータを受信する場合の注意点

各種エミッタで送信されたデータを受信する場合、下記の点に注意してください。

1.送信側で1件ごとにエミッタでデータ送信する場合：

例）
送信側のJSONシリアライザのデータを1件で切替
受信側のMQTTコレクタの誤差許容範囲（秒）を10

この場合、データ登録から10秒以内でデータを受信できれば、元のタイムスタンプのままデータは登録されます。ただし、クラウドへのデータ送信が頻繁に行われる点に注意してください。

2.送信側で集約した値をエミッタで送信する場合：

例）
送信側のJSONシリアライザのデータを1件で切替
送信側のJSONシリアライザでは、入力ポートで集約する、任意のデータの平均値とする、ウィンドウ幅60秒
受信側のMQTTコレクタの誤差許容範囲（秒）を10

この場合、指定された幅で、値の平均値が登録され、エミッタで送信されます。コレクタは、これを受信して、元のタイムスタンプでデータを登録します。全データを送信しないことから、ネットワークへの負荷は小さくなります。

3.送信側で一定時間データをためてエミッタで送信する場合：

例）
送信側のJSONシリアライザのデータを60秒で切替
受信側のMQTTコレクタの誤差許容範囲（秒）を0

この場合、すべてのデータが受信した現在時刻で登録されます。元のタイムスタンプでの登録とならない点に注意してください。

4.送信側で一定時間データをためてエミッタで送信する場合（タイムスタンプも元のまま登録）：

これを実現するには、少なくとも以下の設定が必要です。
例）
送信側のJSONシリアライザのデータを10～30秒程度で切替
受信側のMQTTコレクタの誤差許容範囲（秒）を60
MQTTコレクタに接続されるコンポーネントの入力ポートで、遅延時間を61000（ミリ秒）とする

この設定により、送信されたデータのタイムスタンプが60秒以内に収まっていれば、データは登録できます。ただし、送受信の確認を行い、性能的に問題ないか、受信したデータ処理が期待結果となっているかを十分に検証した上でご利用ください。

5.4.6 演算コレクタ

任意の計算式で演算を行うコンポーネントです。
他のコンポーネントが出力したデータを入力ポートから受け取り、そのデータを任意の計算式で計算した結果を新規データとして出力します。
計算式には四則演算や論理演算の他、種々の関数を用いることができます。



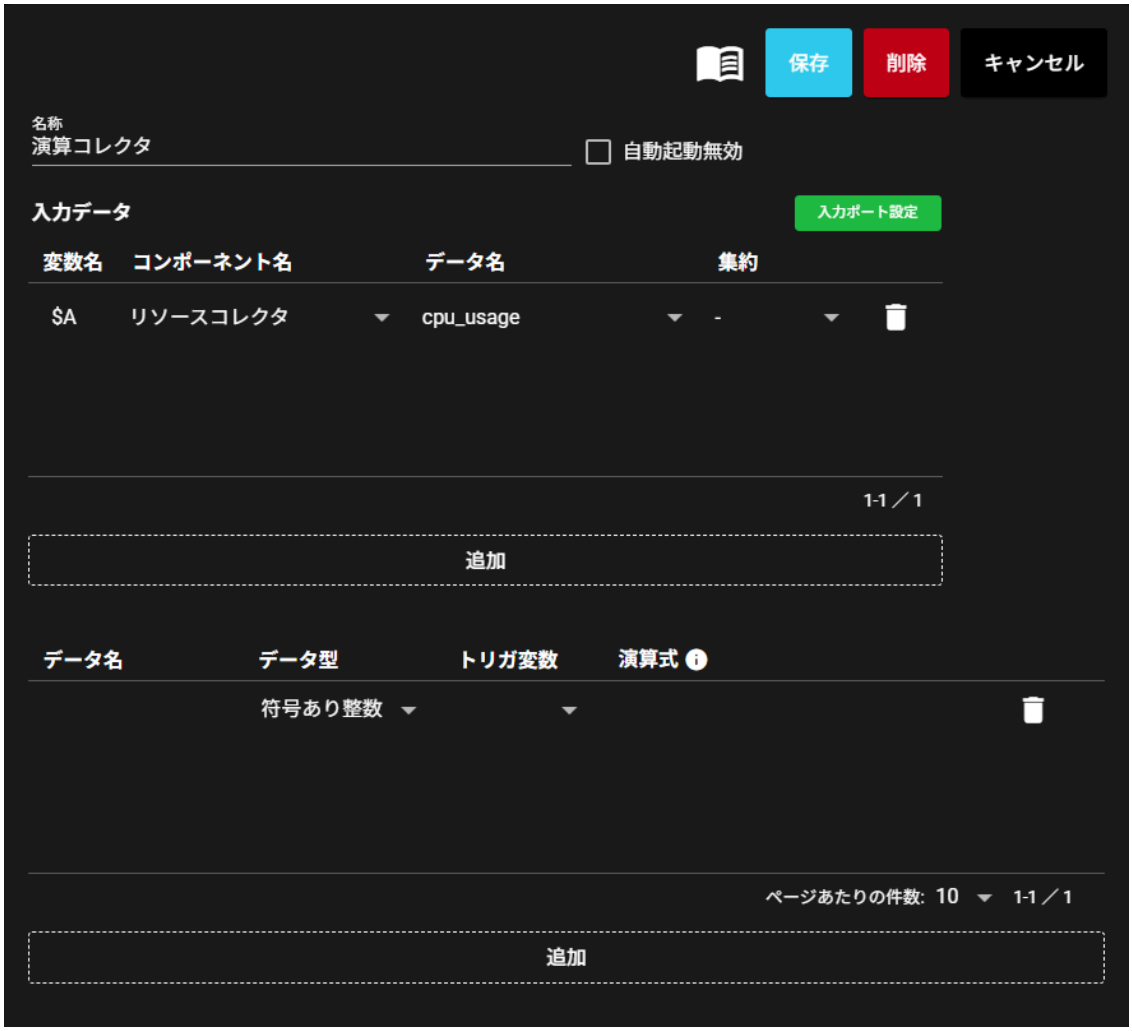
5.4.6.1 基本情報

演算コレクタの基本情報は、次の通りです。

項目	説明
ベース名	演算コレクタ
説明	他のコレクタからデータを受け取って演算を行い、新しい別データとして扱います。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 「設定項目」に従って、演算結果を出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	入力ポートの設定画面を開き、他のコンポーネントから受け取ったデータに集約などの設定を行うことができます。詳細は「入力ポート設定」 をご覧ください。

5.4.6.2 設定項目

演算コレクタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
変数名	入力データに自動で付与される名前
コンポーネント名	入力ポートのデーター覧からコンポーネントを選択
データ名	上で選択したコンポーネントのデーター覧からデータを選択

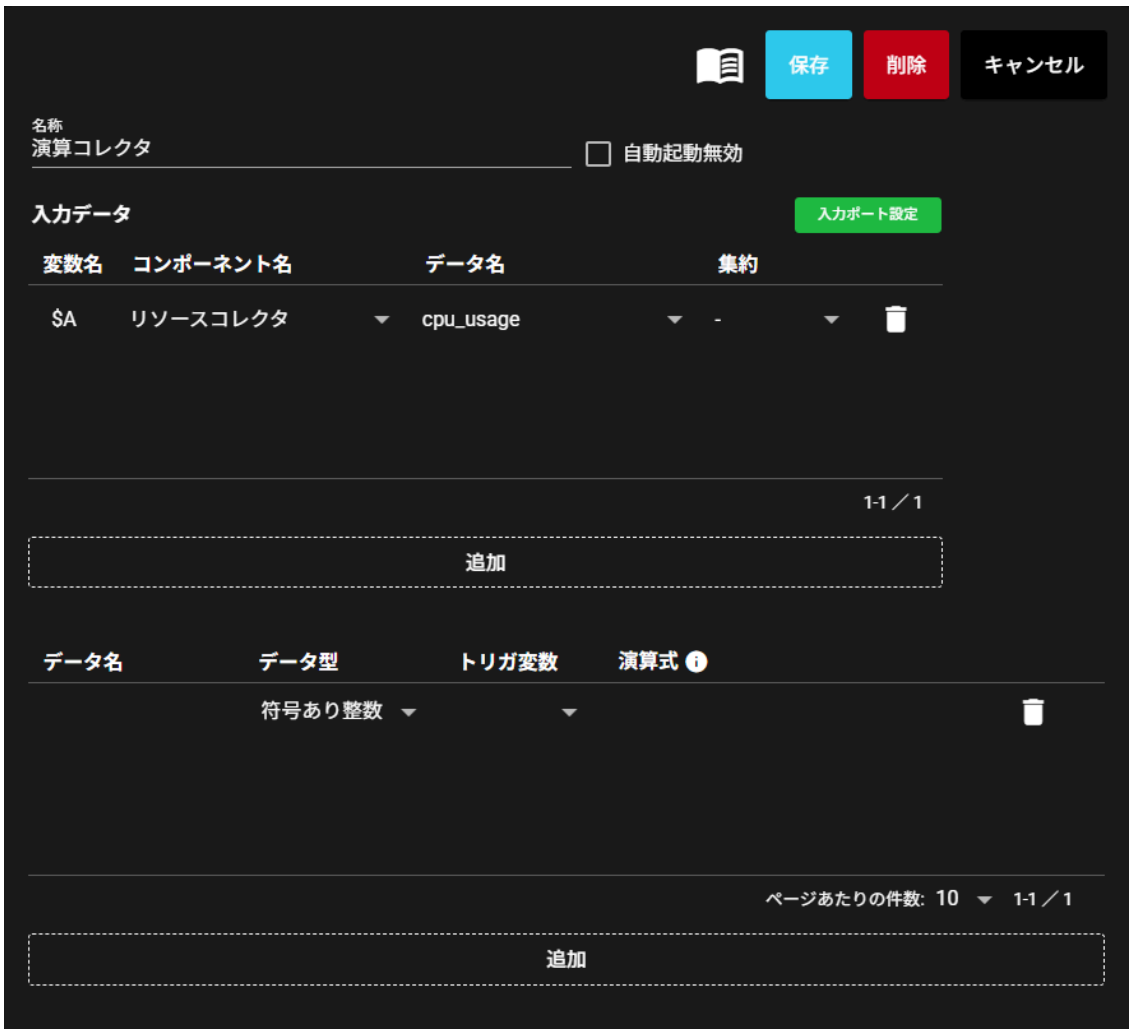
項目	説明
集約	集約値を選択 ※コンポーネントのデータに集約オプションを適用していた場合のみ選択可能です。集約の詳細は「入力ポート」をご覧ください。

- 演算コレクタの出力データは、次の設定項目によって定義されます。

項目	説明
データ名	データの名前を入力
データ型	データの型を選択
トリガ変数	演算を実行するトリガとなる変数 入力データで設定した変数を選択可能です。選択した変数が更新されると演算が実行されます。
演算式	コレクタが実行する演算式を入力 ※表題の「演算式」の右のアイコンをクリックすると、入力例と式に使用できる要素一覧が表示されます。

5.4.6.3 入力ポート設定

追加済みの演算コレクタでは、設定画面で「入力ポート設定」を押下して、演算コレクタの入力ポートの設定画面を開くことができます。 入力ポートの設定画面では、他のコンポーネントから受け取ったデータに集約などの設定を行うことができます。 詳細は「入力ポート」をご覧ください。


保存
削除
キャンセル
名称
演算コレクタ
自動起動無効
入力データ
入力ポート設定
変数名
コンポーネント名
データ名
集約
$A
リソースコレクタ
cpu_usage
-
1-1 / 1
追加
データ名
データ型
トリガ変数
演算式
符号あり整数
ページあたりの件数: 10
1-1 / 1
追加

5.5 シリアライザ

ここでは、各シリアライザコンポーネントについて説明します。

5.5.1 JSONシリアライザ

指定データをJSON形式へ加工するコンポーネントです。
入力ポートからコンポーネントのデータを受け取り、一定時間/件数で「コンポーネント名:データ名」をキーとしたJSON形式に加工して出力します。



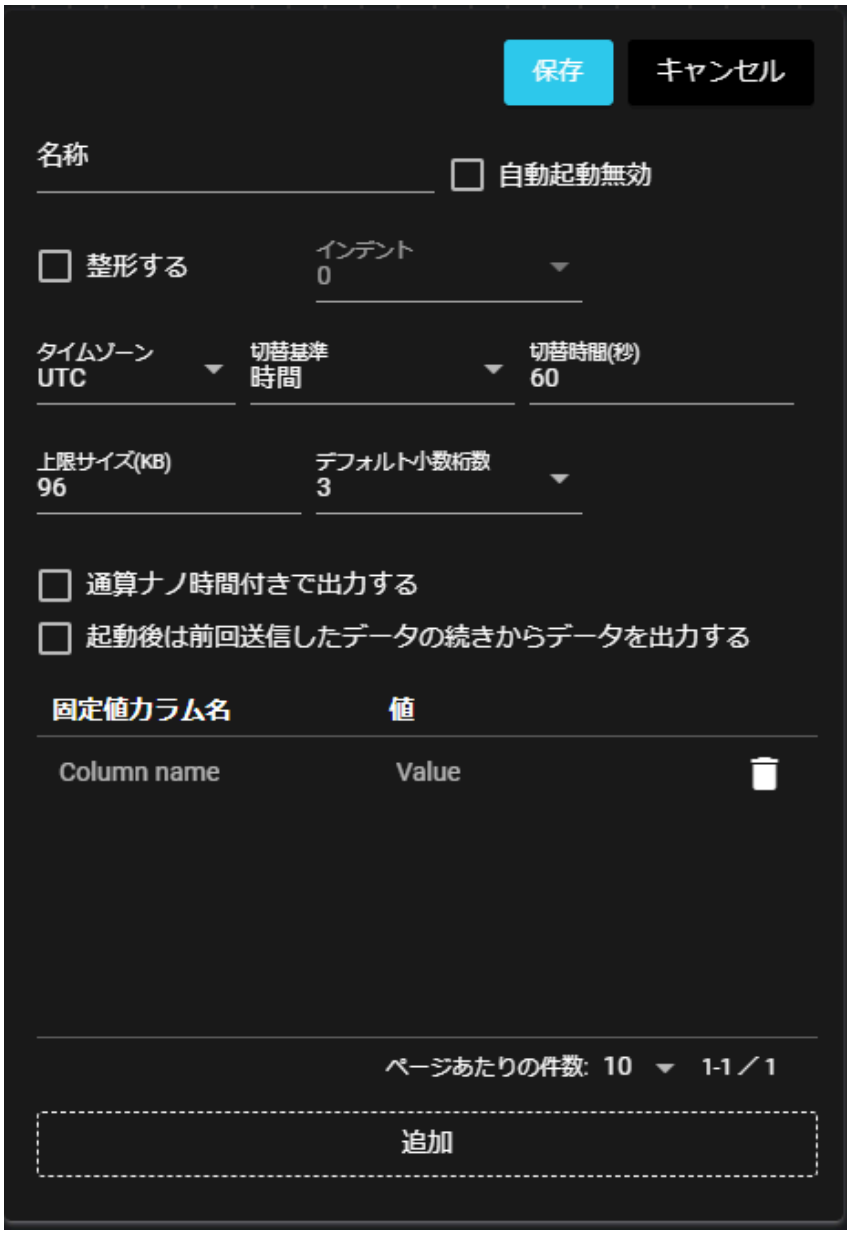
5.5.1.1 基本情報

JSONシリアライザの基本情報は、次の通りです。

項目	説明
ベース名	JSON
説明	コンポーネントのデータを受け取ってJSONフォーマットに加工し、他のコンポーネントに渡します。
消費ポイント	0
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 入力したデータを加工して作成したJSONファイルを出力します。出力するデータの詳細は、「出力データ項目」をご覧ください。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.5.1.2 設定項目

JSONシリアライザの設定項目は、次の通りです。



項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
整形する	JSONを整形する場合ON

項目	説明
インデント	JSONに付与するインデントを指定 ※整形するをONにした場合入力可能
タイムゾーン	JSON内のデータに付与するタイムスタンプのタイムゾーンを選択
切替基準	JSONファイルを切り替える基準を選択
切替時間(秒)	1つのJSONファイルに何秒分のデータを書き込むか入力 ※切替基準が時間の場合入力可能
切替件数	1つのJSONファイルに何件データを書き込むかを入力 ※切替基準が件数の場合入力可能
上限サイズ(KB)	1つのJSONファイルの上限サイズを入力
デフォルト小数桁数	データが浮動小数の場合の送信する桁数を設定 各データに小数桁数が指定していない場合のデフォルト値として利用する
通算ナノ時間付きで出力する	データに通算ナノ秒を追加する
起動後は前回送信するデータの続きからデータを出力する	コンポーネントを停止して、再開させた時に前回送信したデータの続きからデータを出力する

任意でJSONファイルに固定値を出力する事ができます。

項目	説明
固定値カラム名	固定値のカラム名
値	固定値として出力する値

5.5.1.3 出力データ項目

JSONシリアライザは、次のデータを出力します。

項目	カラム名	データ型	説明
JSONデータ出力	output file	FILE	作成したJSONファイルを出力するカラム。メディアタイプは"application/json"固定

以下は、リソースコレクタのI/Oのデータを受け取り、生成されたoutput fileの例です。 "records"に、データが格納されます。 "timestamp"は、データの生成日時です。 "resrc:~~"は、"resrc"という名前のリソースコレクタから受け取った、I/Oのデータです。

```
{
  "records": [
    {
      "timestamp": "2023-10-02T05:54:54.837742200Z",
      "resrc:IOPS": 0,
      "resrc:I_O": "sdb",
      "resrc:kb_r_s": 0,
      "resrc:kb_w_s": 0
    },
    {
      "timestamp": "2023-10-02T05:54:55.837652700Z",
      "resrc:IOPS": 0,
      "resrc:I_O": "sdb",
      "resrc:kb_r_s": 0,
      "resrc:kb_w_s": 0
    },
    ~~~
  ]
}
```

5.5.2 CSVシリアライザ

指定データをCSV形式へ加工するコンポーネントです。
入力ポートからコンポーネントのデータを受け取り、一定時間/件数でCSV形式に加工して出力します。
CSVの並び順は入力ポートの設定で変更できます。



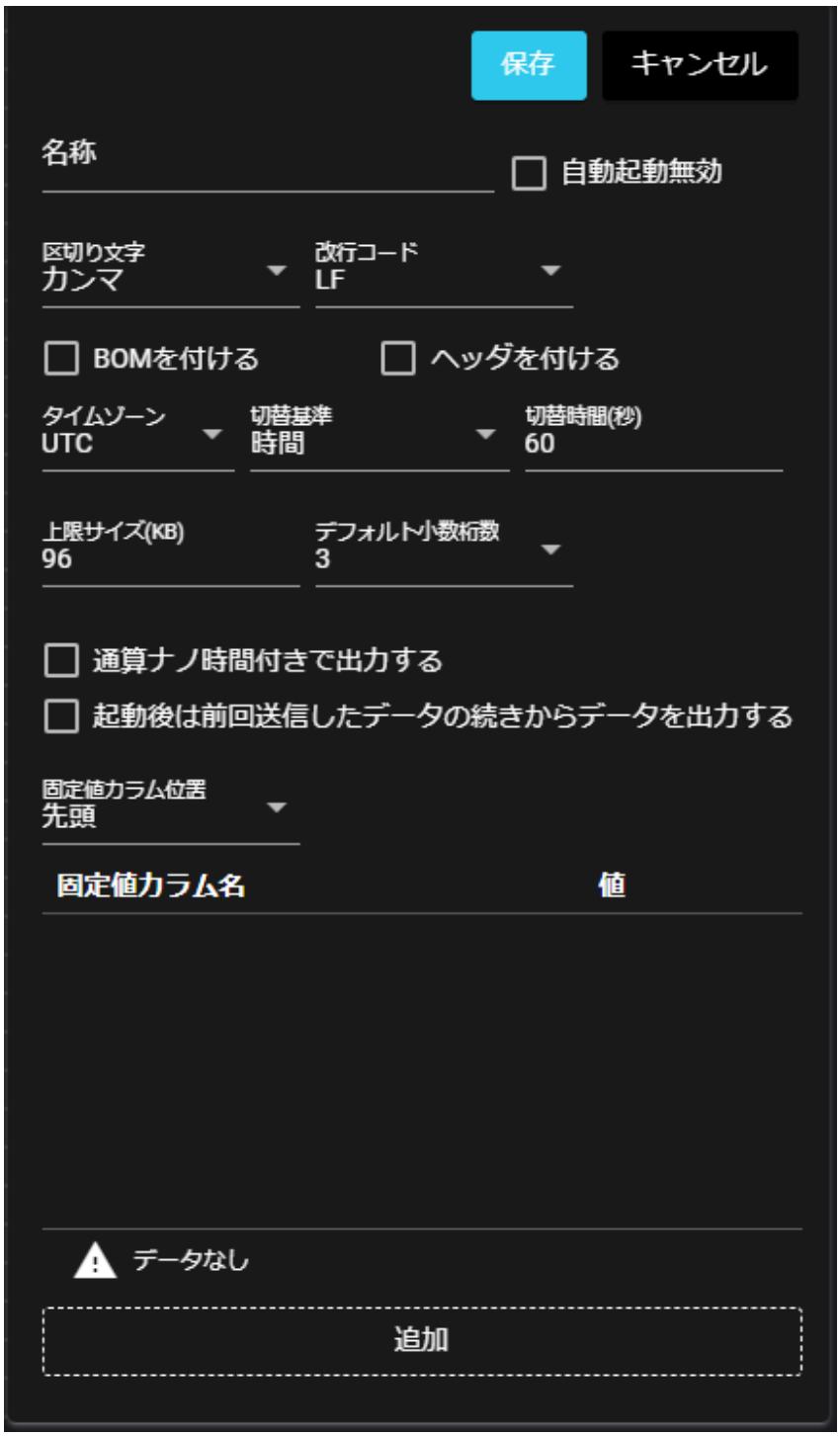
5.5.2.1 基本情報

CSVシリアライザの基本情報は、次の通りです。

項目	説明
ベース名	CSV
説明	コンポーネントのデータを受け取ってCSVフォーマットに加工し、他のコンポーネントに渡します。
消費ポイント	0
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 入力したデータを加工して作成したCSVファイルを出力します。出力するデータの詳細は、「出力データ項目」をご覧ください。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.5.2.2 設定項目

CSVシリアライザの設定項目は、次の通りです。



項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
区切り文字	CSVファイルの区切り文字を選択

項目	説明
改行コード	CSVファイルの改行コードを選択
BOMを付ける	CSVファイルにファイルBOMを付ける場合ON
ヘッダを付ける	CSVファイルにヘッダを付ける場合ON
タイムゾーン	CSVデータに付与するタイムスタンプのタイムゾーンを選択
切替基準	CSVファイルを切り替える基準を選択
切替時間(秒)	1つのCSVファイルに何秒分のデータを書き込むか入力 ※切替基準が時間の場合入力可能
切替件数	1つのCSVファイルに何件データを書き込むかを入力 ※切替基準が件数の場合入力可能
上限サイズ(KB)	1つのJSONファイルの上限サイズを入力
デフォルト小数桁数	データが浮動小数の場合の送信する桁数を設定 各データに小数桁数が指定していない場合のデフォルト値として利用する
通算ナノ時間付きで出力する	データに通算ナノ秒を追加する
起動後は前回送信するデータの続きからデータを出力する	コンポーネントを停止して、再開させた時に前回送信したデータの続きからデータを出力する
固定値カラムの位置	CSVファイル内での固定値カラムの位置を選択

任意でCSVファイルに固定値を出力する事ができます。

項目	説明
固定値カラム名	固定値のカラム名
値	固定値として出力する値

5.5.2.3 出力データ項目

CSVシリアライザは、次のデータを出力します。

項目	カラム名	データ型	説明
CSVデータ出力	output file	FILE	作成したCSVファイルを出力するカラム。メディアタイプは"text/csv"固定

以下は、リソースコレクタのI/Oのデータを受け取り、生成されたoutput fileの例です。 一番左の列は、データのタイムスタンプです。それ以外の列が、リソースコレクタから受け取った、I/Oのデータです。

```
2023-10-02T06:19:52.837670200Z,0.000,"sdb",0.000,0.000
2023-10-02T06:19:53.837656800Z,0.000,"sdb",0.000,0.000
2023-10-02T06:19:54.837805700Z,0.000,"sdb",0.000,0.000
2023-10-02T06:19:55.837734700Z,0.000,"sdb",0.000,0.000
~~~
```

5.6 エミッタ

ここでは、各エミッタコンポーネントについて説明します。

5.6.1 MQTTエミッタ

MQTTプロトコルで任意のデータファイルを送信（パブリッシュ）するコンポーネントです。
入力ポートに繋がれた各種シリアライザからデータを受け取り、そのデータを送信します。



5.6.1.1 基本情報

MQTTエミッタの基本情報は、次の通りです。

項目	説明
ベース名	MQTTエミッタ
説明	MQTTプロトコル（Publish）によるデータ送信を行います。
消費ポイント	2
ポート	入力：1 シリアライザ・コンポーネントから、ファイル形式のデータを受け取ります。受け取ったデータは、MQTTプロトコル（Publish）によって送信されます。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	MQTTブローカーに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.6.1.2 設定項目

MQTTエミッタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	MQTTブローカーのホスト名 or IPアドレスを指定
ポート	MQTTブローカーのポート番号を指定
MQTT Qos	使用するMQTTのQoSを選択（0:受信確認なし、1:受信確認あり、2:受信確認、重複抑止）
常時接続する	サーバー接続を送信の都度、切断しない

項目	説明
匿名アクセス	ブローカーに匿名アクセスする場合はONにする
ユーザー名	匿名アクセスではない場合、ユーザー名を入力
パスワード	匿名アクセスではない場合、パスワードを入力
暗号化	暗号化を行う場合ON
CA証明書	CA証明書をアップロード ※暗号化がONの場合のみアップロード可能
プロキシを利用	MQTTブローカーとの通信をプロキシサーバーを経由して行う場合ON
プロキシサーバー	プロキシサーバーを選択 ※事前にプロキシ設定で使用するサーバーを設定しておく必要があります。
トピック	Publishの際に付与するトピック
クライアントID	MQTTクライアントID ※接続先仕様によっては指定する必要があります。省略時は、ランダムな文字列が使用されます。 ※複数のコンポーネントで、同一MQTTブローカーへ接続する場合、クライアントIDは被らないように設定してください。通信が正しく行われなくなります。
タイムアウト(ミリ秒)	送信処理がタイムアウトするまでの時間をミリ秒単位で入力
メッセージ送信回数	１回のイベントでの送信最大回数
リトライモード	送信に失敗してリトライする際の挙動を選択
リトライ間隔(ミリ秒)	送信リトライの間隔を入力
リトライ回数	送信リトライの回数を入力 ※リトライモードが「リトライ回数分再送」の場合のみ入力可能
起動後は前回送信するデータの続きからデータを出力する	コンポーネントを停止して、再開させた時に前回送信したデータの続きからデータを出力する

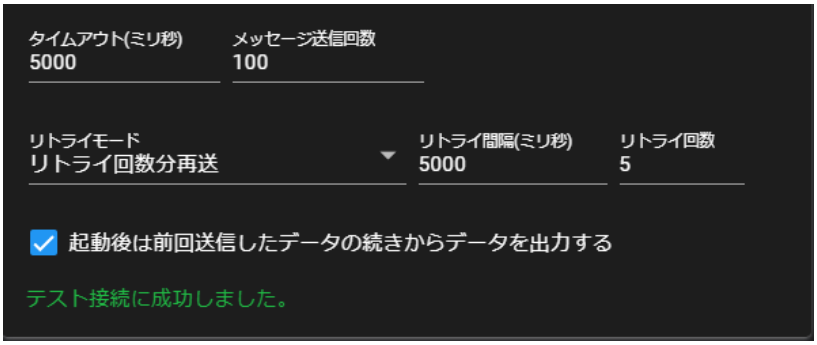
5.6.1.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したMQTTエミッタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



MQTTエミッタのテスト接続に関する注意点

テスト接続は、次の場合のみ操作可能です。
・コンポーネントが停止している場合
・コンポーネントのステータスがエラーの場合
※ステータスについての詳細は「ステータス一覧」をご覧ください。

5.6.2 Fileエミッタ

収集したデータをファイルとして、ストレージへ保存するコンポーネントです。
入力ポートに接続された各種シリアライザからデータを受け取り、そのデータをパラメータで指定されたファイルパスに保存します。
保存されたファイルは、設定ダイアログから時間範囲を指定してダウンロードすることができます。



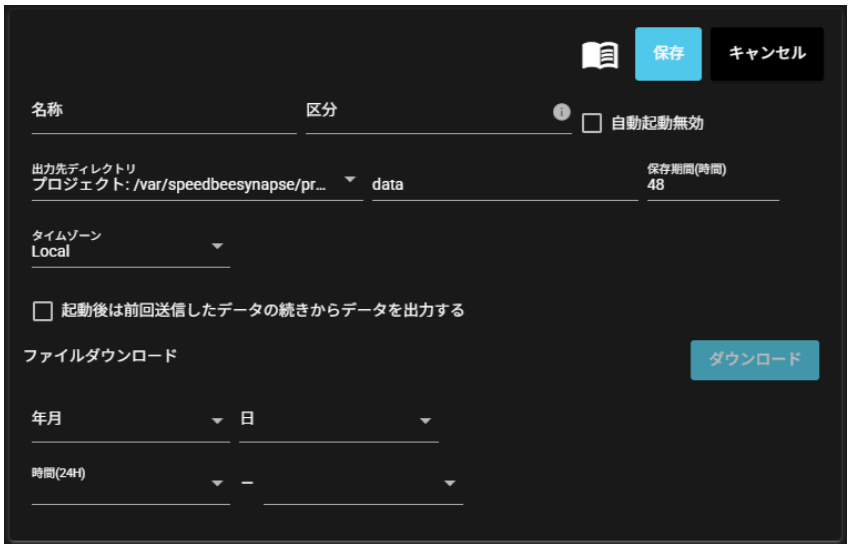
5.6.2.1 基本情報

Fileエミッタの基本情報は、次の通りです。

項目	説明
ベース名	Fileエミッタ
説明	収集データをストレージへ保存します。
消費ポイント	0
ポート	入力：1 シリアライザ・コンポーネントから、ファイル形式のデータを受け取ります。受け取ったデータは、ストレージへ保存されます。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	出力したファイルをダウンロードすることができます。詳細は「ファイルダウンロード」をご覧ください。

5.6.2.2 設定項目

Fileエミッタの設定項目は、次の通りです。

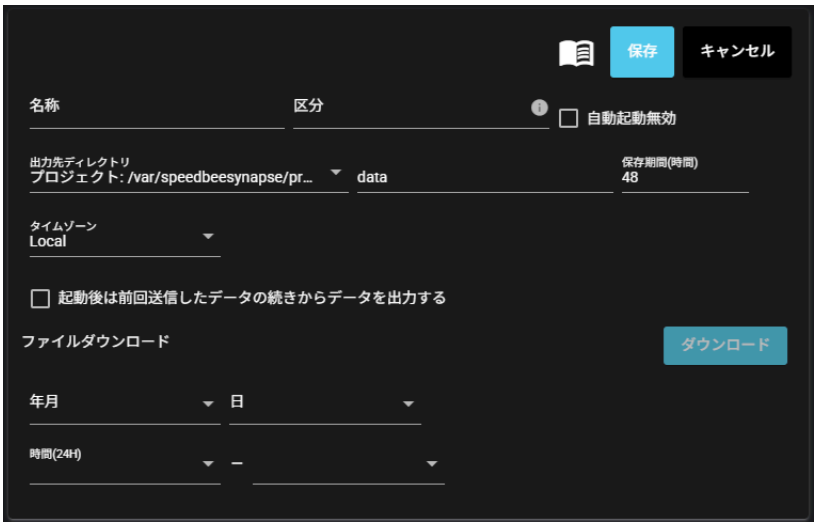


項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
区分	Fileエミッタ毎に一意に付与する任意の値を入力 ※区分は出力先ディレクトリのサブディレクトリ名として使用されます。半角のみ入力可能です。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
出力先ディレクトリ	ファイルを出力するディレクトリ。ベースディレクトリと、そこからの相対パスを指定します。ベースディレクトリは以下から選択可能です。 • プロジェクトパス（プロジェクト配下へ保存されます。プロジェクトを複製などした場合、新しいプロジェクトの配下が保存先となります。） ※一部の環境では、この設定を変更/編集できません。 • 任意のパス（通常はベースディレクトリに対する相対パスを指定しますが、任意の絶対パスを直接指定します。プロジェクトを複製などしても指定したパスが保存先となります。）
保存期間(時間)	ファイルを保存しておく期間を入力
タイムゾーン	ファイルのタイムゾーンを選択
起動後は前回送信するデータの続きからデータを出力する	起動後は前回送信したデータの続きからデータを出力する

5.6.2.3 ファイルダウンロード

次の手順で、Fileエミッタが出力したファイルをダウンロードすることができます。

1. 「ファイルダウンロード」欄で、時間範囲を入力します。



2. 「ダウンロード」を押下すると、上で入力した時間範囲内に出力されたファイルが、zip形式でダウンロードできます。

5.6.3 Cloudエミッタ(AWS用)

AWS IoT Core向けにデータ送信を行うコンポーネントです。
入力ポートで各種シリアライザからデータを受け取り、そのデータをAWS IoT Coreに送信します。



5.6.3.1 基本情報

Cloudエミッタ(AWS用)の基本情報は、次の通りです。

項目	説明
ベース名	Cloudエミッタ(AWS用)
説明	AWS IoT Core向けにデータ送信を行います。
消費ポイント	2

項目	説明
ポート	入力：1 シリアライザ・コンポーネントから、ファイル形式のデータを受け取ります。受け取ったデータは、AWS IoT Core向けに送信されます。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	AWS IoT Coreに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.6.3.2 設定項目

Cloudエミッタ(AWS用)の設定項目は、次の通りです。



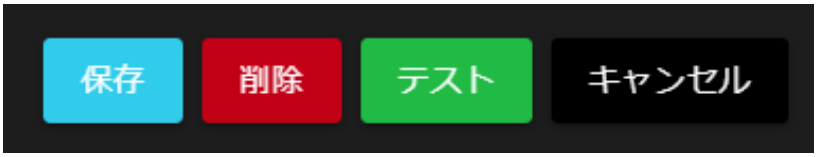
項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
クライアントID	AWS IoT Coreのクライアント ID
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	AWS IoT Coreのホスト名を入力
ポート	AWS IoT Coreのポート番号を選択（445 or 8883）
MQTT Qos	使用するMQTTのQoSを選択（0:受信確認なし、1:受信確認あり）
常時接続する	サーバー接続を送信の都度、切断しない
CA証明書	AmazonルートCA1を選択
SSL証明書	デバイス証明書を選択
秘密鍵	プライベートキーファイルを選択
プロキシを利用	AWS IoT Coreとの通信をプロキシサーバーを経由して行う場合ON
プロキシサーバー	プロキシサーバーを選択 ※事前にプロキシ設定で使用するサーバーを設定しておく必要があります。
トピック	Publishの際に付与するトピック
タイムアウト(ミリ秒)	送信処理がタイムアウトするまでの時間(ミリ秒単位) ※この項目は編集できません。
メッセージ送信回数	１回のイベントでの送信最大回数 ※この項目は編集できません。
リトライモード	送信に失敗してリトライする際の挙動 ※この項目は編集できません。

項目	説明
リトライ間隔(ミリ秒)	送信リトライの間隔 ※この項目は編集できません。
リトライ回数	送信リトライの回数 ※この項目は編集できません。
起動後は前回送信するデータの続きからデータを出力する	起動後は前回送信したデータの続きからデータを出力する

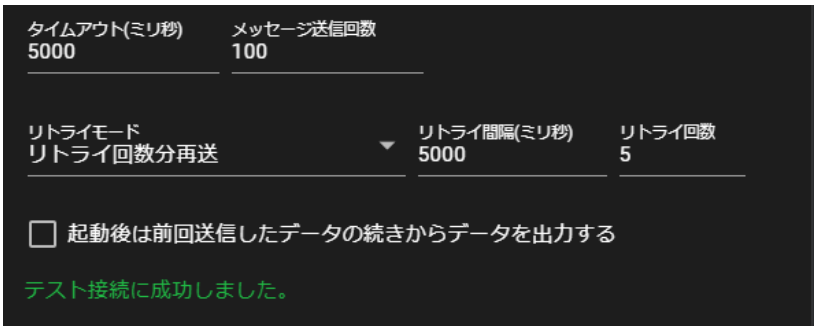
5.6.3.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したCloudエミッタ(AWS用)の設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.6.4 Cloudエミッタ(Azure用)

Azure IoT Hub向けにデータ送信を行うコンポーネントです。
入力ポートで各種シリアライザからデータを受け取り、そのデータをAzure IoT Hubに送信します。



5.6.4.1 基本情報

Cloudエミッタ(Azure用)の基本情報は、次の通りです。

項目	説明
ベース名	Cloudエミッタ(Azure用)
説明	Azure IoT Hub向けにデータ送信を行います。
消費ポイント	2
ポート	入力：1 シリアライザ・コンポーネントから、ファイル形式のデータを受け取ります。受け取ったデータは、Azure IoT Hub向けに送信されます。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	Azure IoT Hubに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.6.4.2 設定項目

Cloudエミッタ(Azure用)の設定項目は、次の通りです。



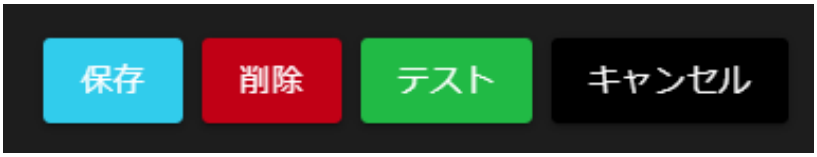
項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
クライアントID	AzureのクライアントID （MQTTで使用するクライアントID）
自動起動無効	コンポーネントの自動起動を無効にする場合ON
接続文字列	Azure IoT Hubの接続文字列を入力
常時接続する	サーバー接続を送信の都度、切断しない
プロキシを利用	Azure IoT Hubとの通信をプロキシサーバーを経由して行う場合ON
プロキシサーバー	プロキシサーバーを選択 ※事前にプロキシ設定で使用するサーバーを設定しておく必要があります。
タイムアウト(ミリ秒)	送信処理がタイムアウトするまでの時間(ミリ秒単位) ※この項目は編集できません。
メッセージ送信回数	１回のイベントでの送信最大回数 ※この項目は編集できません。

項目	説明
リトライモード	送信に失敗してリトライする際の挙動 ※この項目は編集できません。
リトライ間隔(ミリ秒)	送信リトライの間隔 ※この項目は編集できません。
リトライ回数	送信リトライの回数 ※この項目は編集できません。
起動後は前回送信するデータの続きからデータを出力する	起動後は前回送信したデータの続きからデータを出力する

5.6.4.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したCloudエミッタ(Azure用)の設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.6.5 FTP/FTPS/SFTPエミッタ

FTP、FTPSまたはSFTPによるデータ転送を行うコンポーネントです。
入力ポートで各種シリアライザからデータを受け取り、そのデータを受け取ったシリアライザの形式のファイルとしてFTP、FTPSまたはSFTPで転送します。
このコンポーネントは、Proxy Serverを経由した通信には対応していません。



5.6.5.1 基本情報

FTP/FTPS/SFTPエミッタの基本情報は、次の通りです。

項目	説明
ベース名	FTP/FTPS/SFTPエミッタ
説明	FTPまたはSFTPによるデータ転送を行います。
消費ポイント	2
ポート	入力：1 シリアライザ・コンポーネントから、ファイル形式のデータを受け取ります。受け取ったデータは、FTPまたはSFTPによって転送されます。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	FTPまたはSFTPサーバーに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.6.5.2 設定項目

FTP/FTPS/SFTPエミッタの設定項目は、次の通りです。



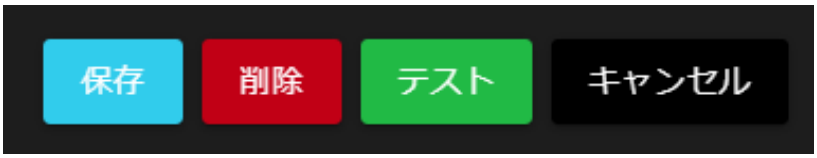
項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
ファイルプレフィックス	ファイル名に付与するプレフィックス
自動起動無効	コンポーネントの自動起動を無効にする場合ON
SFTPを使う	SFTPを使う場合ON
ホスト	送信先サーバーのホスト名かIPアドレスを指定
ポート	送信先サーバーのポート番号を指定
匿名アクセス	ユーザー認証を行わない場合ON ※SFTPを使うがOFFの場合のみ選択可能です。
ユーザー名	認証ユーザー名を入力 ※次の場合は入力可能です。 ・SFTPを使うがOFFかつ匿名アクセスがOFF ・SFTPを使うがON
パスワード	認証パスワードを入力 ※次の場合は入力可能です。 ・SFTPを使うがOFFかつ匿名アクセスがOFF ・SFTPを使うがONかつ公開鍵認証がOFF

項目	説明
パスフレーズ	パスフレーズを入力 ※SFTPを使うがONかつ公開鍵認証がONの場合のみ入力可能です。
公開鍵認証	公開鍵認証を行う場合ON ※SFTPを使うがONの場合のみ選択可能です。
秘密鍵	秘密鍵をアップロード ※SFTPを使うがONかつ公開鍵認証がONの場合のみアップロード可能です。
暗号化	送受信するデータを暗号化する場合ON ※SFTPを使うがOFFの場合のみ選択可能です。
出力先ディレクトリ	送信先サーバー内の出力先ディレクトリを指定
起動後は前回送信するデータの続きからデータを出力する	起動後は前回送信したデータの続きからデータを出力する

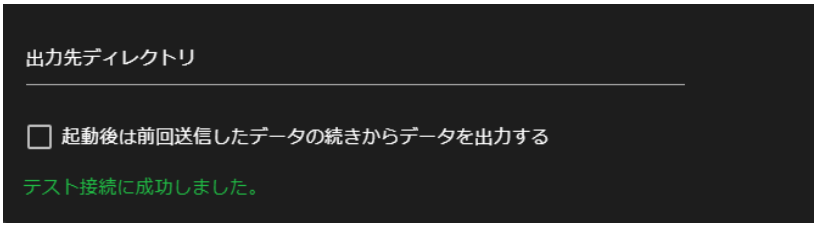
5.6.5.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したFTP/FTPS/SFTPエミッタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。

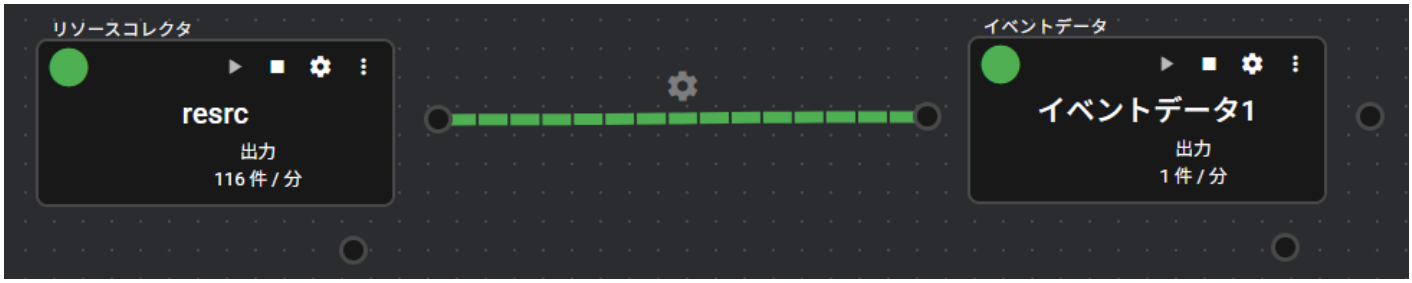


5.7 ロジック（イベント）

ここでは、イベントに関するロジックコンポーネントについて説明します。

5.7.1 イベントデータ

収集したデータを条件判定し、条件合致している区間の情報を出力するコンポーネントです。
入力ポートの「イベントを使用する」と組み合わせて使用します。
入力ポートからデータを受け取り、そのデータに対して都度条件判定を行い、条件に合致している区間の情報をデータとして出力します。



5.7.1.1 基本情報

イベントデータの基本情報は、次の通りです。

項目	説明
ベース名	イベントデータ
説明	収集したデータのうち、条件合致したものだけを、新たに別データとして出力します。 ※別のデータとして登録するため、メモリ使用量等は増加します。高サンプリングレートデータをそのまま扱いたい場合には不向きです。
消費ポイント	2

項目	説明
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 受け取ったデータのうち、条件合致したものを出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	イベントデータを用いた設定例は、「設定例」をご覧ください。

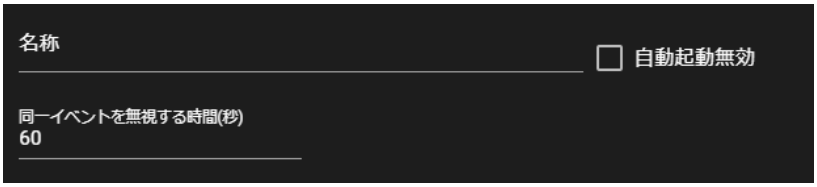
5.7.1.2 設定方法

次の手順で、イベントデータの設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. イベントデータを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



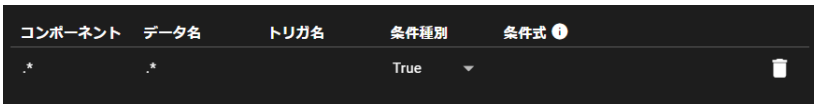
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
同一イベントを無視する時間(秒)	同一イベントを無視する時間を入力

6. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に設定を行う
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して設定を行う

7. 次の項目を入力し、イベント条件を設定します。



項目	説明
コンポーネント	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
トリガ名	トリガの名前を入力
条件種別	True：条件式が真ならトリガ発生 False：条件式が偽ならトリガ発生

項目	説明
	Fall：条件式が真→偽の際にトリガ発生 Raise：条件式が偽→真の際にトリガ発生 Anytime：値の登録時に発生
条件式	トリガの条件式を入力 ※条件種別がAnytimeの場合入力不可

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 「保存」を押下します。



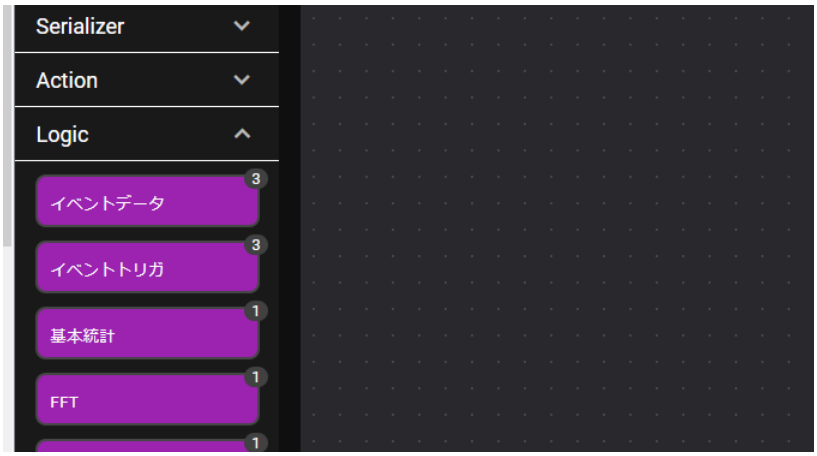
9. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.7.1.3 設定例

イベントデータの設定例は次の通りです。ここではまず、リソースコレクタを用いてCPU使用率を収集します。そこにイベントデータを用いて、CPU使用率が20%を超える場合のみ新たに別データとして登録されるようにします。

1. 事前に、コレクタを追加します。ここでは"resrc"という名前のリソースコレクタを作成し、cpu使用率を出力するように設定します。

2. イベントデータのアイコンを、パネル上にドラッグアンドドロップします。



3. 入力ポートに繋げるコンポーネントを選択します。今回は、"resrc"という名前のリソースコレクタを選択します。



4. 次の項目を入力し、保存を押下します。

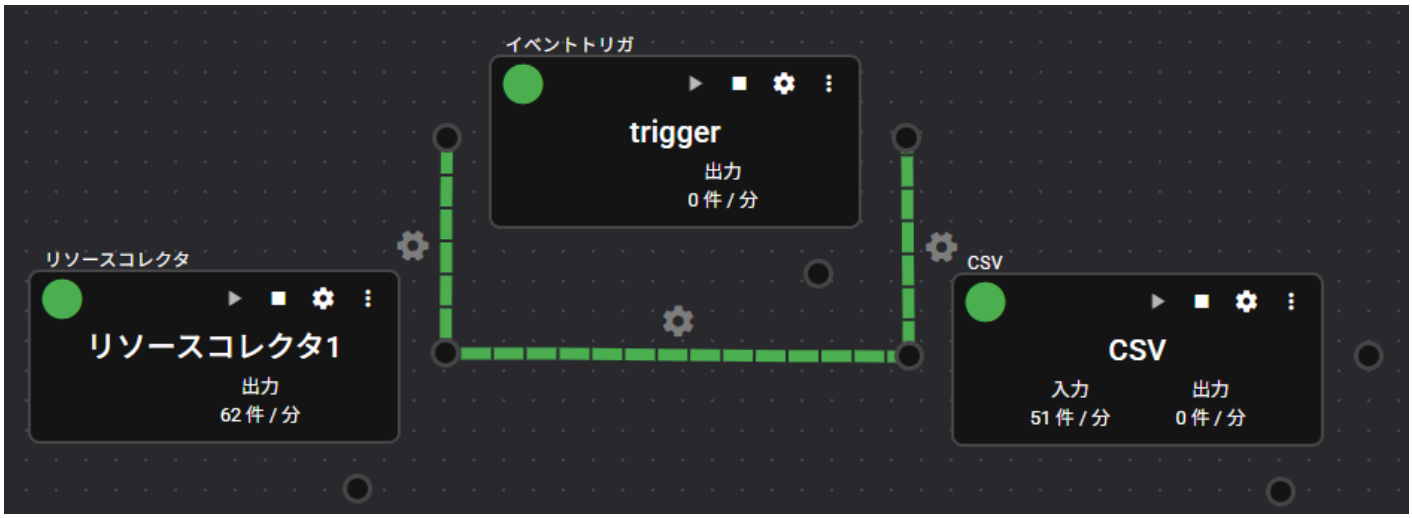


項目	説明
名称	イベントデータ1
自動起動無効	OFF
同一イベントを無視する時間(秒)	60
設定方法	データ定義を受信して設定する
コンポーネント	resource
データ名	cpu_usage
トリガ名	trigger1
条件種別	True
条件式	$VALUE > 20

5. リソースコレクタからcpu使用率を受け取り、20%を超えたときに、新たに別データとして登録されます。登録されたデータは、出力ポートから出力されます。

5.7.2 イベントトリガ

収集したデータを条件判定し、条件合致した場合のみ、新たに別データとして登録するコンポーネントです。
入力ポートからデータを受け取り、そのデータに対して都度条件判定を行い、条件に合致している場合は新たに別データとして出力します。



5.7.2.1 基本情報

イベントトリガの基本情報は、次の通りです。

項目	説明
ベース名	イベントトリガ
説明	収集したデータを条件判定し、条件合致している区間の情報を、データとして出力します。入力ポートの「イベントを使用する」と組み合わせて使うコンポーネントです。イベント中のデータをフローリンク先へ伝達するには、別途フローリンクを接続する必要があります。元データを別で登録するイベントデータとは違い、イベント区間の情報を登録するためメモリ使用量などは抑えれます。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 受け取ったデータを条件判定し、条件に合致したかどうかを真偽値として出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	イベントトリガを用いた設定例は、「設定例」をご覧ください。

5.7.2.2 設定方法

次の手順で、イベントトリガの設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. イベントトリガを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
同一イベントを無視する時間(秒)	同一イベントを無視する時間を入力
基本記録時間(分)	基本記録時間を分単位で入力

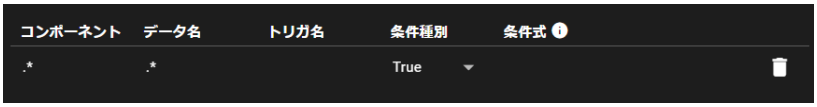
6. 開始イベントの「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に設定を行う

項目	説明
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して設定を行う

7. 次の項目を入力し、開始イベントを設定します。



項目	説明
コンポーネント	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
トリガ名	トリガの名前を入力
条件種別	True：条件式が真ならトリガ発生 False：条件式が偽ならトリガ発生 Fall：条件式が真→偽の際にトリガ発生 Raise：条件式が偽→真の際にトリガ発生 Anytime：値の登録時に発生
条件式	トリガの条件式を入力 ※条件種別がAnytimeの場合入力不可

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 終了イベントを設定する場合、「終了イベント」をONにします。

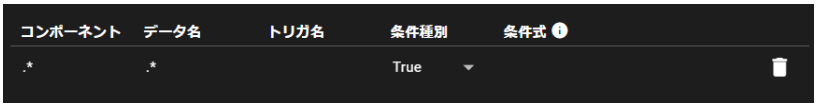


9. 終了イベントを設定する場合、終了イベントの「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に設定を行う
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して設定を行う

10. 終了イベントを設定する場合、次の項目を入力し、終了イベントを設定します。



項目	説明
コンポーネント	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
トリガ名	トリガの名前を入力
条件種別	True：条件式が真ならトリガ発生 False：条件式が偽ならトリガ発生 Fall：条件式が真→偽の際にトリガ発生 Raise：条件式が偽→真の際にトリガ発生 Anytime：値の登録時に発生

項目	説明
条件式	トリガの条件式を入力 ※条件種別がAnytimeの場合入力不可

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

11. 「保存」を押下し、設定を保存します。



12. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
13. イベントトリガで設定したイベントを、他のコンポーネントがデータを受け取る際の条件判定に使用することができます。詳細は「入力ポート」の「イベントの設定」をご覧ください。

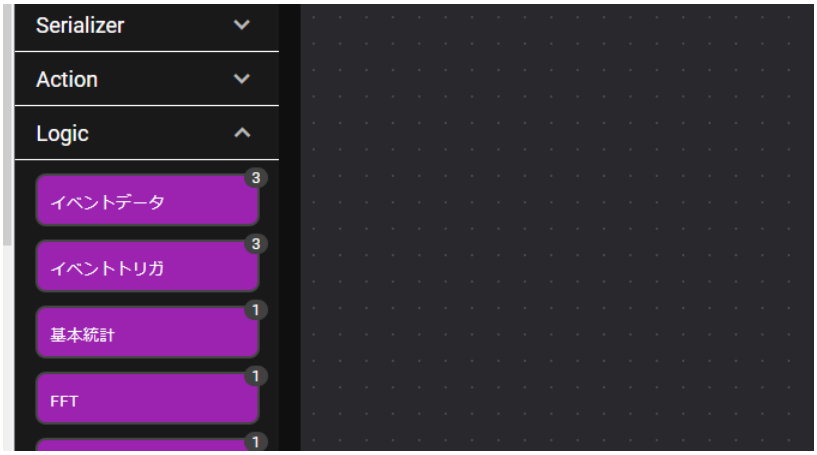
イベントトリガのイベント区間が使えないコンポーネント

下記コンポーネントはイベントトリガに接続可能(設定可能)ですが、イベント区間を利用することはできません。
・ロジックコンポーネント（イベントデータ・基本統計・FFT・移動平均）

5.7.2.3 設定例

イベントトリガの設定例は次の通りです。ここではまず、リソースコレクタが収集したCPU使用率をCSVシリアライザで加工するように設定します。そこにイベントトリガを用いて、CPU使用率が20%を超える場合のみ加工が行われるようにします。

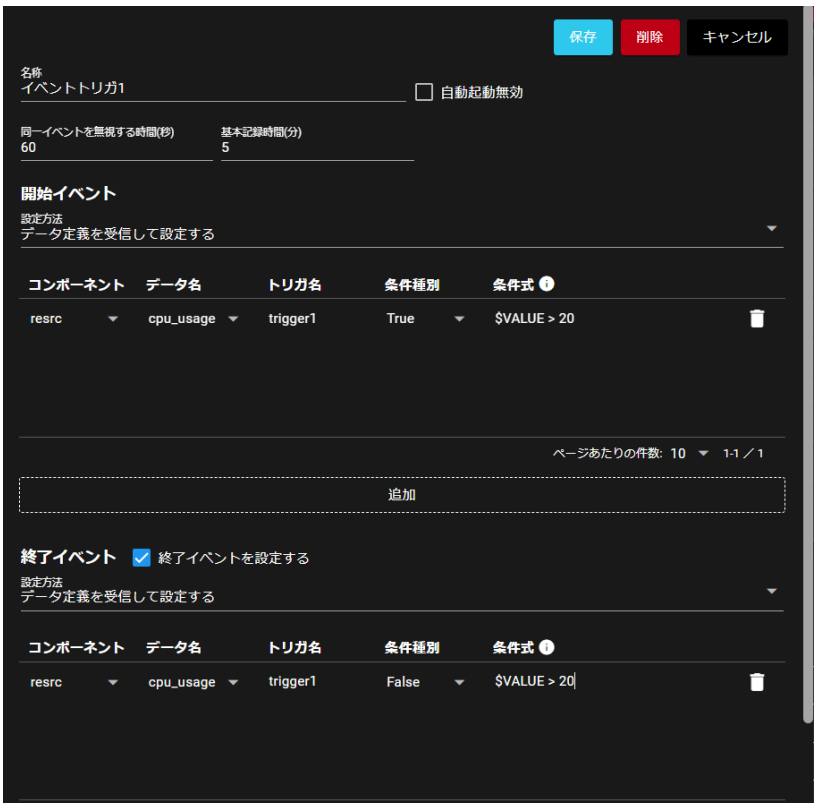
1. 事前に、コレクタを追加します。ここでは"resrc"という名前のリソースコレクタを作成し、cpu使用率を出力するように設定します。
2. 左メニューの中からイベントトリガのアイコンをドラッグし、パネル上にドロップします。



3. 入力ポートに繋げるコンポーネントを選択します。今回は、"resrc"という名前のリソースコレクタを選択します。



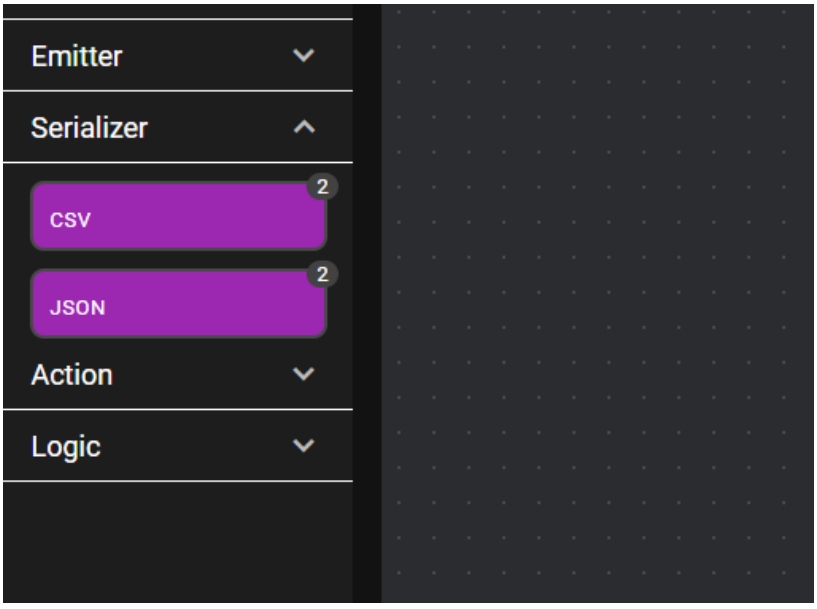
4. 次の項目を入力し、保存を押下します。



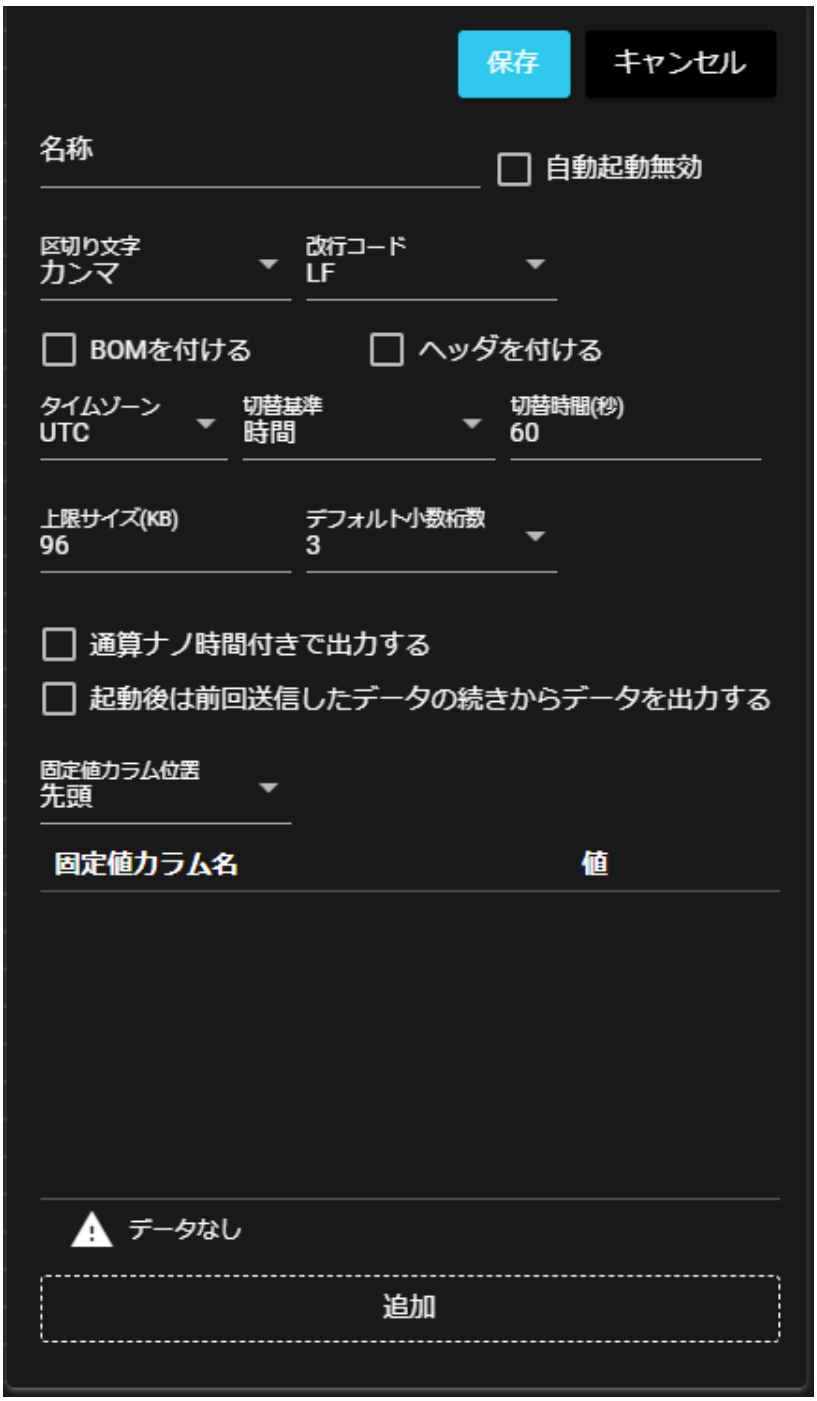
項目	説明
名称	イベントトリガ1
自動起動無効	OFF
同一イベントを無視する時間(秒)	60
基本記録時間(分)	5
設定方法	データ定義を受信して設定する
コンポーネント	resource
データ名	cpu usage
トリガ名	trigger1
条件種別	True
条件式	$VALUE > 20

項目	説明
終了イベントを設定する	ON
設定方法	データ定義を受信して設定する
コンポーネント	resource
データ名	cpu_usage
トリガ名	trigger1
条件種別	False
条件式	$VALUE > 20

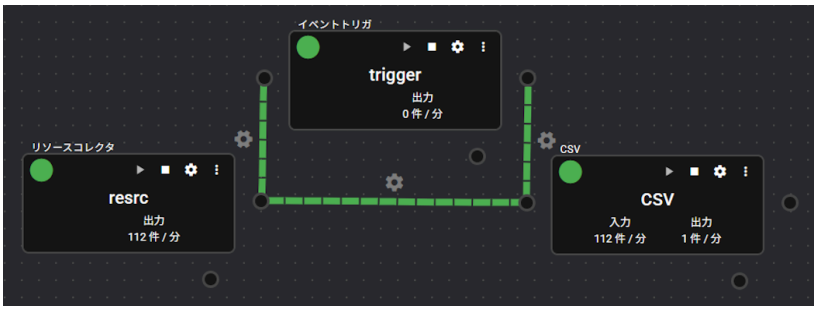
5. 左メニューの中からCSVシリアライザのアイコンをドラッグし、パネル上にドロップします。



6. 任意の設定項目を入力し、保存を押下します。



7. リソースコレクタ・イベントトリガの出力ポートと、CSVシリアライザの入力ポートを、フローリンクでつなぎます。



8. CSVシリアライザの入力ポートを右クリックします。



9. 「イベントを使用する」にチェックを付け、「イベント」で"イベントトリガ1"を選択し、保存を押下します。



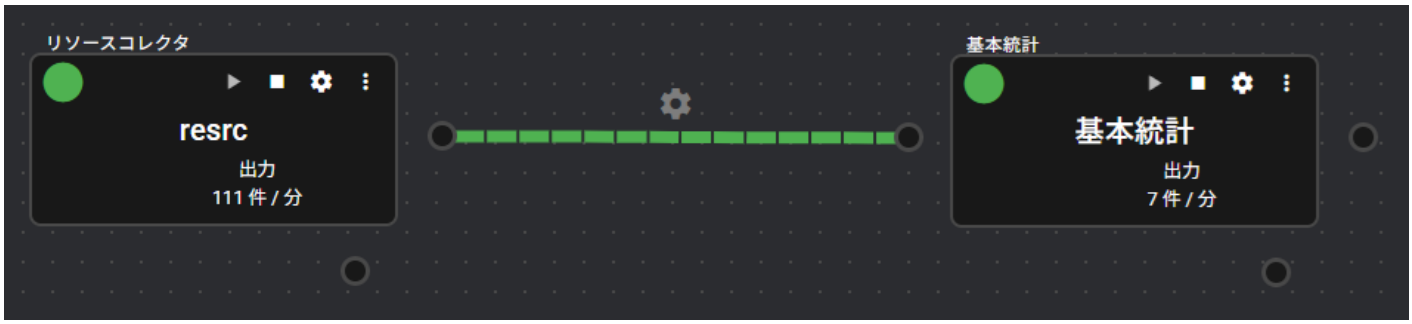
10. この状態で、コンポーネントを稼働させることで、リソースコレクタが収集したCPU使用率が、20%を超えた場合のみ、CSVシリアライザによってCSVファイルに加工されます。

5.8 ロジック（リアルタイム分析/データ操作）

ここでは、リアルタイム分析に関するロジックコンポーネントについて説明します。

5.8.1 基本統計

収集データの基本統計した結果を別データとして登録するコンポーネントです。
入力ポートから受け取ったデータに対し一定の範囲（件数/秒）で基本統計を行い、その結果を別データとして出力します。



5.8.1.1 基本情報

基本統計の基本情報は、次の通りです。

項目	説明
ベース名	基本統計
説明	収集データに基本統計を行った結果を、データとして出力します。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 「設定方法」によって設定された内容をもとに、入力データに基本統計を行った結果を出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.8.1.2 設定方法

次の手順で、基本統計の設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. 基本統計を追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



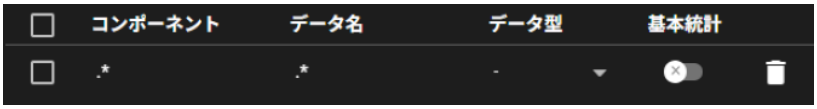
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON

6. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に設定を行う
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して設定を行う

7. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



項目	説明
コンポーネント	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現
データ型	入力ポートで受け取っているデータのうち、設定対象とするもののデータ型

正規表現の入力例

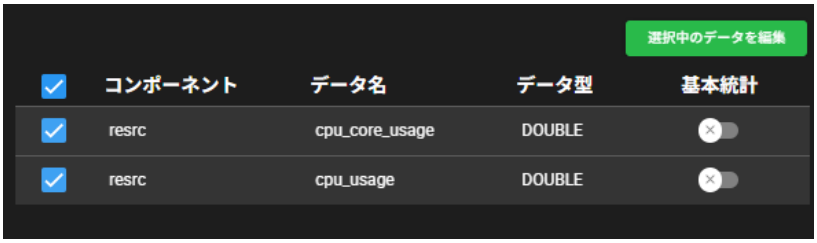
正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 次のいずれかの方法を行い、基本統計の設定ダイアログを開きます。
   - 基本統計の設定を行うデータの「基本統計」のアイコンを押下します。



- 基本統計の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下します。チェックを付けたデータに、同じ設定内容が適用されます。



9. 次の項目を入力し、基本統計の設定を行います。



項目	説明
分析幅	分析の幅を設定
分析単位	分析幅の単位を設定（件数 or 秒）
件数	分析対象のデータ件数
合計	分析対象のデータの合計値
2乗和	分析対象の各データの二乗を合計した値
偏差平方和	分析対象の各データが平均からどれだけ離れているかの偏差を二乗して合計した値
最小値	分析対象のデータ内で最も小さい値
最大値	分析対象のデータ内で最も大きい値

項目	説明
範囲	分析対象のデータの最大値と最小値の差
算術平均	分析対象のデータの平均値
分散	分析対象の各データが平均からどれだけ離れているかの平均の二乗
標準偏差	分散の平方根
不偏分散	分散を件数-1で割った値
標本不偏標準偏差	標本の標準偏差
標準誤差	標本平均の標準偏差
変動係数	標準偏差を平均で割った値
中央値	分析対象のデータを小さい順に並べ、中央にある値
四分位数	分析対象のデータを小さい順に並べ、四等分した位置にある値
最頻値	分析対象のデータ内で最も頻繁に出現する値
平均偏差	分析対象の各データが平均からどれだけ離れているかの平均
調和平均	分析対象の各データの逆数の平均の逆数
幾何学平均	分析対象の各データの積のN乗根 ※Nは件数
尖度	分布の尖り具合
歪度	分布の歪み
信頼区間	分析対象のデータの平均値が含まれると信頼できる数値の範囲

10. 「確定」を押下します。



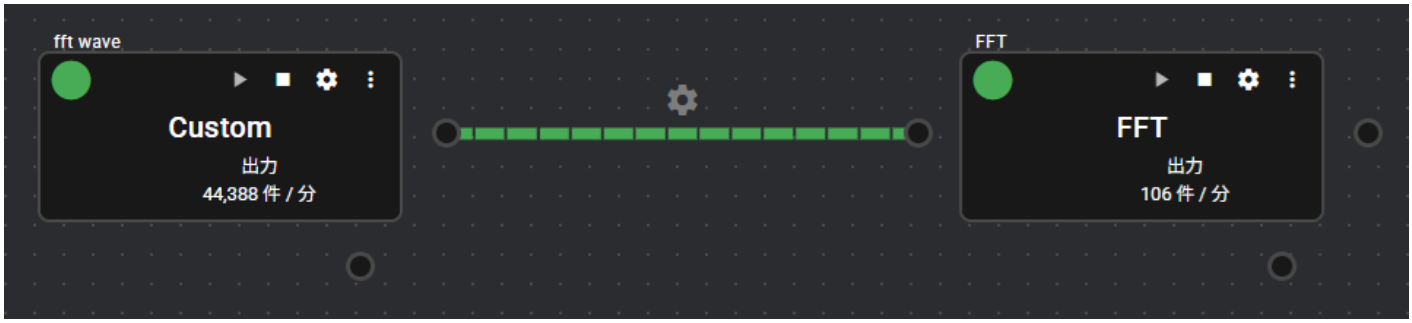
11. 「保存」を押下します。



12. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.8.2 FFT

収集データのFFTした結果をデータとして登録するコンポーネントです。
入力ポートからデータを受け取り、そのデータにFFT解析を行った結果を別データとして出力します。
このコンポーネントは、サンプリングレートが定義されているカラムタイプミドルのデータに対してのみ利用できます。



カラムタイプミドルのデータについて

標準機能のコンポーネントでは、サンプリングレートが定義されているカラムタイプミドルを使用するコンポーネントは存在しません。カラムタイプミドル（サンプリングレートが定周期）のデータを扱うカスタムコンポーネントを開発して使用します。

5.8.2.1 基本情報

FFTの基本情報は、次の通りです。

項目	説明
ベース名	FFT
説明	収集データにFFT解析を行った結果を、データとして出力します。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 「設定方法」によって設定された内容をもとに、入力データにFFT解析を行った結果を出力します。 システム：1

項目	説明
	コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.8.2.2 設定方法

次の手順で、FFTの設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. FFTを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



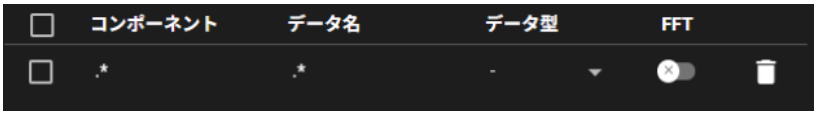
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON

6. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に設定を行う
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して設定を行う

7. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



項目	説明
コンポーネント	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現
データ型	入力ポートで受け取っているデータのうち、設定対象とするもののデータ型

正規表現の入力例

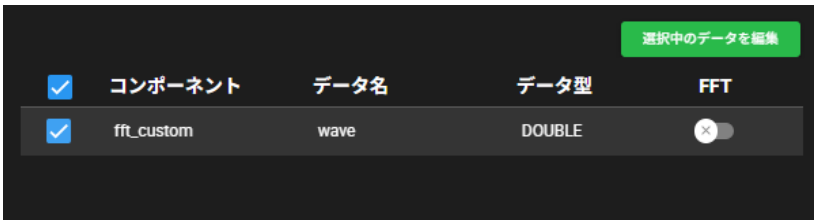
正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 次のいずれかの方法を行い、FFTの設定ダイアログを開きます。
   - FFTの設定を行うデータの「FFT」のアイコンを押下します。



- FFTの設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下します。チェックを付けたデータに、同じ設定内容が適用されます。



9. 「FFT解析を行う」をONにします。



10. 次の項目を入力し、FFTの設定を行います。



項目	説明
件数	FFTを実行するデータの件数
ウィンドウ関数	FFTを実行する前にデータに対して適用される関数

11. 「確定」を押下します。



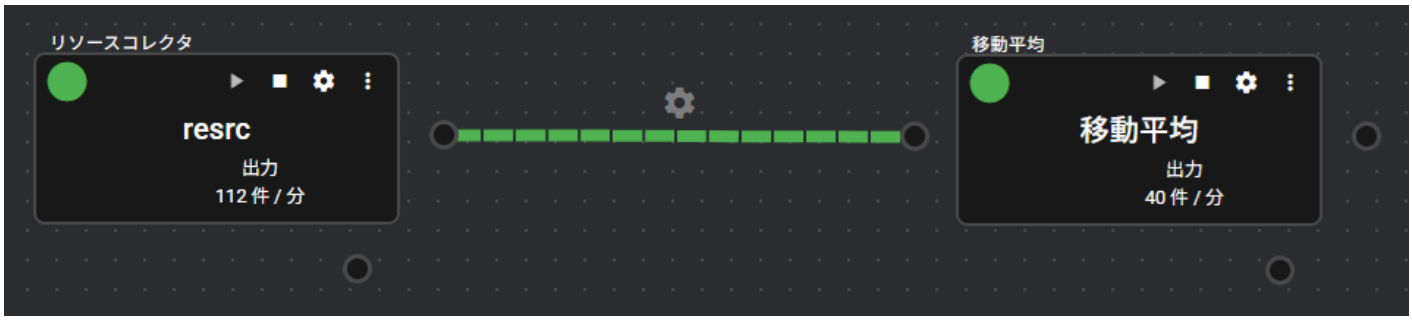
12. 「保存」を押下します。



13. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.8.3 移動平均

収集データの移動平均値をデータとして登録するコンポーネントです。
入力ポートからデータを受け取り、そのデータから期間内の移動平均値を計算して別データとして出力します。



5.8.3.1 基本情報

移動平均の基本情報は、次の通りです。

項目	説明
ベース名	移動平均
説明	収集データの移動平均値を、データとして出力します。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 「設定方法」によって設定された内容をもとに、入力データの移動平均値を出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.8.3.2 設定方法

次の手順で、移動平均の設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

3. 移動平均を追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。

4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



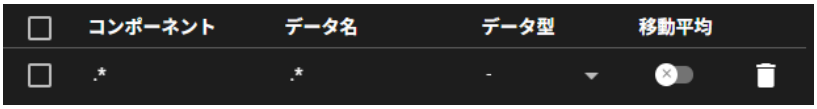
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON

6. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に設定を行う
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して設定を行う

7. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



項目	説明
コンポーネント	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現
データ型	入力ポートで受け取っているデータのうち、設定対象とするもののデータ型

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 次のいずれかの方法を行い、移動平均の設定ダイアログを開きます。
   - 移動平均の設定を行うデータの「移動平均」のアイコンを押下します。



- 移動平均の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下します。チェックを付けたデータに、同じ設定内容が適用されます。



9. 次の項目を入入力し、移動平均の設定を行います。



項目	説明
移動平均値期間1~3	計算対象とする期間
指数加重移動平均値の標準化定数α	指数加重移動平均値のバランスを取るための係数 ※指数加重移動平均値がONの場合のみ入力可能

項目	説明
単純移動平均値	期間内のデータの単純な平均値
線形加重移動平均値	期間内の各データに対して線形な重みをかけて計算した平均値
指数加重移動平均値	全期間のデータに対して指数的な重みをかけて計算した平均値

10. 「確定」を押下します。



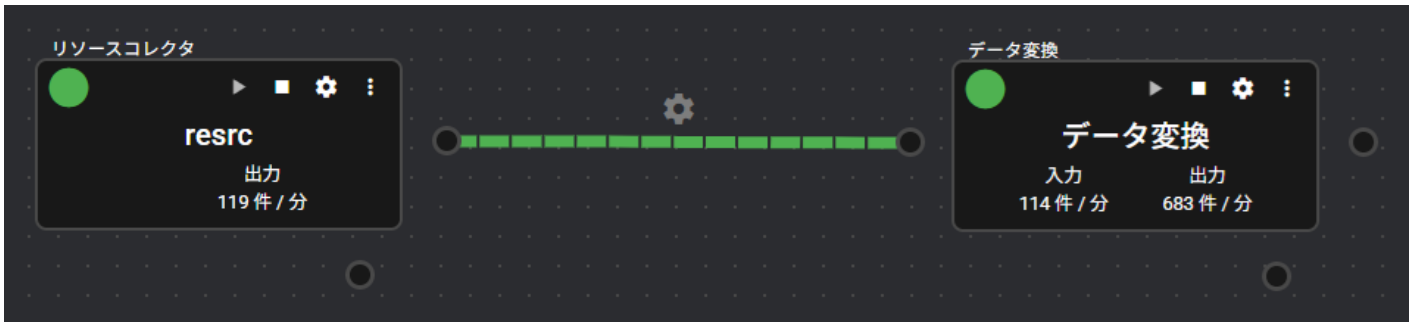
11. 「保存」を押下します。



12. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.8.4 データ変換

入力ポートのカラムから様々な型のデータを取得し、 それに対して各種変換を通して出力ポートのカラムに出力するコンポーネントです。



現在提供しているデータ変換機能は下記の通りです。

変換機能	説明
配列分解	配列カラムのデータを個々の要素に分解して個別のカラムに登録します。 分解後の各カラムの名称はデフォルトで自動設定されます。 [対象カラム] INT8/INT16/INT32/INT64配列 UINT8/UINT16/UINT32/UINT64配列 BOOLEAN/FLOAT/DOUBLE配列
ビット変換	カラムのデータのbitN~bitMまでを整数化してカラムに登録します。 整数型スカラの場合、その整数のbitN~bitMをそのまま扱います。 bool型配列の場合、配列の各要素[N]~[M]をbitN~bitMとして扱います INT8/UINT8型配列の場合、各要素は8bitあるが 0 or !0でbit化し、 各要素[N]~[M]をbitN~bitMとして扱います [対象カラム] 整数型スカラ BOOLEAN型配列 INT8/UINT8型配列 (ただし各要素は0 or 1として扱う)
データ名変更	入力データをそのまま別データとして名称を設定しカラムに登録します。 [対象カラム] 入力データの制限なし

5.8.4.1 基本情報

データ変換の基本情報は、次の通りです。

項目	説明
ベース名	データ変換
説明	様々な型のデータを取得し、各種変換を通して出力します。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 「設定方法」によって設定された内容をもとに、入力データの移動平均値を出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.8.4.2 設定方法(共通)

次の手順で、データ変換の設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. データ変換を追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON

6. 使用するデータ変換機能を選択します。



5.8.4.3 設定方法(配列分解)

1. 追加ボタンを押下して入力データ欄を追加する。



2. 入力するコンポーネント名とデータ名を入力する。
(ドロップダウン選択または手動で入力)



項目	説明
コンポーネント名	入力ポートに接続したコンポーネント名を設定
データ名	入力ポートに接続したコンポーネント名のデータ名を設定 (配列カラムのみ設定可)
有効	設定した入力データ設定を無効にする時はOFFにしてください
削除	設定した入力データ設定を削除する時に押下してください
手動設定	配列分解後の出力データ名を手動設定したい時はONにしてください (ONにすると出力データ設定の展開ボタンが有効になります)

3. (任意)追加ボタンを押下して出力データ欄を追加する。
   (手動設定をONにして出力データ設定の展開ボタンを押下すると追加ボタンが表示されます)



4. (任意)出力するデータ名を入力する。



項目	説明
配列番号	配列番号が自動採番され表示されます (編集不可)
出力データ名	出力データ名を設定
有効	設定した出力データ設定を無効にする時はOFFにしてください
削除	設定した出力データ設定を削除する時に押下してください

5. 「保存」を押下します。



6. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.8.4.4 設定方法(ビット変換)

1. 追加ボタンを押下して入力データ欄を追加する。



2. 入力するコンポーネント名とデータ名を入力する。
(ドロップダウン選択または手動で入力)



項目	説明
コンポーネント名	入力ポートに接続したコンポーネント名を設定
データ名	入力ポートに接続したコンポーネント名のデータ名を設定
有効	設定した入力データ設定を無効にする時はOFFにしてください
削除	設定した入力データ設定を削除する時に押下してください

3. 出力するデータの設定を入力する。



項目	説明
出力データ名	出力データ名を設定
データ型	出力するデータ型を設定
開始BIT	入力するデータから取得するデータの開始BITを設定。 手動入力またはビットレンジ設定を使用してください。 最下位ビットは0、最上位ビットは入力するデータ型のビット幅に応じて、 7, 15, 31, 63と変動いたします
終了BIT	入力するデータから取得するデータの終了BITを設定。 手動入力またはビットレンジ設定を使用してください。 最下位ビットは0、最上位ビットは入力するデータ型のビット幅に応じて、 7, 15, 31, 63と変動いたします
ビットレンジ設定	開始BIT、終了BITと連動しており、どちらかの方法で入力。 ビットレンジの最大値はスカラデータの場合、入力するデータ型に連動、 配列データの場合は入力する配列数に連動

4. 「保存」を押下します。



5. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.8.4.5 設定方法(データ名変更)

1. 追加ボタンを押下して入力データ欄を追加する。



2. 入力するコンポーネント名とデータ名を入力する。
(ドロップダウン選択または手動で入力)



項目	説明
コンポーネント名	入力ポートに接続したコンポーネント名を設定
データ名	入力ポートに接続したコンポーネント名のデータ名を設定
出力データ名	出力データ名を設定
有効	設定した入力データ設定を無効にする時はOFFにしてください

項目	説明
削除	設定した入力データ設定を削除する時に押下してください

3. 出力するデータ名を入力する。



4. 「保存」を押下します。



5. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.9 アクション

ここでは、各アクションコンポーネントについて説明します。

5.9.1 シェルコマンド

収集したデータを使い、任意のシェルコマンドを実行するコンポーネントです。
入力ポートからデータを受け取ると、パラメータで指定された任意のシェルコマンドを実行します。
シェルコマンドを指定する文字列には、受け取ったデータ名や型、値、タイムスタンプなどを用いることができます。



5.9.1.1 基本情報

シェルコマンドの基本情報は、次の通りです。

項目	説明
ベース名	シェルコマンド
説明	データの入力を実行トリガとして、シェルコマンドを実行します。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.9.1.2 設定項目

シェルコマンドの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
コマンド	実行するシェルコマンド ※1 変数を使用すると、対応する情報に変換できます。 ※2 OSによって指定方法が異なります。 **Linux** $HIVE_COMPONENT：データの送信元コンポーネント名 $HIVE_DATANAME：データ名 $HIVE_DATATYPE：データ型 $HIVE_DATAVALUE：値 $HIVE_TIMESTAMP：データのタイムスタンプ $HIVE_DATAARRAYSIZE：データの配列数 **Windows** %HIVE_COMPONENT%：データの送信元コンポーネント名 %HIVE_DATANAME%：データ名 %HIVE_DATATYPE%：データ型 %HIVE_DATAVALUE%：値 %HIVE_TIMESTAMP%：データのタイムスタンプ %HIVE_DATAARRAYSIZE%：データの配列数
タイムアウト(秒)	コマンド実行時のタイムアウト時間を入力

5.9.2 メール送信

収集したデータを使ってメール送信を行うコンポーネントです。
入力ポートからデータを受け取ると、指定された宛先、内容でメールを送信します。



5.9.2.1 基本情報

メール送信の基本情報は、次の通りです。

項目	説明
ベース名	メール送信
説明	データの入力を実行トリガとして、内容と宛先を指定したメールを送信します。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	メールサーバーに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.9.2.2 設定項目

メール送信の設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
SMTPホスト	メール送信を行うSMTPサーバーのホスト名orIPアドレス
SMTPポート	メール送信を行うSMTPサーバーのポート
SSL/TLS	メールの暗号化方式
SMTPユーザー名	メール送信時のSMTP認証に必要なユーザー名

項目	説明
SMTPパスワード	メール送信時のSMTP認証に必要なパスワード
タイムアウト(秒)	メール送信にかける時間の上限
FROM	送信するメールの送り主のメールアドレス
TO	送信するメールの宛先のメールアドレス
CC	送信するメールの宛先(CC)のメールアドレス
BCC	送信するメールの宛先(BCC)のメールアドレス
件名	送信するメールの件名 ※変数を使用すると、対応する情報に変換できます。詳細は「メール送信で使用可能な変数」をご覧ください。
本文	送信するメールの本文 ※変数を使用すると、対応する情報に変換できます。詳細は「メール送信で使用可能な変数」をご覧ください。

5.9.2.2.1 メール送信で使用可能な変数

変数名	説明	変換後の文字列例
$HIVE_COMPONENT	データの送信元コンポーネント名	plc-collector
$HIVE_DATANAME	データ名	D110
$HIVE_DATATYPE	データ型	double
$HIVE_DATAVALUE	値	123.45
$HIVE_DATAARRAYSIZE	データの配列数	0
$HIVE_TIMESTAMP	データのタイムスタンプ	1733356200123456789
$HIVE_TS	データのタイムスタンプ（秒単位）	1733356200

変数名	説明	変換後の文字列例
$HIVE_TS_MSEC	データのタイムスタンプ（ミリ秒単位）	1733356200123
$HIVE_TS_USEC	データのタイムスタンプ（マイクロ秒単位）	1733356200123456
$HIVE_TS_NSEC	データのタイムスタンプ（ナノ秒単位）	1733356200123456789
$HIVE_TS_DATE_LOCAL	データのタイムスタンプの年月日（ローカルタイムゾーン）	2024-12-05
$HIVE_TS_DATE_UTC	データのタイムスタンプの年月日（UTC）	2024-12-04
$HIVE_TS_TIME_LOCAL	データのタイムスタンプの時分秒（ローカルタイムゾーン）	08:50:00
$HIVE_TS_TIME_UTC	データのタイムスタンプの時分秒（UTC）	23:50:00
$HIVE_TS_DATETIME_LOCAL	データのタイムスタンプの年月日時分秒（ローカルタイムゾーン）	2024-12-05T08:50:00
$HIVE_TS_DATETIME_UTC	データのタイムスタンプの年月日時分秒（UTC）	2024-12-04T23:50:00
$HIVE_TS_TZ_DIFF	ローカルタイムゾーンの時差	+0900

変数名の記述について

- 変数名のHIVE_はSYNAPSE_と記述することも可能です。
- 変数の記述形式は$HIVE_COMPONENT、${HIVE_COMPONENT}、%HIVE_COMPONENT%の3種類の記述形式が使用できます。

5.9.2.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したメール送信の設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.9.3 PLC書き込み

PLCのレジスタへ値を書き込むコンポーネントです。
入力ポートからトリガとなるデータを受け取ると、PLCの指定レジスタに任意の値を書き込む事ができます。
書き込む値としては、固定値やトリガとなったデータの値、他コンポーネントのデータの値を指定可能です。



5.9.3.1 基本情報

PLC書き込みの基本情報は、次の通りです。

項目	説明
ベース名	PLC書き込み
説明	データの入力を実行トリガとして、PLCの指定レジスタに、任意の値を書き込みます。

項目	説明
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	PLCに対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

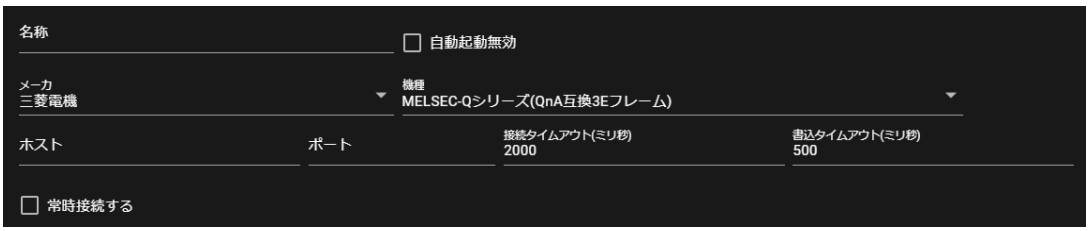
5.9.3.2 設定方法

次の手順で、PLC書き込みの設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. PLC書き込みを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



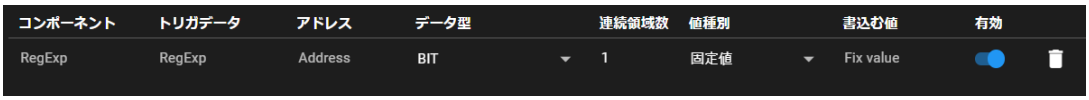
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
メーカ	接続するPLCのメーカを選択 ※選択できるメーカは次の通りです。 ・三菱電機 ・ジェイテクト ・オムロン ・キーエンス ・パナソニック
機種	接続するPLCの機種を選択 ※選択できる機種はメーカによって異なります。
ホスト	接続するPLCのホストorIPアドレス
ポート	接続するPLCのポート
接続タイムアウト（ミリ秒）	PLCへの接続にかける時間の上限
書込タイムアウト（ミリ秒）	値書き込みにかける時間の上限
常時接続する	PLCへの接続を問い合わせの都度、切断しない

6. 「トリガデータ指定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータの一覧から、書き込みのトリガとするデータを選択する
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものを書き込みのトリガとする

7. 設定対象とするデータについて、次の項目を入力します。



項目	説明
(設定方法が「正規表現で設定する」の場合)	
コンポーネント	入力ポートで受け取っているデータのうち、書き込みのトリガとするもののコンポーネント名を表す正規表現
トリガデータ	入力ポートで受け取っているデータのうち、書き込みのトリガとするもののデータ名を表す正規表現
(設定方法が「データ定義を受信して設定する」の場合)	
コンポーネント	入力ポートで受け取っているデータのうち、書き込みのトリガとするものを選択
トリガデータ	入力ポートで受け取っているデータのうち、書き込みのトリガとするものを選択
(共通)	
アドレス	値の書き込み先であるレジスタのアドレス
プログラム	値の書き込み先であるレジスタのプログラム番号 ※メーカがジェイテクトの場合のみ選択可能
データ型	値の書き込み先であるレジスタのデータ型

項目	説明
連続領域数	入力したアドレスを先頭として連続するレジスタの中から、値書込みの対象とする範囲
値種別	・固定値：書込む値として、固定値を入力する ・コンポーネント：書込む値として、入力ポートで受け取ったデータを選択する ・イベント：トリガデータの値が書き込まれる
書込む値	選択した値種別に沿った値を入力 ※値種別がイベントの場合は入力不可
有効	書き込みを行うか否か

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 「保存」を押下します。

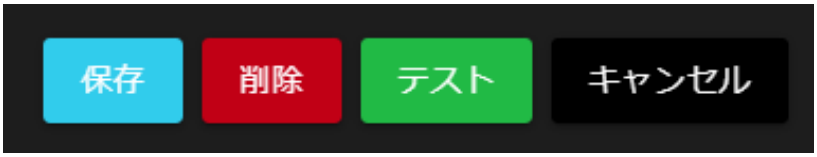


9. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.9.3.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したPLC書き込みの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.9.3.4 アドレス自動設定

一部のPLCでは、ラダー編集ツールで出力したファイルを読み込むことで、アドレスを自動で一括設定する事が出来ます。

1. ラダー編集ツールで、作成したラダーをファイルとして出力します。出力するファイルの形式は、PLCのメーカ・機種によって異なります。ファイルのサンプルは次の通りです。
   - 三菱電機 MELSEC-Qシリーズ(QnA互換3Eフレーム)、MELSEC iQ-Rシリーズ(QnA互換3Eフレーム)

     サンプルファイルのダウンロード
   - オムロン SYSMAC CS/CJ/CP/NSJシリーズ(FINS)

     サンプルファイルのダウンロード
2. 「アドレス自動設定」を押下します。



3. ラダー編集ツールで出力したファイルを選択します。
4. ファイルをもとに、アドレスが自動で設定されます。

5.9.4 Modbus書き込み

Modbus TCP対応機器の指定アドレスへ値を書き込むコンポーネントです。
入力ポートからトリガとなるデータを受け取ると、本コンポーネントがModbusのマスターとなり、Modbusスレーブ機器の指定レジスタに任意の値を書き込みます。
書き込む値としては、固定値やトリガとなったデータの値、他コンポーネントのデータの値を指定可能です。



5.9.4.1 基本情報

Modbus書き込みの基本情報は、次の通りです。

項目	説明
ベース名	Modbus書き込み
説明	データの入力を実行トリガとして、Modbus TCP対応機器の指定レジスタに、任意の値を書き込みます。

項目	説明
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
その他	Modbus TCP対応機器に対するテスト接続を行うことができます。詳細は「テスト接続」をご覧ください。

5.9.4.2 設定方法

次の手順で、Modbus書き込みの設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. Modbus書き込みを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。

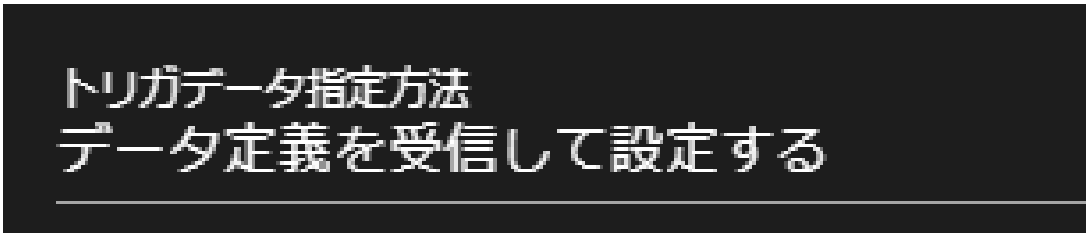


5. 次のコンポーネント情報を入力します。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
エンディアン	書き込む値のバイトオーダ
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	接続する機器のホストorIPアドレス
ポート	接続する機器のポート
接続タイムアウト（ミリ秒）	機器への接続にかける時間の上限
書込タイムアウト（ミリ秒）	値書き込みにかける時間の上限
スレーブID	接続先のスレーブID

6. 「トリガデータ指定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータの一覧から、書き込みのトリガとするデータを選択する
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものを書き込みのトリガとする

7. 設定対象とするデータについて、次の項目を入力します。



項目	説明
(設定方法が「正規表現で設定する」の場合)	
コンポーネント	入力ポートで受け取っているデータのうち、書き込みのトリガとするもののコンポーネント名を表す正規表現
トリガデータ	入力ポートで受け取っているデータのうち、書き込みのトリガとするもののデータ名を表す正規表現
(設定方法が「データ定義を受信して設定する」の場合)	
コンポーネント	入力ポートで受け取っているデータのうち、書き込みのトリガとするものを選択
トリガデータ	入力ポートで受け取っているデータのうち、書き込みのトリガとするものを選択
(共通)	
アドレス	値の書き込み先であるレジスタのアドレス ※Modbusの仕様上、アドレス範囲は通常「1~65536」となっていますが、SpeeDBee Synapseで設定する際のアドレス範囲は「0~65535」のように-1したアドレスを設定する必要があります。
データ型	値の書き込み先であるレジスタのデータ型
連続領域数	入力したアドレスを先頭として連続するレジスタの中から、値書込みの対象とする範囲
種類	値の書き込み先であるレジスタの種類
エンディアン	書き込むデータのバイトオーダ ※「-」の場合、全体のエンディアンが適用される
値種別	・固定値：書込む値として、固定値を入力する ・コンポーネント：書込む値として、入力ポートで受け取ったデータを選択する ・イベント：トリガデータの値が書き込まれる

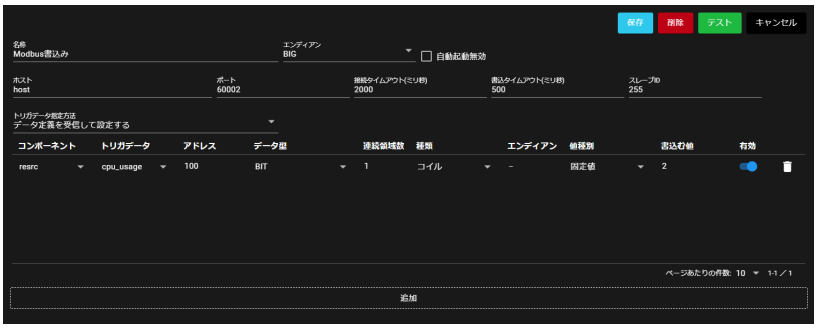
項目	説明
書込む値	選択した値種別に沿った値を入力 ※値種別がイベントの場合は入力不可
有効	書き込みを行うか否か

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 「保存」を押下します。

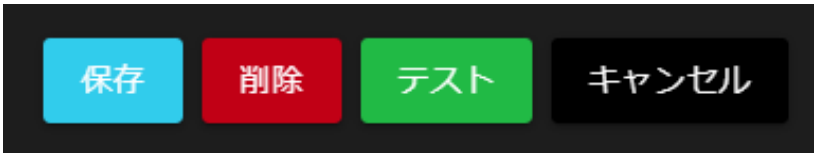


9. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

5.9.4.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したModbus書き込みの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.9.5 EtherNet/IP書き込み

EtherNet/IP対応機器の指定アドレスへ値を書き込むコンポーネントです。
入力ポートからトリガとなるデータを受け取った際、本コンポーネントがオリジネーターとなりEtherNet/IP対応機器に接続し、指定変数に任意の値を書き込みます。



5.9.5.1 基本情報

EtherNet/IP書き込みの基本情報は、次の通りです。

項目	説明
ベース名	EtherNet/IP書き込み
説明	データの入力を実行トリガとして、EtherNet/IP対応機器の指定した変数に、任意の値を書き込みます。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

項目	説明
その他	変数を自動で設定することができます。詳細は「変数自動設定」をご覧ください。

5.9.5.2 設定方法

次の手順で、EtherNet/IP書き込みの設定を行うことができます。

1. 出力ポートが存在する任意のコンポーネントを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
2. 追加したコンポーネントを起動し、データを出力させます。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。
3. EtherNet/IP書き込みを追加します。コンポーネントの追加方法は、「コンポーネントの追加」をご覧ください。
4. 入力ポートに接続するコンポーネントに、追加したコンポーネントを選択します。



5. 次のコンポーネント情報を入力します。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
ホスト	EtherNet/IP対応機器のホスト名またはIPアドレスを入力

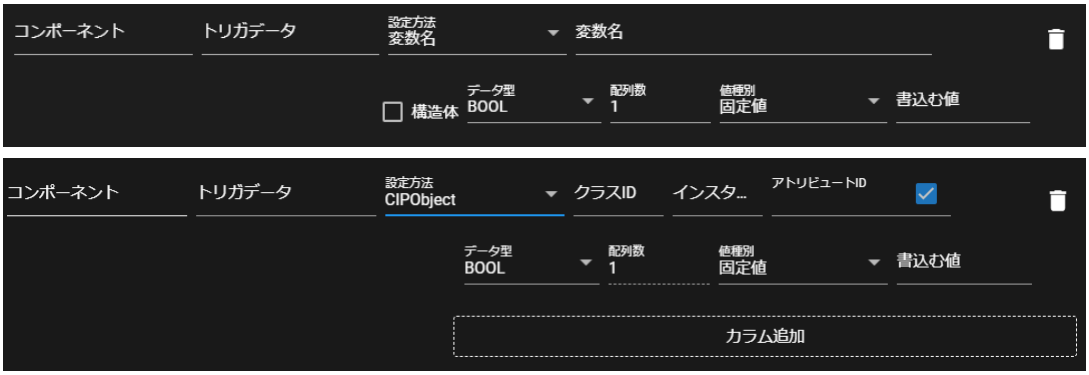
項目	説明
ポート	EtherNet/IP対応機器のポートを入力
接続タイムアウト(ミリ秒)	接続時のタイムアウト時間を入力

6. 「トリガデータ指定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータの一覧から、書き込みのトリガとするデータを選択する
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものを書き込みのトリガとする

7. 設定対象とするデータについて、次の項目を入力します。



項目	説明
(設定方法が「正規表現で設定する」の場合)	
コンポーネント	入力ポートで受け取っているデータのうち、書き込みのトリガとするもののコンポーネント名を表す正規表現
トリガデータ	入力ポートで受け取っているデータのうち、書き込みのトリガとするもののデータ名を表す正規表現

項目	説明
(設定方法が「データ定義を受信して設定する」の場合)	
コンポーネント	入力ポートで受け取っているデータのうち、書き込みのトリガとするものを選択
トリガデータ	入力ポートで受け取っているデータのうち、書き込みのトリガとするものを選択
(共通)	
設定方法	取得するデータの設定方法を選択
変数名	タグ名を入力 ※設定方法が変数名の場合のみ入力可能です。
構造体	複数のメンバを登録する場合ON ※設定方法が変数名の場合のみ設定可能です。
クラスID	クラスIDを入力 ※設定方法がCIPObjectの場合のみ入力可能です。
インスタンスID	インスタンスIDを入力 ※設定方法がCIPObjectの場合のみ入力可能です。
アトリビュートID	アトリビュートIDを入力 ※設定方法がCIPObjectの場合のみ入力可能です。
アトリビュート設定	アトリビュートIDを入力する時にチェック ※設定方法がCIPObjectの場合のみ操作可能です。
データ型	メンバのデータ型
配列数	メンバの配列数 ※設定方法が変数名の場合のみ操作可能です。
値種別	・固定値：書込む値として、固定値を入力する ・コンポーネント：書込む値として、入力ポートで受け取ったデータを選択する ・イベント：トリガデータの値が書き込まれる

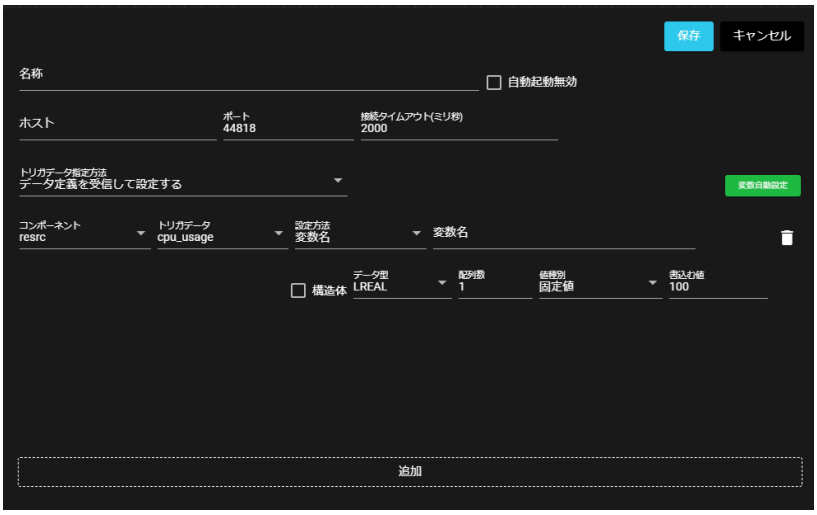
項目	説明
書込む値	選択した値種別に沿った値を入力 ※値種別がイベントの場合は入力不可

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

8. 「保存」を押下します。



9. コンポーネントを起動します。コンポーネントの起動方法は、「コンポーネントの起動/停止」をご覧ください。

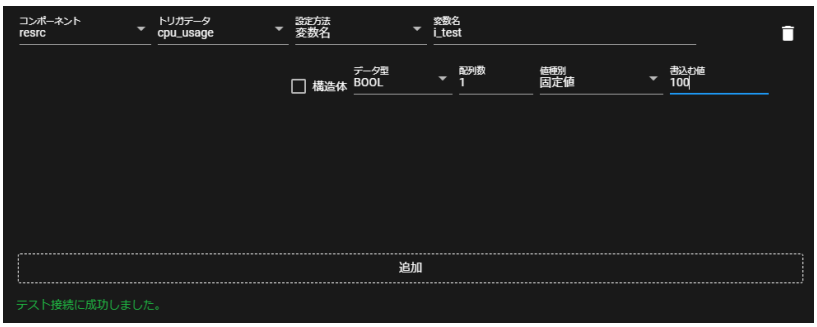
5.9.5.3 テスト接続

次の手順で、設定した接続先に対してテスト接続を行うことができます。

1. 登録したEtherNet/IP書き込みの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定した接続先へのテスト接続が行われ、設定画面の下部に結果が表示されます。



5.9.5.4 変数自動設定

Sysmac Studioがエクスポートするグローバル変数情報などを、指定のフォーマットのテキストファイルに変換して読み込むことで、変数を自動設定することができます。

1. Sysmac Studioがエクスポートするグローバル変数情報などを、次の列情報を持つテキストファイルに変換します。
   - HOST
   - NAME
   - DATATYPE
   - ADDRESS
   - COMMENT
   - TAGLINK
   - RW
   - POU

   例

```
HOST    NAME    DATATYPE    ADDRESS COMMENT TAGLINK RW  POU
        b_test  BOOL        test1           TRUE    RW
        i_test  UINT        test2           TRUE    RW
        s_test  STRING(256) test3           TRUE    RW
```

2. 「変数自動設定」を押下します。



3. 用意したテキストファイルを選択します。

4. ファイルをもとに、変数が自動で設定されます。

5.10 開発用コンポーネント

ここでは、各開発用コンポーネントについて説明します。 開発用コンポーネントは、SCCDEテスト実行でのみ利用可能です。

5.10.1 ダミーコレクタ

ダミーデータの登録を行うコンポーネントです。
指定された運転ファイル(csv)の内容をもとに、カラムを作成しダミーデータを登録します。



5.10.1.1 基本情報

ダミーコレクタの基本情報は、次の通りです。

項目	説明
ベース名	ダミーコレクタ
説明	ダミーデータを登録します。
消費ポイント	0
ポート	入力：0 出力：1 運転ファイル(csv)の内容をもとにダミーデータを出力します。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.10.1.2 設定項目

ダミーコレクタの設定項目は、次の通りです。



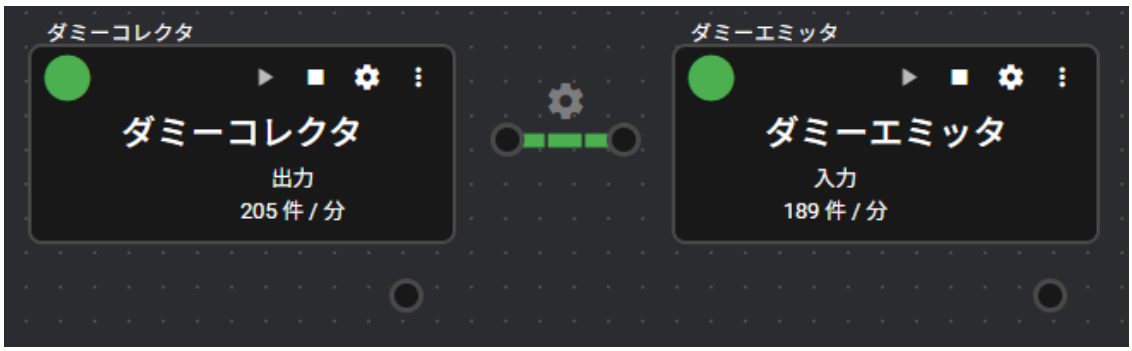
項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
運転ファイル(csv)	作成するカラムや登録するデータを定義したcsv形式のファイルを選択

運転ファイルは、sccdeの作業ディレクトリ内にある`simulation_data`ディレクトリに配置されています。 初期状態では、以下のファイルを選択することが可能ですが、任意のファイルを追加して選択することも可能です。 運転ファイルのフォーマットについては、「運転ファイルのフォーマット」を参照してください。

ファイル名	説明
all_primitive_data_type.csv	プリミティブ型のデータを登録します。
array_data.csv	配列のデータを登録します。

5.10.2 ダミーエミッタ

収集したデータをCSVファイルとして、ストレージへ保存するコンポーネントです。
入力ポートに接続された各種コンポーネントからプリミティブ型のデータを受け取り、そのデータを特定のディレクトリへ保存します。



5.10.2.1 基本情報

ダミーエミッタの基本情報は、次の通りです。

項目	説明
ベース名	ダミーエミッタ
説明	収集データをCSV形式でストレージへ保存します。
消費ポイント	0
ポート	入力：1 コンポーネントから、プリミティブ型のデータを受け取ります。受け取ったデータは、CSV形式でストレージへ保存されます。 出力：0 システム：1 コンポーネントの稼働状況やエラー情報を出力します。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

5.10.2.2 設定項目

ダミーエミッタの設定項目は、次の通りです。



項目	説明
名称	コンポーネントに付与する名前 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
改行コード	CSVファイルの改行コードを選択
BOMを付ける	CSVファイルにファイルBOMを付ける場合ON
タイムゾーン	ファイル、CSVデータに付与するタイムスタンプのタイムゾーンを選択

出力先は、sccdeの作業ディレクトリ内にある`.synapse/var-speedbeesynapse/projects/sccde/data`ディレクトリに、 登録しているダミーエミッタごとにディレクトリが作成され、ファイルが出力されます。

5.10.2.3 ファイルの切替タイミング

ダミーエミッタは起動ごとにファイルが新しく切り替わります。 起動後は、基本的に同じファイルへ出力し続けますが、以下の場合のみファイルを新しく切り替えます。

- ファイルのサイズが上限(10MB)を超えた場合。
- 入力ポートで受け取ったデータのカラム定義（カラム数、カラム名など）が変化した場合。

※出力されたファイルは、自動削除されないので、必要に応じて削除してください。

5.11 システムコンポーネント

ここでは、各システムコンポーネントについて説明します。システムコンポーネントは、他のコンポーネントと違い複数のインスタンスを設定することができません。システムパネルへ配置されているコンポーネントを設定して利用します。

5.11.1 DBクエリサービス

SpeeDBee Synapse内で集めたデータへの問い合わせを受け付けるコンポーネントです。このコンポーネントは、データ収集やデータ参照を行いません。外部からのSQLを処理するために使用します。詳細は、DBクエリ（SQL）を参照してください。
クエリ実行を行う際は、DBクエリサービス利用時の注意点の一読を推奨します。

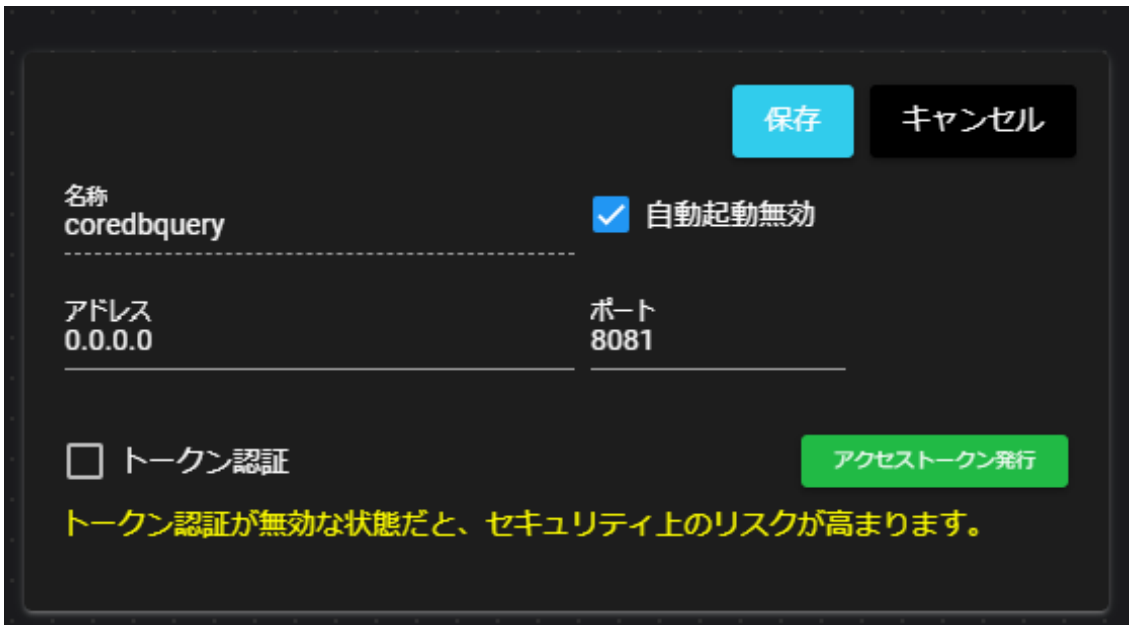
5.11.1.1 基本情報

DBクエリサービスの基本情報は、次の通りです。

項目	説明
ベース名	DBクエリサービス
説明	データベースへの問い合わせを受け付けるコンポーネントです。起動するとHTTP経由でDBへのクエリを発行できます。 DBクエリについての詳細は、「DBクエリ（SQL）」をご覧ください。 このコンポーネントは、SpeeDBee Synapseで１つしか設定できません。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
消費ポイント	0
ポート	入力：0 出力：0

5.11.1.2 設定項目

DBクエリサービスの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前（coredbquery固定）
自動起動無効	コンポーネントの自動起動を無効にする場合ON
アドレス	受信側のIPアドレスを入力、0.0.0.0であればすべてのネットワークからの通信を許可 ※一部の環境では、この設定を変更/編集できません。
ポート	DBクエリを受け付けるポートを入力 ※一部の環境では、この設定を変更/編集できません。
トークン認証	WebAPIリクエスト時の認証用アクセストークンを選択
アクセストークン発行	アクセストークン発行画面を表示 ※アクセストークン画面の詳細は、「アクセストークン発行」をご覧ください。

5.11.2 コアインターナル

SpeeDBee Synapse内の内部ステータスを管理するコンポーネントです。このコンポーネントは、ユーザーが起動したり停止を行うことはできません。自動的に起動され、常時動作する特殊なコンポーネントです。
稼働マシンの電断によりデータの欠損があった場合、検知したエラーをメールで通知することができます。手順の詳細は、トラブルシュートを参照してください。



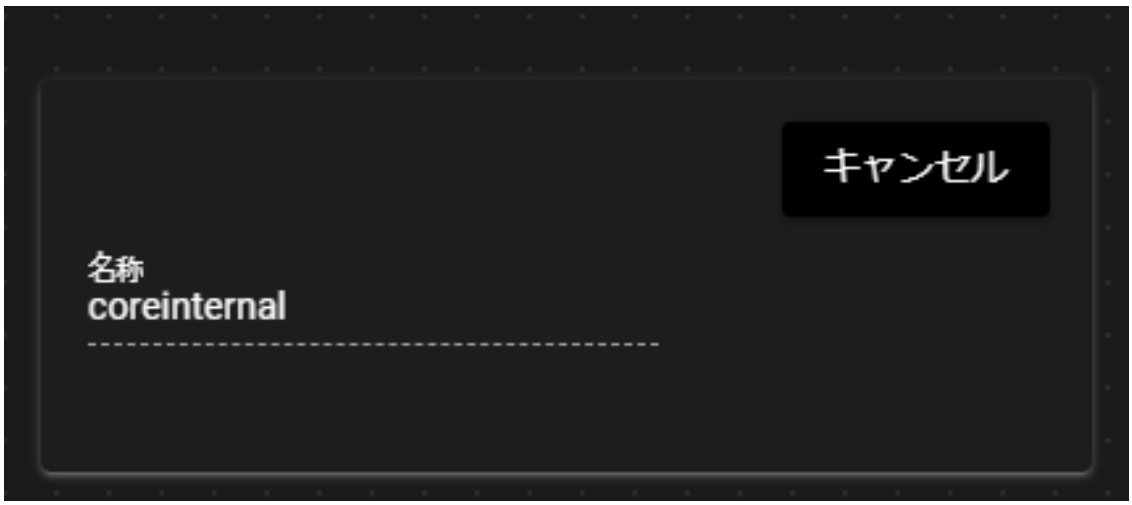
5.11.2.1 基本情報

コアインターナルの基本情報は、次の通りです。

項目	説明
ベース名	コアインターナル
説明	コア内で定期的な実行が必要な処理を行うコンポーネントです。 このコンポーネントは、SpeeDBee Synapseで１つしか存在しません。また、設定変更および停止など制御することもできません。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
消費ポイント	0
ポート	入力：0 出力：0

5.11.2.2 設定項目

コアインターナルは設定を変更することができない特殊なコンポーネントです。



項目	説明
名称	コンポーネントの名前（coreinternal固定）

5.11.3 エラーマネージャー

SpeeDBee Synapse内で発生した全コンポーネントのエラーを検知できるコンポーネントです。
検知したエラーをメールで通知することができます。手順の詳細は、トラブルシュートを参照してください。



5.11.3.1 基本情報

エラーマネージャーの基本情報は、次の通りです。

項目	説明
ベース名	エラーマネージャー

項目	説明
説明	SpeeDBee Synapse内で発生した全エラーを検知するコンポーネントです。 このコンポーネントは、SpeeDBee Synapseで１つしか存在しません。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
消費ポイント	0
ポート	入力：0 出力：1

5.11.3.2 設定項目

エラーマネージャーの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前（errormanager固定）
自動起動無効	コンポーネントの自動起動を無効にする場合ON
処理間隔	エラー情報を収集する処理間隔（秒単位）
タイムアウト	エラー情報を収集する処理のタイムアウト時間（秒単位）

5.11.3.3 出力データ項目

エラーマネージャーは、次のデータを出力します。

項目	カラム名	データ型	説明
停止コンポーネント数	standby_component_count	uint16	前回のエラー情報収集から今回までの、停止しているコンポーネントの数の変化
稼働コンポーネント数	running_component_count	uint16	前回のエラー情報収集から今回までの、稼働しているコンポーネントの数の変化
警告コンポーネント数	warning_component_count	uint16	前回のエラー情報収集から今回までの、警告状態で稼働しているコンポーネントの数の変化
異常コンポーネント数	error_component_count	uint16	前回のエラー情報収集から今回までの、エラー状態で停止しているコンポーネントの数の変化
警告コンポーネント名一覧	warning_components	string	前回のエラー情報収集から今回までの、警告状態で稼働しているコンポーネントの名の一覧（カンマ区切り）の変化
異常コンポーネント名一覧	error_components	string	前回のエラー情報収集から今回までの、エラー状態で停止しているコンポーネントの名の一覧（カンマ区切り）の変化
最新の停止コンポーネント数	latest_standby_component_count	uint32	エラー情報収集時の、停止しているコンポーネントの数
最新の稼働コンポーネント数	latest_running_component_count	uint32	エラー情報収集時の、稼働しているコンポーネントの数
最新の警告コンポーネント数	latest_warning_component_count	uint32	エラー情報収集時の、警告状態で稼働しているコンポーネントの数
最新の異常コンポーネント数	latest_error_component_count	uint32	エラー情報収集時の、エラー状態で停止しているコンポーネントの数

項目	カラム名	データ型	説明
最新の警告コンポーネント名一覧	latest_warning_components	string	エラー情報収集時の、警告状態で稼働しているコンポーネントの名の一覧（カンマ区切り）
最新の異常コンポーネント名一覧	latest_error_components	string	エラー情報収集時の、エラー状態で停止しているコンポーネントの名の一覧（カンマ区切り）

5.11.4 イベントデータマネージャー

このコンポーネントは、i-Triggerで記録されるイベントデータの定期削除などを行う特殊なコンポーネントです。i-Triggerを使用しない場合は、起動する必要はありません。

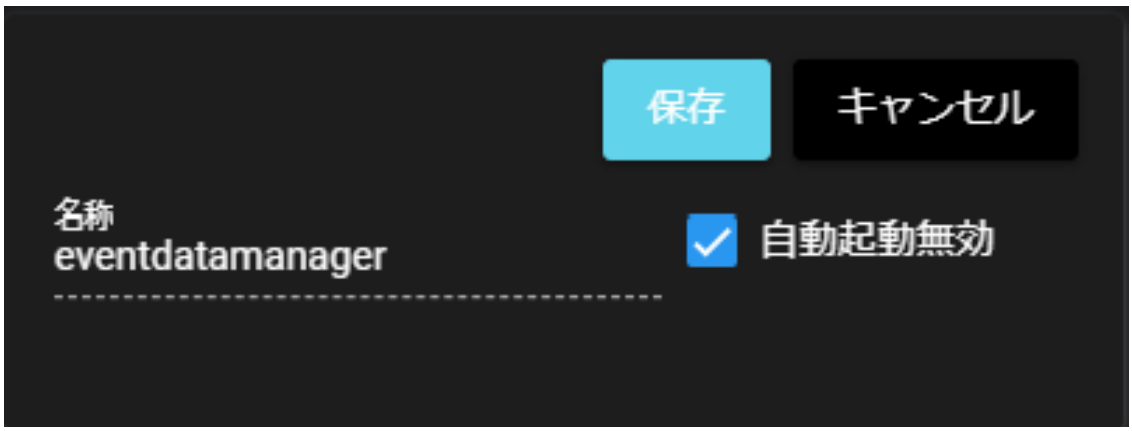
5.11.4.1 基本情報

イベントデータマネージャーの基本情報は、次の通りです。

項目	説明
ベース名	イベントデータマネージャー
説明	イベントデータのデータ管理を行います。Event Recorderによって保存された各種データを定期削除します。また、内部データベースのバックアップ、リストアについても処理します。このコンポーネントは、SpeeDBee Synapseで１つしか存在しません。出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。
消費ポイント	0
ポート	入力：0 出力：0

5.11.4.2 設定項目

イベントデータマネージャーの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前（eventdatamanager固定）
自動起動無効	コンポーネントの自動起動を無効にする場合ON

5.11.4.3 データ管理の動作仕様

イベントデータマネージャーは、データ管理を次のように行います。本項は、イベントデータマネージャーのデータ管理について、確認したい場合にのみ参照してください。

Event Recorderの保存したデータの定期削除

Event Recorderの設定には、「保存時間」の設定項目があります。 Event Recorderが保存したデータ（イベントデータJSONファイル、動画ファイル）を、指定された保存時間に応じて削除します。

- 保存時間を途中で変更した場合、最新の保存時間を基準に削除を行います
- Event Recorderのインスタンスを削除しても、イベントデータマネージャーが起動している限り、保存時間を過ぎたデータは削除されます

ビューワ用内部データベースファイルのバックアップ・リストア

Event Recorderは、イベントの記録時のデータを内部のデータベースで管理します。このデータベースは、i-Trigger専用です。このデータベースを定期的にバックアップして、破損時はリストアを行います。

- バックアップは、1日単位で処理されます（日時の切替り10分以内に処理されます）
- 破損時のリストアは、1日前のデータベースより復元されるため、最新のデータは消失します
- ディスクフル時は、バックアップ/リストアもエラーとなります

5.12 コンポーネント共通の設定項目

ここでは、コンポーネントのポートの設定、フローリンクのフィルターの設定について説明します。

5.12.1 出力ポート

出力ポートは、フローリンクの接続元であり、コンポーネントがデータを出力するポートです。 出力ポートでは、出力されるデータに対して、永続化、登録条件、数値変換の有無等の設定を行う事ができます。

5.12.1.1 永続化の設定

永続化を有効にしたデータは、ストレージ内に保存され、保存されている間、SQLからの参照、Grafana連携での参照が可能となります。次の手順で、コンポーネントの出力データに対して永続化の設定を行うことができます。

1. 次のいずれかの方法で出力ポートのダイアログを開きます。
   - メニューアイコンを左クリックして「出力ポート設定」を選択



   - 出力ポートを右クリック

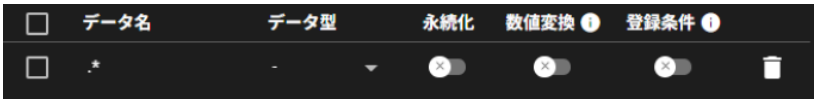


2. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	出力データを一覧で表示し、個別に永続化の設定を行う
正規表現で設定する	正規表現を入力し、出力データの中でマッチしたものに対して、永続化の設定を行う

3. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



項目	説明
データ名	出力データのうち、設定対象とするもののデータ名を表す正規表現
データ型	出力データのうち、設定対象とするもののデータ型

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

4. 次のいずれかの方法を行い、永続化の設定ダイアログを開きます。
   - 永続化の設定を行うデータの「永続化」のアイコンを押下します。



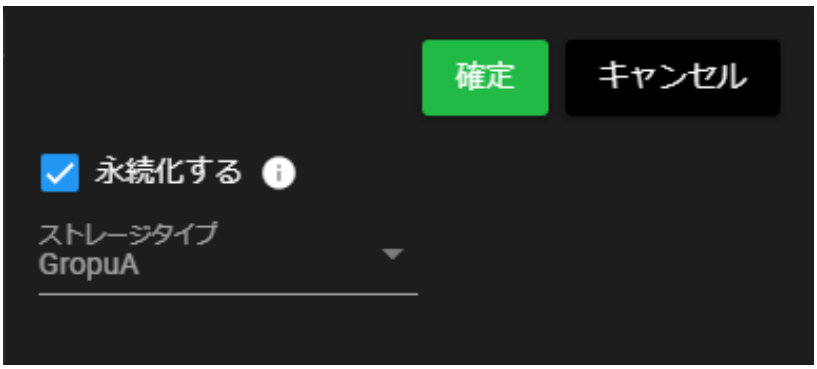
   - 永続化の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下し「永続化」を選択します。チェックを付けたデータに、同じ設定内容が適用されます。



5. 「永続化する」をONにします。

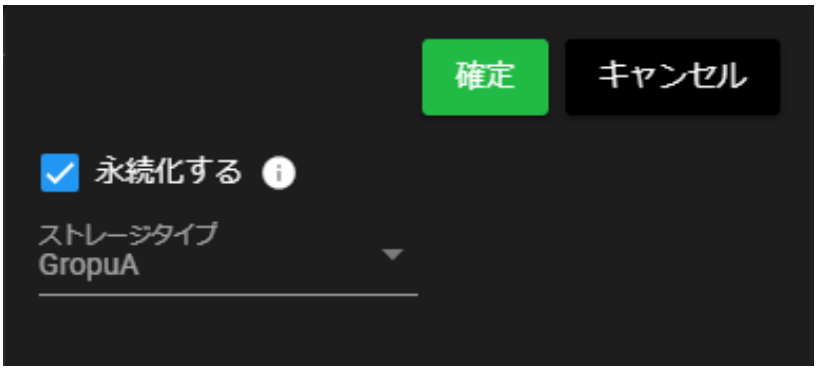


6. 次の項目を入力します。



項目	説明
ストレージタイプ	時系列データベースに保存する期間。 サーバー設定の「グループ単位で保存期間を指定」で、GroupA・GroupB・GroupCの保存期間を確認・変更する事ができます。「グループ単位で保存期間を指定」がONの場合のみ変更できます。 デフォルト値 • GroupA : 30分 • GroupB : 24時間 • GroupC : 7日間

7. 「確定」を押下します。



8. 「保存」を押下します。



9. 永続化の設定が適用されます。

5.12.1.2 数値変換の設定

数値変換は、元のデータをそのまま登録するのではなく、演算した結果をデータ登録するための機能です。次の手順で、コンポーネントの出力データに対して数値変換の設定を行うことができます。

1. 次のいずれかの方法で出力ポートのダイアログを開きます。
   - メニューアイコンを左クリックして「出力ポート設定」を選択



   - 出力ポートを右クリック



2. 「設定方法」を、次の中から選択します。

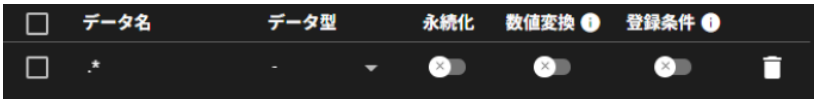


項目	説明
データ定義を受信して設定する	出力データを一覧で表示し、個別に数値変換の設定を行う
正規表現で設定する	正規表現を入力し、出力データの中でマッチしたものに対して、数値変換の設定を行う

配列型の出力データは対象外

配列型の出力データは、数値変換の対象外です。
正規表現の条件で、データ名が該当する場合でも、配列型であった場合は数値変換は処理されません。
例えば、リソースコレクタの"cpu_core_usage"が該当します。

3. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



項目	説明
データ名	出力データのうち、設定対象とするもののデータ名を表す正規表現

項目	説明
データ型	出力データのうち、設定対象とするもののデータ型

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

4. 次のいずれかの方法を行い、数値変換の設定ダイアログを開きます。
   - 数値変換の設定を行うデータの「数値変換」のアイコンを押下します。



   - 数値変換の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下し「数値変換」を選択します。チェックを付けたデータに、同じ設定内容が適用されます。



5. 「数値変換を行う」をONにします。



6. 次の項目を入力します。



項目	説明
変換後の型	数値変換後のデータ型
変換内容	どのような数値変換式を記述するかの選択 ※選択できる変換内容は次の通りです。 ・任意の数式 ・BCD(4桁) → INT ・BCD(8桁) → INT ・BCD(16桁) → INT ※「任意の数式」を選択すると数値変換で実行する変換式を任意で入力することができます。 ※「BCD(4桁) → INT」、「BCD(8桁) → INT」、「BCD(16桁) → INT」を選択するとBCD値を整数値に変換する数式が自動で入力されます。これらを選択している状態では変換式の編集を行うことができません。変換式の編集を行う場合には「任意の数式」を選択してください ※BCD値から整数値に変換したデータは整数型となります。PLC書込みやEtherNet/IP書込みで変換後の値をBCDを扱うアドレスに書き込むと正しい値となりません
変換式	数値変換で実行する式 ※入力エリア右端のアイコンをクリックすると、入力例と式に使用できる要素一覧が表示されます。

7. 「確定」を押下します。



8. 「保存」を押下します。

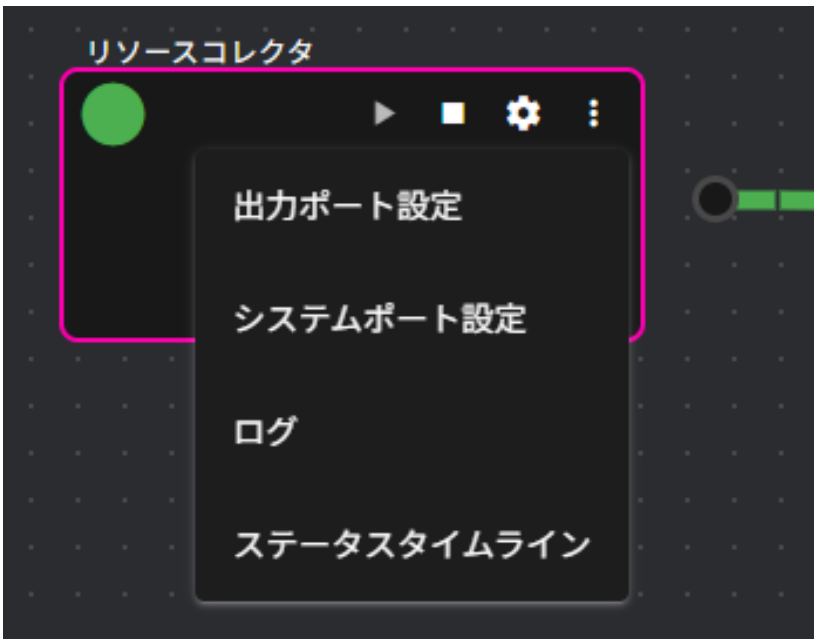


9. 数値変換の設定が適用されます。

5.12.1.3 登録条件の設定

登録条件は、条件に合致したデータのみを登録する機能です。条件に合致しないデータは、非該当のデータとして登録されません。次の手順で、コンポーネントの出力データに対して登録条件の設定を行うことができます。

1. 次のいずれかの方法で出力ポートのダイアログを開きます。
   - メニューアイコンを左クリックして「出力ポート設定」を選択



- 出力ポートを右クリック



2. 「設定方法」を、次の中から選択します。

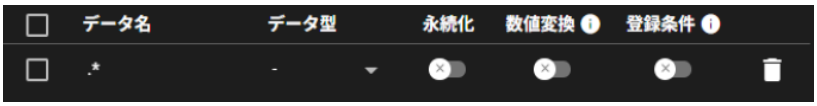


項目	説明
データ定義を受信して設定する	出力データを一覧で表示し、個別に登録条件の設定を行う
正規表現で設定する	正規表現を入力し、出力データの中でマッチしたものに対して、登録条件の設定を行う

配列型の出力データは対象外

配列型の出力データは、登録条件の対象外です。
正規表現の条件で、データ名が該当する場合でも、配列型であった場合は登録条件は処理されません。
例えば、リソースコレクタの"cpu_core_usage"が該当します。

3. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



項目	説明
データ名	出力データのうち、設定対象とするもののデータ名を表す正規表現
データ型	出力データのうち、設定対象とするもののデータ型

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

4. 次のいずれかの方法を行い、登録条件の設定ダイアログを開きます。
   - 登録条件の設定を行うデータの「登録条件」のアイコンを押下します。



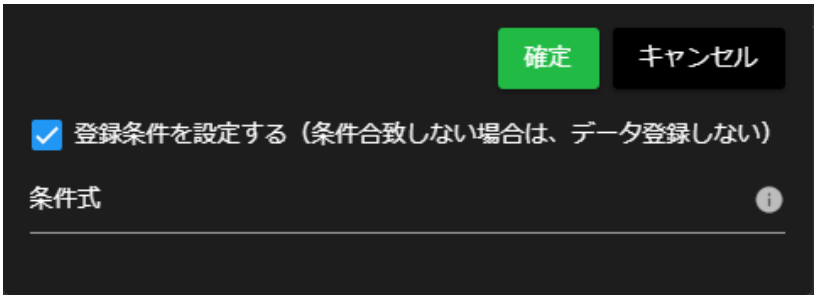
   - 登録条件の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下し「登録条件」を選択します。チェックを付けたデータに、同じ設定内容が適用されます。



5. 「登録条件を設定する」をONにします。



6. 次の項目を入力します。



項目	説明
条件式	登録するデータを判定する条件式 ※入力エリア右端のアイコンをクリックすると、入力例と式に使用できる要素一覧が表示されます。

7. 「確定」を押下します。



8. 「保存」を押下します。



9. 登録条件の設定が適用されます。

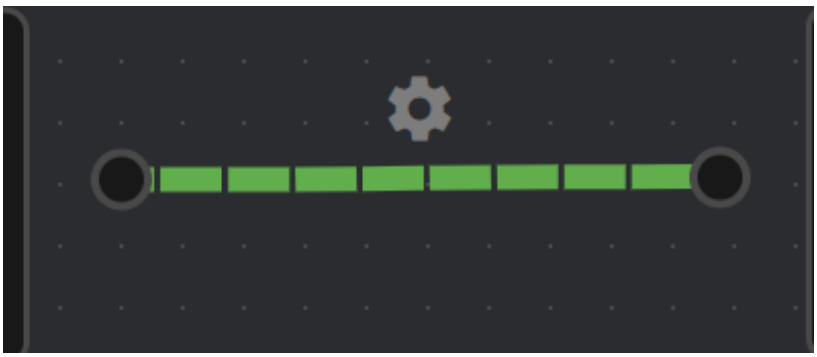
5.12.2 フィルタ

フィルタは、各データをフローリンク上で通過させるかどうか選択することができる機能です。

5.12.2.1 設定方法

次の手順で、フィルタを設定することができます。

1. フローリンクを右クリックします。



2. 「フィルタ」を選択します。

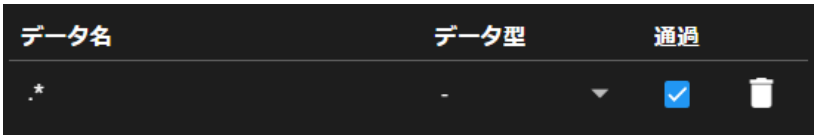


3. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	フローリンクの接続元のデータを一覧で表示し、個別にフィルタの設定を行う
正規表現で設定する	正規表現を入力し、フローリンクの接続元のデータの中でマッチしたものに対して、フィルタの設定を行う

4. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



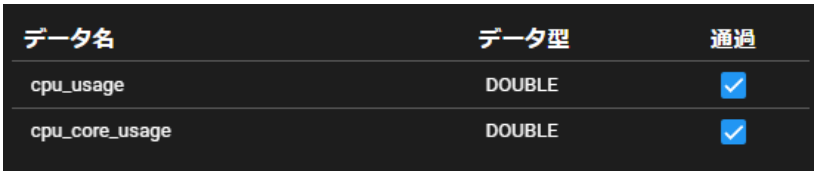
項目	説明
データ名	フローリンクの接続元のデータのうち、設定対象とするもののデータ名を表す正規表現
データ型	フローリンクの接続元のデータのうち、設定対象とするもののデータ型

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

5. 通過させ、フローリンクの接続先に渡すデータの「通過」をONにします。



6. 「保存」を押下します。



7. フィルタの設定が適用されます。設定が適用されたフローリンクは、アイコンの色が変化します。



5.12.2.2 削除

次の手順で、フィルタを削除することができます。

1. フィルタが設定されたフローリンクを右クリックします。



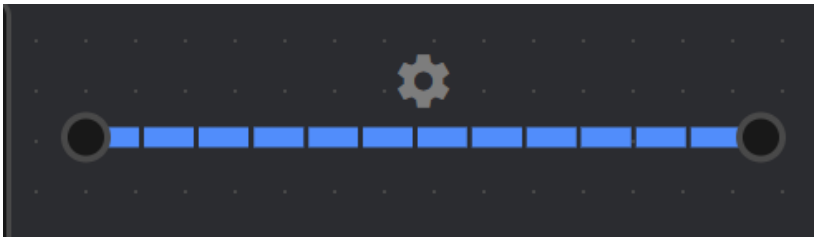
2. 「フィルタ」を選択します。



3. 「削除」を押下し、確認ダイアログで「はい」を選択します。



4. フィルタの設定が削除されます。アイコンの色が変化します。



5.12.3 フローリンクアイコン

フローリンクの接続時に出現するアイコンを左クリックすると、フローリンクを右クリックした際と同様の操作が可能となります。 フィルタ設定に関する詳細は「フィルタ」をご覧ください。 フローリンク削除に関する詳細は「削除」をご覧ください。



5.12.4 入力ポート

入力ポートは、フローリンクの接続先であり、コンポーネントがデータを受け取るポートです。 入力ポートでは、受け取ったデータに対して、イベントの使用・集約の有無等の設定を行う事ができます。

5.12.4.1 遅延時間の設定

次の手順で、コンポーネントが受け取ったデータに対して遅延時間の設定を行うことができます。

1. 次のいずれかの方法で入力ポートのダイアログを開きます。
   - メニューアイコンを左クリックして「入力ポート設定」を選択



   - 入力ポートを右クリック



2. 「遅延時間(ミリ秒)」に、データの収集遅延時間をミリ秒単位で入力します。
   詳しくは「データ収集に関連する動作仕様について」の「データ参照時の遅延時間について」を参照してください



3. 「保存」を押下します。



4. 遅延時間の設定が適用されます。

5.12.4.2 イベントの設定

次の手順で、コンポーネントが受け取ったデータに対してイベントの設定を行うことができます。

1. イベントトリガを登録します。詳細は「イベントトリガ」をご覧ください。
2. イベントトリガからコンポーネントにフローリングをつなぎます。詳細は「フローリンクの追加」をご覧ください。
3. 次のいずれかの方法で入力ポートのダイアログを開きます。
   - メニューアイコンを左クリックして「入力ポート設定」を選択



- 入力ポートを右クリック



4. 「イベントを使用する」をONにします。



5. 「イベント」に、登録したイベントトリガを選択します。



6. 次の項目を入力します。



項目	説明
イベント発生前の取得範囲(秒)	イベント発生前のデータを取得する範囲
イベント終了後の取得範囲(秒)	イベント発生後のデータを取得する範囲

7. 「保存」を押下します。



8. イベントの設定が適用されます。

5.12.4.3 集約の設定

次の手順で、集約の設定を行うことができます。

1. 次のいずれかの方法で入力ポートのダイアログを開きます。
   - メニューアイコンを左クリックして「入力ポート設定」を選択



   - 入力ポートを右クリック



2. 「集約する」をONにします。



3. 「ウィンドウ関数」を次の中から選択します。



項目	説明
タンブリングウィンドウ	指定した時間幅で、集約対象のデータの範囲を区切る

4. ウィンドウ幅を指定します。選択したウィンドウ関数によって、入力する項目が異なります。



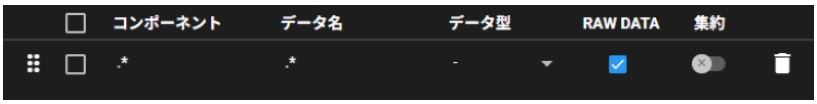
項目		説明
(タンブリングウィンドウを選択した場合)	ウィンドウ幅	集約対象の時間幅を入力(秒単位)

5. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	入力ポートで受け取っているデータを一覧で表示し、個別に集約の設定を行う
正規表現で設定する	正規表現を入力し、入力ポートで受け取っているデータの中でマッチしたものに対して、集約の設定を行う

6. 設定対象とするデータについて、次の項目を入力します。



項目	説明
コンポーネント名	入力ポートで受け取っているデータのうち、設定対象とするもののコンポーネント名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
データ名	入力ポートで受け取っているデータのうち、設定対象とするもののデータ名を表す正規表現 ※設定方法が「正規表現で設定する」の場合のみ入力可能
データ型	入力ポートで受け取っているデータのうち、設定対象とするもののデータ型 ※設定方法が「正規表現で設定する」の場合のみ選択可能
出力名	既存のデータ名を上書きする新しい名前 ※設定方法が「データ定義を受信して設定する」の場合のみ選択可能
RAW DATA	加工していないデータを取得する

正規表現の入力例

正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

7. 次のいずれかの方法を行い、集約の設定ダイアログを開きます。
   - 集約の設定を行うデータの「集約」のアイコンを押下します。



   - 集約の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下します。チェックを付けたデータに、同じ設定内容が適用されます。



8. 取得する集約値にチェックを付けます。



項目	説明
最古値	対象データ内の最も古い値
最新値	対象データ内の最も新しい値
件数	対象のデータ件数
合計	対象のデータの合計値
2乗和	対象の各データの二乗を合計した値
偏差平方和	対象の各データが平均からどれだけ離れているかの偏差を二乗して合計した値
最小値	対象のデータ内で最も小さい値
最大値	対象のデータ内で最も大きい値
範囲	対象のデータの最大値と最小値の差
算術平均	対象のデータの平均値
分散	対象の各データが平均からどれだけ離れているかの平均の二乗
標準偏差	分散の平方根
不偏分散	分散を件数-1で割った値

項目	説明
標本不偏標準偏差	標本の標準偏差
標準誤差	標本平均の標準偏差
変動係数	標準偏差を平均で割った値

9. 「確定」を押下します。



10. 「保存」を押下します。



11. 集約の設定が適用されます。

5.12.4.4 データの並べ替え

データの左端のアイコンをクリックして移動させる事で、データの並び順を変更する事ができます。



5.12.5 システムポート

システムポートは、フローリンクの接続元であり、コンポーネントのシステム情報(エラー情報や稼働情報)を出力するポートです。 システムポートでは、システム情報に対して、永続化の有無等の設定を行う事ができます。

システムポートの設定変更について

このシステムポートで設定できる設定内容については、エラーの記録期間などに影響するため、必要がない限りは変更しないでください。

5.12.5.1 設定可能なシステム情報

システムポートで出力される、コンポーネントのシステム情報は次の通りです。

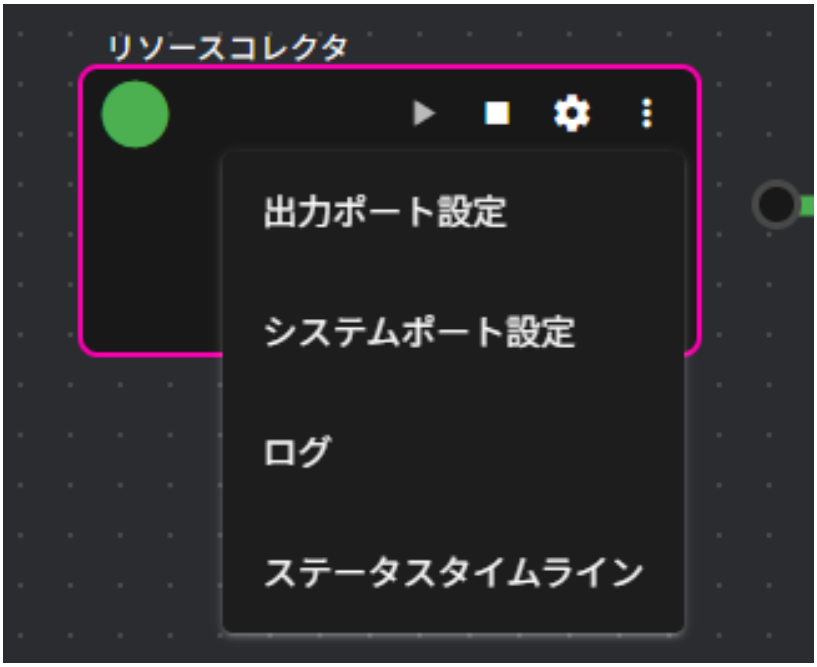
項目	データ型	説明
errors	COMPONENT_STATUS	エラー情報
errors_run_status	UINT8	整数値による稼働状態 0: 停止 1: 実行中（エラーなし） 2: 実行中（エラーあり） 3: エラー停止
total_input_data_count	UINT64	入力データの件数
total_input_data_bytes	UINT64	入力データのバイト数
total_output_data_count	UINT64	出力データの件数
total_output_data_bytes	UINT64	出力データのバイト数

項目	データ型	説明
total_received_count	UINT64	コンポーネントが受信したデータの件数
total_received_bytes	UINT64	コンポーネントが受信したデータのバイト数
total_emission_count	UINT64	コンポーネントが送信したデータの件数
total_emission_bytes	UINT64	コンポーネントが送信したデータのバイト数

5.12.5.2 永続化の設定

永続化を有効にしたデータは、ストレージ内に保存され、保存されている間、SQLからの参照、Grafana連携での参照が可能となります。次の手順で、永続化の設定を行うことができます。

1. 次のいずれかの方法でシステムポートのダイアログを開きます。
   - メニューアイコンを左クリックし、「システムポート設定」を選択



   - コンポーネントのシステムポートを右クリック

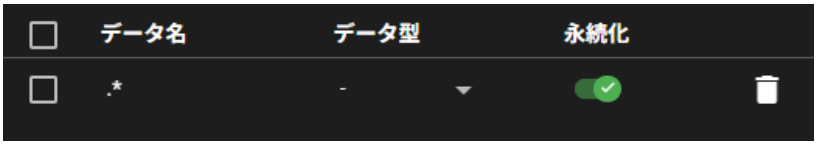


2. 「設定方法」を、次の中から選択します。



項目	説明
データ定義を受信して設定する	システム情報を一覧で表示し、個別に永続化の設定を行う
正規表現で設定する	正規表現を入力し、システム情報の中でマッチしたものに対して、永続化の設定を行う

3. 「設定方法」が「正規表現で設定する」の場合のみ、設定対象とするデータについて、次の項目を入力します。



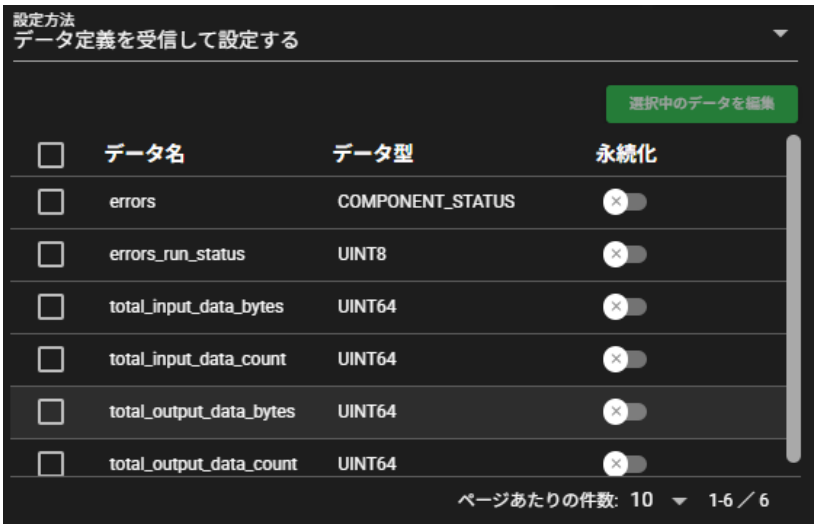
項目	説明
データ名	システム情報のうち、設定対象とするもののデータ名を表す正規表現
データ型	システム情報のうち、設定対象とするもののデータ型

正規表現の入力例

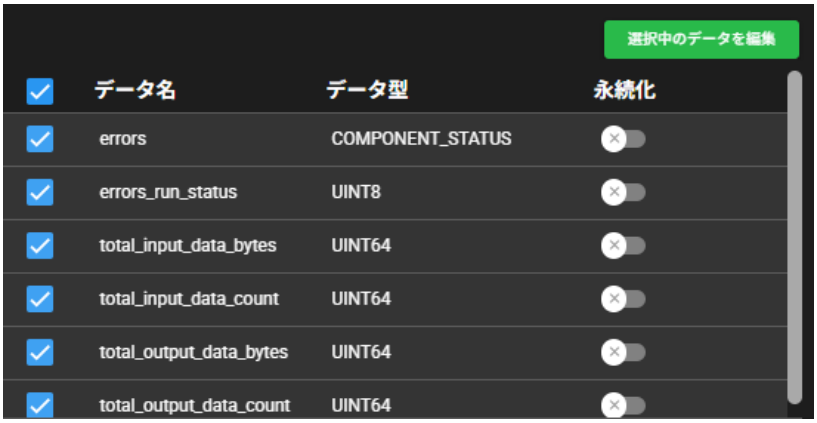
正規表現の入力例は次の通りです。

- 「data1」にのみマッチ： data1
- 名前に「abc」を含むデータにマッチ： .*abc.*
- 名前が「abc」で始まるデータにマッチ： abc.*
- 名前が「abc」で終わるデータにマッチ： .*abc

4. 次のいずれかの方法を行い、永続化の設定ダイアログを開きます。
   - 永続化の設定を行うデータの「永続化」のアイコンを押下します。



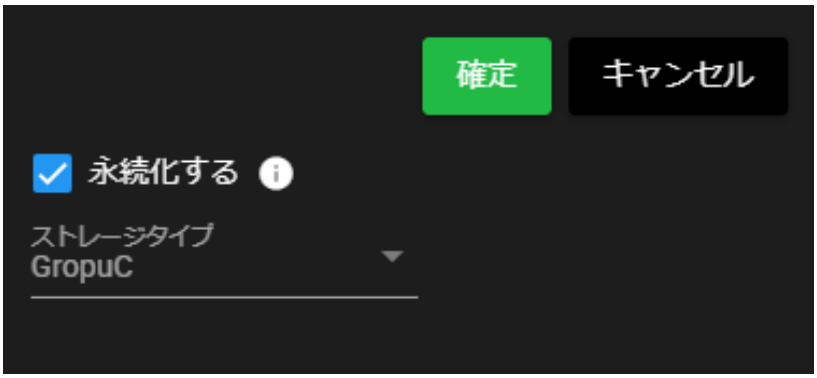
   - 永続化の設定を行うデータにチェックを付けて、「選択中のデータを編集」を押下します。チェックを付けたデータに、同じ設定内容が適用されます。



5. 「永続化する」をONにします。

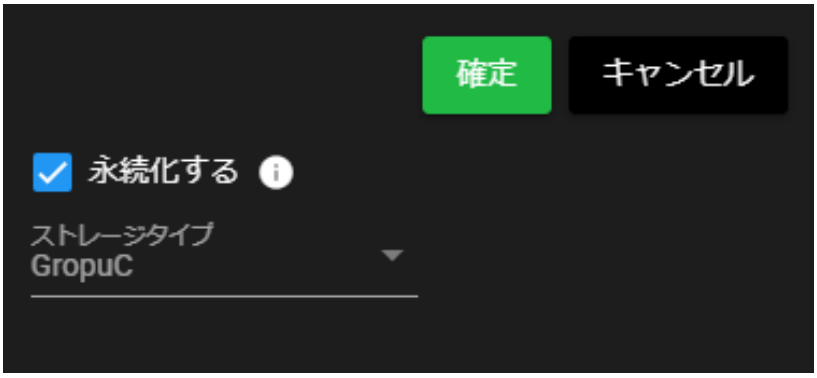


6. 次の項目を入力します。



項目	説明
ストレージタイプ	時系列データベースに保存する期間。 サーバー設定の「グループ単位で保存期間を指定」で、GroupA・GroupB・GroupCの保存期間を確認・変更する事ができます。「グループ単位で保存期間を指定」がONの場合のみ変更できます。

7. 「確定」を押下します。



8. 「保存」を押下します。



9. 永続化の設定が適用されます。

5.13 ステータスタイムライン

ここでは、稼働ステータスの一覧、ステータスタイムラインについて説明します。

5.13.1 ステータス一覧

コンポーネントのステータスランプから、稼働ステータスを確認することができます。



稼働ステータスの種類は、次の通りです。

ランプ	ステータス名	説明
	正常	コンポーネントが正常に稼働している
	警告	問題が発生したが、コンポーネントの稼働は止まっていない
	エラー	問題が発生し、コンポーネントの稼働が止まっている
	停止	コンポーネントが停止している
	停止中	コンポーネントが稼働していた状態から、停止状態に移行している

ランプ	ステータス名	説明
▬	ロード失敗	カスタムコンポーネントが登録されていない、もしくはロード時にエラーとなっており、コンポーネントを利用できない

5.13.2 ステータスタイムライン

コンポーネントの稼働ステータスの遷移履歴を確認することができます。 ステータスが警告・エラーの場合は、エラーコードを確認することができます。

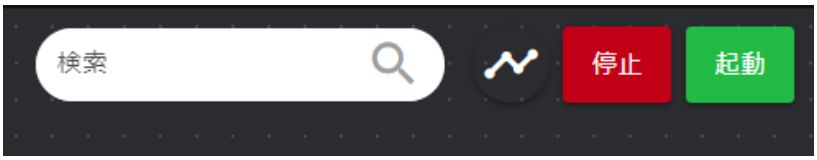
5.13.2.1 閲覧

ここでは、ステータスタイムラインの閲覧方法を説明します。 ステータスタイムラインに全コンポーネントの情報を表示させる方法と、個別コンポーネントの情報を表示させる方法があります。

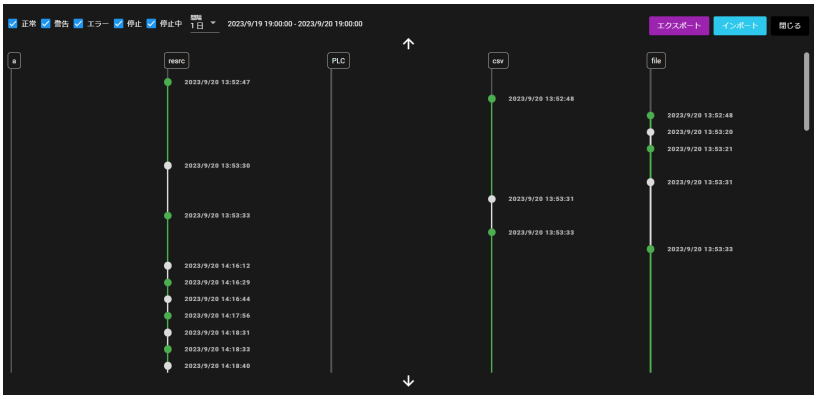
5.13.2.1.1 全コンポーネント

次の手順で、全コンポーネントの情報を表示したステータスタイムラインを表示することができます。

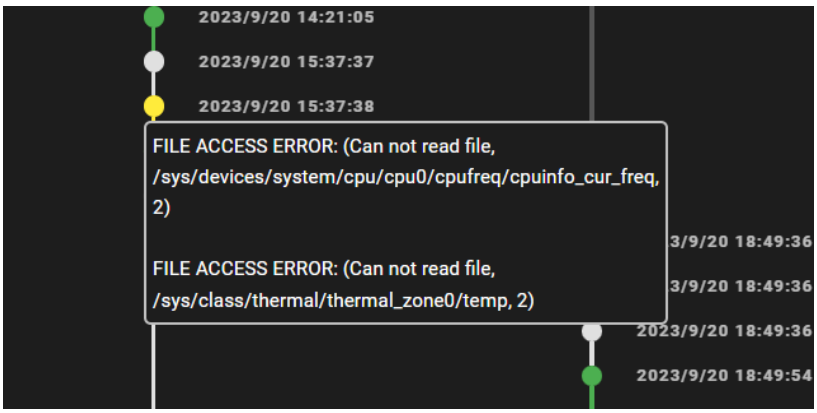
1. ステータスタイムラインのアイコンを押下します。



2. 全コンポーネントのステータスの遷移履歴が表示されます。



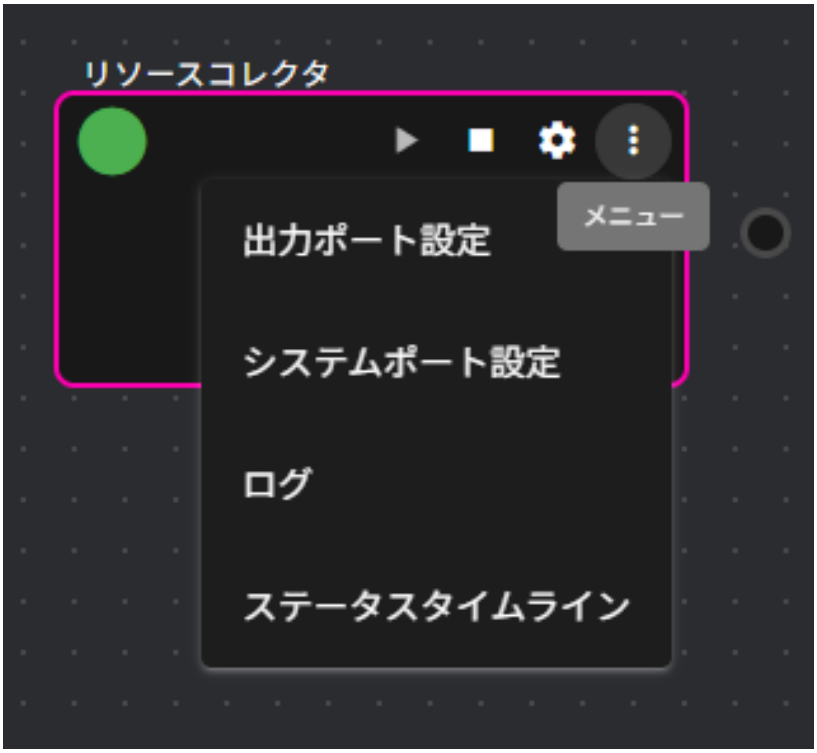
3. 警告・エラーのステータスにマウスを合わせると、エラーコードが表示されます。



5.13.2.1.2 個別コンポーネント

次の手順で、個別コンポーネントの情報を表示したステータスタイムラインを表示することができます。

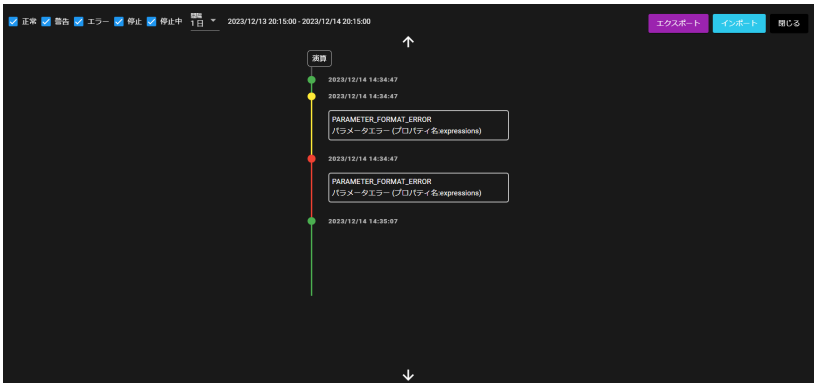
1. 以下のいずれかでステータスタイムラインのダイアログを開きます。
   - メニューアイコンを左クリックして「ステータスタイムライン」を選択



   - コンポーネントを右クリックして「ステータスタイムライン」を選択



2. 個別コンポーネントのステータスの遷移履歴が表示されます。警告・エラーのステータスには、エラーコードが表示されます。



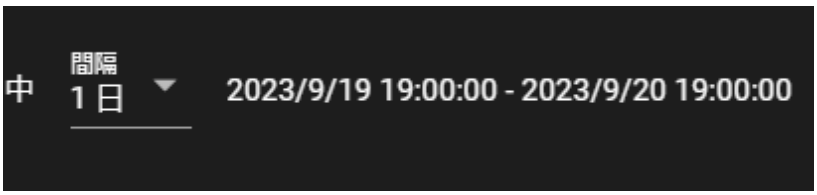
5.13.2.2 絞り込み

ここでは、ステータスタイムラインに表示させる内容を絞り込む方法について説明します。

表示間隔

次の手順で、ステータスタイムラインの表示間隔を変更することができます。

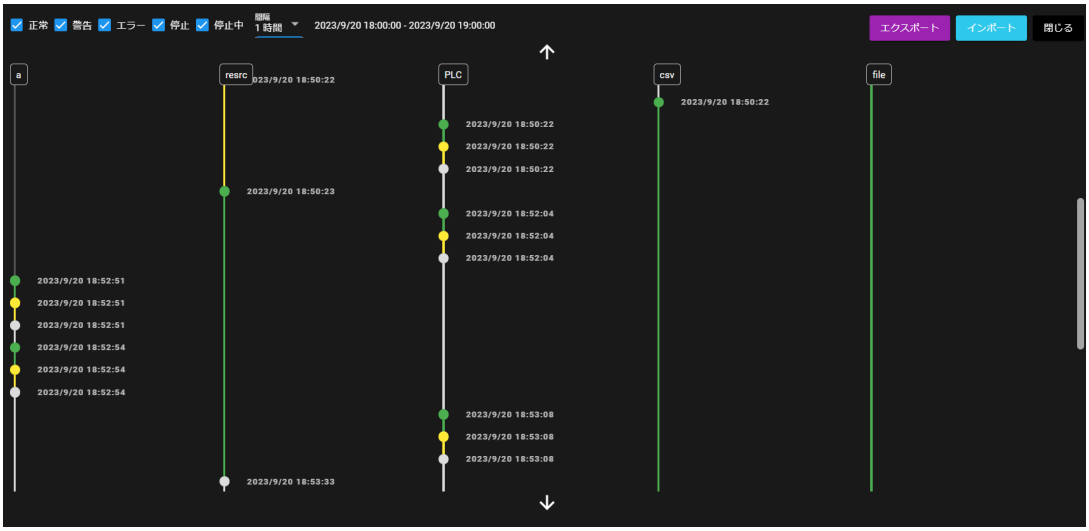
1. 「間隔」を押下します。



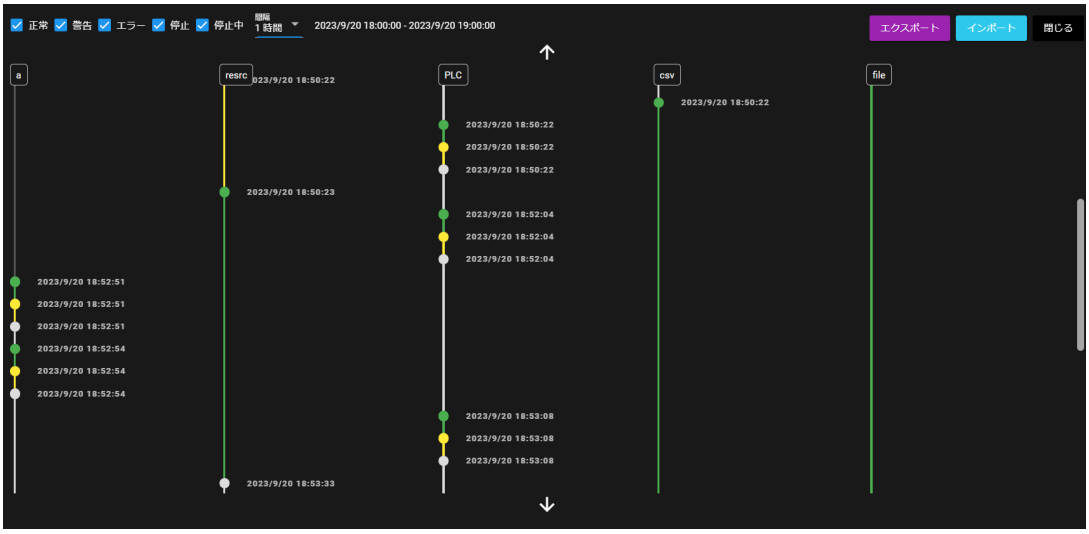
2. 時間単位を選択します。



3. 選択した時間単位の範囲の分だけ、ステータスの遷移履歴が表示されます。



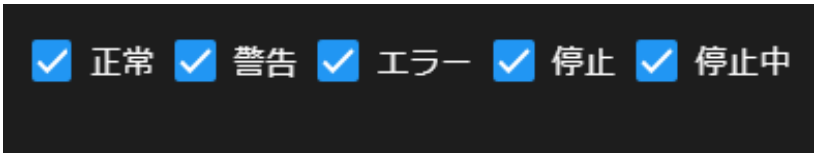
4. 上矢印アイコンを押下すると過去、下矢印アイコンを押下すると未来の履歴を表示します。



ステータス

次の手順で、指定したステータスのみステータスタイムラインに表示させることができます。

1. ステータスにチェックを付けるか、または外します。



2. ステータスの遷移履歴のうち、チェックを付けたものだけが表示されます。

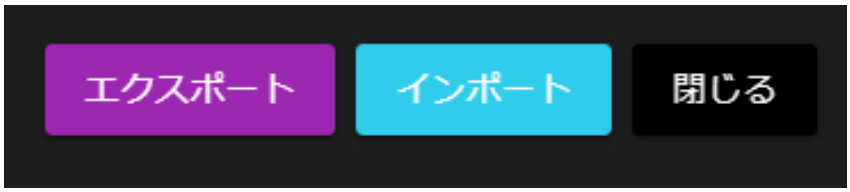


5.13.2.3 エクスポート・インポート

ステータスタイムラインの表示内容を、ファイルとして出力することができます。出力したファイルを別の環境のSpeeDBee Synapseに取り込んで表示させることで、稼働ステータスの情報を共有することが可能になります。

エクスポート

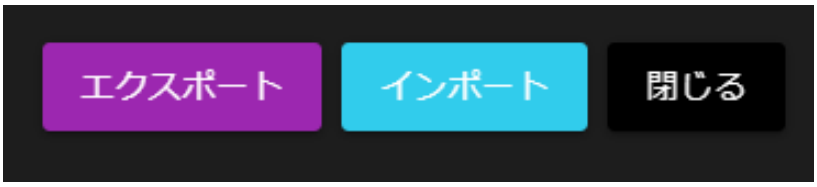
「エクスポート」を押下すると、ステータスタイムラインの表示内容を、ファイルとして出力することができます。



インポート

次の手順で、エクスポートしたファイルから情報を取得し、ステータスタイムラインを表示することができます。

1. 「インポート」を押下します。



2. エクスポートしたファイルを選択します。
3. ファイルから情報を取得し、ステータスタイムラインが表示されます。

5.14 各種ログ

ここでは、SpeeDBee Synapseの各種ログについて説明します。

5.14.1 ログの種別

SpeeDBee Synapseが扱うログの種別は次の通りです。

種別	説明	画面での閲覧	ダウンロード
SpeeDBee Synapseの操作ログ	ユーザーなどによるSpeeDBee Synapseの操作履歴が記録されます。 ※詳細は「操作ログ」をご覧ください。	○	○
WEBUIの運用ログ	WEBUIの運用情報が記録されます。	×	○
コア、コンポーネント、データベースの運用ログ	コア、コンポーネント、データベースの運用情報が記録されます。	×	○
コンポーネント個別の運用ログ	コンポーネントの運用情報が記録されます。	○	○

5.14.2 操作ログ

ユーザーがSpeeDBee Synapse上で行った操作はすべて、操作ログに記録されます。操作ログには、SpeeDBee Synapseや外部APIによって行われた操作も記録されます。

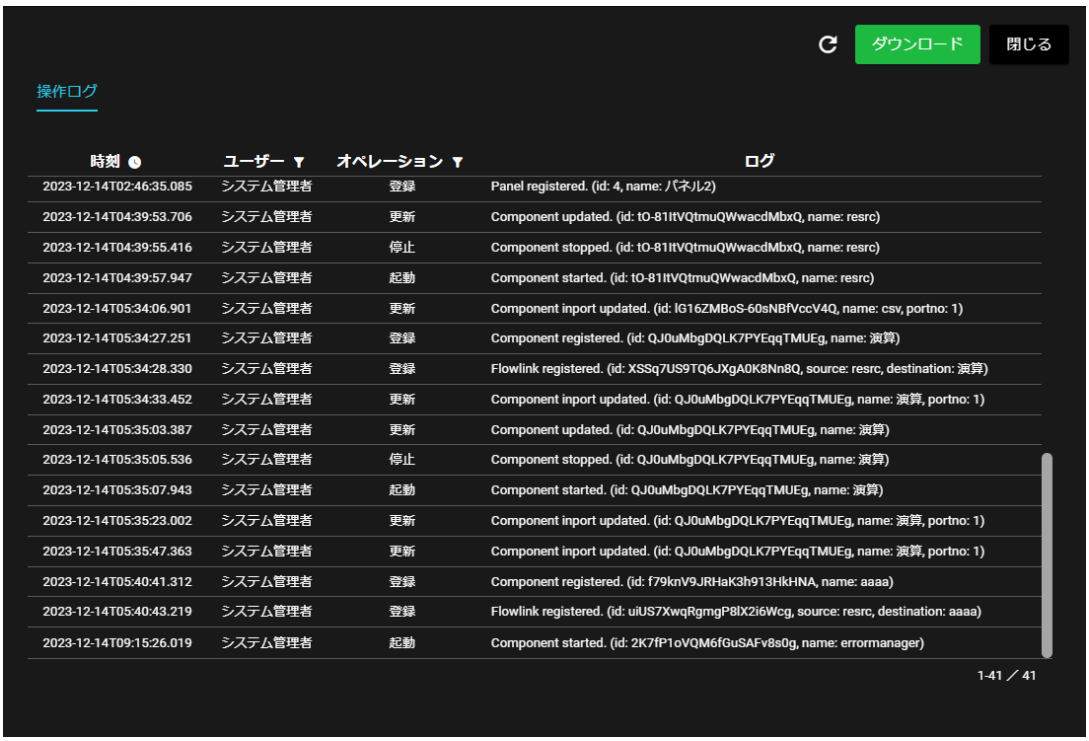
5.14.2.1 閲覧

次の手順で、操作ログを閲覧することができます。

1. ヘッダのログアイコンを押下します。



2. 操作ログが表示されます。



操作ログに表示される内容は、次の通りです。

項目	説明
時刻	操作を行った時刻
ユーザー	操作を行ったユーザー ※1 "システム"はSpeeDBee Synapseによって行われた操作を表します ※2 "外部API"は外部APIによって行われた操作を表します。
オペレーション	操作の種類
ログ	操作の内容

5.14.2.2 リフレッシュ

リフレッシュボタンを押下すると、最新のログを取得できます。



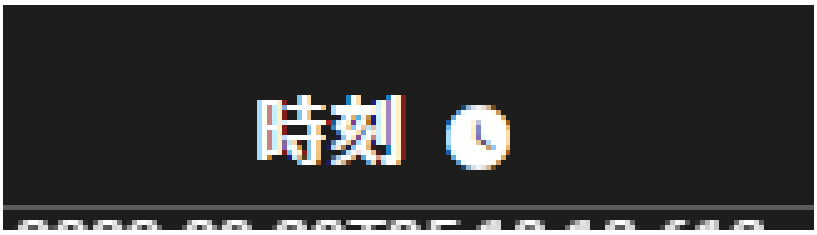
5.14.2.3 絞り込み

時刻、ユーザー、操作の種類により、閲覧する操作ログを絞り込むことができます。

時刻

次の手順で、指定した時間範囲の操作ログのみを閲覧することができます。

1. 「時刻」の横のアイコンを押下します。



2. 時間範囲を入力し、「確定」を押下します。



3. 入力した時間範囲のログが表示されます。



時刻	ユーザー	オペレーション	ログ
2023-09-20T04:52:05.421	システム管理者	登録	Component registered. (id: 6395813463879043, name: csv)
2023-09-20T04:52:07.265	システム管理者	登録	Flowlink registered. (id: 6647666766917860, source: resrc, destination: csv)
2023-09-20T04:52:16.121	システム管理者	登録	Component registered. (id: 4670249239335872, name: file)
2023-09-20T04:52:18.375	システム管理者	登録	Flowlink registered. (id: 2846928861372731, source: csv, destination: file)
2023-09-20T04:52:41.618	システム管理者	更新	license setting updated. (key: ... -> U7QJP-4PDYQ-...)
2023-09-20T04:52:48.606	システム管理者	起動	Component started. (id: 2161262164829785, name: resrc)
2023-09-20T04:52:48.609	システム管理者	起動	Component started. (id: 6395813463879043, name: csv)
2023-09-20T04:52:48.613	システム管理者	起動	Component started. (id: 4670249239335872, name: file)
2023-09-20T04:53:20.288	システム管理者	停止	Component stopped. (id: 4670249239335872, name: file)
2023-09-20T04:53:21.995	システム管理者	起動	Component started. (id: 4670249239335872, name: file)
2023-09-20T04:53:31.257	システム管理者	停止	Component stopped. (id: 2161262164829785, name: resrc)
2023-09-20T04:53:31.260	システム管理者	停止	Component stopped. (id: 6395813463879043, name: csv)
2023-09-20T04:53:31.262	システム管理者	停止	Component stopped. (id: 4670249239335872, name: file)
2023-09-20T04:53:33.329	システム管理者	起動	Component started. (id: 2161262164829785, name: resrc)
2023-09-20T04:53:33.331	システム管理者	起動	Component started. (id: 6395813463879043, name: csv)
2023-09-20T04:53:33.332	システム管理者	起動	Component started. (id: 4670249239335872, name: file)

1-19 / 19

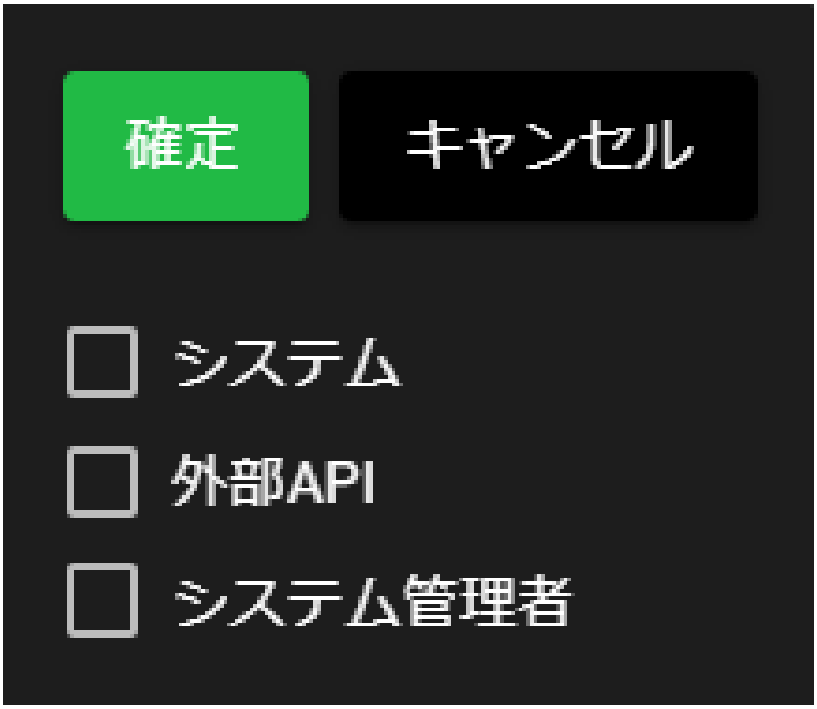
ユーザー

次の手順で、指定したユーザーの操作ログのみを閲覧することができます。

1. 「ユーザー」の横のアイコンを押下します。



2. ユーザーを選択し、「確定」を押下します。



3. 選択したユーザーが行った操作のログが表示されます。

時刻	ユーザー	オペレーション	ログ
2023-09-20T06:23:20.199	システム管理者	登録	Component registered. (id: 1454937385528078, name: aaaa)
2023-09-20T06:23:20.385	システム管理者	登録	Flowlink registered. (id: 4493995553124316, source: resrc, destination: aaaa)
2023-09-20T06:35:58.379	システム管理者	削除	Component deleted. (id: 1454937385528078, name: aaaa)
2023-09-20T06:37:36.406	システム管理者	更新	Component updated. (id: 2161262164829785, name: resrc, baseid: a6706d88-a3af-4c4f-b3aa-c164abb008d8 -> a6706d88-a3af-4c4f-b3aa-c164abb008d8)
2023-09-20T06:37:37.929	システム管理者	停止	Component stopped. (id: 2161262164829785, name: resrc)
2023-09-20T06:37:38.973	システム管理者	起動	Component started. (id: 2161262164829785, name: resrc)
2023-09-20T06:38:19.145	システム管理者	登録	Component registered. (id: 8658106277169131, name: aaaa)
2023-09-20T06:38:19.235	システム管理者	登録	Flowlink registered. (id: 3243777045516557, source: resrc, destination: aaaa)
2023-09-20T06:38:19.269	システム管理者	登録	Flowlink registered. (id: 3597906071617880, source: csv, destination: aaaa)
2023-09-20T06:38:26.446	システム管理者	削除	Flowlink deleted. (id: 3597906071617880, source: csv, destination: aaaa)
2023-09-20T06:42:09.537	システム管理者	削除	Component deleted. (id: 8658106277169131, name: aaaa)
2023-09-20T07:18:44.612	システム管理者	登録	Component registered. (id: 3131939484492041, name: fft)
2023-09-20T07:18:44.751	システム管理者	登録	Flowlink registered. (id: 9200368006240683, source: resrc, destination: fft)
2023-09-20T08:40:40.354	システム管理者	登録	User registered. (id: 2, name: user1)
2023-09-20T08:40:55.803	システム管理者	登録	User registered. (id: 3, name: user2)
2023-09-20T08:41:11.471	システム管理者	登録	User registered. (id: 4, name: user3)

1-47 ／ 47

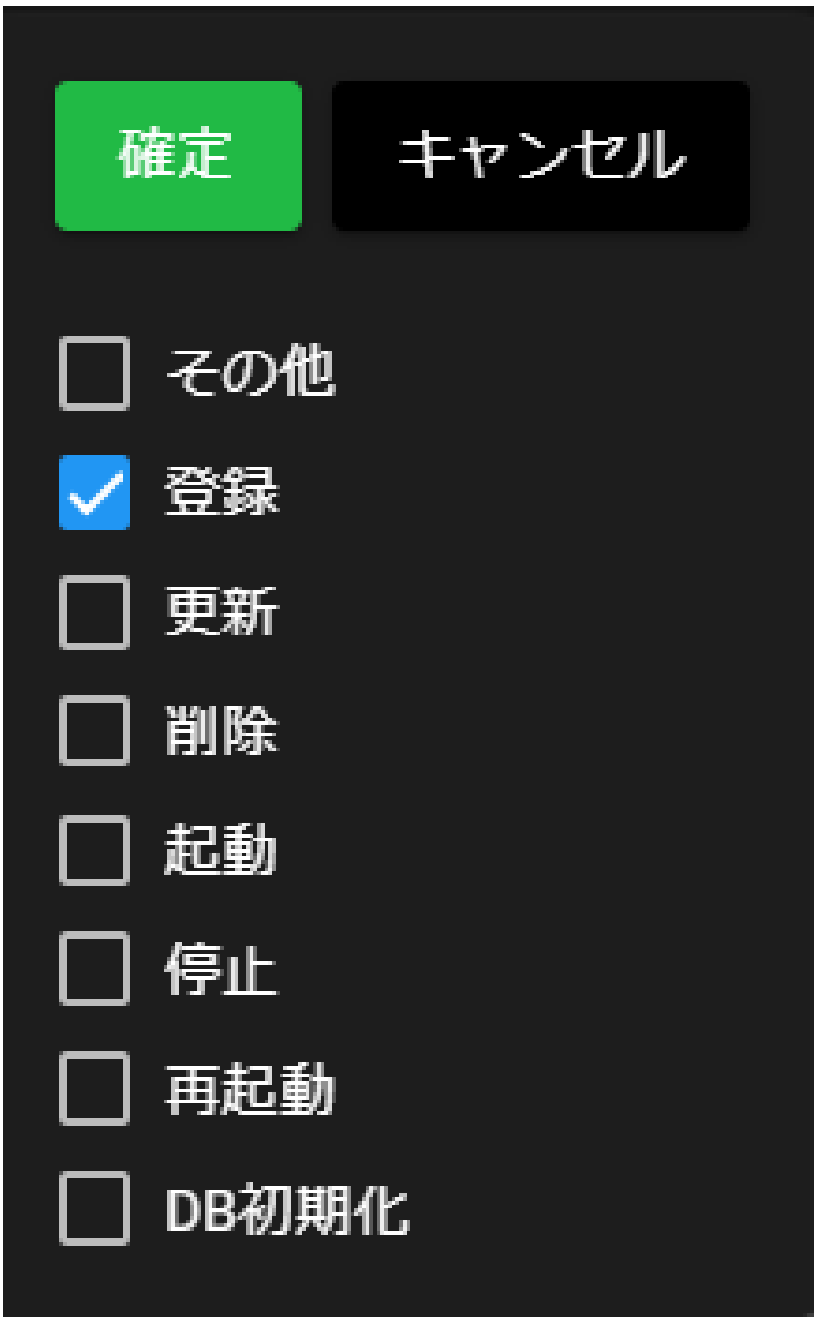
オペレーション

次の手順で、指定した種類の操作のログのみを閲覧することができます。

1. 「オペレーション」の横のアイコンを押下します。



2. 操作ログの種類を選択し、「確定」を押下します。



3. 選択した種類の操作ログが表示されます。



時刻	ユーザー	オペレーション	ログ
2023-09-20T04:52:05.421	システム管理者	登録	Component registered. (id: 6395813463879043, name: csv)
2023-09-20T04:52:07.265	システム管理者	登録	Flowlink registered. (id: 6647666766917860, source: resrc, destination: csv)
2023-09-20T04:52:16.121	システム管理者	登録	Component registered. (id: 4670249239335872, name: file)
2023-09-20T04:52:18.375	システム管理者	登録	Flowlink registered. (id: 2846928861372731, source: csv, destination: file)
2023-09-20T04:59:30.207	システム管理者	登録	Panel registered. (id: 4, name: パネル2)
2023-09-20T04:59:40.234	システム管理者	登録	Component registered. (id: 0381571032388029, name: resrc1)
2023-09-20T06:23:20.199	システム管理者	登録	Component registered. (id: 1454937385528078, name: aaaa)
2023-09-20T06:23:20.385	システム管理者	登録	Flowlink registered. (id: 4493995553124316, source: resrc, destination: aaaa)
2023-09-20T06:38:19.145	システム管理者	登録	Component registered. (id: 8658106277169131, name: aaaa)
2023-09-20T06:38:19.235	システム管理者	登録	Flowlink registered. (id: 3243777045516557, source: resrc, destination: aaaa)
2023-09-20T06:38:19.269	システム管理者	登録	Flowlink registered. (id: 3597906071617880, source: csv, destination: aaaa)
2023-09-20T07:18:44.612	システム管理者	登録	Component registered. (id: 3131939484492041, name: fft)
2023-09-20T07:18:44.751	システム管理者	登録	Flowlink registered. (id: 9200368006240683, source: resrc, destination: fft)
2023-09-20T08:40:40.354	システム管理者	登録	User registered. (id: 2, name: user1)
2023-09-20T08:40:55.803	システム管理者	登録	User registered. (id: 3, name: user2)
2023-09-20T08:41:11.471	システム管理者	登録	User registered. (id: 4, name: user3)

1-18 ／ 18

5.14.3 コンポーネント個別ログ

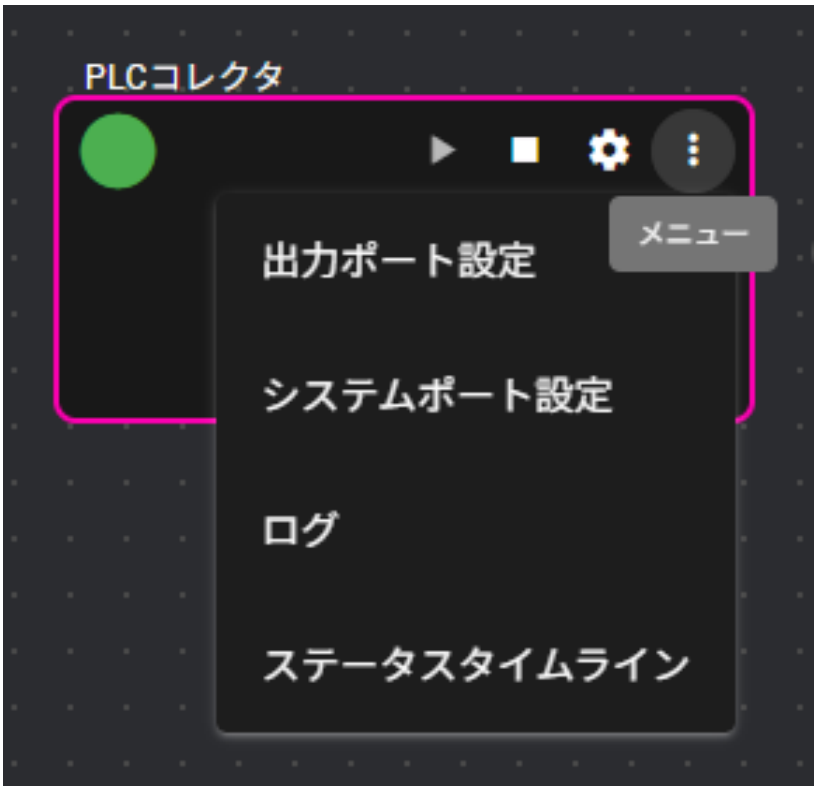
コンポーネントによって出力されたログが記録されます。

また本機能のログ閲覧画面からコンポーネント個別でログ出力のレベルを設定することが可能です。
設定方法は「個別ログレベル設定」をご覧ください。

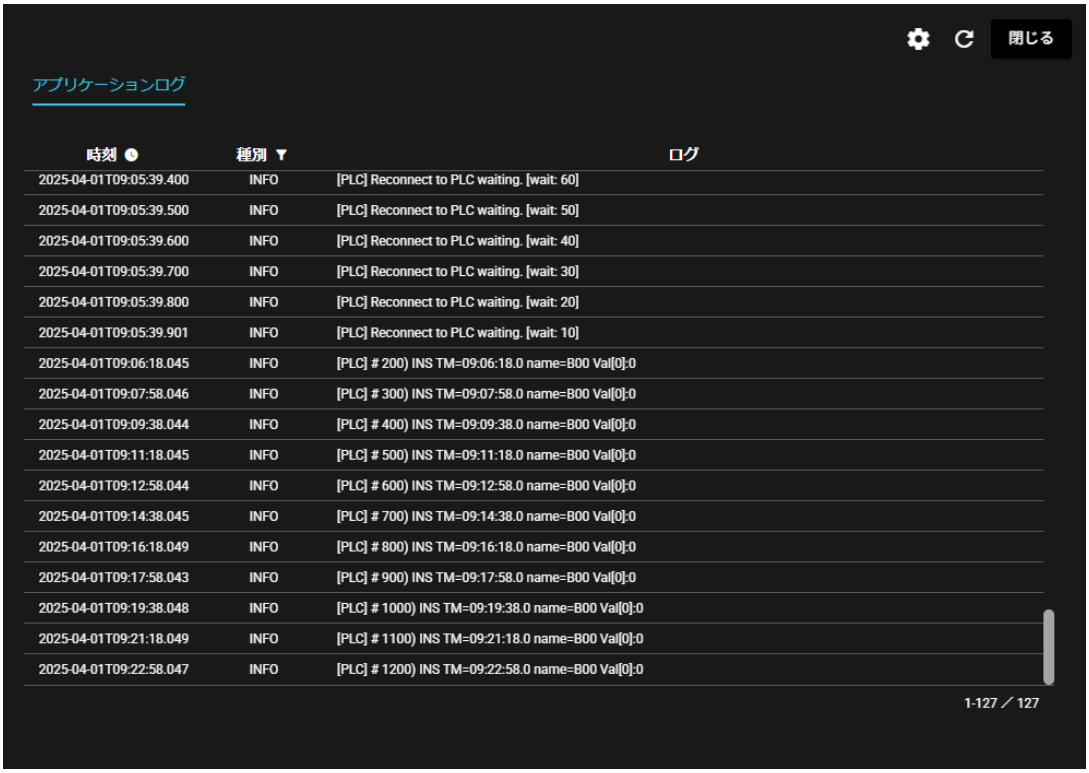
5.14.3.1 閲覧

次の手順でコンポーネント個別ログを閲覧することができます。

1. メニューアイコンを左クリックして「ログ」を選択。



2. コンポーネント個別ログが表示されます。

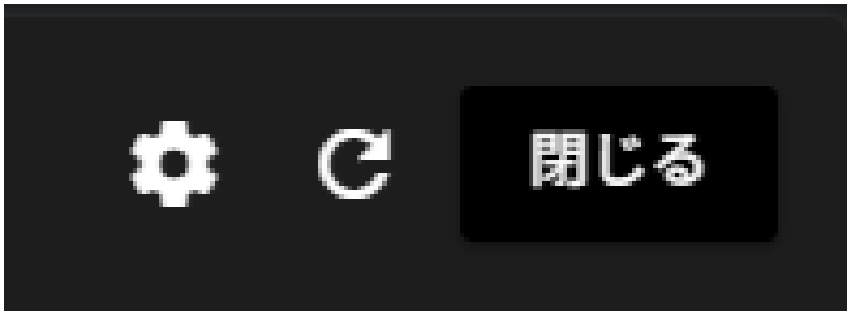


コンポーネント個別ログに表示される内容は、次の通りです。

項目	説明
時刻	コンポーネントによりログが出力された時刻
種別	ログ出力のレベル
ログ	ログの内容

5.14.3.2 リフレッシュ

リフレッシュボタンを押下すると、最新のログを取得できます。



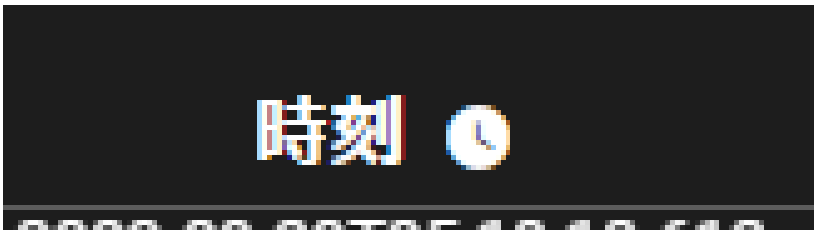
5.14.3.3 絞り込み

時刻、種別により、閲覧するコンポーネント個別ログを絞り込むことができます。

時刻

次の手順で、指定した時間範囲のコンポーネント個別ログのみを閲覧することができます。

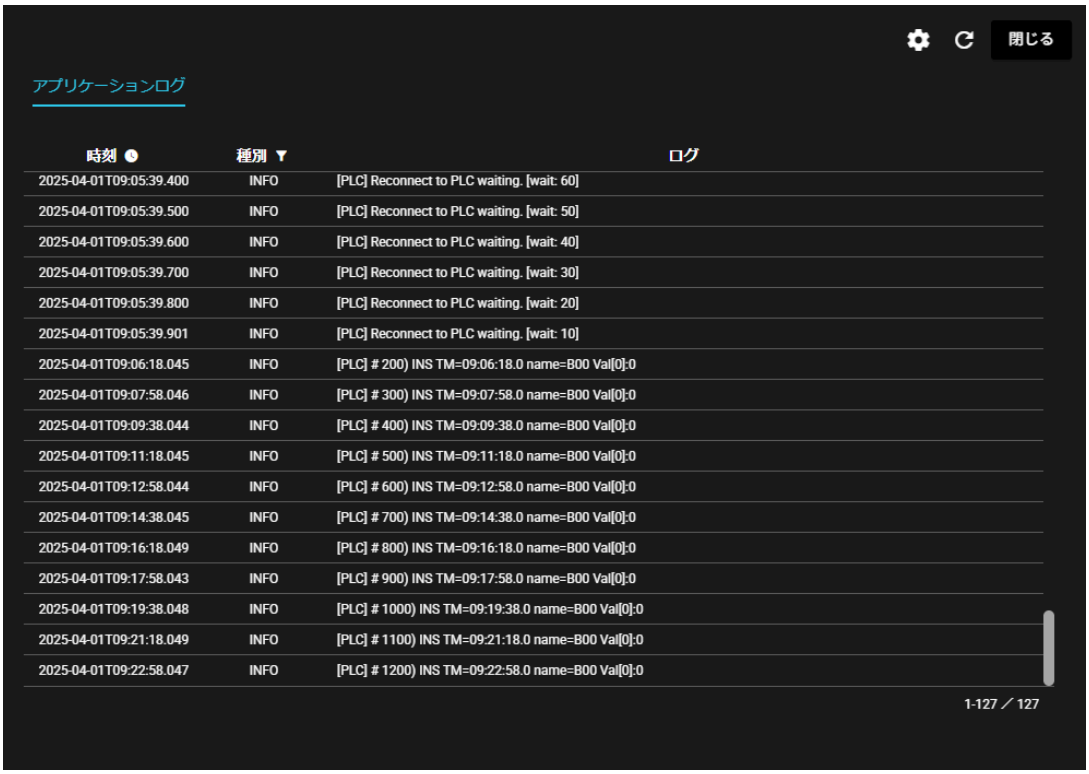
1. 「時刻」の横のアイコンを押下します。



2. 時間範囲を入力し、「確定」を押下します。



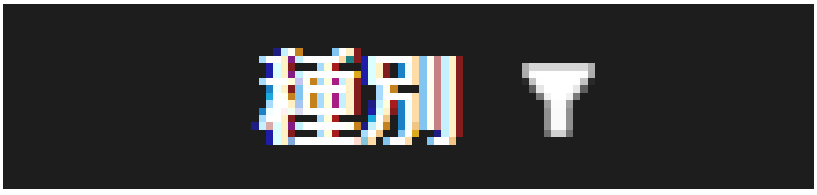
3. 入力した時間範囲のログが表示されます。



種別

次の手順で、指定した種別（ログレベル）のコンポーネント個別ログのみを閲覧することができます。

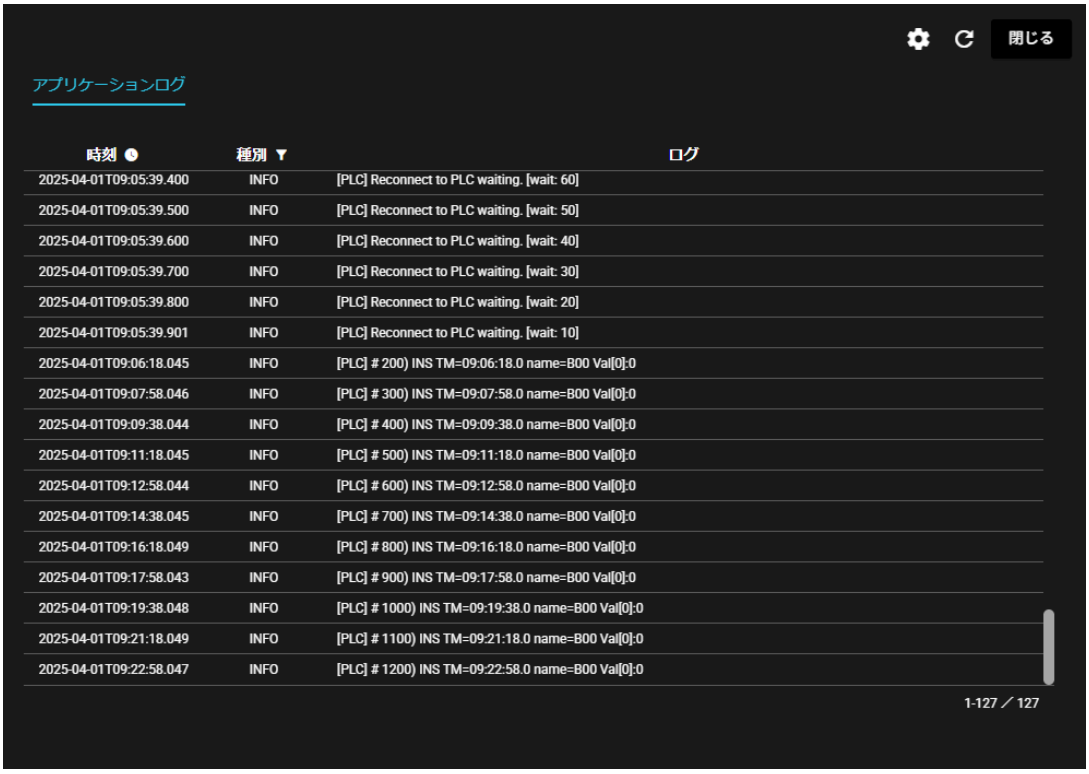
1. 「種別」の横のアイコンを押下します。



2. 種別（ログレベル）を選択し、「確定」を押下します。



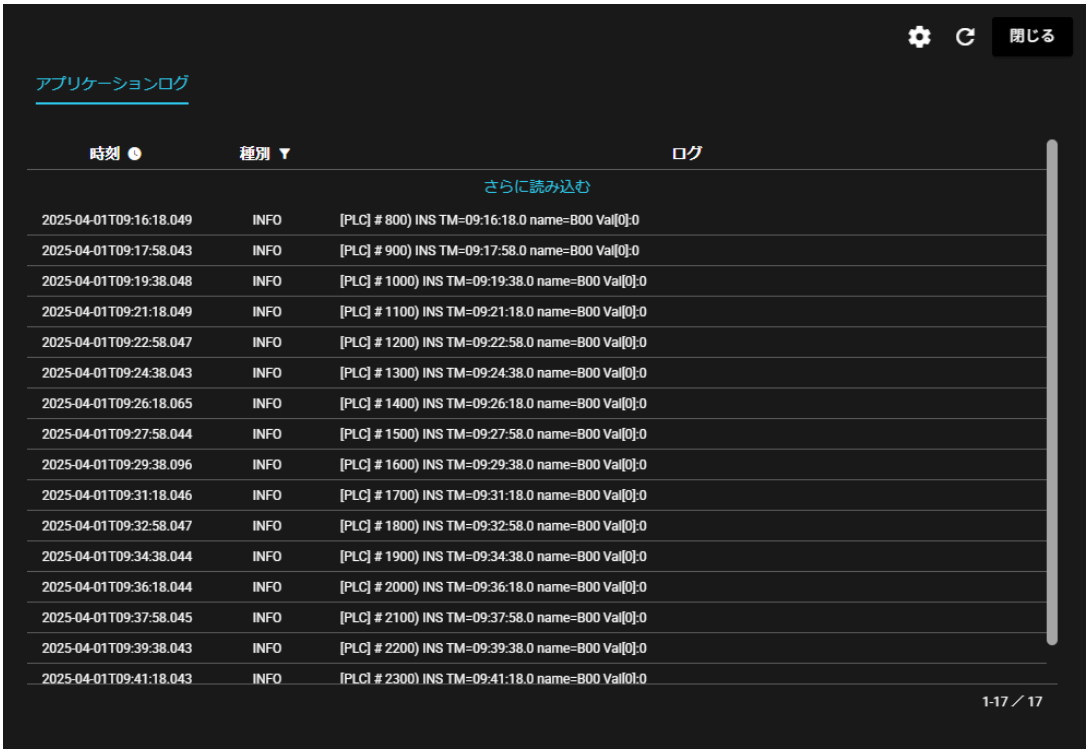
3. 選択した種別（ログレベル）のコンポーネント個別ログが表示されます。



5.14.3.4 さらに読み込む

次の手順で、現在表示されているコンポーネント個別ログよりさらに過去のログを追加で表示することができます。

1. コンポーネント個別ログを一番上までスクロールすると「さらに読み込む」が表示されるので押下します。



※すでに一番過去のログまで表示している場合、「さらに読み込む」は表示されません。

5.14.3.5 個別ログレベル設定

コンポーネント個別でログ出力のレベルを設定することができます。

この設定を有効にした場合には「全般的な設定」の「ログ設定」で設定されたログレベルに影響されず、 個別で設定したログレベル未満のログは出力されなくなります。

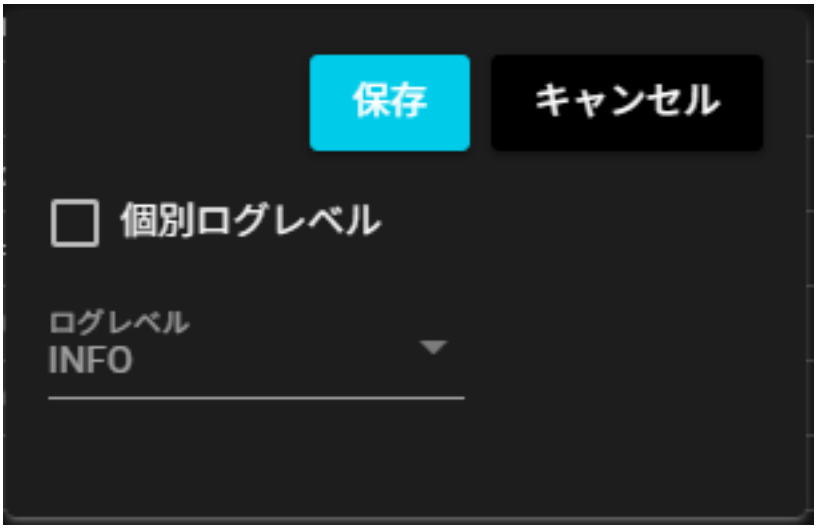
この設定を無効とした場合には「全般的な設定」の「ログ設定」で設定されたログレベルに従いログが出力されます。

次の手順で、現在表示されているコンポーネント個別のログレベルを設定することができます。

1. 設定アイコンを押下します。



2. 個別ログレベル設定が表示されます。



個別ログレベル設定に表示される内容は、次の通りです。

項目	説明
保存	設定内容を保存
キャンセル	設定内容を破棄
個別ログレベル	個別ログレベルの有効/無効
ログレベル	ログ出力のレベル

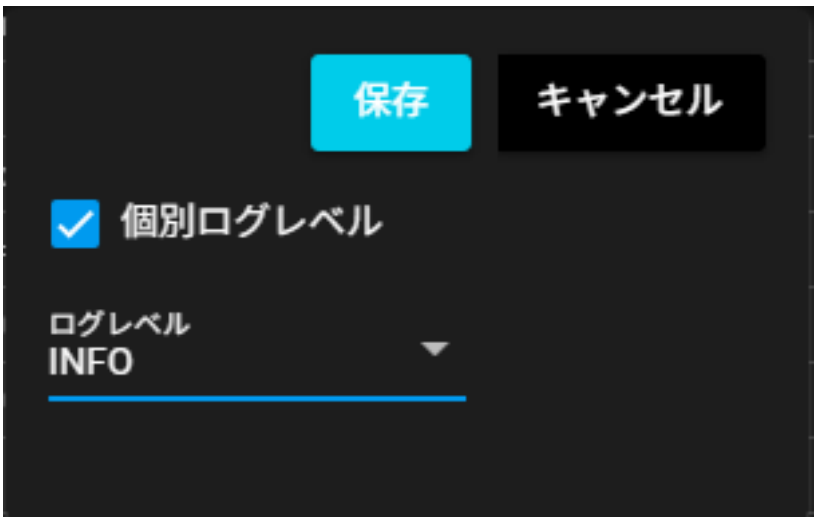
3. 「個別ログレベル」を有効にします。



4. 「ログレベル」を選択します。



5. 「保存」を押下します。



設定を変更しない場合は「キャンセル」を押下してください。

この操作はEdit Modeでのみ行うことができます。

5.14.4 ダウンロード

次の手順で、SpeeDBee Synapseの各種ログをダウンロードすることができます。 コンポーネント個別ログもダウンロード対象に含まれますが、全てのコンポーネントやSpeeDBee Synapseの稼働ログも含まれたファイルになります。

1. ダウンロードボタンを押下します。



2. ZIPファイルがダウンロードされます。ZIPファイルは下記の構成でログファイルを格納しています。

ZIPファイル構成

```
├─webui
│  ├─operation_log.YYYYMM.csv（SpeeDBee Synapseの操作ログ）
│  └─speedbeesynapse-webui-backend.log（WEBUIの運用ログ）
├─ speedbeesynapse.log（コア、コンポーネント、データベースの運用ログ）
└─ speedbeesynapse-YYYYMMDDHHMMSS.log（speedbeesynapse.logの古いバージョン）
```

5.15 DBクエリ（SQL）

ここでは、DBクエリ（SQL）について説明します。

5.15.1 DBクエリの概要

DBクエリサービスを起動すると、時系列データベースに対して Web-API（HTTP）経由でSQL（Structured Query Language）による問い合わせが可能です。 構文は、SpeeDBee Synapse独自のものですが、一般的なRDBで使用されるSQLと近い構文仕様となっています。

5.15.2 DBクエリの詳細

ここでは、Web-API仕様および構文について説明します。

5.15.2.1 Web-API仕様

DBクエリサービスのWeb-API仕様は、こちらをご覧ください。

5.15.2.2 構文仕様

5.15.2.2.1 SELECT

SELECT句は、時系列データベース内に登録されているデータを参照する構文です。

1. 例文：リソースコレクタr1のcpu_usageを現在時刻までを範囲として、5件取得する

```
SELECT "r1:cpu_usage" FROM SPDB WHERE _ts < NOW LIMIT 5;
```

2. 例文：リソースコレクタr1のcpu_usageを現在時刻までを範囲とし、かつcpu_usageが10以上のデータを取得する

```
SELECT "r1:cpu_usage" FROM SPDB WHERE _TS < NOW AND "r1:cpu_usage" >= 10
```

3. 例文：PLCコレクタplc1のdataAの最大値と最小値について、分析の時間幅5秒で取得する

```
SELECT MAX("plc1:dataA"),MIN("plc1:dataA") FROM SPDB WHERE _ts < NOW GROUP BY TUMBLING(5,sec)
LIMIT 10;
```

構文仕様

```
SELECT <select_list> FROM <table> [WHERE <condition>] [GROUP BY <window_condition> [HAVING
<condition>]] [ORDER BY _TS [ASC/DESC]] [LIMIT <number> [OFFSET <start>]|LIMIT <start>,<row_count>]
```

予約語については、大文字、小文字を区別しません。

- <select_list>

抽出するカラムや式を指定します。指定はカンマ区切りで複数の指定が可能です。 なお、SpeeDBee Synapseで使用するカラムは、必ずダブルクオーテーション（"）でカラム名を囲んで指定して下さい。 またダブルクォーテーションで囲まれたカラム名は大文字・小文字を区別します。

例1：カラム名

```
"plcA:data1", "computeB:DATAa"
```

例2：整数、浮動小数、文字列

```
3, -5, 1.52, 0.32e2, 'TEST-STR'
```

SQL内の整数はint64として扱います。浮動小数はdoubleとして扱い、指数表現も可能です。 文字列は、シングルクォーテーションで囲みます。ダブルクォーテーションは、カラム名などの識別子として扱われます。

例3：式

```
"plcA:data2" + 3
```

使用可能な演算子	説明
+	加算
-	減算

使用可能な演算子	説明
*	乗算
/	除算
%	余り

例4：関数

```
MAX("compute1:dataA"), AVG("compute2:dataB")
```

使用可能な数学関数	戻り値	説明
ABS	INT64	絶対値 (整数)
ACOS	DOUBLE	三角関数 逆余弦
ASIN	DOUBLE	三角関数 逆正弦
ATAN	DOUBLE	三角関数 逆正接
CEIL	DOUBLE	小数点以下切り上げ
COS	DOUBLE	三角関数 余弦
COSH	DOUBLE	双曲線余弦
EXP	DOUBLE	底がネイピア数 e である指数関数
FABS	DOUBLE	絶対値 （倍精度浮動小数）
FLOOR	DOUBLE	小数点以下切り捨て
FMOD	DOUBLE	剰余
LOG	DOUBLE	自然対数
LOG10	DOUBLE	10を底とする対数
POW	DOUBLE	べき乗

使用可能な数学関数	戻り値	説明
ROUND	DOUBLE	小数点以下四捨五入
SIN	DOUBLE	三角関数 正弦
SINH	DOUBLE	双曲線正弦
SQRT	DOUBLE	平方根
TAN	DOUBLE	三角関数 正接
TANH	DOUBLE	双曲線正接

使用可能な集約・統計関数	戻り値	説明
AVEDEV	DOUBLE	平均偏差
AVG	DOUBLE	算術平均
COUNT	INT64	件数
CV	DOUBLE	変動係数
DOF	INT64	自由度
MAX	DOUBLE	最大
MIN	DOUBLE	最小
SE	DOUBLE	標準誤差
STDEV	DOUBLE	不偏標準偏差
STDEVP	DOUBLE	標準偏差
SUM	DOUBLE	合計
SUMSQ	DOUBLE	2乗和

使用可能な集約・統計関数	戻り値	説明
VAR	DOUBLE	不偏分散
VARP	DOUBLE	標本分散

使用可能な代表値	戻り値	説明
LAST	値のデータ型	ウィンドウ最終値
TOP	値のデータ型	ウィンドウ先頭値

例5：定数

```
M_PI
```

使用可能な定数	戻り値	説明
M_E	DOUBLE	ネイピア数、自然底数の底（e）
M_LOG2E	DOUBLE	log_e(e)
M_LOG10E	DOUBLE	log_10(e)
M_LN2	DOUBLE	log_e(2)
M_LN10	DOUBLE	log_e(10)
M_PI	DOUBLE	円周率（Pi）
M_PI_2	DOUBLE	Pi/2
M_PI_4	DOUBLE	Pi/4
M_1_PI	DOUBLE	1/Pi
M_2_SQRTPI	DOUBLE	2/Pi
M_SQRT2	DOUBLE	sqrt(2)

使用可能な定数	戻り値	説明
M_SQRT1_2	DOUBLE	1/sqrt(2)

例6：疑似列

```
_WID, _TS
```

使用可能な疑似列	説明
_WID	ウィンドウID（GROUP BY使用時の分析幅単位で変化するID）
_TS	タイムスタンプ（ナノ秒）

- <table>

時系列データの検索対象を指定します。

メモリとファイルの両方を検索する場合

```
SPDB
```

メモリのみ検索する場合

```
SPDB.MEM
```

ファイルのみ検索する場合

```
SPDB.FILE
```

- <condition>

条件式を指定します。

例1：メモリ使用率が50未満　または　CPU使用率が30以下

```
"r1:mem_usage" < 50 OR "r1:cpu_usage" <= 30
```

使用可能な比較演算子	説明
=	等価である
!=	不一致

使用可能な比較演算子	説明
<>	不一致
<	未満
>	超過
<=	以下
>=	以上
IS [NOT] NULL	NULLである、NULLでない
BETWEEN AND	範囲内
BETWEEN EXCLUSIVE AND EXCLUSIVE	指定範囲を含めない

使用可能な論理演算子	説明
AND	論理積
OR	論理和
NOT	否定

_TSに指定可能な演算子	説明
_TS = 100000000	指定したナノ秒と等価
_TS > 100000000	指定したナノ秒を超過
_TS >= 100000000	指定したナノ秒以上
_TS < 100000000	指定したナノ秒未満
_TS <= 100000000	指定したナノ秒以下
_TS BETWEEN 1000000000 AND 9000000000	指定したナノ秒の範囲

_TSに指定可能な演算子	説明
_TS BETWEEN 1000000000 EXCLUSIVE AND 9000000000 EXCLUSIVE	指定したナノ秒を除く範囲
_TS > '20230808T01:23:45.111222333'	ISO8601フォーマットでの文字列指定での比較
_TS < NOW	現在時刻との比較
_TS < NOW + 1 hour	現在時刻+1時間までを範囲とする
_TS < NOW + 10 minutes	現在時刻から+10分までを範囲とする
_TS < NOW + 5 sec	現在時刻から+5秒までを範囲とする
_TS < NOW - 100 msec	現在時刻から-100ミリ秒までを範囲とする
_TS < NOW - 100 usec	現在時刻から-100マイクロ秒までを範囲とする
_TS < NOW - 100 nsec	現在時刻から-100ナノ秒までを範囲とする

※_TSは左辺にのみ使用できます。

また、時間条件は１つのSELECT句で複数指定することはできません。

他の条件と使用する場合は、ANDで接続します。OR接続はできません。

- <window_condition>

ウィンドウ分析の条件を指定します。ウィンドウは2種類あり、時間幅で分析するウィンドウを区切るタンブリングウィンドウと値が変化したことでウィンドウを区切るバリューチェンジウィンドウが使用できます。

例1：タンブリングウィンドウ（5秒で分析ウィンドウを分割）

```
GROUP BY TUMBLING(5,sec)
```

例2：バリューチェンジウィンドウ（plcA:pressureが変化したら分析ウィンドウを区切る）

```
GROUP BY VCHG("plcA:pressure")
```

例3：バリューチェンジウィンドウ（plcA:pressureが変化、または30秒経過で分析ウィンドウを区切る）

```
GROUP BY VCHG("plcA:pressure", 30, sec)
```

例4：タンブリングウィンドウ＋HAVING（5秒で分析ウィンドウを分割、plcA:pressureの最大値が10以上のものを抽出）

```
GROUP BY TUMBLING(10,sec) HAVING MAX("plcA:pressure") >= 10
```

例5：バリューチェンジウィンドウ+HAVING（plcA:pressureが変化、または30秒経過で分析ウィンドウを区切り、plcA:pressureの最小値が5未満のものを抽出）

```
GROUP BY VCHG("plcA:pressure", 30, sec) HAVING MIN("plcA:pressure") < 5
```

- <row_count>

出力行数を整数で指定します。

取得するデータ件数を3件とする場合

```
LIMIT 3
```

- <start>

開始位置を飛ばす行数を整数で指定します。

先頭から5件目以降の3件を取得する場合

```
LIMIT 5,3
```

- ORDER BYについて

ORDER BYは_TSのみ指定可能です。ORDER BYの指定がない場合、ORDER BY _TS ASCによる昇順取得が指定されたことになります。

降順指定

```
ORDER BY _TS DESC
```

- 配列型のカラムに対する指定について

例1：com1:aが配列型の場合、すべての配列要素を参照します。

```
SELECT "com1:a" FROM SPDB;
```

例2：com1:aが配列型の場合、先頭要素と2番目の要素を参照します。
さらに条件式に配列要素を指定しています。

```
SELECT "com1:a"[0], "com1:a"[1] FROM SPDB WHERE "com1:a"[0] > 1;
```

例3：com1:aが配列型の場合、条件式に配列を指定しています。
この場合、暗黙的にcom1:a[0]と同じ意味となります。

```
SELECT "com1:a"[0], "com1:a"[1] FROM SPDB WHERE "com1:a" > 1;
```

例4：配列検索ANYの例です。com1:aの要素数は2とします。次の2つの例文は、同じ意味となります。
ANYはすべての要素との比較を行い、いずれか一つでも真となる場合に、真が成立する指定です。

```
SELECT "com1:a"[0], "com1:a"[1] FROM SPDB WHERE 1 < ANY("com1:a");
```

```
SELECT "com1:a"[0], "com1:a"[1] FROM SPDB WHERE 1 < "com1:a"[0] OR 1 < "com1:a"[1];
```

※ANYは右辺にのみ使用できます。

例5：配列検索ALLの例です。com1:aの要素数は2とします。次の2つの例文は、同じ意味となります。
ALLはすべての要素との比較を行い、すべてが真となる場合に、真が成立する指定です。

```
SELECT "com1:a"[0], "com1:a"[1] FROM SPDB WHERE 1 < ALL("com1:a");
```

```
SELECT "com1:a"[0], "com1:a"[1] FROM SPDB WHERE 1 < "com1:a"[0] AND 1 < "com1:a"[1];
```

※ALLは右辺にのみ使用できます。

5.15.2.2.2 SHOW

SHOW句は、時系列データベース内のメタ情報を参照する構文です。 現在は、エイリアス名（SpeeDBee Synapseで作成されるカラム名に相当）のみ参照が可能です。
ORDER BYを省略すると、並び順は保証されません。

- SHOW ALIAS

例1：エイリアス名の一覧を取得（昇順）

```
SHOW ALIAS ORDER BY ALIAS_NAME ASC
```

例2：エイリアス名の一覧を取得（降順）

```
SHOW ALIAS ORDER BY ALIAS_NAME DESC
```

構文仕様

```
SHOW ALIAS [ORDER BY ALIAS_NAME [ASC/DESC]]
```

5.15.2.2.3 SET

グローバル変数へ値を代入する構文です。 SQL内では、グローバル変数の先頭には、必ず$を付与する必要があります。

例1：整数の代入

```
SET $GVAR0 TO 10;
```

例2：マイナス値の代入

```
SET $PVAR0 -111;
```

例3：浮動小数の代入

```
SET $GVAR1 1.235;
```

例4：指数表現での代入

```
SET $PVAR1 -1.234E-2
```

例5：文字列の代入

```
SET $PVAR2 'abcd'
```

代入したグローバル変数の参照例

```
SELECT \"com1:a\",$GVAR1, $PVAR2 FROM SPDB WHERE _ts BETWEEN NOW - 1 sec AND NOW ORDER BY _TS DESC
LIMIT 1;
```

構文仕様

```
SET グローバル変数名 [TO] リテラル値
```

5.15.2.2.4 INSERT

INSERT句は、時系列データベースへデータを登録する構文です。 存在するコンポーネントのカラムに対して、登録が可能です。登録可能なカラム名の一覧は、SHOWで取得できます。
INSERT文の[INTO]直後のテーブル指定は、SPDB固定となります。

例1：リソースコレクタ「resource1(リソースコレクタ)」のカラム「cpu_usage」に、値「2」を挿入する

```
INSERT INTO SPDB (\"resource1:cpu_usage\") VALUES (2);
```

※タイムスタンプを省略した場合、登録されるタイムスタンプは稼働マシンのシステム時刻が使用されます。

例2：リソースコレクタ「resource1(リソースコレクタ)」のカラム「cpu_usage」に時刻「2025年04月03日 13時02分09秒」を指定して、値「10」を挿入する

ISO8606形式で指定する場合

```
INSERT INTO SPDB (_TS, \"resource1:cpu_usage\") VALUES ('2025-04-03T13:02:09', 10);
```

エポックナノ秒で指定する場合

```
INSERT INTO SPDB (_TS, \"resource1:cpu_usage\") VALUES (1743685329000000000, 10);
```

例3：整数型の値「10」を挿入する

```
INSERT INTO SPDB (\"customcomp1:columnA\") VALUES (10);
```

例4：浮動小数型の値「3.14」を挿入する

```
INSERT INTO SPDB (\"customcomp1:columnB\") VALUES (3.14);
```

例5：文字型の値「ABCDE」を挿入する

```
INSERT INTO SPDB (\"customcomp1:columnC\") VALUES ('ABCDE');
```

例6：配列の値{3,4}を挿入する

```
INSERT INTO SPDB (\"customcomp1:columnD\") VALUES ({3,4});
```

例7：一つのコレクタ＆複数のカラムへ、複数のデータを挿入する

```
INSERT INTO SPDB (\"customcomp1:columnA\",
\"customcomp1:columnB\",\"customcomp1:columnC\",\"customcomp1:columnD\",\"customcomp1:columnE\")
VALUES (10,3.14,'ABCDE',{3,4},'00000000000000000000000000')
```

例8：複数のコレクタ＆複数のカラムへ、複数のデータを挿入する

```
INSERT INTO SPDB (\"resource1:cpu_usage\", \"customcomp1:columnA\",
\"customcomp1:columnB\",\"customcomp1:columnC\",\"customcomp1:columnD\",\"customcomp1:columnE\")
VALUES (99,10,3.14,'ABCDE',{3,4},'00000000000000000000000000');
```

構文仕様

```
INSERT [INTO] SPDB (_TS, <column_name>,...) VALUES (タイムスタンプ値, value,...)
```

5.15.3 DBクエリサービス利用時の注意点

DBクエリサービス利用時の注意点を説明します。

SpeeDBee SynapseのDBクエリサービスは、高負荷時においても安定稼働を維持する設計を目指しておりますが、DBクエリの実行に関しては、システムの負荷状況に大きく左右される特性があります。
特に、システムが高負荷で動作している場合、DBクエリの応答性能が一時的に低下する可能性があります。

クエリ条件が及ぼすシステムへの負荷

データベースに対するクエリは、指定するデータの対象範囲が広くなればなるほど、システムへの負荷が高まります。たとえば期間を広くとるクエリや、絞り込み条件が少ないクエリは、大量のデータを走査する必要があるため、応答時間が長くなるだけでなく、他の処理にも影響を及ぼす可能性があります。

安定したシステム運用のため、クエリ実行の際は以下の点にご留意ください。

クエリ実行の留意事項

- アクセストークンを設定する
  アクセストークンを発行して、DBクエリサービスへ適用することで、DBクエリを利用できる利用者を制限することが出来ます。
- 必要最小限の範囲でクエリを実行する
  必要なデータに絞り込み、期間や条件をできるだけ具体的に設定してください。
- 高負荷時を避ける
  可能であれば、システム利用が集中する時間帯を避けてクエリを実行することを推奨します。

5.16 演算式・条件式

ここでは、SpeeDBee Synapseで利用可能な演算式、条件式について説明します。

5.16.1 演算式・条件式の概要

SpeeDBee Synapse内で、演算式と条件式が使える機能を示します。

項目	演算式	条件式	説明
演算コレクタ	○	-	任意の演算結果をコレクタのデータとして登録
イベントデータ	-	○	条件式が真となる場合に、条件に合致したデータとして登録
イベントトリガ	-	○	条件式が真となる場合に、そのイベント区間の情報を出力
出力ポート数値変換	○	-	任意の演算結果をコレクタのデータとして登録
出力ポート登録条件	-	○	条件式が真となる場合に、対象のデータを登録

5.16.2 演算式と条件式（共通）

演算式と条件式、共通で使用できる式について、説明します。

項目	
例	($VALUE + 10) * ABS(5)
使用可能な比較演算	=,<>,!=,<,>,<=,>=
使用可能な算術演算	(),+,-,*,/,%
使用可能な論理演算	AND,OR,NOT
制御文	IF

使用可能な変数	説明
$VALUE	コレクタから受け取った値、もしくは、変換式による計算結果
$PVALUE	前回DBに登録された値
$GVAR0～63	グローバル変数
$PVAR0～63	永続化グローバル変数
	※グローバル変数についてはこちらを参照してください。

使用可能な定数	説明
M_E	ネイピア数、自然数の底（e）
M_LOG2E	log_e(e)
M_LOG10E	log_10(e)
M_LN2	log_e(2)
M_LN10	log_10(e)
M_PI	円周率（Pi）
M_PI_2	Pi/2
M_PI_4	Pi/4
M_1_PI	1/Pi
M_2_SQRTPI	2/Pi
M_SQRT2	sqrt(2)
M_SQRT1_2	1/sqrt(2)

使用可能な関数	説明
ABS	絶対値（整数）
ACOS	三角関数　逆余弦
ASIN	三角関数　逆正弦
ATAN	三角関数　逆正接
CEIL	小数点以下切り上げ
COS	三角関数　余弦
COSH	双曲線余弦
EXP	底がネイピア数eである指数関数
FABS	絶対値（倍精度浮動小数）
FLOOR	小数点以下切り捨て
FMOD	剰余
LOG	自然対数
LOG10	10を底とする対数
POW	べき剰
ROUND	小数点以下四捨五入
SIN	三角関数　正弦
SINH	双曲線正弦
SQRT	平方根
TAN	三角関数　正接
TANH	双曲線正接

5.16.3 演算式

演算式について、説明します。演算式は、任意の演算結果を算出する場合に指定する式です。コレクタから取得した値に、何らかの演算が必要な場合などに利用できます。

項目	
例	($VALUE + 10) * ABS(5)

グローバル変数の代入について（参考）

GVAR1 :=(式)

- 代入するグローバル変数の先頭には$は必要ありません。
- "GVAR1 := $VALUE * 2"のように代入を指定すると、演算結果としても$VALUEに2を乗算した値が返されます。

例
GVAR1 := ($VALUE * 3)
($VALUEを2と仮定した場合)
①GVAR1 には 2 * 3 = 6 の値が格納されます。
②演算結果は、GVAR1に代入された値（6）と同じになります。

IF文を使った代入

IF(条件式,真の場合の値,偽の場合の値)

GVAR1 := IF($VALUE = 1,$GVAR1 + 1, $GVAR1)

上記の場合、$VALUEが1の場合に、真となれば$GVAR1+1の結果が返されます。
偽の場合は、$GVAR1が元の値のまま代入されます。

注意点として、「真の場合の値」と「偽の値の場合」内で、代入を行うと、真偽に関係なく毎回評価されてしまいます。
IF($VALUE=1,GVAR1 := $GVAR1 + 1, $GVAR1)
このような指定をすると、真偽に関係なく「真の場合の値」が式として評価され、毎回代入されてしまいます。意図せぬ値となる可能性があるので、代入は、IF文の外で指定してください。

5.16.4 条件式

条件式について、説明します。条件式は、任意の条件の真偽を得る場合に指定する式です。真偽を得ることで、その先の出力への制御に利用できます。

項目	
例	($VALUE + 10) * ABS(5) < 100

グローバル変数の代入について（参考）

1 < (PVAR5 :=(式))

- 代入するグローバル変数の先頭には$は必要ありません。
- "PVAR5 := ($VALUE * 2)"のように代入のみを指定することはできません。条件式としての指定が必要です。

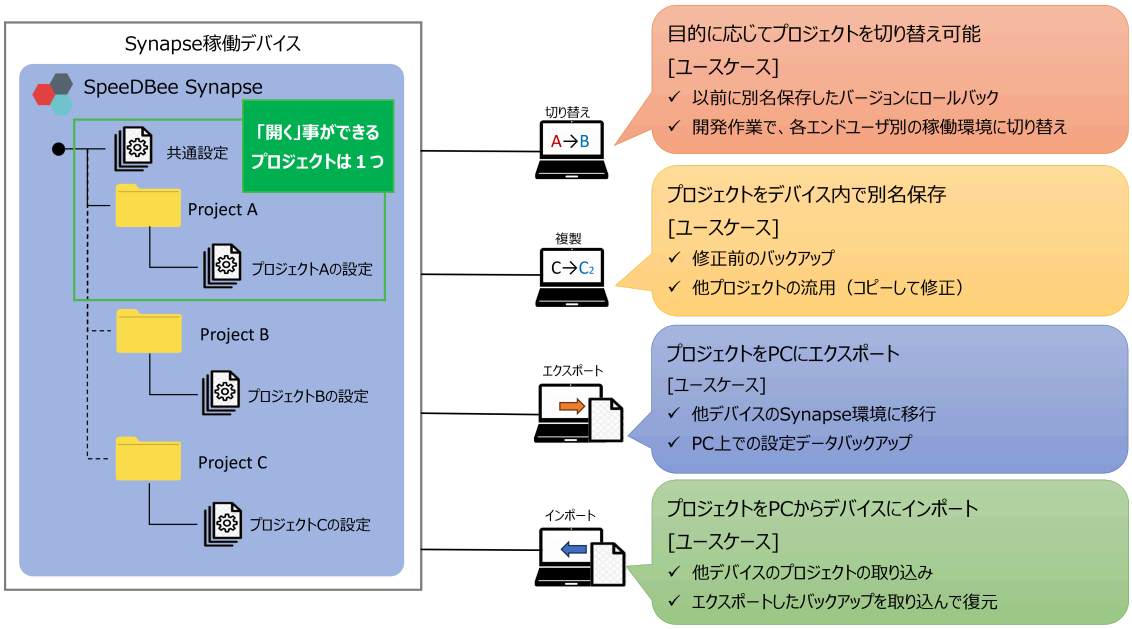
例
GVAR1 := ($VALUE * 3)
($VALUEを2と仮定した場合)
①GVAR1 には 2 * 3 = 6 の値が格納されます。
②演算結果は、GVAR1に代入された値（6）と同じになります。

5.17 設定情報の管理(プロジェクト)

5.17.1 プロジェクトの概要

プロジェクトは、SpeeDBee Synapseの設定内容を管理する仕組みです。 SpeeDBee Synapseの設定内容は、共通設定とプロジェクト個別の設定に分けられます。 プロジェクトを切り替えることで、同一のデバイス上で複数の設定を管理することが可能になります。 またプロジェクト設定をエクスポートし、他のデバイスにインポートするなどの機能があります。

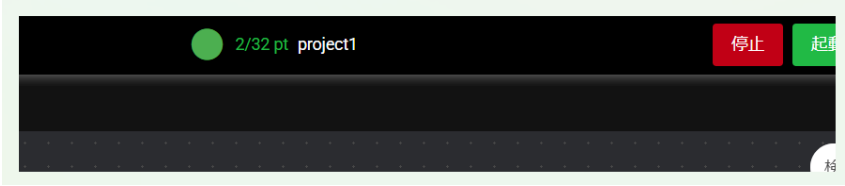
以下はプロジェクト機能のイメージ図です。



以下は共通設定とプロジェクト個別の設定の一覧です。

共通/プロジェクト固有	設定内容
共通	・ライセンス情報 ・ユーザー管理/認証 ・暗号化通信 ・サーバー設定 ・ログ ・データベース ・アクセストークン
プロジェクト固有	・設定されているデータフロー(パネル、コンポーネント、フローリンク) ・登録したカスタムコンポーネント ・登録した依存モジュール（Python） ・コンポーネントが出力したデータ ・Grafana設定

SpeeDBee Synapseの動作中は、必ず１つのプロジェクトが選択されています。 現在選択中のプロジェクトはヘッダーに表示されます。



現在選択中のプロジェクトに対して以下の操作を行うことはできません。
・削除

5.17.2 新規作成

次の手順で、プロジェクトを作成することができます。

1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



2. 「新規作成」ボタンを押下します。



3. 新規作成するプロジェクトの名前を入力します。



4. 「保存」を押下します。



5. 新しいプロジェクトが作成されます。

5.17.3 複製

次の手順で、プロジェクトのコピーを作成することができます。

1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



2. 複製するプロジェクトの「コピーアイコン（複製）」を押下します。



3. 複製のプロジェクト名を入力します。



4. 「保存」を押下します。



5. プロジェクトの複製が作成されます。

5.17.4 インポート

次の手順で、プロジェクトをインポートすることができます。

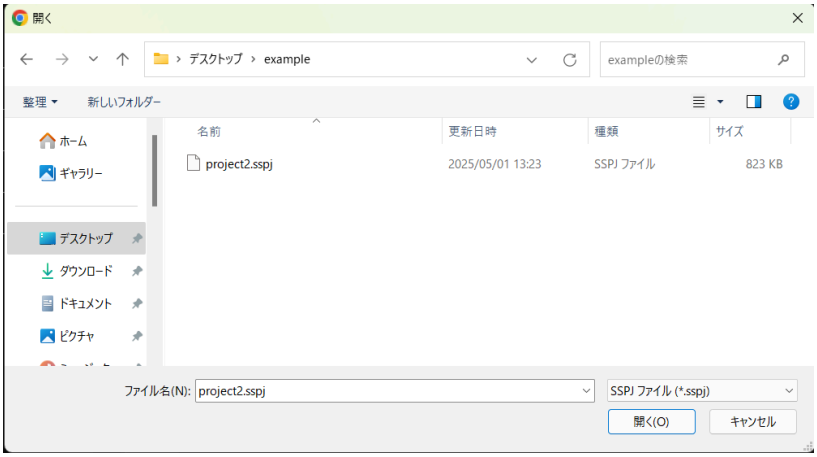
1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



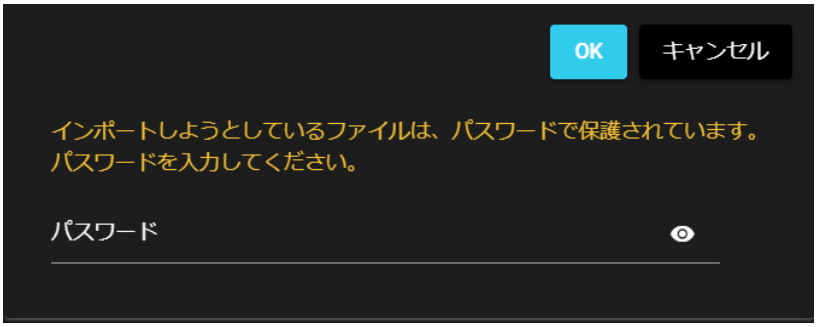
2. 「インポート」ボタンを押下します。



3. 「エクスポート」の手順でエクスポートしたファイルを選択します。



4. ファイルがパスワードで保護されている場合は、パスワードを入力して「OK」を選択します。



5. プロジェクトがインポートされます。

注意事項

・Pythonカスタムコンポーネントの依存モジュールは、インポート先のプラットフォーム又はPythonバージョンが異なる場合はインターネット経由でインストールされます。インターネット接続されていない場合は、Pythonモジュールはインストールされません。
・C言語カスタムコンポーネントは、インポート先のプラットフォームが異なる場合はインポートされません。

5.17.5 エクスポート

次の手順で、プロジェクトをエクスポートすることができます。

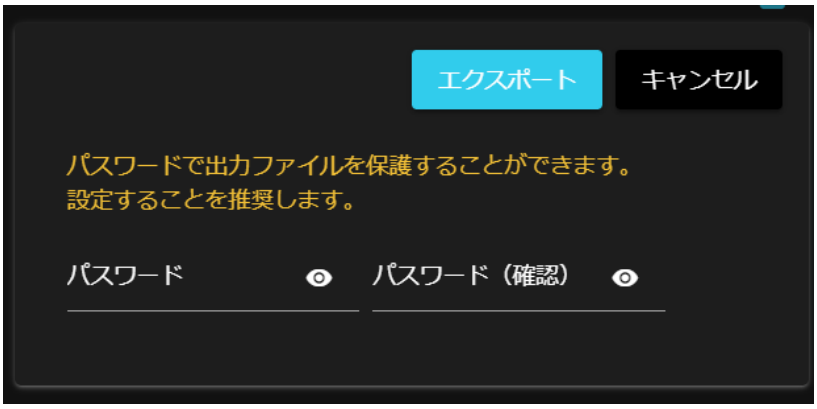
1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



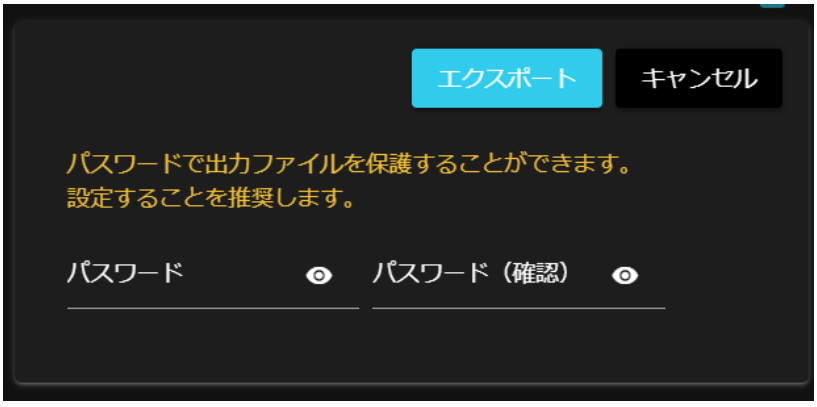
2. エクスポートするプロジェクトの「exportアイコン（エクスポート）」アイコンを押下します。



3. エクスポートするプロジェクトをパスワードで保護する場合は、任意のパスワードを入力します。
※この手順は省略可能です。



4. 「エクスポート」を選択します。



5. プロジェクトがエクスポートされます。

5.17.6 切替

次の手順で、プロジェクトを切り替えることができます。

1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



2. 切り替え先のプロジェクトのラジオボタンを選択します。



3. 確認ダイアログで「はい」を選択します。
4. プロジェクトが切り替わります。

5.17.7 情報更新

次の手順で、プロジェクトの情報を更新することができます。更新できる情報はプロジェクト名のみです。

1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



2. 情報を更新するプロジェクトの「エディットアイコン（編集）」アイコンを押下します。



3. 新しいプロジェクト名を入力します。



4. 「保存」を押下します。



5. プロジェクト名が更新されます。

5.17.8 削除

次の手順で、プロジェクトを削除することができます。

1. 設定メニューアイコンを押下し、「プロジェクト設定」を選択します。



2. 削除するプロジェクトの「ゴミ箱アイコン（削除）」を押下します。



※現在選択中のプロジェクトを削除することはできません。

3. 確認ダイアログで「はい」を選択します。
4. プロジェクトが削除されます。

6 応用的な利用方法

6.1 応用的な利用方法

本章では、応用的な利用方法について説明します。

ここで説明する内容は、より発展的なシステムを構築するために役立つものとなります。記載内容は次のとおりです。

1. カスタムコンポーネントの作成・利用
2. 制御プログラムとの連携（WebAPI）
3. Grafanaとの連携
4. グローバル変数

6.2 カスタムコンポーネントの作成・利用

6.2.1 カスタムコンポーネントの作成・利用

ここでは、カスタムコンポーネントの作成・利用方法について説明します。

このカスタム機能を利用することで、独自の機能をSpeeDBee Synapseへ追加することが可能となります。記載内容は次のとおりです。

1. カスタムコンポーネントの概要
2. Pythonカスタムコンポーネント作成
3. C言語カスタムコンポーネント作成
4. カスタムUI作成
5. カスタムコンポーネント開発環境(SCCDE)

6.2.2 カスタムコンポーネントの概要

6.2.2.1 カスタムコンポーネントとは

SpeeDBee Synapseでは、データの収集や送信を担うモジュールのことを「コンポーネント」と呼びます。

標準ではPLC等のデータを収集するコンポーネントや、クラウドにデータを送信するコンポーネントが用意されていますが、ユーザーが独自にコンポーネントを開発、運用することも可能です。

コンポーネントの開発にはC、もしくは、Pythonを使用できますが、本章では言語によらず、カスタムコンポーネントの開発に共通して必要な基本概念を説明します。

6.2.2.2 コンポーネントベースとコンポーネントインスタンス

通常はひとまとめに「コンポーネント」と呼称しますが、カスタムコンポーネントを実装する上では、「コンポーネントベース」と「コンポーネントインスタンス」の違いを理解する必要があります。



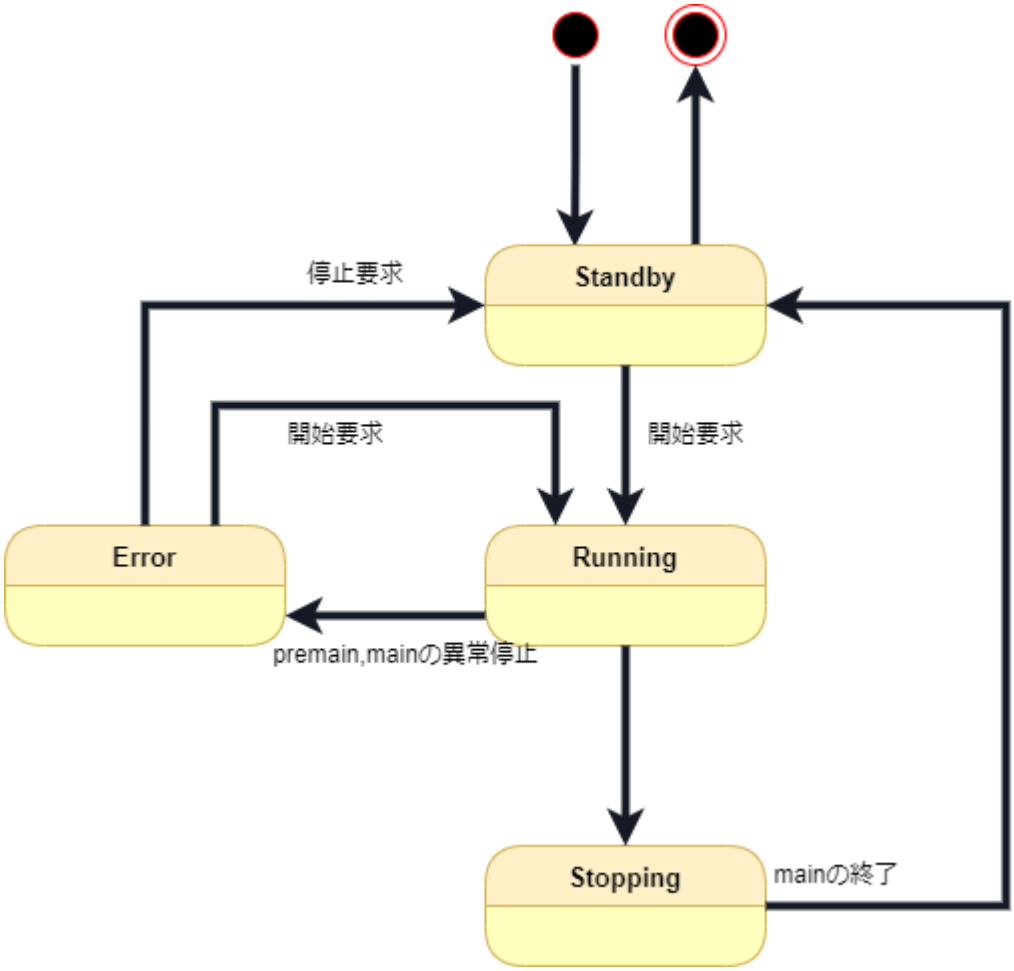
左側に１列に並べられている項目が、「コンポーネントベース」、中央～右側にある３つの要素が「コンポーネントインスタンス」です。

「コンポーネントインスタンス」は、いずれかの「コンポーネントベース」をドラッグアンドドロップすることで作成されます。 特殊なものを除き、１つの「コンポーネントベース」から複数の「コンポーネントインスタンス」を作成することができます。

カスタムコンポーネントを実装する場合もこれは同様になります。 １つのカスタムコンポーネントを作成すると、画面の左側の列に新しい「コンポーネントベース」が１つ追加されます。 この追加されたコンポーネントベースをドラッグアンドドロップすることで、「コンポーネントインスタンス」を複数作成することができます。

6.2.2.3 コンポーネントインスタンスのライフサイクル

すべてのコンポーネントインスタンスは、作成されてから削除されるまで下図のライフサイクルで動作します。

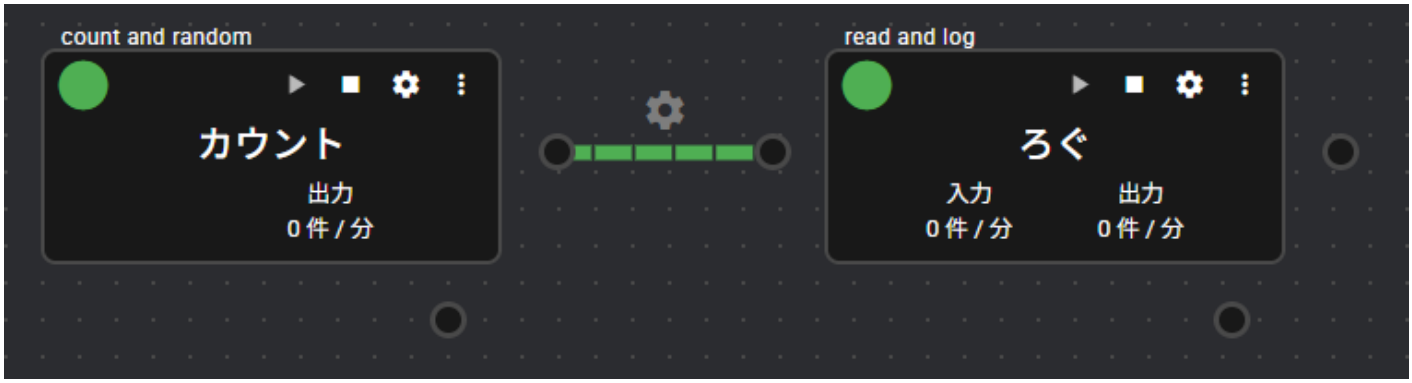


- コンポーネントインスタンスの作成直後は`Standby`状態です
  この状態ではコンポーネントは何も動作していません。何のデータも出力せず、何のデータも入力されません。
- コンポーネントインスタンスの起動アイコンをクリックすると、`Running`状態に遷移します
  この状態にあると、コンポーネントインスタンスは継続的にデータを出力したり、他のコンポーネントからデータを受け取ったりすることができます。
- コンポーネントインスタンスの停止アイコンをクリックすると、`Stopping`状態になり、その後すべての処理を完了すると`Standby`状態に戻ります。
- `Running`途中でコンポーネントインスタンスの処理が継続できないエラー等が発生した場合は、`Error`状態に移行し、処理を停止します

カスタムコンポーネントの開発では、このライフサイクルの中で、様々な処理をプログラミングすることができます。 以降の章ではプログラミング言語ごとの実装方法を説明します。

6.2.2.4 コンポーネントの入力ポートと出力ポート

各コンポーネントは、出力ポートと入力ポートを備えています。 出力ポートは、そのコンポーネントが生成したり、外部にアクセスしたりして収集したデータをSpeeDBee Synapseに出力するための要素。 入力ポートは、別のコンポーネントが出力したデータを受け入れて、そのコンポーネントで利用するための要素です。



カスタムコンポーネントの実装においては、それぞれのポート数を指定する必要があります。 データを出力するコンポーネントの場合は出力ポート数を1に、データを受け入れるコンポーネントの場合は入力ポート数を1に、それぞれのポートが必要のない場合は0としてください。 将来的には複数のポートを扱えるようにする可能性がありますが、現時点では各ポートは最大１つに制限されています。

6.2.3 Pythonカスタムコンポーネント作成

本項では、Pythonによるカスタムコンポーネントの作成方法について説明します。

6.2.3.1 前提知識

カスタムコンポーネントを開発する上での前提知識として、Python言語のチュートリアルの内容は大方把握できていること、併せてクラス定義、メソッド定義、クラスの継承等について理解していることを想定しています。 入門レベルの情報については、下記の Python公式ドキュメントを参照してください。

Python公式ドキュメント： https://docs.python.org/ja/3/

6.2.3.2 サンプルカスタムコンポーネント

Pythonによるカスタムコンポーネントの実装方法説明のため、シンプルな仕様のカスタムコンポーネントをサンプルとして提供しています。

サンプルカスタムコンポーネントのダウンロード

もしくは、SpeeDBee Synapseをインストール済みの場合、カスタムコンポーネントサンプルも下記の場所にインストールされています。

```
$ ls /usr/local/speedbeesynapse/share/sample_component_py/
sin_and_random.py  read_and_log.py
```

6.2.3.2.1 サンプルの内容

カスタムコンポーネントのサンプルとして本章では下記のサンプルを説明します。

- `sin_and_random.py`

  出力ポートに対して以下の２つのカラムを作成し、データを出力し続けるコンポーネントです。

カラム名	データ型	内容
sin	DOUBLE	1HzのSIN波の値を秒間10件登録します。SIN波の周期はパラメータで変更可能です
rand	INT32	１秒に１回、[0, 100)の範囲内のいずれかの値をランダムに登録します

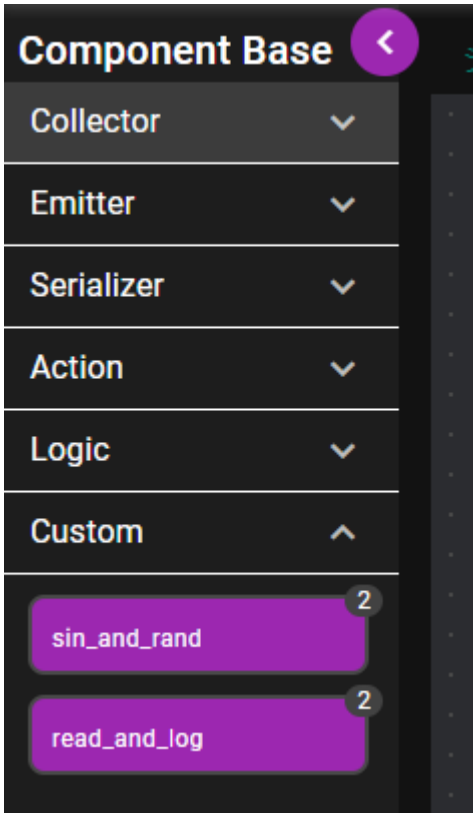
- `read_and_log.py`

  入力ポートからデータを受け取り、それを文字列型カラムとログに出力します。

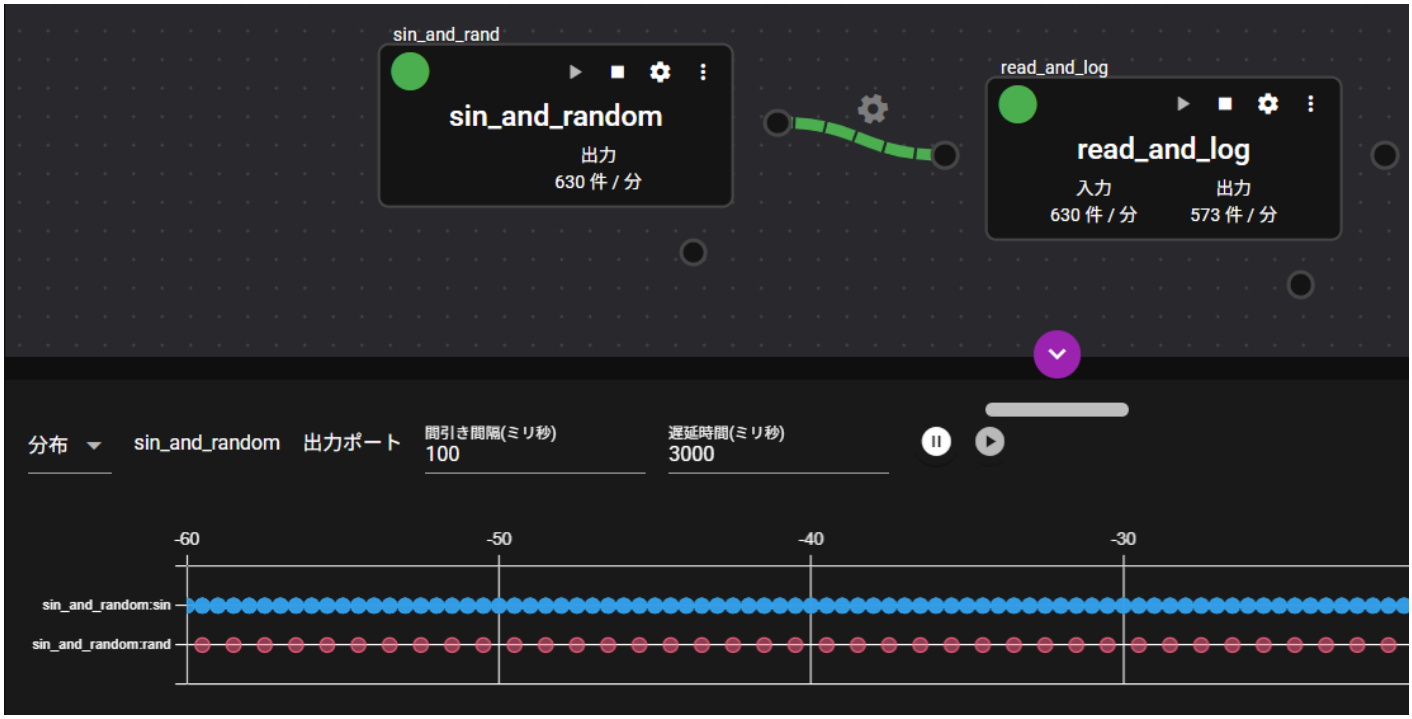
6.2.3.2.2 実行

`*.py`ファイルをそのまま画面から登録することで、エラー等がなければコンポーネントベースが追加された状態になります。登録方法は「カスタムコンポーネントを使用する」を参照してください。

登録後は、通常のコンポーネントと同様にD&Dでコンポーネントインスタンスを生成できます。



`sin_and_random`および`read_and_log`を繋いで実行することで、それぞれの出力したカラムを表示できます。 下記は`sin_and_random`の出力ポートをモニタリングしています。値の変化を確認するには、「間引き間隔」を100ミリ秒へ変更してください。



6.2.3.3 カスタムコンポーネント実装詳細

以降では、サンプルの`sin_and_random`、および、`read_and_log`のソースコードを例に、カスタムコンポーネントの実装内容を説明していきます。

6.2.3.3.1 カスタムコンポーネントクラス定義

Pythonによるカスタムコンポーネントは、下記の通り`HiveComponentBase`クラスを継承した新しいクラスを定義することでそのコンポーネントベースの名前や識別情報、実行時の処理を定義することができます。

```
from speedbeesynapse.component.base import HiveComponentBase, HiveComponentInfo, DataType
  :

@HiveComponentInfo(uuid='00000000-0000-0000-0000-000000000000', name='コンポーネント名', inports=0,
outports=1)
class HiveComponent(HiveComponentBase):
    def main(self, param):
        :
```

クラス名は必ず`HiveComponent`としてください。 また、コンポーネントの情報を`@HiveComponentInfo`デコレータでラップする必要があります。 ここに定義した情報により、画面でもコンポーネントベースの情報を扱うことができるようになります。 下記の説明を参考に設定してください。

メンバ名	定義情報	説明
`uuid`	UUID	コンポーネントを識別するためのUUIDです。ランダムなIDを生成してここに設定してください。

メンバ名	定義情報	説明
`name`	コンポーネント名	コンポーネントの名前を設定します。画面ではこの名前が表示されますのでなるべく他のコンポーネントと被らず、分かりやすい名前にしてください。
`tag`	タグ	コンポーネントへタグを付与します。"collector","emitter","serializer","action","logic"のいずれかの指定によりコンポーネントベースの配置先を指定できます。省略した場合は、Customへ配置されます。
`outports`	出力ポート数	このコンポーネントが持つ出力ポートの数を設定します。他のコンポーネントの入力ポートにデータを渡す場合はこれを1としてください。
`inports`	入力ポート数	このコンポーネントが持つ入力ポートの数を設定します。他のコンポーネントの出力ポートからデータを受け取る場合は、これを1としてください。

UUIDについては必ずランダムなIDを生成してください。 他のコンポーネントと同じUUIDにしてしまうと、どちらのコンポーネントも使えなくなることがあります。

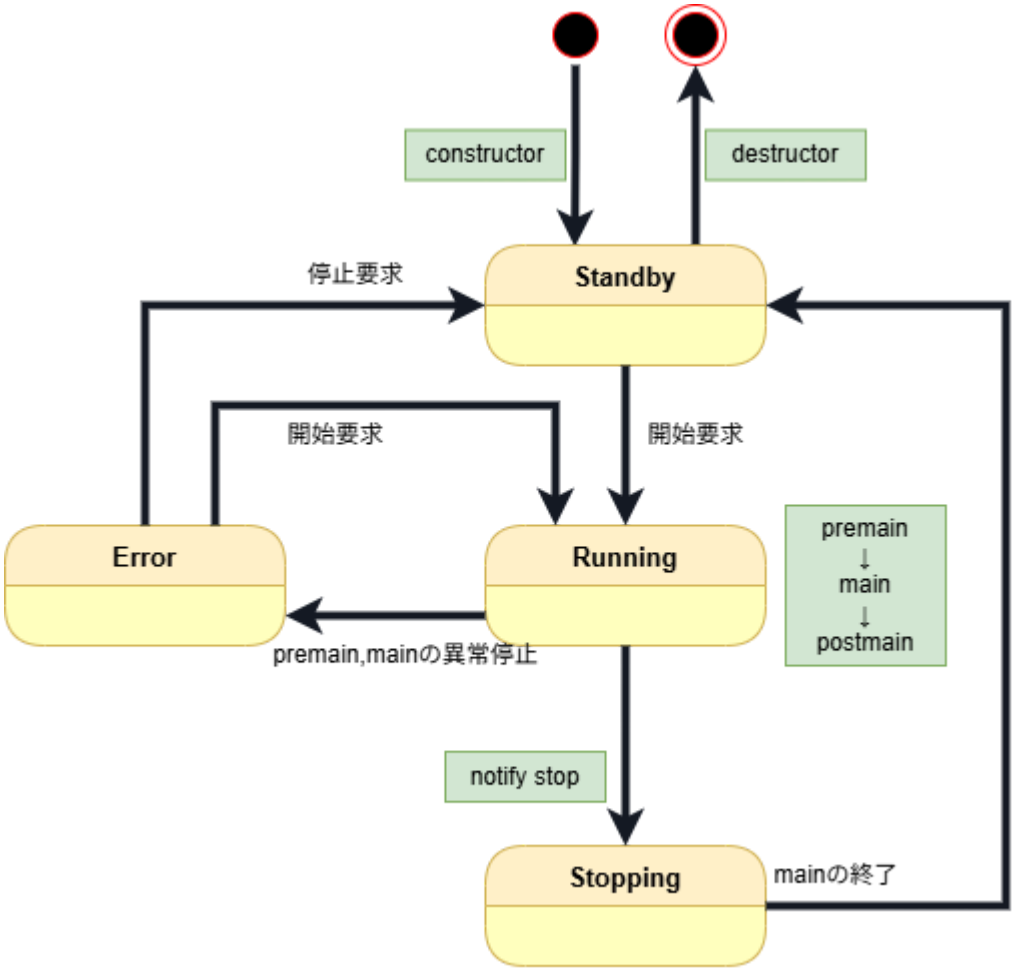
6.2.3.3.2 `HiveComponent` クラスのライフサイクル

ここで定義した`HiveComponent`クラスが、１つのコンポーネントベースとなります。 Pythonスクリプトファイル内で、このクラスをインスタンス化するような処理は必要ありません。 かわりに、画面上でコンポーネントベースをD&Dしてコンポーネントインスタンスを生成すると、自動でここで定義した`HiveComponent`クラスのインスタンスが生成されます。 また、画面上でコンポーネントインスタンスを削除すると、このクラスのインスタンスが破棄されます。

そのため、`HiveComponent`クラス内に`__init__()`メソッドや`__del__()`メソッドを定義しておけば、画面上でのコンポーネントインスタンスの生成・削除に合わせてそのメソッドが実行されることになります。

```
@HiveComponentInfo(uuid='00000000-0000-0000-0000-000000000000', name='サンプルコンポ', inports=0,
outports=0)
class HiveComponent(HiveComponentBase):
    def __init__(self):
        # called at creating component instance
        pass
    def __del__(self):
        # called at deleting component instance
        pass
    def premain(self, param):
        # called before main()
        pass
    def postmain(self, param):
        # called after main() finished
        pass
    def main(self, param):
        # called at starting component instance
        while self.is_runnable():
          pass
    def notify_stop(self):
        # called before the component received stop request from system
        pass
```

他にも、画面からのコンポーネントの操作により、コールされるメソッドがあります。 コールされるタイミングとそのメソッドについては下図、下表を参照してください。



コールバック関数	説明
`__init__`	コンポーネントのインスタンスが生成されたときにコールされます
`__del__`	コンポーネントのインスタンスが破棄されるときにコールされます
`premain`	コンポーネントを開始するときにmainの前にコールされます
`main`	コンポーネントのメインの処理を実装する関数です コンポーネントの終了要求を受けるまで、この関数は終了せずにループ処理し続ける必要があります
`postmain`	コンポーネントの終了時、mainの終了直後にコールされます
`notify_stop`	mainの実行中、コンポーネントの停止要求を受けたときに実行されます このメソッドは`main()`の実行中に呼ばれるため、別スレッドで実行されることに注意してください。

`main`は、コンポーネントの処理の本体になりますので、必ず定義する必要があります。 以降ではサンプルプログラムの`main`関数の実装方法を説明します。

6.2.3.3.3 出力ポートのカラム作成/データ登録(`sin_and_random`)

下記はカスタムコンポーネントのサンプル`sin_and_random`のメインの処理です。 これをもとにデータを出力する流れを説明します。

```
def main(self, param):
     :
    count = 0

    # Create columns.
    self.sin = self.out_port1.Column('sin', DataType.DOUBLE)
    self.rand = self.out_port1.Column('rand', DataType.INT32)

    # Repeat at 100msec intervals.
    for [ts, skip] in self.interval_iteration(100000000):

        # Compute sin wave value and insert it into 'sin' column.
        val = math.sin(2*math.pi*sin_hz*count/10)
        self.sin.insert(val, ts)

        # Insert random value into 'rand' column only once in 10.
        if count % 10 == 0:
            self.rand.insert(random.randrange(0, 100), ts)

        count += 1
```

1. カラムの作成

   初めに、カラムを作成します。

```
self.sin = self.out_port1.Column('sin', DataType.DOUBLE)
self.rand = self.out_port1.Column('rand', DataType.INT32)
```

   このコンポーネントでは倍精度浮動小数点数のカラム`sin`と、32bit整数のカラム`rand`を利用しますので、`self.out_port1`にある`Column()`メソッドにてカラム名、データ型等を指定して、カラムオブジェクトを生成しています。 戻り値は後にカラムにデータを登録する際に使用しますので、変数に格納しておきます。

   コンポーネントが出力するすべてのデータは、このように生成したカラムに対して登録していく必要があります。

2. 定期的な処理の繰り返し

   定期的にデータを登録するため、for文で繰り返し処理を実装しています。

```
for [ts, skip] in self.interval_iteration(100000000):
    :
```

   ループの判定条件で呼び出している`self.interval_iteration()`は、引数に指定した時間間隔で繰り返し値を返すジェネレータになっています。 それにより、このfor文の内部は一定間隔で実行されることになります。

   引数に指定するのはナノ秒単位になりますので、この例の場合は0.1秒(100000000ナノ秒)に１回、ループ処理が実行されることになります。

   画面からコンポーネントを停止しない限り、このfor文のループは終わることはありません。

3. カラムへのデータ登録

   繰り返し処理の中で、カラムに対してデータを登録します。

```
val = math.sin(2*math.pi*sin_hz*count/10)
self.sin.insert(val, ts)

if count % 10 == 0:
    self.rand.insert(random.randrange(0, 100), ts)
```

   データの登録は、事前に作成したカラムオブジェクトの`insert()`メソッドの呼び出しで行なえます。 第一引数には登録する値を、第二引数にはその値を登録する時刻を指定してください。第二引数は省略すれば現在時刻として扱われます。

> **insertで登録できる最大サイズ**
>
> insert()で登録できるデータの最大サイズは、512バイト（STRINGはNULLを含まない511バイト）です。上限サイズを超えた値の登録は、例外がスローされます。

6.2.3.3.4 入力ポートからのデータ読み込み（`read_and_log`）

下記はカスタムコンポーネントのサンプル`read_and_log`のメインの処理です。 これをもとに別のコンポーネントが出力したデータを受け取る処理の流れを説明します。

```
def main(self, param):
    # Create a column for logging received data.
    self.logclm = self.out_port1.Column('log', DataType.STRING)

    # Logging column information received from 'in_port1'.
    columns = self.in_port1.get_columns()
    for column in columns:
        self.log.info(column)

    # Start receving column values from 'in_port1'.
    with self.in_port1.ContinuousReader(start=self.get_timestamp()) as reader:
        # Loop until the component received 'stop' request.
        while self.is_runnable():
            # Get a group of records of column data (called as window_data).
            window_data = reader.read()
            if not window_data:
                # window_data may be None.
                self.log.info("no data yet")
                continue

            # Process all records in a window_data one by one.
            for record in window_data.records:
                # Convert timestamp as a string.
                ts = datetime.datetime.fromtimestamp(record.timestamp / 1000000000)

                # Process all column's value in a record one by one.
                for columnvalue in record.data:
                    if columnvalue.value:
                        # Make string from columnvalue, logging it and insert it into the column
                        logmsg = f"{columnvalue.column}[{str(ts)}]: {columnvalue.value}"
                        self.log.info(logmsg)
                        self.logclm.insert(logmsg)
                    else:
                        pass # ignore None
```

先頭で作成している`log`カラムは、文字列として情報を出力するためのものです。 別コンポーネントからデータを受けるだけなら必須ではないため割愛します。

1. 入力ポートのカラム情報の取得

   `in_port1`にある`get_columns()`メソッドにより、そのコンポーネントの入力ポートに接続されたコンポーネントが出力しているデータのカラム情報を取得できます。

   ```
   columns = self.in_port1.get_columns()
   for column in columns:
       self.log.info(column)
   ```

   上記の例では取得したカラムをイテレーションループでログに出力しています。

   コンポーネントへの接続を変えたり、接続されたコンポーネントを停止したりすると、この`get_columns()`で取得できるカラム情報が変化します。 常に最新の情報が必要な場合には定期的に取得して内容をチェックするようにしてください。

2. `ContinuousReader`の開始とループ

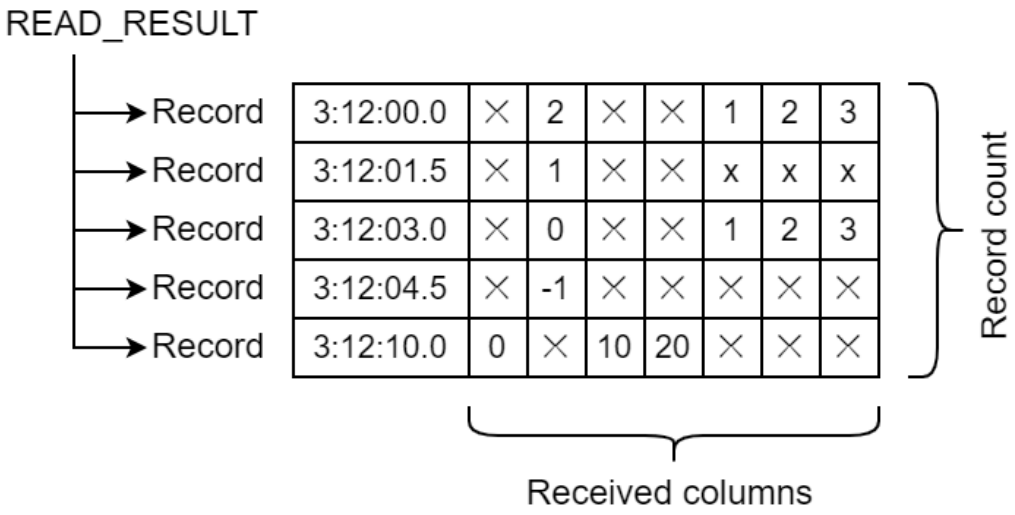
   入力ポートから継続的にデータを取得し続ける場合、`ContinuousReader`オブジェクトを利用します。 下記のようにwith句で使用し、さらに繰り返し`read()`をコールすることで入力ポートに届いたデータを取得することができます。

```
with self.in_port1.ContinuousReader(start=self.get_timestamp()) as reader:
    while self.is_runnable():
        window_data = reader.read()
        if not window_data:
            self.log.info("no data yet")
            continue
        :
```

   `read()`の戻り値はNoneになることもあるのでその場合は読み飛ばしてください

3. 取得したデータのレコードを参照

   `ContinuousReader`の`read()`メソッドは、`HiveWindowData`オブジェクトを返します。 このオブジェクトは、入力ポートに入ってきたデータをある時間幅分まとめて保持したもので、下図のように同一時刻のデータを収めた複数のレコードからなり、１つのレコードに複数のカラムのデータが収められています。



   そのため、下記のように二重のforループ文によりすべてのデータにアクセスすることができます。

```
for record in window_data.records:
    ts = datetime.datetime.fromtimestamp(record.timestamp / 1000000000)

    for columnvalue in record.data:
        if columnvalue.value:
            logmsg = f"{columnvalue.column}[{str(ts)}]: {columnvalue.value}"
            self.log.info(logmsg)
            self.logclm.insert(logmsg)
        else:
            pass # ignore None
```

   各データの詳細は`HiveRecordData`オブジェクトを参照してください。

6.2.3.3.5 ログ出力

カスタムコンポーネントでは、下記のように`self.log`のメソッドを利用することで任意のタイミングでログを出力することができます。 詳細は`HiveLog`を参照してください。

```
self.log.info('create columns')
self.log.info(f'insert value={val} into {column_name}')
self.log.error('invalid data received')
```

出力したログは、画面からダウンロードすることができます。「各種ログ」を参照してください。

6.2.3.3.6 コンポーネントパラメータ

画面上からコンポーネントインスタンスを生成すると、その設定項目としてコンポーネントの実行時パラメータを設定することができます。 インスタンスごとにパラメータ設定を変えることで、同じコンポーネントベースの実装でも、コンポーネントの動作を変えることができます。



画面で指定する際、パラメータのタイプはSTRING、もしくは、JSONを選択できます。実装時に扱いやすいタイプを選択してください。

パラメータタイプ	説明
STRING	任意の文字列を設定できます。 コンポーネントの各メソッドの第二引数では`str`型として参照できます。
JSON	JSON文字列を設定できます。 コンポーネントの各メソッドの第二引数では`dict`型として参照できます。

コンポーネントインスタンスに設定したパラメータは、定義した`HiveComponent`クラスの`premain()`,`main()`,`postmain()`メソッドの第二引数として取得することができます。

```
@HiveComponentInfo(uuid='00000000-0000-0000-0000-000000000000', name='サンプルコンポ', inports=0,
outports=0)
class HiveComponent(HiveComponentBase):
    def premain(self, param):
        // use 'param' here.
        pass
    def postmain(self, param):
        // use 'param' here.
        pass
    def main(self, param):
        // use 'param' here.
        while ...
```

これらの扱いはカスタムコンポーネントの実装者次第です。 例として`sin_and_random`では、パラメータに数値文字列を`sin`カラムの波形の周波数として利用していますので参考にしてください。

6.2.3.4 カスタムコンポーネント用API

本項で紹介した関数以外にも、多数のAPI関数が用意されています。 詳細は付録の「カスタムコンポーネントPython-APIリファレンス」を参照してください。

6.2.3.5 カスタムコンポーネントを使用する

ここでは、作成したカスタムコンポーネントをSpeeDBee Synapseで使用する方法について説明します。

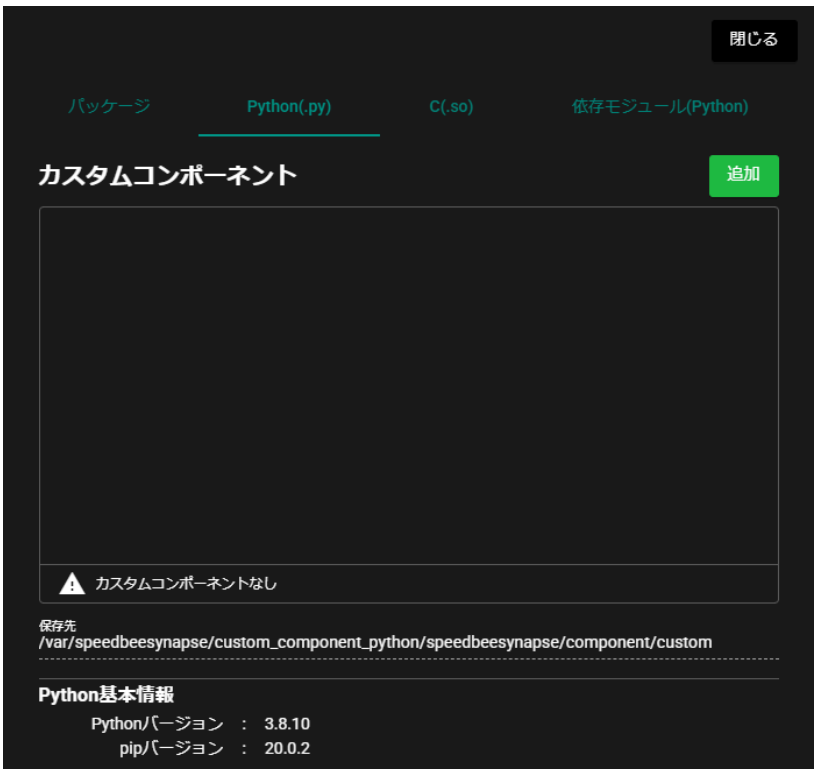
6.2.3.5.1 登録

次の手順で、作成したカスタムコンポーネントをSpeeDBee Synapseに登録することができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「Python(.py)」を選択します。



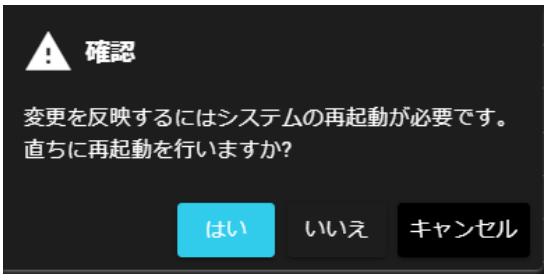
3. カスタムコンポーネントの「追加」を押下し、カスタムコンポーネントのファイルを登録します。



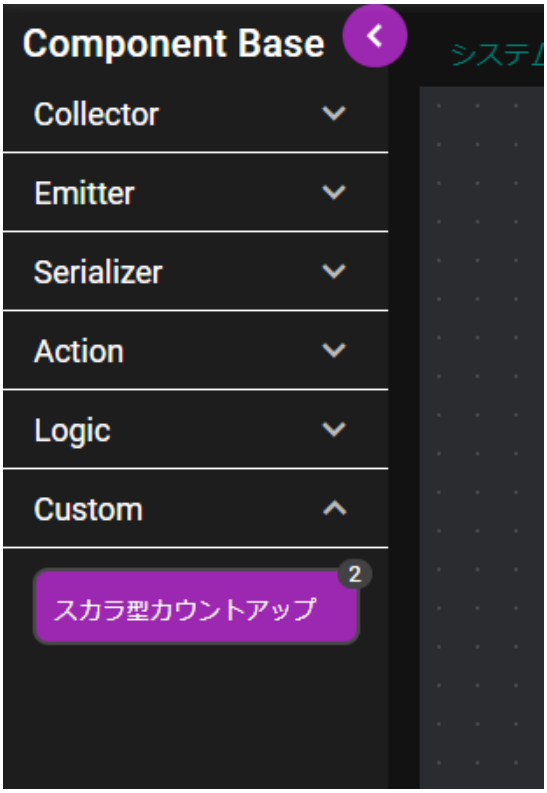
4. 「閉じる」を押下します。



5. 確認ダイアログで「はい」を選択し、変更を反映するために、システムを再起動させます。



6. 左メニューに、登録したカスタムコンポーネントが反映されます。



ファイルの保存先

登録したカスタムコンポーネントのファイルは、次のディレクトリに保存されます。

Linux

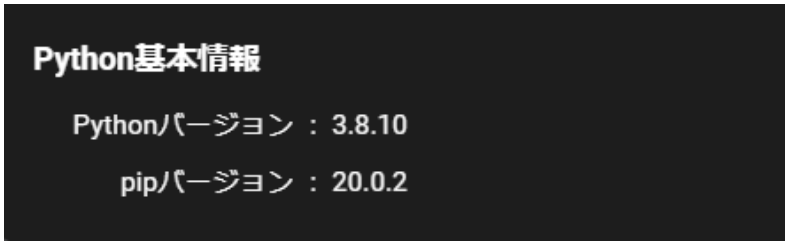
/var/speedbeesynapse/custom_component_python/speedbeesynapse/component/custom

保存先のディレクトリは、画面項目「保存先」に表示されます。



Pythonバージョン

カスタムコンポーネントを実行するPythonのバージョンは、画面項目「Python基本情報」の「Pythonバージョン」に表示されます。

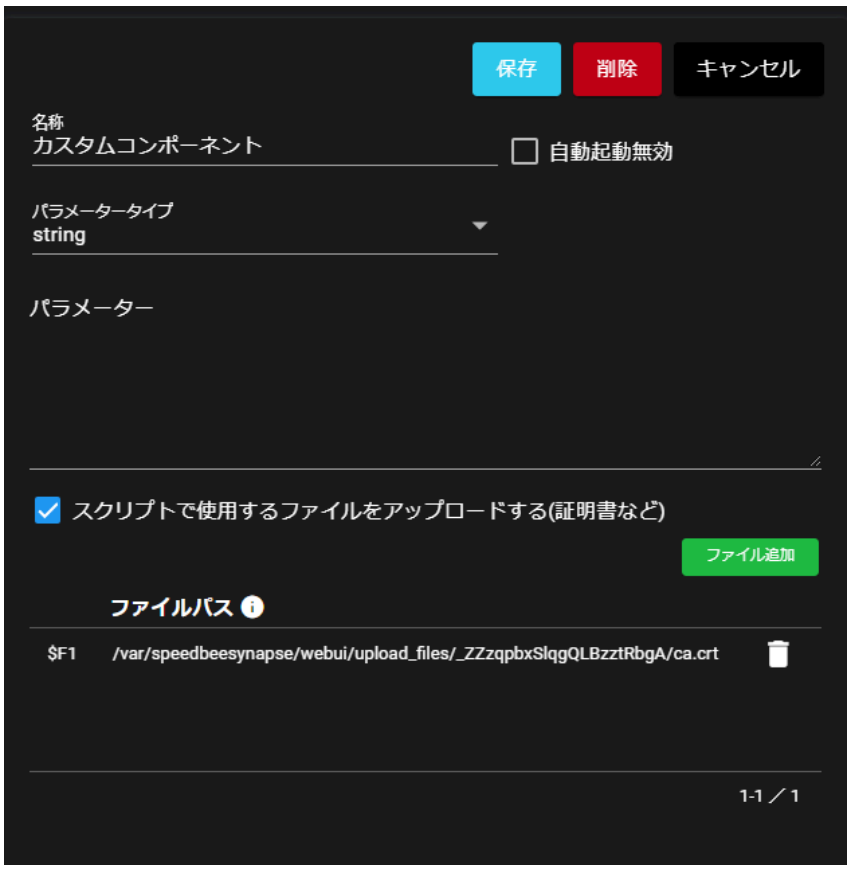


Pythonファイル名

カスタムコンポーネントはSpeeDBee Synapseではモジュールとして扱っており、ファイル名をモジュール名としています。 したがって、ファイル名に"-"を使用した場合正しく扱えません。
Pythonの命名規則に従いファイル名は短くすべて小文字で区切りは"_"を使用してください。

6.2.3.5.2 設定画面

カスタムコンポーネントの設定画面の項目は次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
パラメータータイプ	パラメータの種類を次の中から選択 ・STRING：文字列 ・JSON：JSON文字列
パラメーター	コンポーネントパラメータを入力
スクリプトで使用するファイルをアップロードする(証明書など)	カスタムコンポーネントの処理で使用する、証明書などのファイルをアップロードする場合ON
ファイル追加	押下すると、スクリプトで使用するファイルをアップロードすることができます。 ※新しく追加されたファイルは、コンポーネントの設定を保存した時にアップロードされます。

項目	説明
ファイルパス	アップロードしたファイルの保存先 ※パラメータでファイルパスを使用する場合、変数を用いることができます。詳細は、表題の「ファイルパス」の右のアイコンをクリックすると確認することができます。

設定画面のパラメータ入力部分に、ユーザーが定義した画面項目を表示するように変更することができます。 詳細は「カスタムUI作成」をご覧ください。

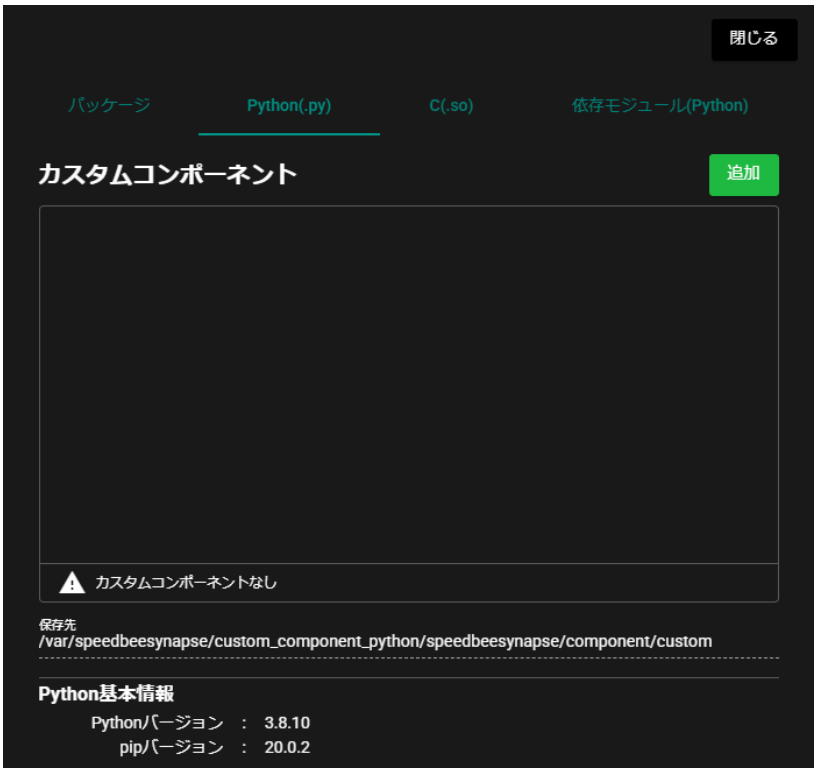
6.2.3.5.3 削除

次の手順で、登録したカスタムコンポーネントをSpeeDBee Synapseから削除することができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「Python(.py)」を選択します。



3. カスタムコンポーネントの「ゴミ箱アイコン（削除）」を押下します。



4. 確認ダイアログで「はい」を選択します。



5. カスタムコンポーネントが削除されます。

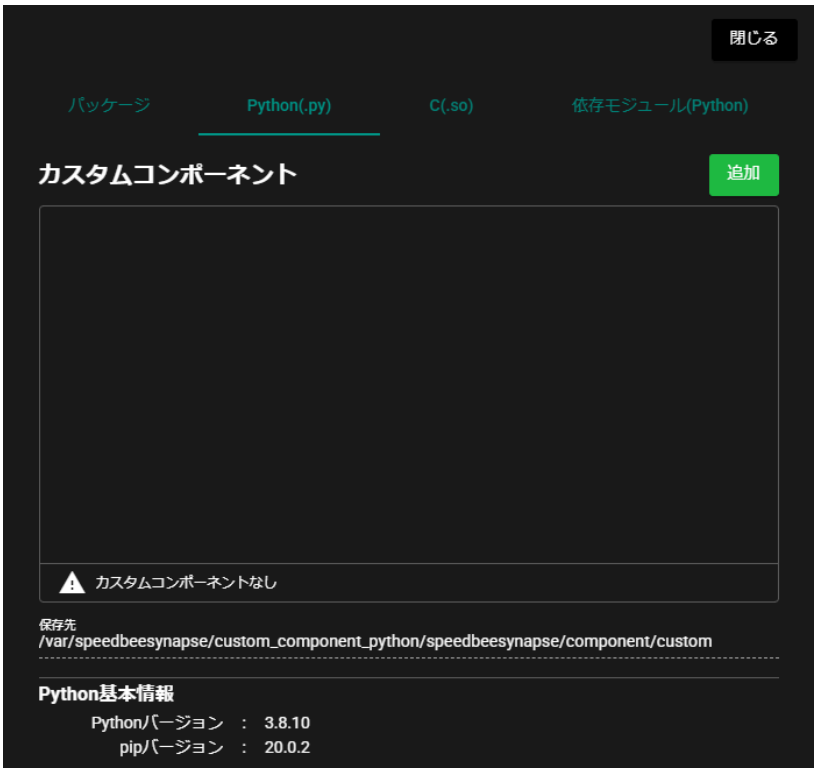
6.2.3.5.4 表示

次の手順で、登録したカスタムコンポーネントの内容を表示することができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「Python(.py)」を選択します。



3. カスタムコンポーネントの「ビジビリティアイコン（表示）」を押下します。



4. カスタムコンポーネントのソースコードが画面に表示されます。



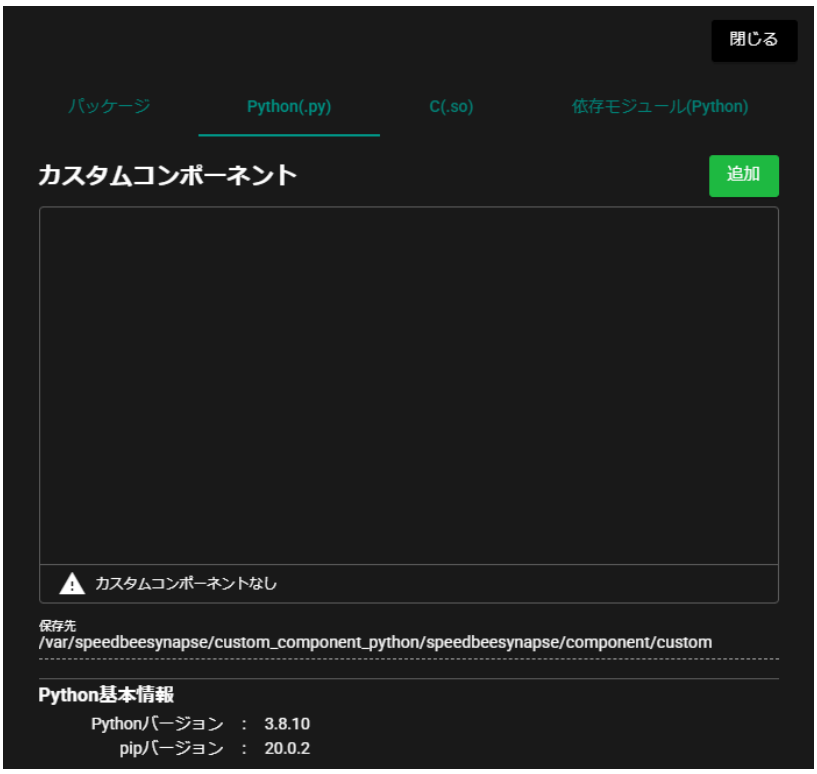
6.2.3.5.5 ダウンロード

次の手順で、登録したカスタムコンポーネントをダウンロードすることができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「Python(.py)」を選択します。



3. カスタムコンポーネントの「ダウンロードアイコン（取得）」を押下します。



4. カスタムコンポーネントのファイルがダウンロードされます。

6.2.3.5.6 保護の付与

登録したコンポーネントには保護をかけることができます。
保護されたコンポーネントは次の操作が行えなくなるため、意図しない更新を防ぐことができます。

- 登録
- 削除

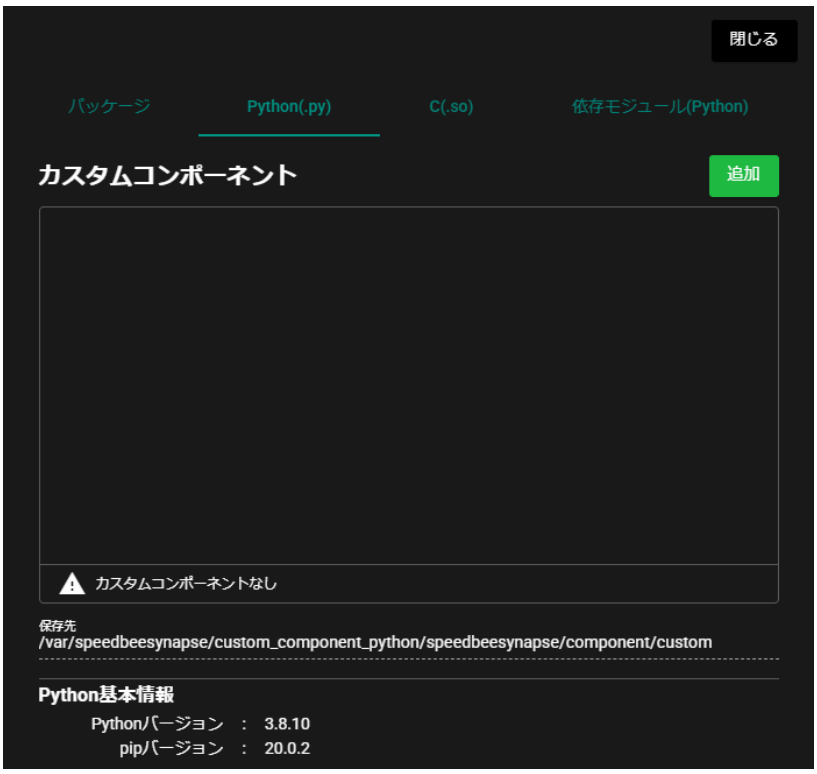
- 表示
- ダウンロード
- カスタムUIの設定

次の手順で、カスタムコンポーネントに保護をかけることができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「Python(.py)」を選択します。



3. カスタムコンポーネントの「鍵アイコン（保護）」を押下します。

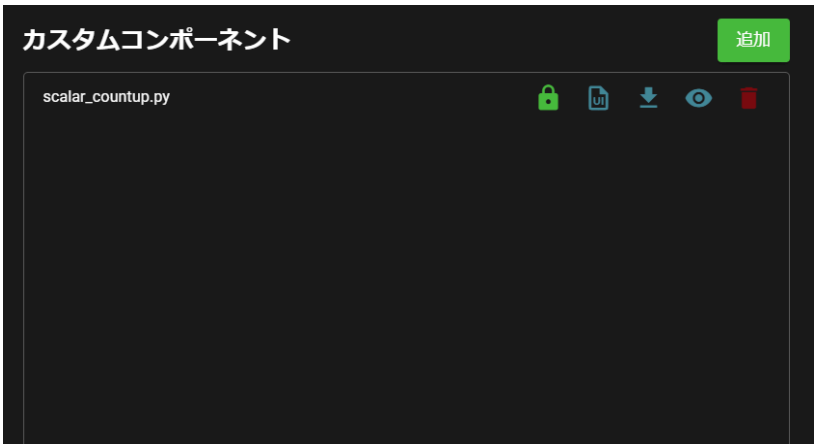


4. 保護解除する際に必要となるパスワードを入力して、「保護」を押下します。



> **警告**
>
> - パスワードは復元できない形式で保存され、一度設定すると保護を解除するまで再設定できません。
> - 保護設定したパスワードを忘れた場合、該当のカスタムコンポーネントは操作不能となります。
>
> パスワードの取り扱いには十分ご注意ください。

5. カスタムコンポーネントが保護状態になります。



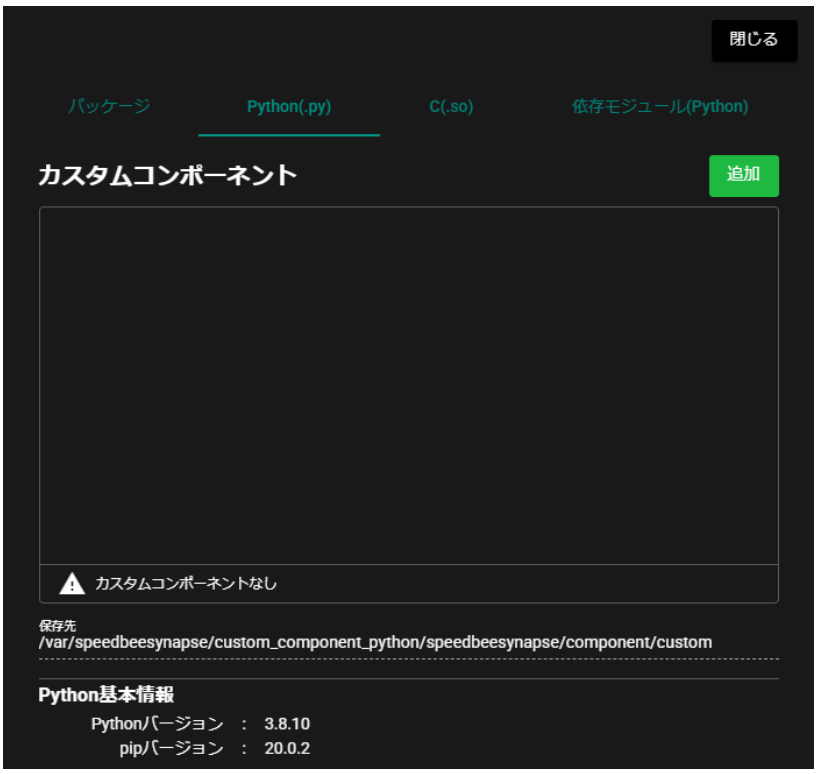
6.2.3.5.7 保護の解除

次の手順で、保護されたカスタムコンポーネントの保護を解除することができます。

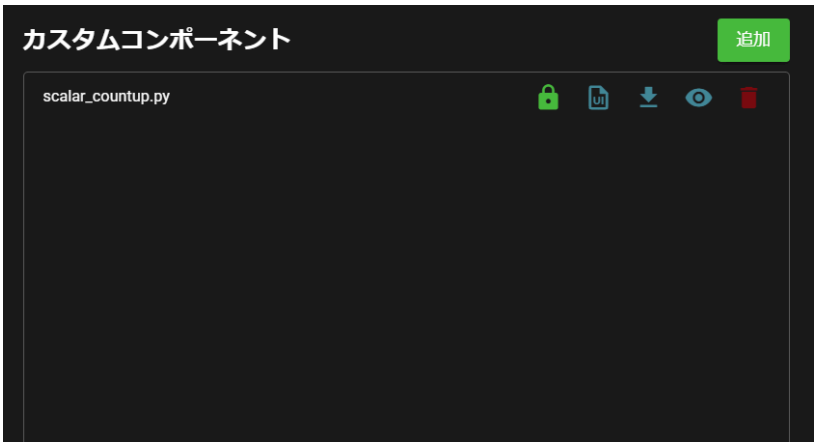
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「Python(.py)」を選択します。



3. カスタムコンポーネントの「鍵アイコン（保護解除）」を押下します。



4. 保護をかけた際に設定したパスワードを入力して、「保護解除」を押下します。



5. カスタムコンポーネントの保護が解除されます。



6.2.3.5.8 外部ライブラリの追加

次の手順で、カスタムコンポーネントが使用する外部ライブラリを、SpeeDBee Synapseに追加することができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「依存モジュール(Python)」を選択します。



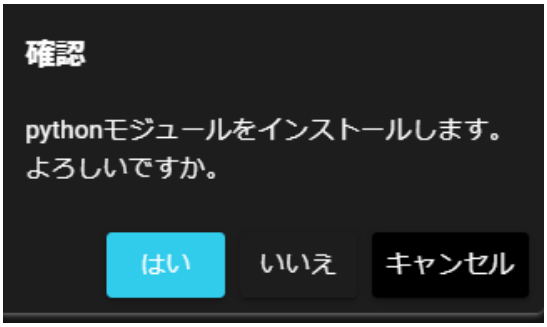
3. Pythonモジュールの入力欄に、登録するライブラリの名前を入力します。



4. Pythonモジュールの「追加」を押下します。



5. 確認ダイアログで「はい」を選択します。



6. ライブラリが登録されます。

外部ライブラリの保存先

登録した外部ライブラリは、次のディレクトリに保存されます。

Linux

/var/speedbeesynapse/dynlibs/pyvenv

保存先のディレクトリは、画面項目「保存先」に表示されます。



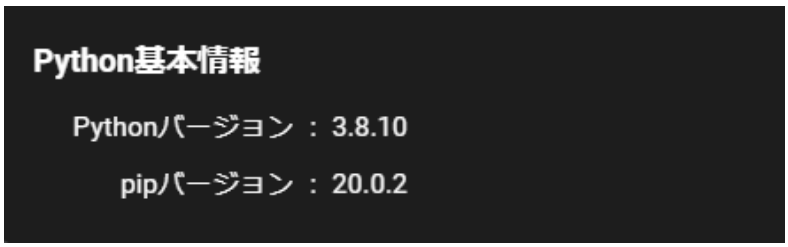
.whlファイルから追加する

Pythonモジュールの「whl」を押下し、.whlファイルを選択します。



pipバージョン

外部ライブラリの管理で使用するpipのバージョンは、画面項目「Python基本情報」の「pipバージョン」に表示されます。



6.2.3.5.9 外部ライブラリの削除

次の手順で、追加した外部ライブラリを、SpeeDBee Synapseから削除することができます。

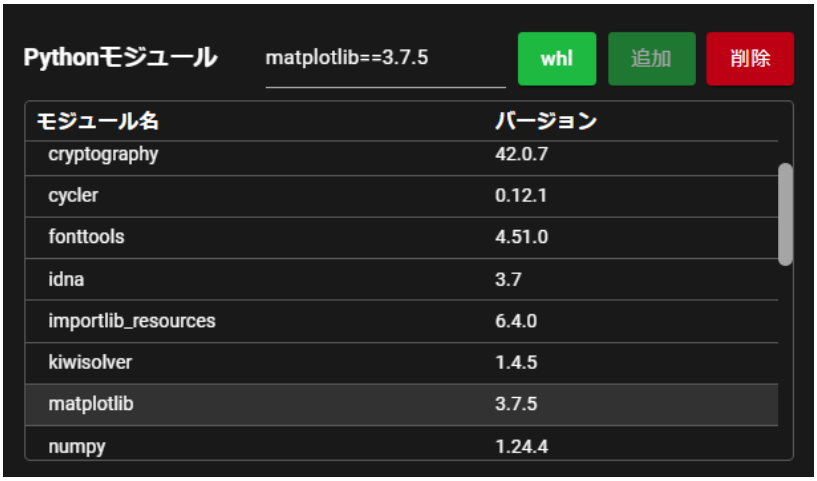
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



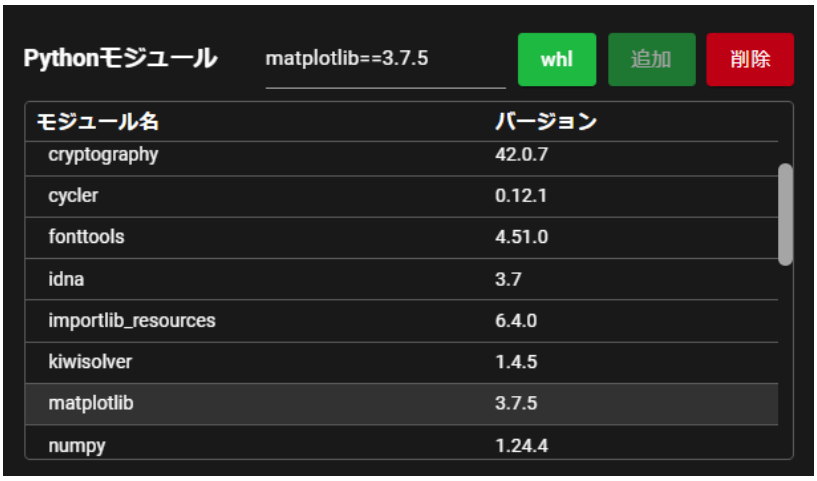
2. ダイアログ上部のタブの「依存モジュール(Python)」を選択します。



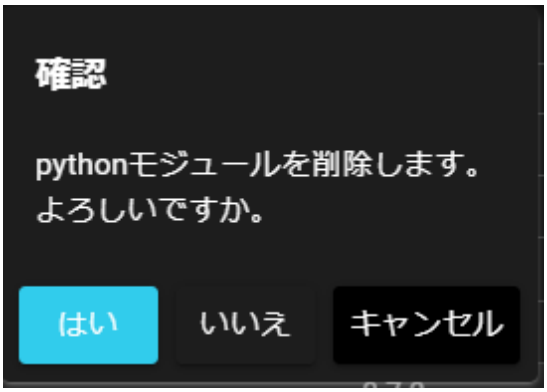
3. 削除するライブラリをクリックします。



4. Pythonモジュールの「削除」を押下します。



5. 確認ダイアログで「はい」を選択します。



6. ライブラリが削除されます。

6.2.4 C言語カスタムコンポーネント作成

本項では、C言語によるカスタムコンポーネントの作成方法について説明します。

6.2.4.1 前提知識

本項の内容は以下に記す知識・経験を前提としています。

- C言語を用いた一般的なプログラムの開発経験
- C言語の開発に必要なツール類の準備
- Makefileによるビルドの実行方法

また、本項ではSpeeDBee Synapseが稼働する環境内でCのプログラムをビルドする前提で記載しています。
SpeeDBee Synapseが稼働する環境とは別の環境でプログラムをビルドして移動する場合は、その環境に応じてクロスビルドが必要になります。クロスビルドの方法は稼働する環境のマニュアル等をご確認ください。

6.2.4.2 サンプルカスタムコンポーネント

C言語によるカスタムコンポーネントの実装方法説明のため、シンプルな仕様のカスタムコンポーネントをサンプルとして提供しています。

サンプルカスタムコンポーネントのダウンロード

もしくは、SpeeDBee Synapseをインストール済みの場合、カスタムコンポーネントサンプルも下記の場所にインストールされています。

```
$ ls /usr/local/speedbeesynapse/share/sample_component_c/*
/usr/local/speedbeesynapse/share/sample_component_c/cmake_project:
CMakeLists.txt  README.md  countandrandom.c  readandlog.c

/usr/local/speedbeesynapse/share/sample_component_c/makefile_project_for_linux:
Makefile  README.md  countandrandom.c  readandlog.c

/usr/local/speedbeesynapse/share/sample_component_c/makefile_project_for_windows:
Makefile  README.md  countandrandom.c  readandlog.c
```

サンプルカスタムコンポーネントのディレクトリは、ビルド環境に合わせて下記の３種類用意されています。

- `cmake_project`

  CMakeを使ってビルドするためのプロジェクトディレクトリです。 Linux/Windowsどちらも同じ手順でビルドできます。

- `makefile_project_for_linux`

  Makeを使ってビルドするためのLinux環境専用プロジェクトディレクトリです。

- `makefile_project_for_windows`

  WindowsのNMAKEを使ってビルドするためのプロジェクトディレクトリです。

6.2.4.2.1 サンプルコンポーネントのビルド

各プロジェクトディレクトリでのビルド方法は、それぞれのREADME.mdファイルを参照してください。 どのプロジェクトディレクトリも、ビルドして生成されるコンポーネントは同一のものになります。

SpeeDBee Synapseを実行する環境がWindowsの場合、`cmake_project`もしくは`makefile_project_for_windows`のどちらかを選択してください。
実行する環境がLinuxの場合、`cmake_project`もしくは`makefile_project_for_linux`のどちらかを選択してください。

6.2.4.2.2 サンプルの内容

カスタムコンポーネントのサンプルとして本項では下記のサンプルを説明します。

- `count_and_random`(countandrandom.c)

  出力ポートに対して以下の２つのカラムを作成し、データを出力し続けるコンポーネントです。

カラム名	内容
count	３秒に１回、1から順にカウントアップした値を登録していきます
random	３秒に１回、[0.0, 100.0)の範囲内の値をランダムに登録します

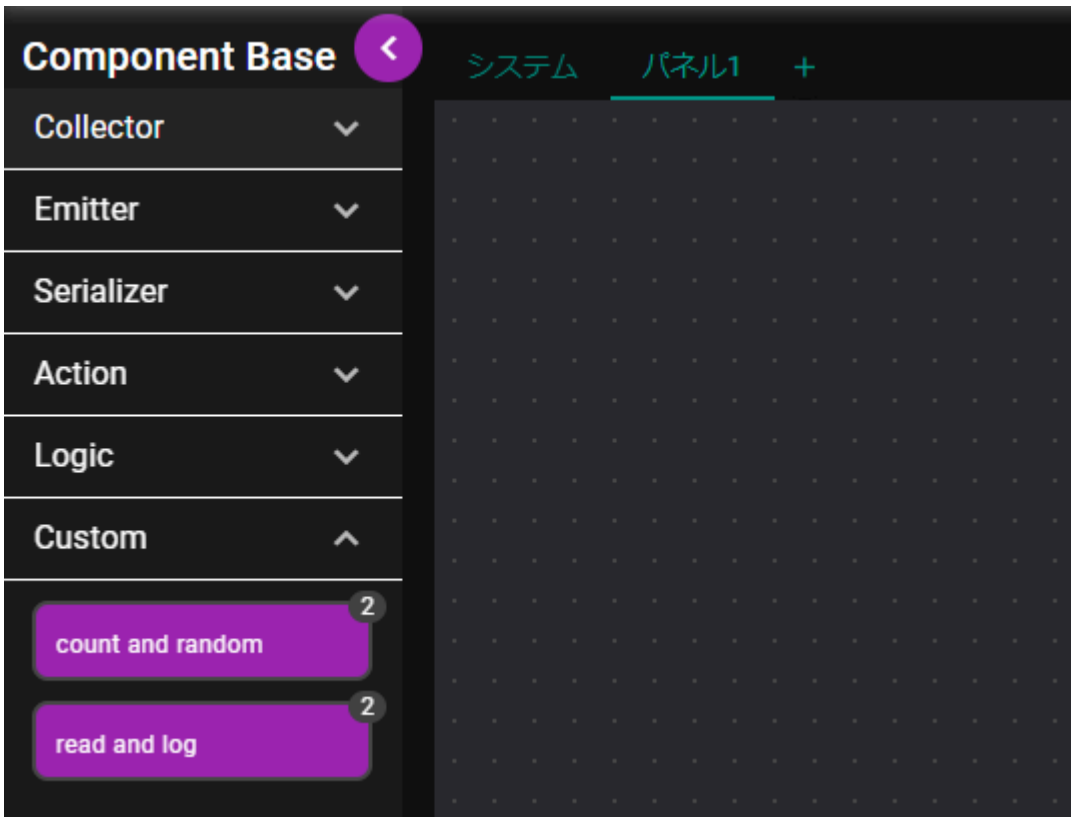
- `read_and_log`(readandlog.c)

  入力ポートからデータを受け取り、それを文字列型カラムとログに出力します。

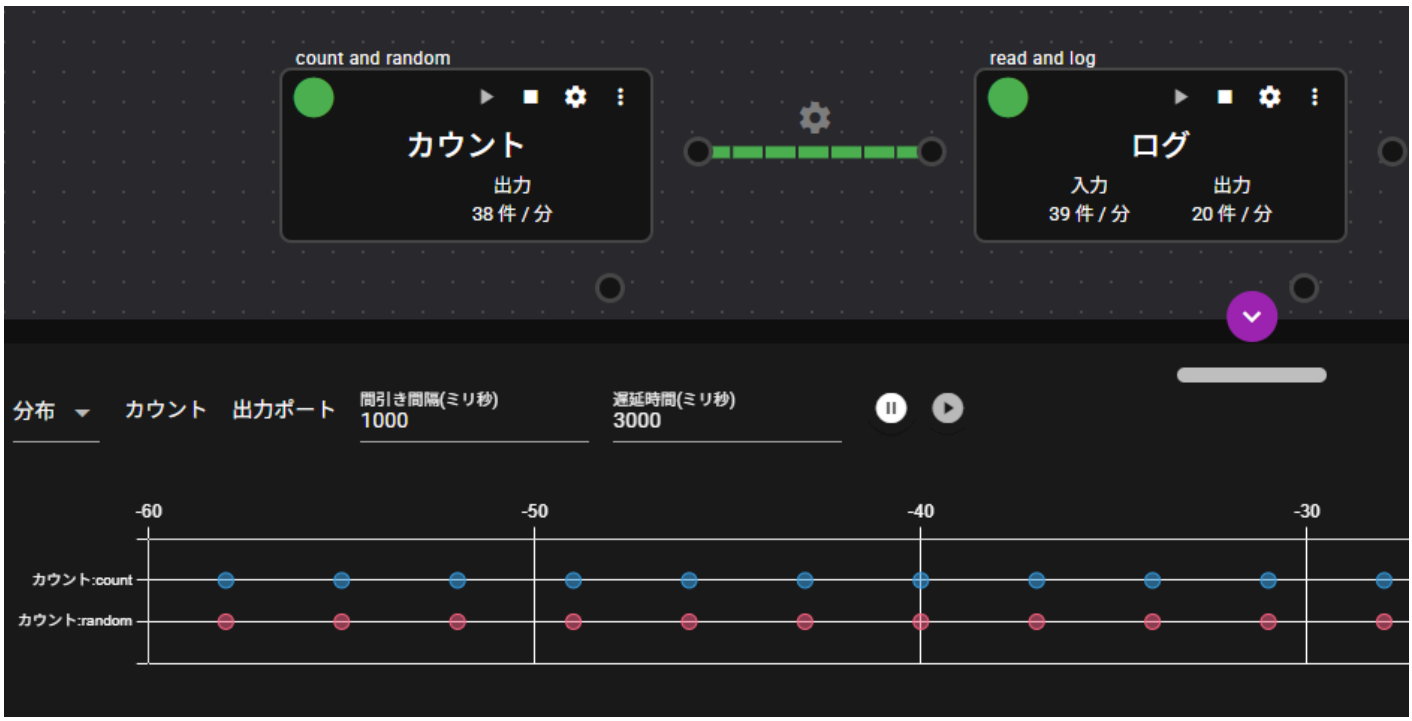
6.2.4.2.3 実行

サンプルコンポーネントのビルドに成功すると、Linuxなら`.so`ファイルが、Windowsなら`.dll`ファイルがコンポーネントごとに生成されます。 生成された`.so` or `.dll`ファイルをWEBUIから登録することで、SpeeDBee Synapseからコンポーネントとして利用できるようになります。 登録手順は カスタムコンポーネントを使用する を参照してください。

ファイルを登録してコアが再起動されると、SpeeDBee Synapseの画面から、他のコンポーネントと同様に新しいカスタムコンポーネントを使用できるようになっています。



`count_and_random`、および、`read_and_log`を繋いで実行することで、それぞれの出力したカラムを表示できます。 下記は`count_and_random`の出力ポートをモニタリングしています。



6.2.4.3 カスタムコンポーネント実装詳細

以降では、サンプルの`count_and_random`、および、`read_and_log`のソースコードを例に、カスタムコンポーネントの実装内容を説明していきます。

6.2.4.3.1 インクルードヘッダ

Cによるカスタムコンポーネントの開発では、ヘッダファイル`hiveframework.h`をインクルードする必要があります。

```
#include <hiveframework.h>

 :  以降、カスタムコンポーネントの実装
```

6.2.4.3.2 カスタムコンポーネント情報定義

Cによるカスタムコンポーネントのソースコードでは、下記のグローバル変数定義により、そのコンポーネントの名前や識別情報、ライフサイクルコールバック関数の定義が必要になります。

```
  :  カスタムコンポーネントの各種関数の定義

const static HIVE_COMPONENT_DEFINITION info = {
  .uuid            = "2e80ad02-1730-4bdc-b321-2fbc2d1b942e",
  .name            = "count and random",
  .parameter_type  = HIVE_COMPONENT_PARAMETER_NONE,
  .in_ports        = 0,
  .out_ports       = 1,
  .functions     = {
    .constructor         = NULL,      // optional
    .premain             = premain,   // optional
    .main                = mainloop,  // required
    .postmain            = postmain,  // optional
    .destructor          = NULL,      // optional
    .stop                = NULL,      // optional
  },
};
EXPORT_COMPONENT_DEFINITION(info);
```

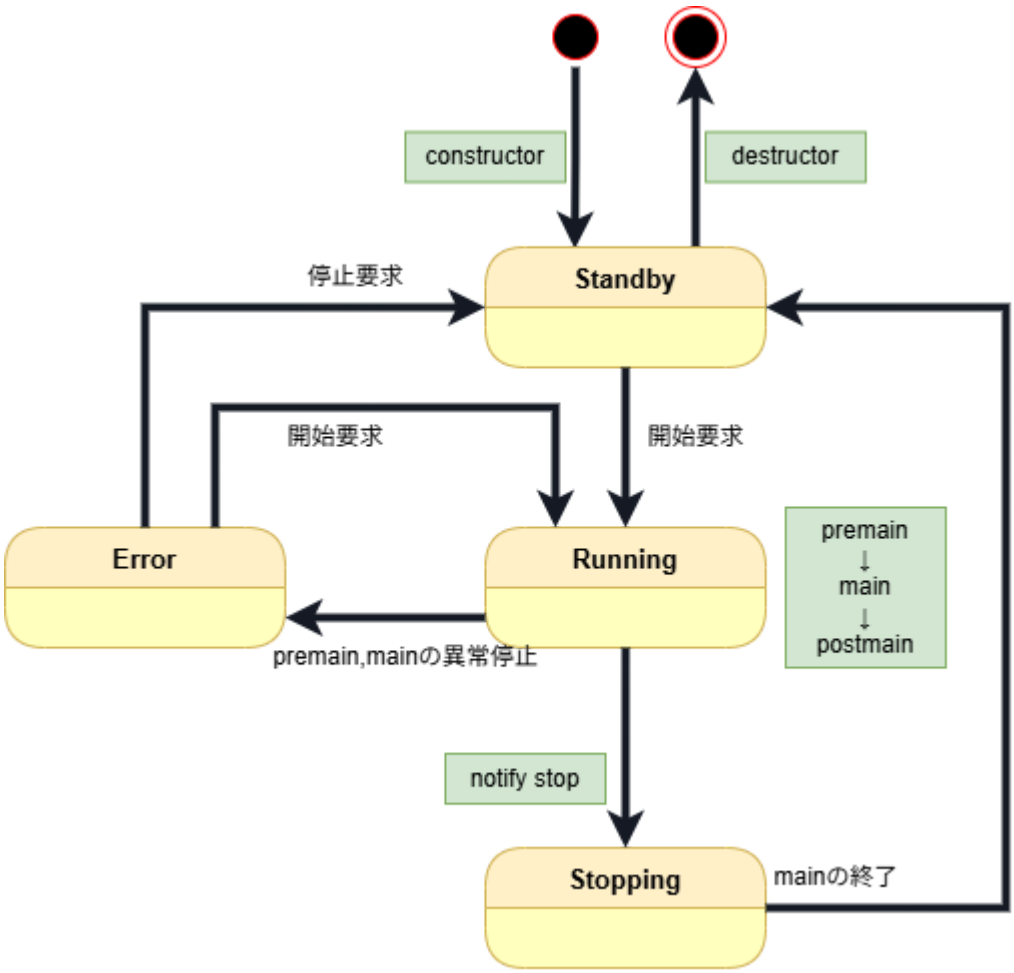
メンバ名	定義情報	説明
`uuid`	UUID	コンポーネントを識別するためのUUIDです。ランダムなIDを生成してここに設定してください。
`name`	コンポーネント名	コンポーネントの名前を設定します。画面にはこの名前が表示されますのでなるべく他のコンポーネントと被らない方が良いです。

メンバ名	定義情報	説明
`tag`	タグ	コンポーネントへタグを付与します。"collector","emitter","serializer","action","logic"のいずれかの指定によりコンポーネントベースの配置先を指定できます。省略した場合は、Customへ配置されます。
`parameter_type`	パラメータ種別	このコンポーネントに渡すパラメータの形式をNONE, TEXT, JSONから選択できますが、現状はこの項目は使われていません。
`out_ports`	出力ポート数	このコンポーネントが持つ出力ポートの数を設定します。他のコンポーネントの入力ポートにデータを渡す場合はこれを1としてください。
`in_ports`	入力ポート数	このコンポーネントが持つ入力ポートの数を設定します。他のコンポーネントの出力ポートからデータを受け取る場合は、これを1としてください。
`functions`	コールバック関数リスト	このコンポーネントの動作を実装した関数のリストを設定します。次節参照。

UUIDについては必ずランダムなIDを生成してください。 他のコンポーネントと同じUUIDにしてしまうと、どちらのコンポーネントも使えなくなることがあります。

6.2.4.3.3 ライフサイクルコールバック定義

前節の`functions`メンバは、コンポーネントインスタンスのライフサイクルに基づいて コールされる関数ポインタを設定することができます。



コールバック関数	説明
`constructor`	コンポーネントのインスタンスが生成されたときにコールされます
`destructor`	コンポーネントのインスタンスが破棄されるときにコールされます
`premain`	コンポーネントを開始するときにmainの前にコールされます
`main`	コンポーネントのメインの処理を実装する関数です コンポーネントの終了要求を受けるまで、この関数は終了せずにループ処理し続ける必要があります
`postmain`	コンポーネントの終了時、mainの終了直後にコールされます
`stop`	mainの実行中、コンポーネントの停止要求を受けたときに実行されます

前節の `functions` 定義は、サンプルコンポーネントの `count_and_random` の末尾に定義されています。

このコンポーネントでは`constructor`, `destructor`は不要なため、省略（NULL指定）しています。`premain`, `postmain`も不要であれば省略することができます。（このサンプルでも、実際にはログ出力しか実装されていません）

`main`は、コンポーネントの処理の本体になりますので、必須です。 実際の関数定義名は`mainloop`など、別の名前にしてください。`main`だとCの実行形式本体の`main`関数と衝突してしまいます。

以降ではサンプルプログラムの`main`関数の実装内容を説明します。

6.2.4.3.4 出力ポートのカラム作成/データ登録（`count_and_random`）

下記はカスタムコンポーネント`count_and_random`のメインの処理です。 これをもとにデータを出力する流れを説明します。

```
static bool mainloop(HIVE_COMPONENT comp, const char *param, HIVE_STATUS status) {
  HIVE_LOG_INFO("create columns");
  HIVE_OUTCOLUMN clm_count = hive_outport_create_column(comp, OUTPORT1, "count",
HIVE_DATA_SCALAR(HIVE_TYPE_INT32), HIVE_COLUMN_OPTION_NONE);
  HIVE_OUTCOLUMN clm_random = hive_outport_create_column(comp, OUTPORT1, "random",
HIVE_DATA_SCALAR(HIVE_TYPE_DOUBLE), HIVE_COLUMN_OPTION_NONE);

  int count = 1;
  while (hive_component_runnable(comp)) {
    HIVE_LOG_INFO("insert count");
    if (!hive_outcolumn_insert(clm_count, &count)) {
      HIVE_API_ERROR err = hive_get_api_error();
      HIVE_LOG_ERROR("insert 'count' failed: errcode=0x%08x", err.code);
    }

    HIVE_LOG_INFO("insert random");
    double r = 100 * (double)rand() / (double)RAND_MAX;
    if (!hive_outcolumn_insert(clm_random, &r)) {
      HIVE_API_ERROR err = hive_get_api_error();
      HIVE_LOG_ERROR("insert 'count' failed: errcode=0x%08x", err.code);
    }

    usleep(3000000);
    count++;
  }

  return true;
}
```

1. カラムの作成

   初めに、カラムを作成しています。

   ```
   HIVE_OUTCOLUMN clm_count = hive_outport_create_column(comp, OUTPORT1, "count",
   HIVE_DATA_SCALAR(HIVE_TYPE_INT32), HIVE_COLUMN_OPTION_NONE);
   HIVE_OUTCOLUMN clm_random = hive_outport_create_column(comp, OUTPORT1, "random",
   HIVE_DATA_SCALAR(HIVE_TYPE_DOUBLE), HIVE_COLUMN_OPTION_NONE);
   ```

このコンポーネントでは32bit整数のカラム `count` と倍精度浮動小数点のカラム `random` を利用しますので、`hive_outport_create_column()` 関数を使用して引数にカラム名、データ型等を指定して、カラムを生成しています。 戻り値は後にカラムにデータを登録する際に使用しますので、変数に保持しておいてください。

コンポーネントが出力するすべてのデータは、このように生成したカラムに対して登録していく必要があります。

2. 定期的な処理の繰り返し

   定期的にデータを登録するため、whileループで繰り返し処理を実装しています

```
int count = 1;
while (hive_component_runnable(comp)) {
  :

  usleep(3000000);
  count++;
}
```

   ループの判定条件で呼び出している `hive_component_runnable()` は、コンポーネントが実行状態の間はtrueを返します。画面からコンポーネントの停止を要求すると、この関数がfalseを返すようになりますので、その場合には速やかにこの `mainloop` 関数を終わらせる必要があります。

   繰り返し処理にはこのようなwhileループの他に、 `hive_component_interval_call()` を使用することもできます。 使用方法はリンク先を参照してください。

3. カラムへのデータ登録

   繰り返し処理の中で、カラムに対してデータを登録します。

```
if (!hive_outcolumn_insert(clm_count, &count)) {
  HIVE_API_ERROR err = hive_get_api_error();
  HIVE_LOG_ERROR("insert 'count' failed: errcode=0x%08x", err.code);
}

double r = 100 * (double)rand() / (double)RAND_MAX;
if (!hive_outcolumn_insert(clm_random, &r)) {
  HIVE_API_ERROR err = hive_get_api_error();
  HIVE_LOG_ERROR("insert 'count' failed: errcode=0x%08x", err.code);
}
```

   データの登録は `hive_outcolumn_insert()` で行えます。 第一引数には事前に作成したカラム変数を、第二引数には登録するデータのポインタを指定します。 このときのポインタは、事前にカラムを作成した時に指定したデータ型と合わせる必要があります。異なる型のデータを指定すると、ソフトウェアが異常終了する原因となりえますので注意してください。

> **insertで登録できる最大サイズ**
>
> hive_outcolumn_insert()で登録できるデータの最大サイズは、512バイト（STRINGはNULLを含まない511バイト）です。上限サイズを超えた値の登録は、戻り値がfalseとなります。

6.2.4.3.5 入力ポートからのデータ読み込み（`read_and_log`）

下記はカスタムコンポーネント`read_and_log`のメインの処理です。（コメントやログ出力は省略）

```
static bool mainloop(HIVE_COMPONENT comp, const char *param, HIVE_STATUS status) {
  HIVE_OUTCOLUMN clm_log = hive_outport_create_column(comp, OUTPORT1, "log",
HIVE_DATA_SCALAR(HIVE_TYPE_STRING), HIVE_COLUMN_OPTION_NONE);

  HIVE_CONTINUOUS_READER *creader
    = hive_inport_continuous_reader(comp, INPORT1, hive_timestamp()+5000000000UL);

  hive_time_t last_updated = 0;
  while (hive_component_runnable(comp)) {
    const HIVE_COLUMN_READ_RESULT *read_result = hive_continuous_reader_read(creader);
    if (!read_result) {
      continue;
    }

    if (last_updated < read_result->incolumn_list->updated_at) {
      last_updated = read_result->incolumn_list->updated_at;
    }

    HIVE_RECORD_ITERATOR iter = HIVE_GET_RECORD_ITERATOR(read_result);
    const HIVE_RECORD *record;
    while ((record = HIVE_RECORD_ITERATOR_GET_NEXT(&iter)) != NULL) {
      for (int i=0; i < record->data_count; i++) {
        const HIVE_RECORD_DATA *rd = HIVE_RECORD_GET_DATA(record, i);
        if (rd && rd->data_size != 0) {
          log_record(clm_log, record->timestamp, &read_result->incolumn_list->incolumns[i], rd);
        }
      }
    }
  }
  hive_continuous_reader_release(creader);

  return true;
}
```

1. カラムの作成

   このコンポーネントでも、初めにカラムを作成しています。

   ```
   HIVE_OUTCOLUMN clm_log = hive_outport_create_column(comp, OUTPORT1, "log",
   HIVE_DATA_SCALAR(HIVE_TYPE_STRING), HIVE_COLUMN_OPTION_NONE);
   ```

   このカラムは文字列を出力するためのものです。

2. データ取得ループ

   入力ポートからのデータを繰り返し受け取る処理が下記になります。

```
HIVE_CONTINUOUS_READER *creader
  = hive_inport_continuous_reader(comp, INPORT1, hive_timestamp()+5000000000UL);

:
while (hive_component_runnable(comp)) {
  const HIVE_COLUMN_READ_RESULT *read_result = hive_continuous_reader_read(creader);
  if (!read_result) {
    continue;
  }

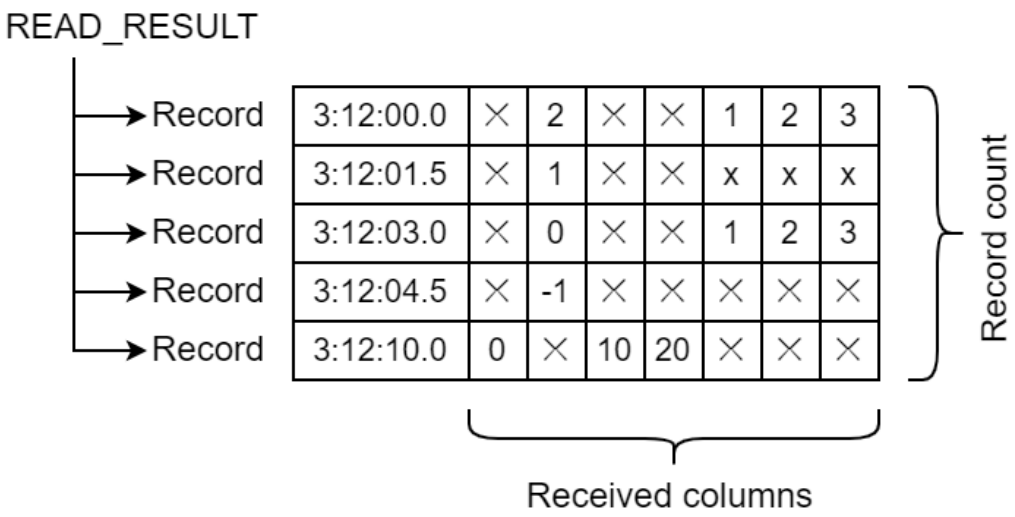
  :
}
hive_continuous_reader_release(creader);
```

ここでは４つの関数を利用しています

- `hive_inport_continuous_reader()`
  継続的に入力ポートからのデータを取得するためのハンドルを作成します
- `hive_continuous_reader_release()`
  データの取得を停止する場合、この関数でハンドルを解放します
- `hive_continuous_reader_read()`
  作成したハンドルからデータを取得します。取得したデータは`HIVE_COLUMN_READ_RESULT`構造体のポインタとして返されますが、データがまだない場合には`NULL`が返されることもありますので、その場合は無視してください。
- `hive_component_runnable()`
  こちらは前節でも使用しました。 コンポーネントが実行状態の間はtrueを返します。画面からコンポーネントの停止を要求すると、この関数がfalseを返すようになりますので、その場合には速やかにこの`mainloop`関数を終わらせる必要があります。

3. 取得したデータのレコードを参照

`hive_continuous_reader_read()`関数により、入力ポートに入ってきたデータをある時間幅分まとめて取得し、`HIVE_COLUMN_READ_RESULT`構造体のポインタ、`read_result`に格納します。`HIVE_COLUMN_READ_RESULT`構造体は、下図のように同一時刻のデータを収めた複数のレコードからなり、１つのレコードに複数のカラムのデータが収められています。



下記のような二重ループにより、この構造体から１レコード、１データずつ処理することができます。
`HIVE_GET_RECORD_ITERATOR ()`と`HIVE_RECORD_ITERATOR_GET_NEXT`がレコードのループ、その内部にある変数`i`によるfor文がレコード内のデータ処理の繰り返しです

```
HIVE_RECORD_ITERATOR iter = HIVE_GET_RECORD_ITERATOR(read_result);
const HIVE_RECORD *record;
while ((record = HIVE_RECORD_ITERATOR_GET_NEXT(&iter)) != NULL) {
  for (int i=0; i < record->data_count; i++) {
    const HIVE_RECORD_DATA *rd = HIVE_RECORD_GET_DATA(record, i);
    if (rd && rd->data_size != 0) {
      // ここでrdが１カラム分のデータを保持している
      log_record(clm_log, record->timestamp, &read_result->incolumn_list->incolumns[i], rd);
    }
  }
}
```

4. レコード内の１データの扱い

   １つのデータの情報は、`HIVE_RECORD`構造体と`HIVE_COLUMN_READ_RESULT`構造体の`incolumn_list`メンバから取得することができます。

項目	型	本サンプルプログラムでの参照例	内容
データ名	const char*	`read_result->incolumn_list->incolumns[i]->data_name`	データを登録しているカラム名
コンポーネント名	const char*	`read_result->incolumn_list->incolumns[i]->source_name`	データを生成したコンポーネント名
データ型	HIVE_DATA_TYPE	`read_result->incolumn_list->incolumns[i]->data_type`	データを登録しているカラムのデータ型
データサイズ	uint32_t	`HIVE_RECORD_GET_DATA(record, i)->data_size`	実データのデータサイズ
データアドレス	char *	`HIVE_RECORD_GET_DATA(record, i)->data`	実データを格納したアドレス

   上記の通り、実データは`const char`ポインタとなっています。様々なデータ型を受ける可能性があるため、`HIVE_DATA_TYPE`で識別してそれぞれの型に応じてキャストして参照する必要が有ることにご注意ください。

6.2.4.3.6 ログ出力

コンポーネントでは任意のタイミングでログを出力することができます。

```
  HIVE_LOG_INFO("create columns");
  HIVE_LOG_INFO("x=%d, y=%f, z=%s", x, 12.34, "aiueo");
  HIVE_LOG_ERROR("insert 'count' failed: errcode=0x%08x", err.code);
```

上記のように、標準ライブラリの`printf()`と同じ形式でログを出力できます。 出力したログは、画面からダウンロードすることができます。 「各種ログ」を参照してください。

6.2.4.3.7 コンポーネントパラメータ

画面上からコンポーネントインスタンスを生成すると、その設定項目としてコンポーネントの実行時パラメータを設定することができます。



パラメータの種別としてはJSONでもSTRINGでも利用可能です。 どちらを指定した場合でも、Cで実装されたコンポーネントにおいては、main関数の第二引数に`const char*`型の文字列として受け渡されますので、これを標準ライブラリのsscanfで解析したり、直接文字列として扱ったりすることが可能です。

```
static bool mainloop(HIVE_COMPONENT comp, const char *param, HIVE_STATUS status) {
  int val1, val2;
  sscanf(param, "%d,%d", &val1, &val2); // perform parameter as a number string

  // some process using 'val1' and 'val2'.
}
```

JSON形式のパラメータを扱う場合も、paramに文字列としてJSON形式が格納されますので、これをパースする必要があります。 JSONの扱いは本システムのAPIとしては用意していませんので、汎用的なライブラリの使用を検討してください。

参考

- json-c
- jansson

6.2.4.4 カスタムコンポーネント用API

本項で紹介した関数以外にも、多数のAPI関数が用意されています。 詳細は付録のカスタムコンポーネントC-APIリファレンスを参照してください。

6.2.4.5 カスタムコンポーネントを使用する

ここでは、作成したカスタムコンポーネントをSpeeDBee Synapseで使用する方法について説明します。

6.2.4.5.1 登録

次の手順で、ビルドしたカスタムコンポーネントをSpeeDBee Synapseに登録することができます。

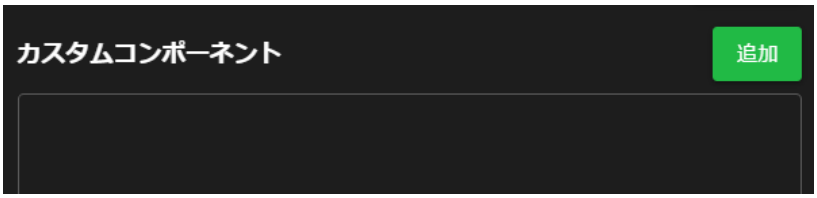
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



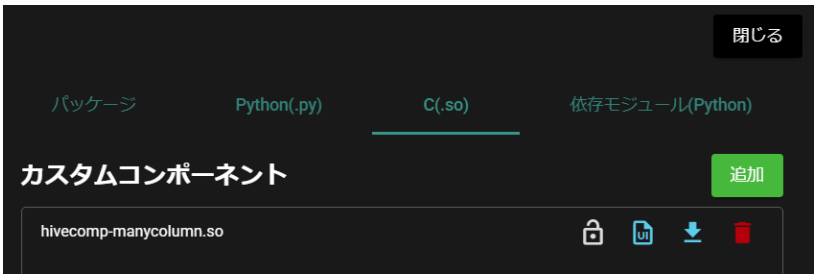
2. ダイアログ上部のタブの「C(.so)」を選択します。



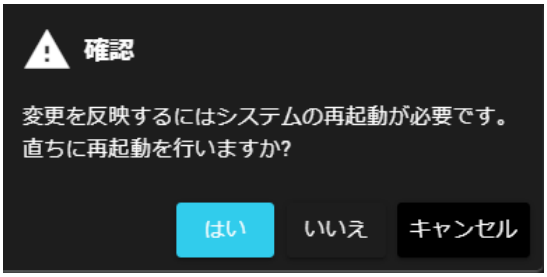
3. 「追加」を押下し、ビルド済みのカスタムコンポーネントのsoファイルを登録します。



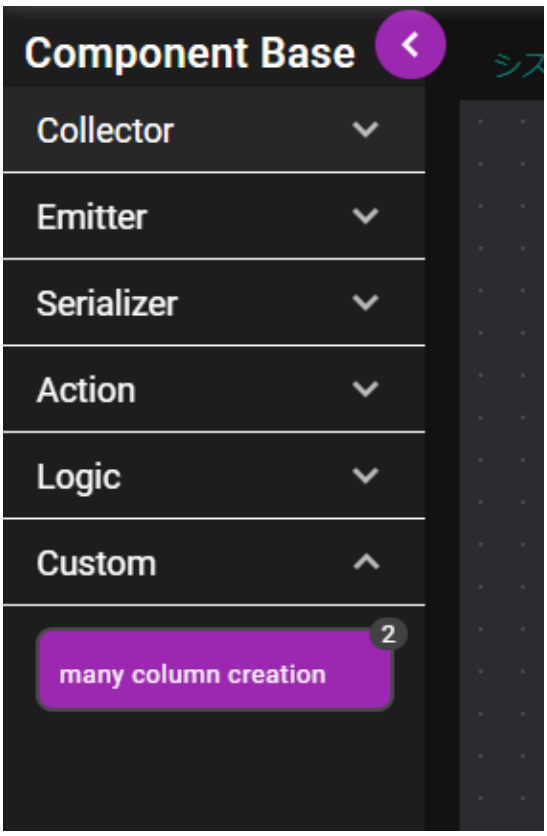
4. 「閉じる」を押下します。



5. 確認ダイアログで「はい」を選択し、変更を反映するために、システムを再起動させます。



6. 左メニューに、登録したカスタムコンポーネントが反映されます。



ファイルの保存先

登録したカスタムコンポーネントのファイルは、次のディレクトリに保存されます。

Linux

/var/speedbeesynapse/custom_component_so

保存先のディレクトリは、画面項目「保存先」に表示されます。



6.2.4.5.2 設定画面

カスタムコンポーネントの設定画面の項目は次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合ON
パラメータータイプ	パラメータの種類を次の中から選択 ・STRING：文字列 ・JSON：JSON文字列
パラメーター	コンポーネントパラメータを入力
スクリプトで使用するファイルをアップロードする(証明書など)	カスタムコンポーネントの処理で使用する、証明書などのファイルをアップロードする場合ON

項目	説明
ファイル追加	押下すると、スクリプトで使用するファイルをアップロードすることができます。 ※新しく追加されたファイルは、コンポーネントの設定を保存した時にアップロードされます。
ファイルパス	アップロードしたファイルの保存先 ※パラメータでファイルパスを使用する場合、変数を用いることができます。詳細は、表題の「ファイルパス」の右のアイコンをクリックすると確認することができます。

設定画面のパラメータ入力部分に、ユーザーが定義した画面項目を表示するように変更することができます。 詳細は「カスタムUI作成」をご覧ください。

6.2.4.5.3 削除

次の手順で、登録したカスタムコンポーネントをSpeeDBee Synapseから削除することができます。

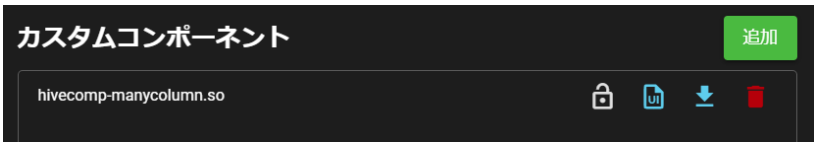
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



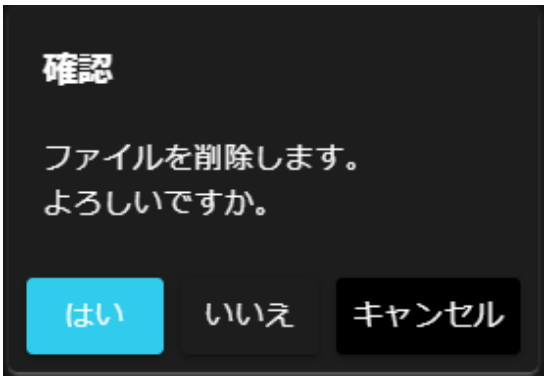
2. ダイアログ上部のタブの「C(.so)」を選択します。



3. カスタムコンポーネントの「ゴミ箱アイコン（削除）」を押下します。



4. 確認ダイアログで「はい」を選択します。



5. カスタムコンポーネントが削除されます。

6.2.4.5.4 ダウンロード

次の手順で、登録したカスタムコンポーネントをダウンロードすることができます。

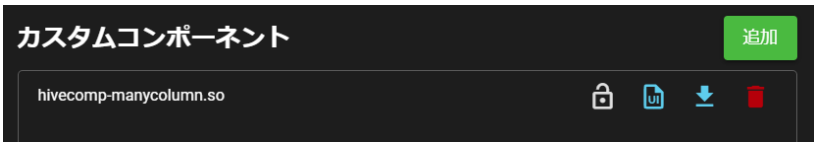
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「C(.so)」を選択します。



3. カスタムコンポーネントの「ダウンロードアイコン（取得）」を押下します。



4. カスタムコンポーネントのファイルがダウンロードされます。

6.2.4.5.5 保護の付与

登録したコンポーネントには保護をかけることができます。
保護されたコンポーネントは次の操作が行えなくなるため、意図しない更新を防ぐことができます。

- 登録
- 削除
- ダウンロード
- カスタムUIの設定

次の手順で、カスタムコンポーネントに保護をかけることができます。

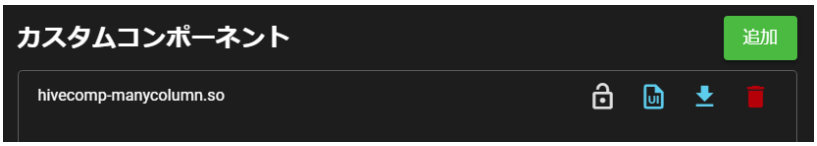
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「C(.so)」を選択します。



3. カスタムコンポーネントの「鍵アイコン（保護）」を押下します。



4. 保護解除する際に必要となるパスワードを入力して、「保護」を押下します。

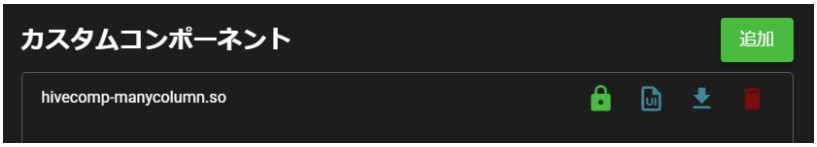


警告

- パスワードは復元できない形式で保存され、一度設定すると保護を解除するまで再設定できません。
- 保護設定したパスワードを忘れた場合、該当のカスタムコンポーネントは操作不能となります。

パスワードの取り扱いには十分ご注意ください。

5. カスタムコンポーネントが保護状態になります。



6.2.4.5.6 保護の解除

次の手順で、保護されたカスタムコンポーネントの保護を解除することができます。

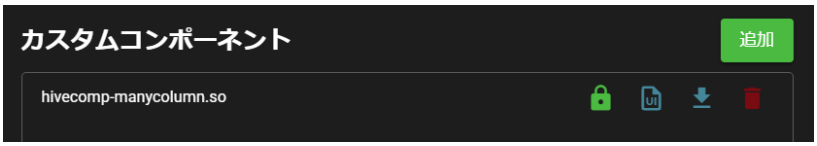
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. ダイアログ上部のタブの「C(.so)」を選択します。



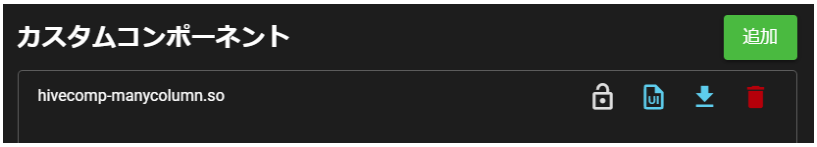
3. カスタムコンポーネントの「鍵アイコン（保護解除）」を押下します。



4. 保護をかけた際に設定したパスワードを入力して、「保護解除」を押下します。



5. カスタムコンポーネントの保護が解除されます。



6.2.5 カスタムUI作成

カスタムコンポーネントの設定画面はデフォルトだと、パラメーターとしてJSON、もしくは文字列を自由に入力できるようになっています。しかし、これだと作成者と使用者が異なる場合、パラメーターに設定する項目について調べるにはカスタムコンポーネントのソースコードを確認する必要があり、手間がかかります。



そのため、カスタムコンポーネントは設定画面のパラメータの入力欄をカスタマイズすることができます。パラメータの入力欄として、どういった項目を配置するかユーザーが定義して、使用者が入力する項目をわかりやすくすることができます。

6.2.5.1 カスタムUIの作成方法

カスタムUIは以下の方法で作成できます。 カスタムUIには以下の２つの形式があり、それぞれ作成方法が異なります。

- SynapseカスタムUI用JSON

  コンポーネントへの設定項目を、JSONファイルに記載する方法です。 作成するのはJSONファイルだけであり、修正が必要な時もJSONファイルを変更するだけで手軽に行えます。 その代わり、Synapseが提供する機能の範囲でしか設定項目は定義できず、複雑なUIの作成はできません。

- SynapseカスタムUI用HTML

  コンポーネントへの設定項目を、HTMLとJavaScriptで作成します。
  HTMLファイルの内容はユーザーが好きなように作成でき、JavaScript, CSS, 画像なども活用することで複雑なUIを実装することができます。 ただし、作成には基本的なHTML/JavaScriptの知識が必要になります。 また、単純な実装だとSynapse本体の画面とはルック＆フィールが異なってしまいます。

カスタムUIとして作成できる範囲は画像の赤枠内にある「パラメーター」に関連する要素だけであり、外側の要素は編集できません。



- 「保存」ボタン

  入力したコンポーネントの設定を保存します。
- 「キャンセル」ボタン

  設定ダイアログを閉じます。
- 名称、自動起動無効

  コンポーネント稼働に必須な設定項目です。
- スクリプトで使用するファイルをアップロードする(証明書など)

  コンポーネント実行時に必要なファイル（証明書など）を管理します。

6.2.5.2 SynapseカスタムUI用JSONファイルの作成

ここでは、カスタムUI用JSONファイルの作成方法について説明します。

6.2.5.2.1 作成方法

次の手順で、カスタムUI用JSONファイルを作成することができます。

1. テキストエディタで空のjsonファイルを作成します。
2. jsonファイル内に、"parameter_defines"という配列を定義します。
3. "parameter_defines"の中に、カスタムコンポーネントの画面に表示する入力要素を定義します。その他、"window_size"、"layout"など必要に応じて定義してください。

定義ファイル(json)

```
{
  "window_size": "auto",
  "parameter_defines": [
    {
      // 入力要素1
    },
    {
      // 入力要素2
    },
    {
      // 入力要素3
    }
    ...
  ]
}
```

画面に表示する入力要素は、以下の項目によって定義します。

項目	値の種類	必須	説明
window_size	string	×	ウィンドウのサイズ { "window_size" : "small" }のように指定 ○ auto/指定なし - min-width: "400px" ○ small - min-width: "33%" ○ middle - min-width: "50%" ○ large - min-width:"80%"
parameter_defines	object	○	項目の親要素 （最上位の階層から１つ下の階層まで入れ子指定可能 参考）
name	string	○	入力した値を設定ファイルに保持する際のキー
label	string	○	画面に表示される項目名
type	string	○	入力する値の種類を、以下の中から選択 string：文字列 number：整数 real：実数 select：選択肢
max	number	○	入力の上限 ※typeがnumber・realの場合のみ有効

項目	値の種類	必須	説明
min	number	×	入力の下限 ※typeがnumber・realの場合のみ有効
select	array	×	選択肢 ※1 typeがselectの場合のみ有効 ※2 選択肢は、"label"と"value"のプロパティを持つオブジェクト配列によって定義してください。 例：[{ "label":"option1", "value":1}, { "label":"option2", "value":2}]
default	any	×	デフォルト値
require	bool	○	入力必須か否か
array	bool	×	配列化するか否か
layout	object	×	各項目の幅指定 参考 { "layout" : { "col" : 2 }}のようにcolへ1～12を指定 省略時はデフォルト6 array,object省略時はデフォルト12

6.2.5.2.2 サンプル

カスタムUI定義ファイルのサンプルは次の通りです。

```
{
  "parameter_defines": [
    {
      "name": "host",
      "label": "ホスト",
      "type": "string",
      "default": "localhost",
      "require": true,
      "max":256
    },
    {
      "name": "port",
      "label": "ポート",
      "type": "number",
      "max": 65535,
      "min": 1,
      "default": "12",
      "require": true
    },
    {
      "name": "select",
      "label": "選択",
      "type": "select",
      "select": [
        { "label":"aaa", "value":1},
        { "label":"bbb", "value":2}
      ],
      "default": 1,
      "require": false
    }
  ]
}
```

次のようなレイアウトとなります。



6.2.5.2.3 入れ子を使ったサンプル

カスタムUI定義ファイルの入れ子を使ったサンプルは次の通りです。

```
{
  "window_size": "middle",
  "parameter_defines": [
    {
      "name": "nested",
      "label": "入れ子",
      "type": "object",
      "array": true,
      "parameter_defines": [
        {
          "name": "name",
          "label": "波名",
          "type": "string",
          "require": true,
          "max": 32,
          "default": "signal_5"
        },
        {
          "name": "freq",
          "label": "周波数",
          "type": "select",
          "require": true,
          "select": [
            { "label": "1.0", "value": 1.0 },
            { "label": "2.0", "value": 2.0 }
          ],
          "default": 1.0
        },
        {
          "name": "amp",
          "label": "振幅",
          "type": "real",
          "require": true,
          "max": 100.0,
          "min": 0.1,
          "default": 1.0
        }
      ],
      "require": false,
      "default": [
        {
          "name": "signal_1",
          "freq": 1,
          "amp": 1.1
        },
        {
          "name": "signal_2",
          "freq": 2,
          "amp": 1.2
        }
      ]
    }
  ]
}
```

次のようなレイアウトとなります。



6.2.5.2.4 項目の幅を指定したサンプル

カスタムUI定義ファイルの項目幅を指定したサンプルは次の通りです。

```
{
  "parameter_defines": [
    {
      "name": "item1",
      "label": "項目1",
      "type": "string",
      "require": false,
      "max": 32,
      "default": "こうもく１",
      "layout": {
        "col": 2
      }
    },
    {
      "name": "item2",
      "label": "項目2",
      "type": "real",
      "require": false,
      "max": 32,
      "default": "12.3",
      "mask": true,
      "layout": {
        "col": 4
      }
    },
    {
      "name": "item3",
      "label": "項目3",
      "type": "select",
      "require": false,
      "max": 32,
      "default": 2,
      "select": [
        { "label": "A", "value": 1},
        { "label": "B", "value": 2},
        { "label": "C", "value": 3}
      ],
      "layout": {
        "col": 6
      }
    }
  ]
}
```

次のようなレイアウトとなります。



6.2.5.3 SynapseカスタムUI用HTML作成、適用方法

ここでは、カスタムUIをHTMLとJavaScriptで作成する方法について説明します。

6.2.5.3.1 作成方法

HTML形式でのカスタムUIを利用するには、下記の環境が必要です。 まずはこの環境を準備してください。

- カスタムコンポーネント開発環境(SCCDE)

以下のいずれかのコマンドで、HTML形式のカスタムUIを含むサンプルコンポーネントを追加できます。

```
$ sccde add -t collector -u html
$ sccde add -t emitter -u html
```

実際のHTMLファイルは、`parameter_ui`配下のコンポーネント個別のディレクトリ内に配置されます。 作成されたサンプルを元に、html, js, cssファイルをカスタマイズして利用してください。

6.2.5.3.2 HTML形式の詳細

- HTMLファイル(index.html)

  通常のHTMLと同様に、HTMLタグ、BODYタグを使って定義できます。 このファイルで定義した内容が、カスタムコンポーネントのパラメータ設定ダイアログの文字列入力部の代わりに表示されます。

表示例



- JavaScriptファイル(index.js)

  ダイアログ表示時、および、パラメータ保存時に、所定の関数呼び出しが必要なため、JavaScriptでその関数を定義する必要があります。 このJavaScriptファイルはHTMLファイル内の`<script>`タグで読み込むようにしてください。

  詳細は後述

- scc-info.jsonファイル

  カスタムコンポーネントでカスタムUI(HTML)を使用することをシステムに伝えるため、対象コンポーネントの下記の定義が必要です。

プロパティ	説明
parameter-ui-type	'html'を指定
parameter-ui-size	パラメータダイアログの画面サイズ指定。'small','middle','large','auto'から選択可能。（デフォルトは'middle'）
parameter-ui	HTMLファイルやJavaScriptファイルを含むディレクトリへのパスを指定

6.2.5.3.3 JavaScriptファイルの記述

以降では、サンプルの`index.js`のソースコードを例に、カスタムUIをHTMLで実装する際のJavaScriptにおいての処理方法を説明します。

JavaScriptでSynapseの情報を取り扱えるようにするため、Synapse専用のAPIオブジェクトを提供しています。 このAPIオブジェクトは、'window.customUiLib`として参照できます。

このオブジェクトに特定のプロパティ名で関数を登録することで、Synapse本体の画面とデータの受け渡しが可能になります。

```
async function onLoad(editable, param, isNewComponent, componentId) {
  :
}
async function onSave() {
  :
}

// コールバック関数の登録
window.customUiLib.onSave = onSave;
window.customUiLib.onLoad = onLoad;
 :
```

その他にもSynapseの情報を取得できるAPIがあります。

これらAPIの詳細については、カスタムUI JS-APIリファレンスを参照してください。

6.2.5.3.4 パラメータダイアログが表示されたときに実行される関数

カスタムコンポーネントのパラメータ設定ダイアログが表示される際には、カスタムUIのJavaScriptで登録した `onLoad()` 関数がコールされます。 この関数の引数として、以下の情報が渡されますので、必要に応じてhtml内の要素に反映してください。

- ダイアログの編集可否

  Run Modeにおいてはダイアログの入力項目を編集不可にするため、引数 `editable` に `false` が渡されます。 詳細は編集制限の制御を参照してください。

- 設定済みパラメータ

  すでに作成済みのコンポーネントパラメータを編集する際、以前設定されたパラメータが引数 `param` に渡されます。

- 新規作成か更新か

  コンポーネントの新規作成時には `true` 、更新時は `false` が引数 `isNewComponent` に渡されます。

- コンポーネントID

  コンポーネントのIDが引数 `componentId` に渡されます。この情報はコンポーネントの新規作成時にはありません。

6.2.5.3.5 パラメータダイアログの保存時に実行される関数

カスタムコンポーネントのパラメータ設定ダイアログで**保存**ボタンがクリックされた際には、カスタムUIのJavaScriptで登録した `onSave()` 関数がコールされます。 この関数から保存すべきパラメータを返すようにしてください。

6.2.5.3.6 編集制限の制御

Synapseには**Run Mode**、**Edit Mode**の二つの画面があり、Run Modeではデータ収集フローの運用を、Edit Modeではデータ収集フローの設定などが行えます。

通常のコンポーネントやJSONファイルで定義したカスタムUIはRun Modeでは編集できないように制限がかけられていますが、HTMLで実装したカスタムUIではその制限ができません。その代わり、編集可能か否かの情報を`onLoad()`関数の引数`editable`から受け取れますので、それをもとにユーザーが編集できないように実装してください。

基本的にはHTML要素のdisableをtrueに設定して非活性とすれば問題ありません。

```
async function onLoad(editable, param, isNewComponent, componentId) {
  // editableがfalseならフォームの'host'項目を非活性にする
  document.forms['form']['host'].disabled = !editable;
}
window.customUiLib.onLoad = onLoad;
```

参考：HTML属性: disabled

6.2.5.3.7 npmを利用したサンプル

Synapseでは、`Vue`や`Quaser`を利用して画面を作成しています。
素のHTMLやCSSでは違和感のある画面となってしまいますが、`npm`を利用することで、`Vue`や`Quaser`といった外部パッケージを利用したUIを作成することができます。 ここまでの説明で利用したサンプルでは素のHTML, Javascript, CSSだけを利用していましたが、Vue, Quaserを利用するようにしたサンプルも提供しています。

`Vue`、`Quaser`を利用したサンプル

サンプルをダウンロードしましたら、下記のコマンドを実行することでhtmlやjsファイルが`dist`ディレクトリ配下に生成されます。 これらのファイルを通常のhtmlと同様にSCCDE環境内に配置して使用可能です。

```
$ ls
sample_custom_ui_vue.zip
$ unzip sample_custom_ui_vue.zip
$ cd sample_custom_ui_vue
$ npm install
$ npm run build
$ ls dist
assets  index.html
```

npmの利用

`npm`を利用するには`Node.js`をインストールする必要があります。 インストール方法を含めたNode.jsについての詳細はNode.js公式サイトを参照してください
Node.js公式サイト：https://nodejs.org/ja

6.2.5.4 Pythonカスタムコンポーネントへのカスタム UI(JSON)の適用

「SynapseカスタムUI用JSONファイルの作成」で作成した定義ファイルを登録することで、 Pythonカスタムコンポーネントにカスタム UIを設定することができます。

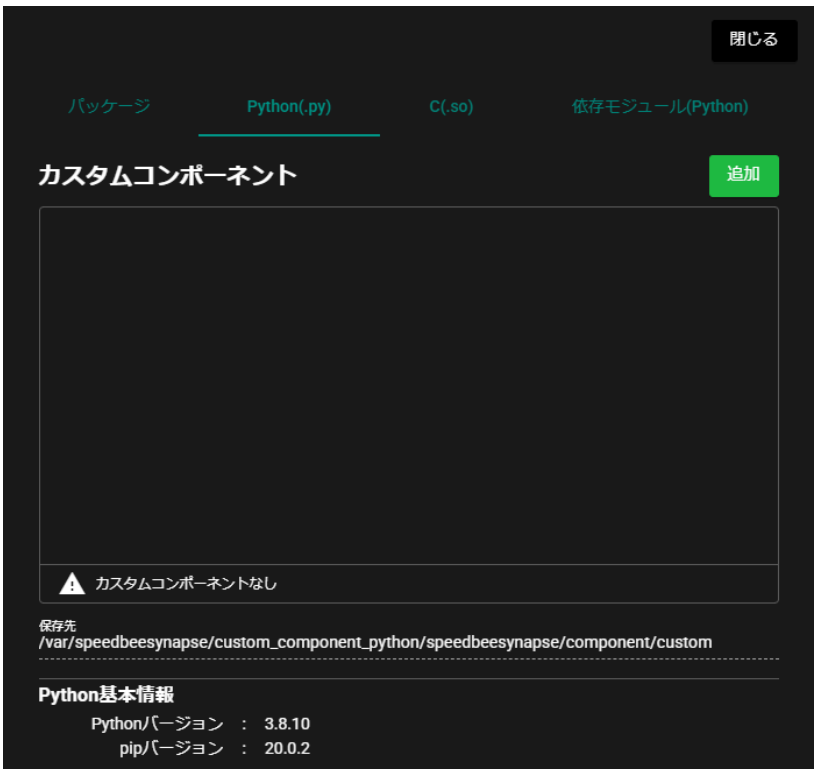
6.2.5.4.1 登録

次の手順で、カスタムUIを登録することができます。

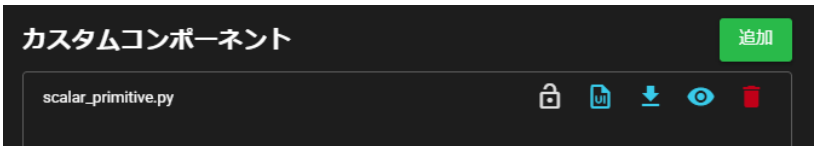
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



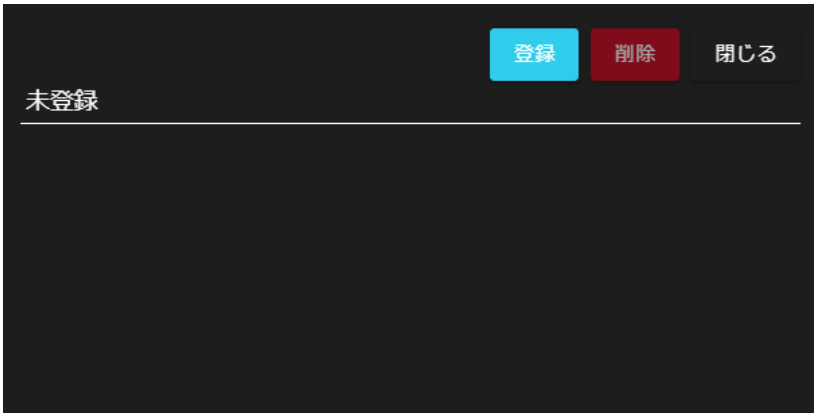
2. ダイアログ上部のタブの「Python(.py)」を選択します。



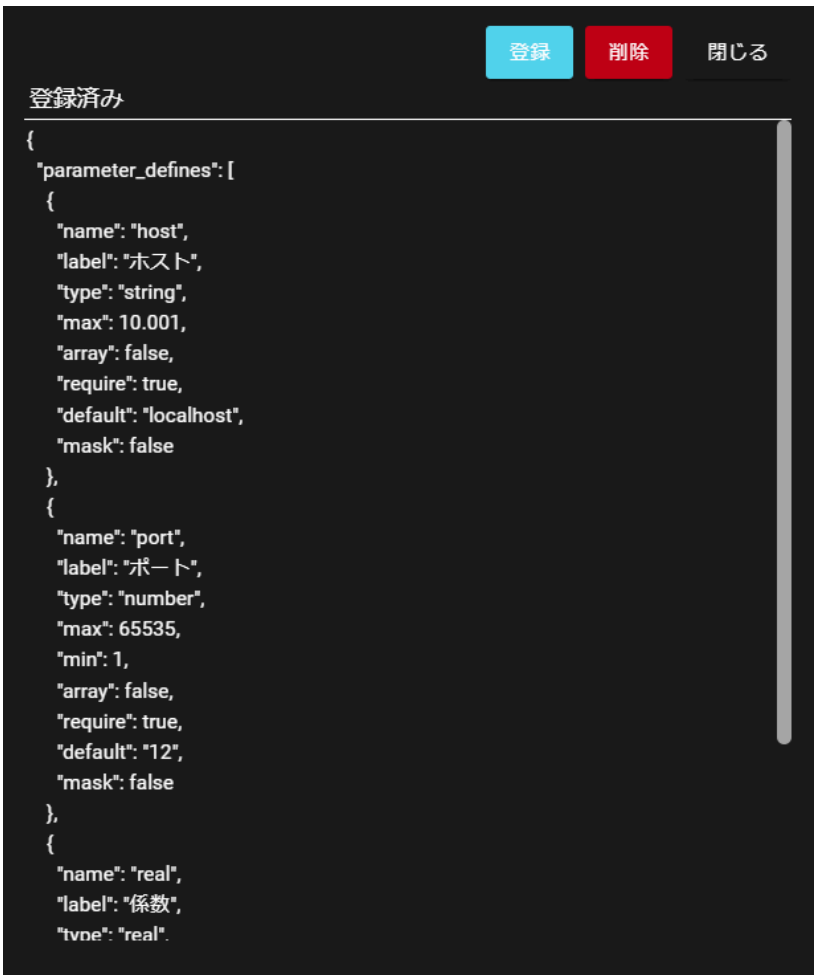
3. 任意のカスタムコンポーネントの「UIアイコン（UI設定）」を押下します。



4. 「登録」を押下し、定義ファイルを登録します。



5. 「登録済み」という文言と、定義ファイルの内容が表示されます。



6. 定義ファイルを登録したカスタムコンポーネントの設定画面に、ファイルで定義した項目が表示されます。



パラメーター表示

1. 「パラメーター表示」を押下します。



2. カスタムUIに入力した内容から生成される、コンポーネントの実行時パラメータを確認することができます。



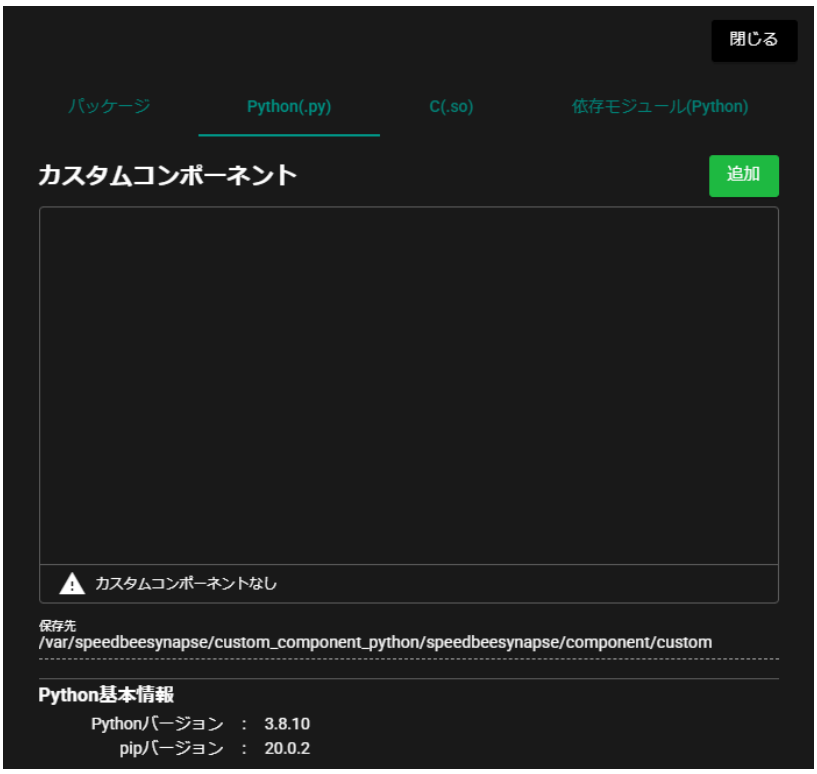
6.2.5.4.2 削除

次の手順で、登録した定義ファイルを削除し、画面をデフォルトの状態にもどすことができます。

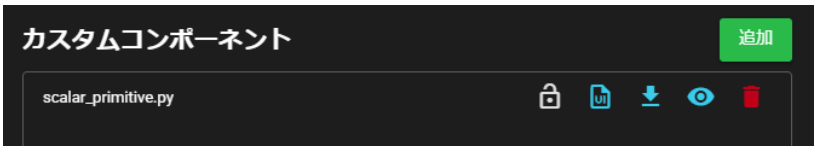
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



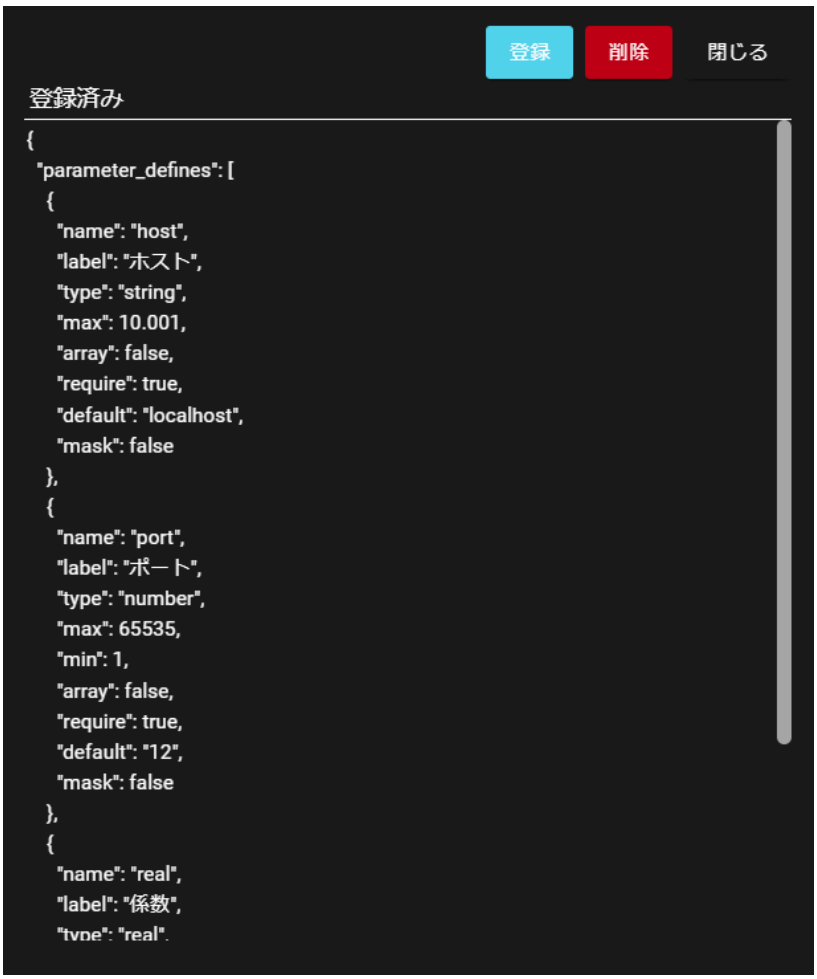
2. ダイアログ上部のタブの「Python(.py)」を選択します。



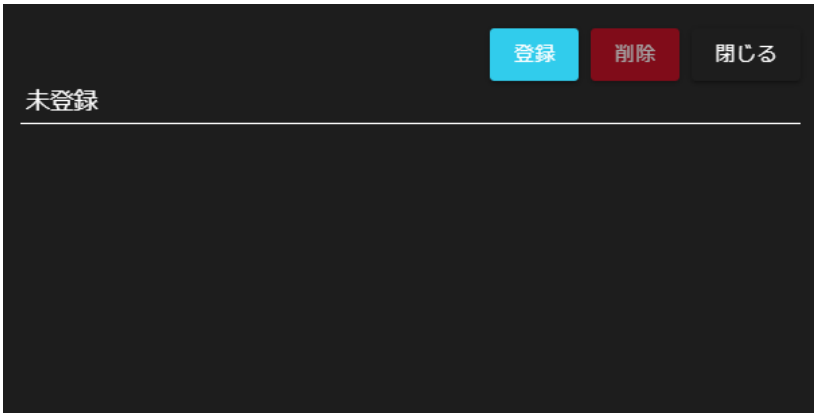
3. 定義ファイルを登録したカスタムコンポーネントの「UIアイコン（UI設定）」を押下します。



4. 「削除」を押下します。



5. 登録した定義ファイルが削除され、「未登録」と表示されます。



6. カスタムコンポーネントの設定画面が、デフォルトの状態に戻ります。



6.2.5.5 C言語カスタムコンポーネントへのカスタムUI(JSON)の適用

「SynapseカスタムUI用JSONファイルの作成」で作成した定義ファイルを登録することで、 C言語カスタムコンポーネントにカスタムUIを設定することができます。

6.2.5.5.1 登録

次の手順で、カスタムUIを登録することができます。

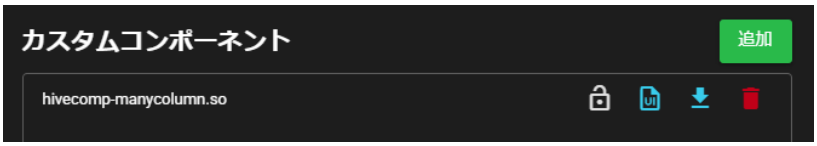
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



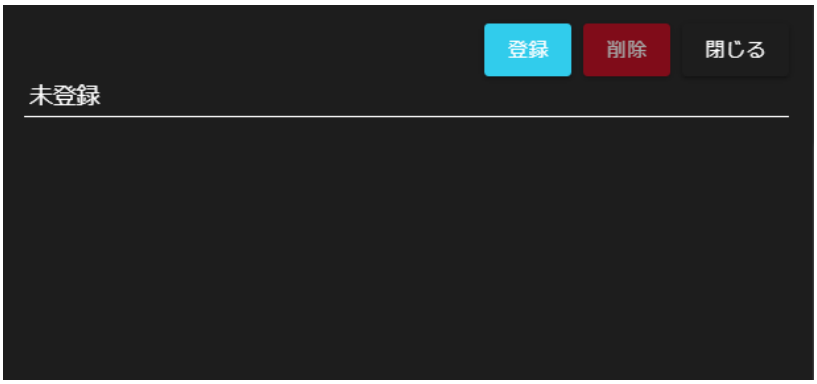
2. ダイアログ上部のタブの「C(.so)」を選択します。



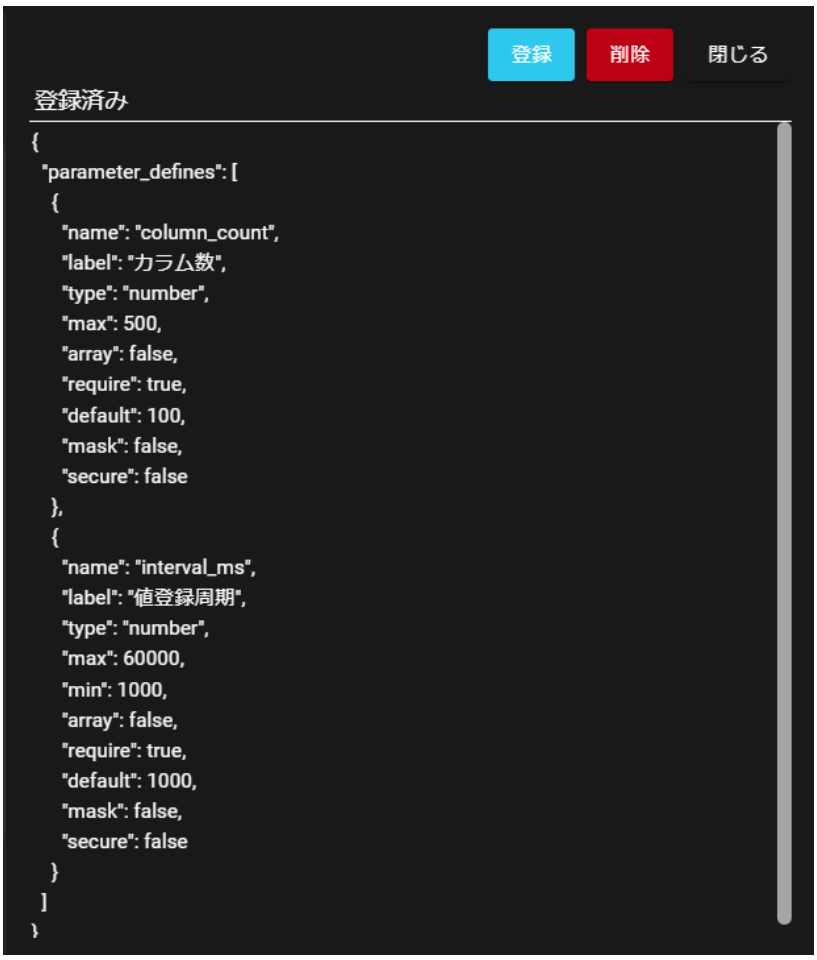
3. 任意のカスタムコンポーネントの「UIアイコン（UI設定）」を押下します。



4. 「登録」を押下し、定義ファイルを登録します。



5. 「登録済み」という文言と、定義ファイルの内容が表示されます。



6. 定義ファイルを登録したカスタムコンポーネントの設定画面に、ファイルで定義した項目が表示されます。

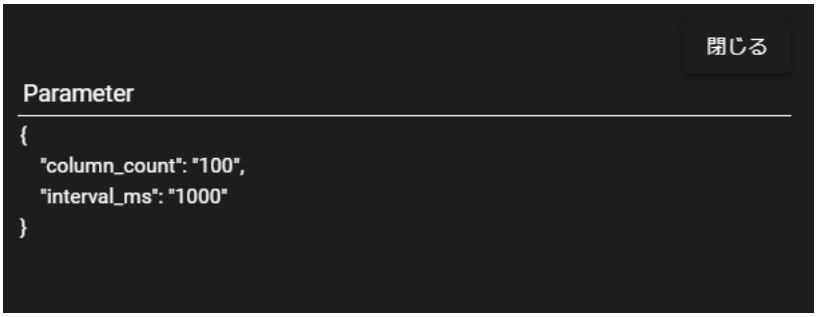


パラメーター表示

1. 「パラメーター表示」を押下します。



2. カスタムUIに入力した内容から生成される、コンポーネントの実行時パラメータを確認することができます。



6.2.5.5.2 削除

次の手順で、登録した定義ファイルを削除し、画面をデフォルトの状態にもどすことができます。

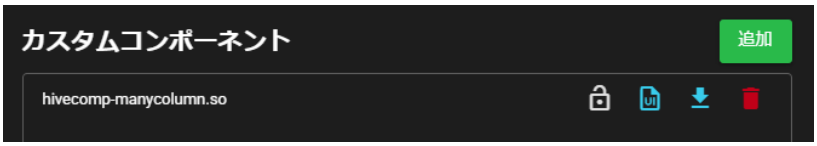
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



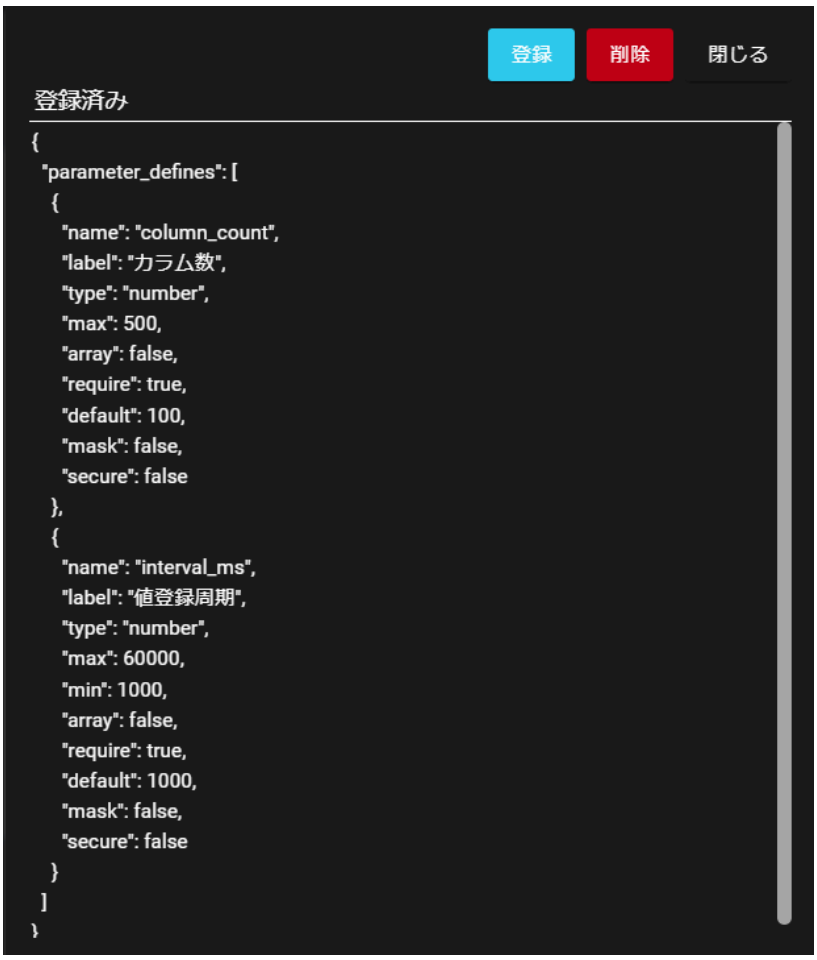
2. ダイアログ上部のタブの「C(.so)」を選択します。



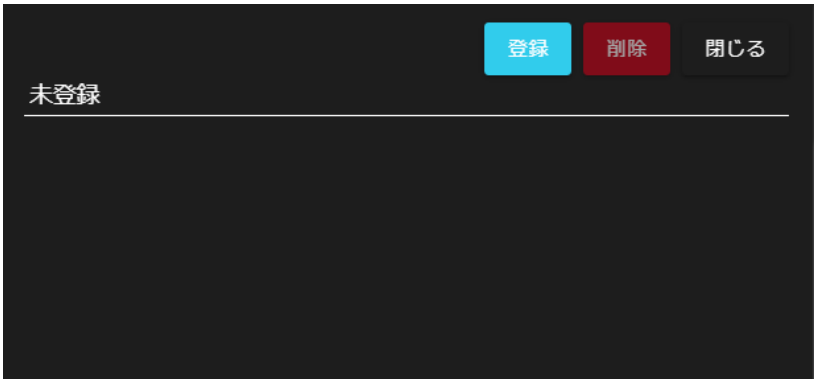
3. 定義ファイルを登録したカスタムコンポーネントの「UIアイコン（UI設定）」を押下します。



4. 「削除」を押下します。



5. 登録した定義ファイルが削除され、「未登録」と表示されます。



6. カスタムコンポーネントの設定画面が、デフォルトの状態に戻ります。


保存
キャンセル
名称
自動起動無効
パラメータータイプ
json
パラメーター
スクリプトで使用するファイルをアップロードする(証明書など)

6.2.6 カスタムコンポーネント開発環境(SCCDE)

本項では、Synapseカスタムコンポーネント開発環境(以下SCCDE)について説明します。

6.2.6.1 SCCDEの概要

Synapseカスタムコンポーネント開発環境(SCCDE)は、Synapseのカスタムコンポーネントを開発するための 様々なツール・パッケージを集約した環境です。現在のところ以下の機能を有しています。

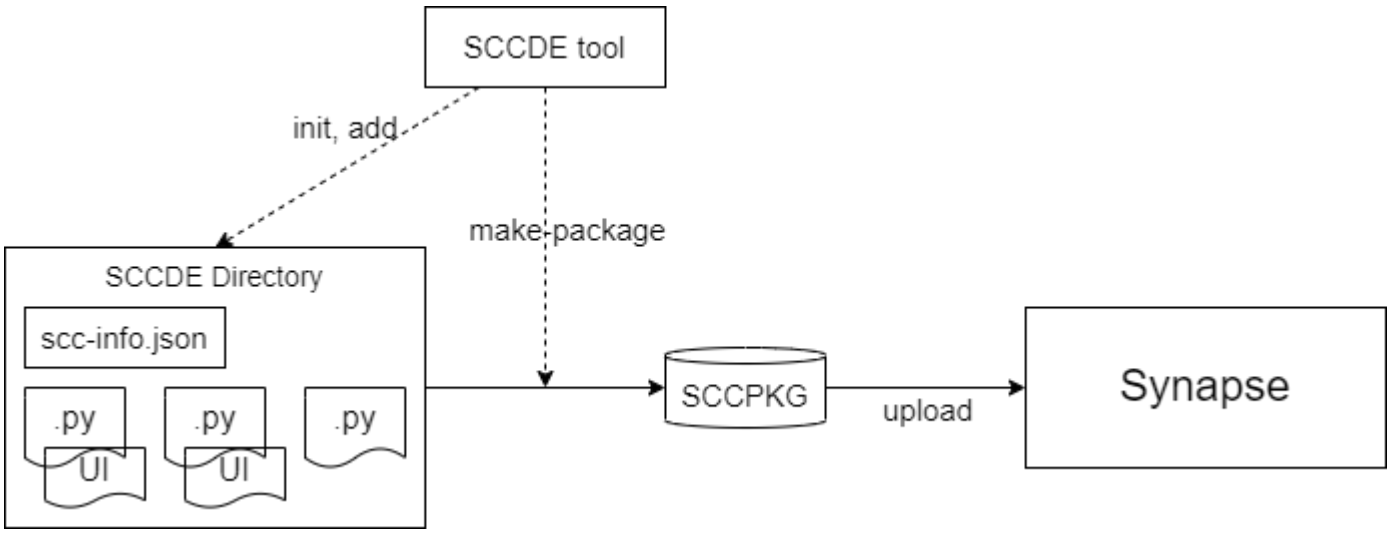
- カスタムコンポーネント管理ディレクトリの初期化(init)

  管理用のディレクトリは、GitやSubversionなどのバージョン管理システムで利用しやすい形態になっています。
- カスタムコンポーネントのサンプルプログラムの追加(add)

  コレクタ、シリアライザ、エミッタなどの簡易的なサンプルプログラムを追加できます。
- カスタムコンポーネントのパッケージング(make-package)

  コンポーネントのパッケージングを行うと、複数のカスタムコンポーネントやカスタムUIを まとめて１つのファイル(SCCPKG)に収めることができます。 このSCCPKGファイルを、Synapseの管理画面からアップロードすることで、 複数のカスタムコンポーネントをまとめて管理することができます。
- カスタムコンポーネントのテスト環境の実行(serve)

  作成したカスタムコンポーネントを簡易的に実行するためのテスト環境を起動できます。



現時点ではPython実装のカスタムコンポーネントのみに対応しています。 C言語実装のコンポーネントの対応は未定です。

6.2.6.2 インストール方法

SCCDEは以下の方法でインストールできます。

- Synapseと一緒に配布されるwhlパッケージファイルを使ってインストール
- PyPI（pipの標準インデックス）からインストール

インストールする環境はWindowsでもLinuxでも構いません。 ただしテスト環境の実行(serve)機能は、同一デバイスにSynapseがインストールされている必要があります。

Linux

```
$ python3 -m venv ~/sccde
$ . ~/sccde/bin/activate
(sccde) $ pip3 install speedbeesynapse-sccde
```

参考：venv

Windows(pwsh)

```
PS> python -m venv "$HOME\sccde"
PS> . "$HOME\sccde\Scripts\Activate.ps1"
(sccde) PS> pip3 install speedbeesynapse-sccde
```

参考：venv

本パッケージは標準的なPythonパッケージですので、 上記のpip, venvを使ったインストール方法以外にも様々なパッケージ管理ツール （Poetry, uvなど） を使用可能です。お好みのツールをご利用ください。

※以降で説明する手順は、上記のactivateを行った`(sccde)`環境内で実施することを想定しています。

6.2.6.3 SCCDEの使い方とパッケージング

6.2.6.3.1 カスタムコンポーネント管理ディレクトリの初期化

カスタムコンポーネントを管理するために、空のディレクトリを用意して初期化する必要があります。 以下の手順で初期化してください。

Linux

```
(sccde) $ mkdir sccpack-test
(sccde) $ cd sccpack-test
(sccde) $ sccde init
(sccde) $ ls
parameter_ui  source  scc-info.json
```

Windows(pwsh)

```
(sccde) PS> mkdir sccpack-test
(sccde) PS> cd sccpack-test
(sccde) PS> sccde init
(sccde) PS> ls
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----          2025/xx/yy    hh:mm                parameter_ui
d----          2025/xx/yy    hh:mm                source
-a---          2025/xx/yy    hh:mm            644 scc-info.json
```

初期化を行うと現在のディレクトリに対して以下の処理が行われます。

- `scc-info.json`の生成
- カスタムコンポーネント(collector)のサンプルプログラムの追加

`scc-info.json`は、このディレクトリ内で管理するカスタムコンポーネントに関する情報を記載するためのファイルです。 JSON形式になっていますので、必要に応じてテキストエディタ等で編集してください。

主要なプロパティを下記に記載いたします。

プロパティ	説明
package-name	パッケージの名称です。任意の名前を記載できます。
package-version	パッケージのバージョンを示す文字列です。基本的には"."区切りの３つの数値を記載してください。 このバージョン文字列はSemantic Versioning 2.0.0に沿った形式が望ましいです。
package-uuid	パッケージを識別するためのIDです。初期状態ではランダムなUUIDが設定されています。基本的には変更せずに使用してください
package-description	パッケージの説明文です。任意の文字列を記載して下さい。
author	パッケージ作成者の名前を記載できます。

プロパティ	説明
license	このパッケージを利用する際のライセンス種別を記入できます。（BSD, MIT, UNLICENSEDなど）。不要な場合は空文字列としてください。
license-file	このパッケージのライセンス詳細を記したファイルの相対パスを指定できます。不要な場合は空文字列としてください。
python-components-source	このディレクトリ内で管理するPythonのカスタムコンポーネントの配置パスです。基本的に変更する必要はありません。
components	複数のカスタムコンポーネントの定義を記載するためのオブジェクトです。keyにはカスタムコンポーネントのUUIDを指定する必要があります。
components.*.name	カスタムコンポーネントの名称です。任意の名前を記載できます。
components.*.description	カスタムコンポーネントの説明文です。
components.*.component-type	カスタムコンポーネントの実装種別です。'python' or 'c'を指定する必要があります。
components.*.moduelname	Pythonカスタムコンポーネントのモジュール名です。 `python-components-source`で指定したパスを基準とした場合のカスタムコンポーネントをimportするためのモジュール名を記載してください。
components.*.parameter-ui-type	カスタムUIの定義ファイルへ種別を指定します。'json' or 'html'を指定可能です。
components.*.parameter-ui	カスタムUIへの相対パスを指定してください。 'json'形式の場合はそのファイルへのパスを、'html'形式の場合はhtmlファイルが配置されたディレクトリへのパスを指定してください。

6.2.6.3.2 カスタムコンポーネントの追加

管理ディレクトリを初期化した後は、`add`サブコマンドによりカスタムコンポーネントのサンプルプログラムを追加できます。

Linux

```
(sccde) $ sccde add -t collector
```

Windows(pwsh)

```
(sccde) PS> sccde add -t collector
```

`-t`の後の文字列は、`collector`の他に`serializer`, `emitter`などに置き換え可能です。

追加したサンプルプログラムは初期状態では`source/python配下に配置されます。`このサンプルはそのままでも実行可能なようになっていますが、これをベースに 自由にカスタムコンポーネントを改変して独自のコンポーネントを作ることができます。

カスタムコンポーネントの実装方法については下記を参照してください。

- Pythonカスタムコンポーネント作成
- カスタムコンポーネントPython-APIリファレンス

また、追加されたサンプルコンポーネントにはカスタムUIのサンプルも付属しています。 こちらも必要に応じて改変してご利用ください。

- カスタムUI作成

デフォルトではjson形式のカスタムUIが付与されます。 追加時のオプションで`-u html`を指定することで、HTML形式のカスタムUIも作成可能です。

6.2.6.3.3 カスタムコンポーネントのテスト実行

SCCDEコマンドをインストールした環境に、Synapseもインストールされている場合は`serve`コマンドでカスタムコンポーネントをテスト実行することができます。

Linux

```
(sccde) $ sccde serve --port 8123
Preparing Synapse data directories..
Synapse server ready: http://127.0.0.1:8123
:
```

Windows(pwsh)

```
(sccde) PS> sccde serve --port 8123
Preparing Synapse data directories..
Synapse server ready: http://127.0.0.1:8123
:
```

初回実行ではデータディレクトリの初期化にしばらく時間がかかります。`ready: http://..`のメッセージが表示されたら、ブラウザでそのURLを開いてください。

URLを開くと、Synapseのフローエディタの Edit Mode と同様の画面が表示されます。 左側のコンポーネントベース一覧から、作成したカスタムコンポーネントを選択可能になっています。

テスト用のSynapseは、`Ctrl-C` キーで停止できます。

オプション

オプション	説明
`--host`	テスト用Synapseサーバーが利用するIPアドレスを指定できます。稼働しているデバイス外からもアクセスしたい場合は `0.0.0.0` を指定してください。デフォルトは `127.0.0.1` となっています。
`--port`	テスト用Synapseサーバーが利用するTCPポート番号指定できます。他のサービスとポート番号が被らないように任意の番号を指定してください。デフォルトは `8000` となっています。
`--verbose`	テスト用Synapseの実行ログを表示します。

データディレクトリとログ

テスト用Synapseの実行中のデータは、sccdeの作業ディレクトリ内にある `.synapse/var-speedbeesynapse` ディレクトリに配置されます。

Linux

```
(sccde) $ ls .synapse/var-speedbeesynapse/
certs  config  dblog  hivecore_exec_status.json  logs  projects  sdts.db  sdts.file  webui_tmp
```

Windows(pwsh)

```
(sccde) PS> ls .\.synapse\var-speedbeesynapse\
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----          2025/xx/yy    hh:mm                certs
d----          2025/xx/yy    hh:mm                config
d----          2025/xx/yy    hh:mm                dblog
d----          2025/xx/yy    hh:mm                logs
d----          2025/xx/yy    hh:mm                projects
d----          2025/xx/yy    hh:mm                sdts.db
d----          2025/xx/yy    hh:mm                sdts.file
d----          2025/xx/yy    hh:mm                webui_tmp
-a---          2025/xx/yy    hh:mm            105 hivecore_exec_status.json
```

データをクリアして初期状態に戻したい場合は、`.synapse` フォルダごと削除してからテスト用Synapseを起動してください。

また、カスタムコンポーネントのデバッグ等でログを参照したい場合は、`.synapse/var-speedbeesynapse/logs` にある `speedbeesynase.log` を参照できます。

開発用コンポーネント

テスト用Synapse実行中は、専用の開発用コンポーネントを利用できます。 開発用コンポーネントの詳細については、「開発用コンポーネント」を参照してください。

6.2.6.3.4 パッケージング

カスタムコンポーネントを追加した後は、`make-package`サブコマンドにより、パッケージを作成できます。

Linux

```
(sccde) $ sccde make-package
(sccde) $ ls
custom_component_package_example.sccpkg  parameter_ui  scc-info.json  source
```

Windows(pwsh)

```
(sccde) PS> sccde make-package
(sccde) PS> ls
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----           2025/xx/xx    hh:mm                parameter_ui
d----           2025/xx/xx    hh:mm                source
-a---           2025/xx/xx    hh:mm           2481 custom_component_package_example.sccpkg
-a---           2025/xx/xx    hh:mm            644 scc-info.json
```

`.sccpkg`という拡張子がついたファイルがSCCPKGです。 このファイルをSynapseに登録することができます。

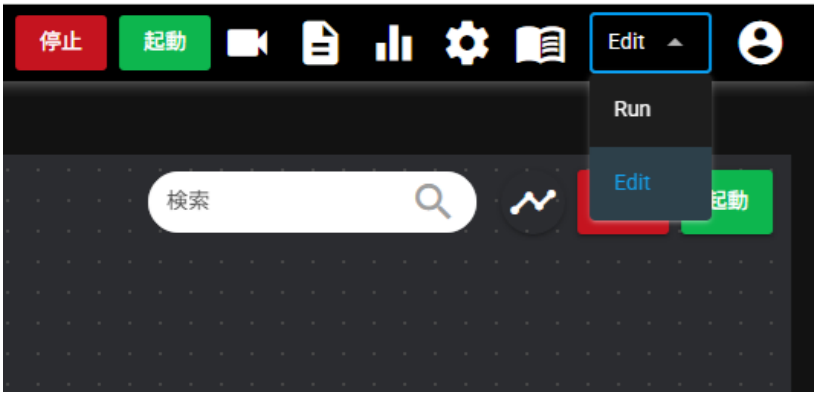
6.2.6.4 SynapseへのSCCPKGファイルの登録、および、削除

前項で説明したパッケージングにより作成したパッケージファイル(SCCPKG)を、 Synapseの管理画面から登録することで、パッケージに含まれるカスタムコンポーネントを利用できるようになります。

6.2.6.4.1 登録/更新

次の手順で、作成したSCCPKGファイルをSpeeDBee Synapseに登録することができます。

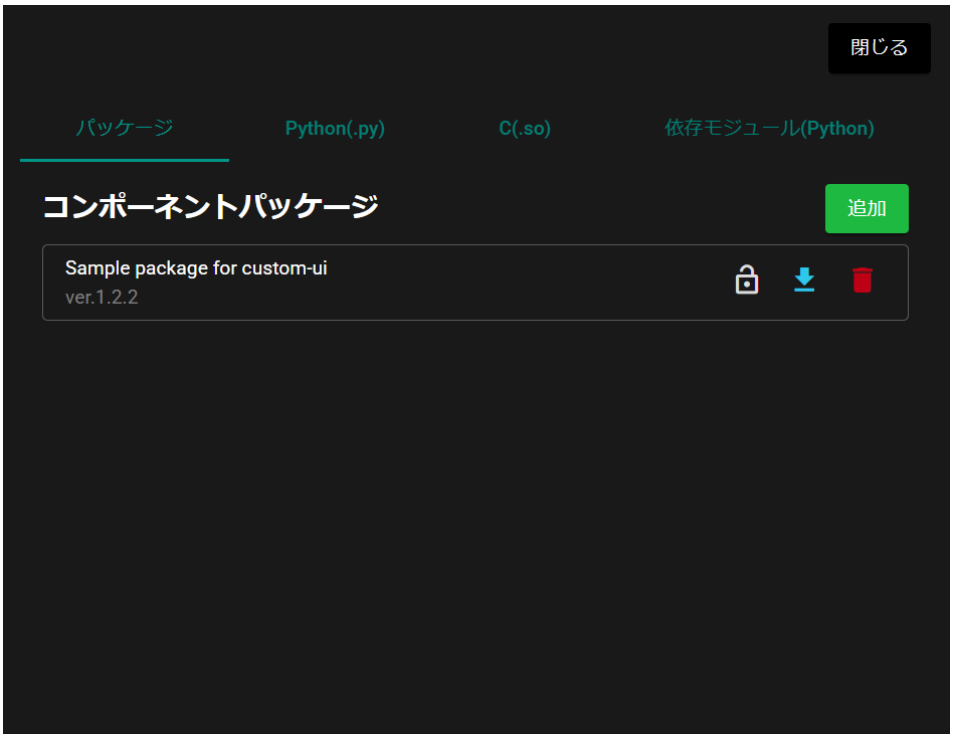
1. ヘッダのモード選択で"Edit Mode"を選択し、Editモードの画面を開きます。



2. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。

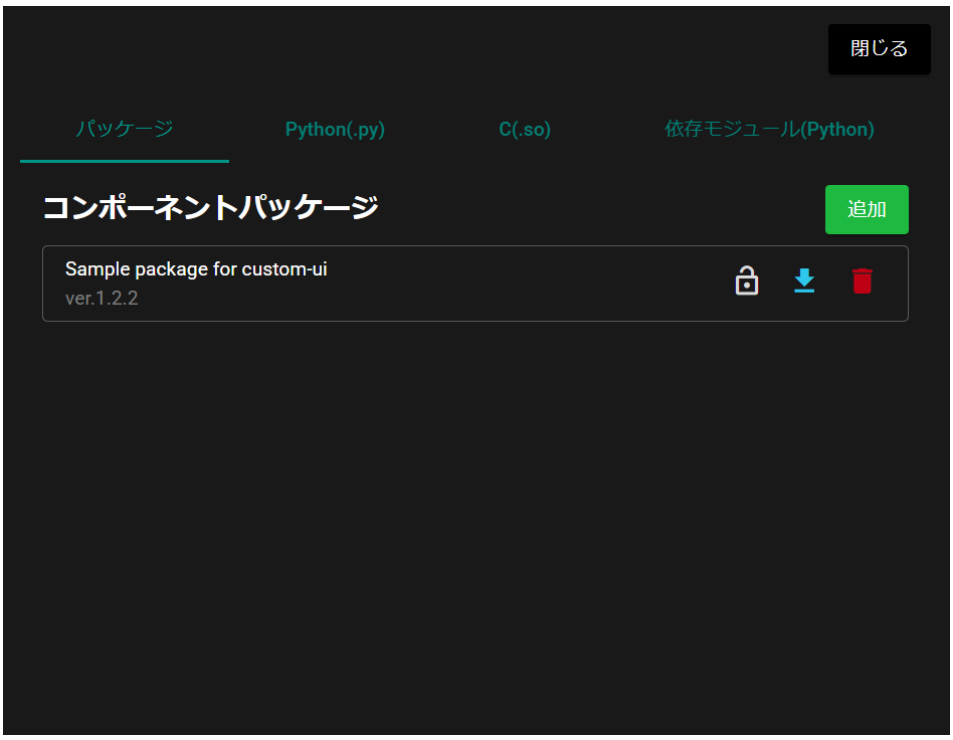


3. コンポーネントパッケージの「追加」を押下し、カスタムコンポーネントのファイルを登録します。

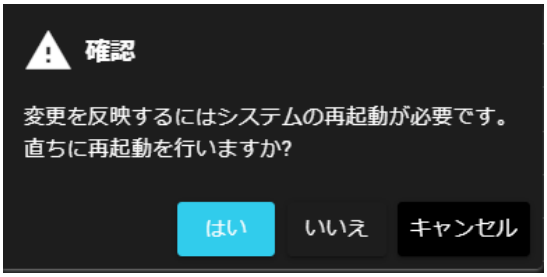


既に同一のパッケージが登録済みの場合、ここで上書きを確認するダイアログが表示されます。 上書きする場合は「はい」を選択してください。

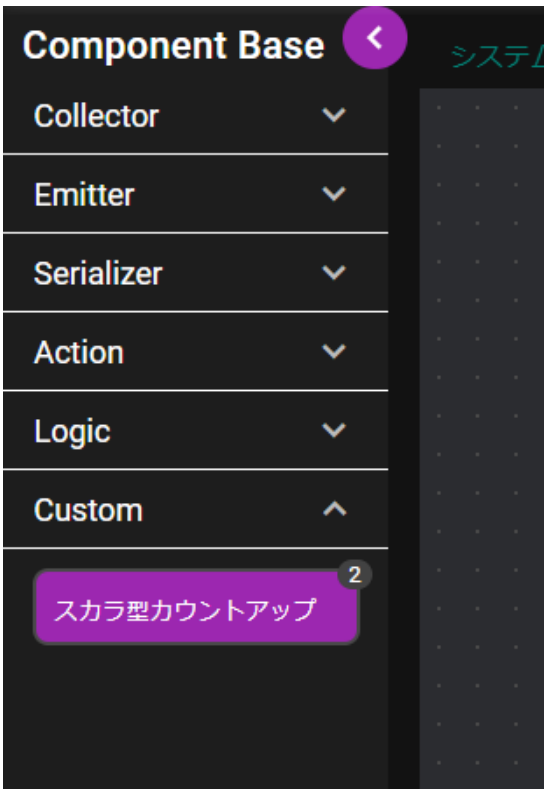
4. 「閉じる」を押下します。



5. 確認ダイアログで「はい」を選択し、変更を反映するために、システムを再起動させます。



6. 左メニューに、登録したカスタムコンポーネントが反映されます。



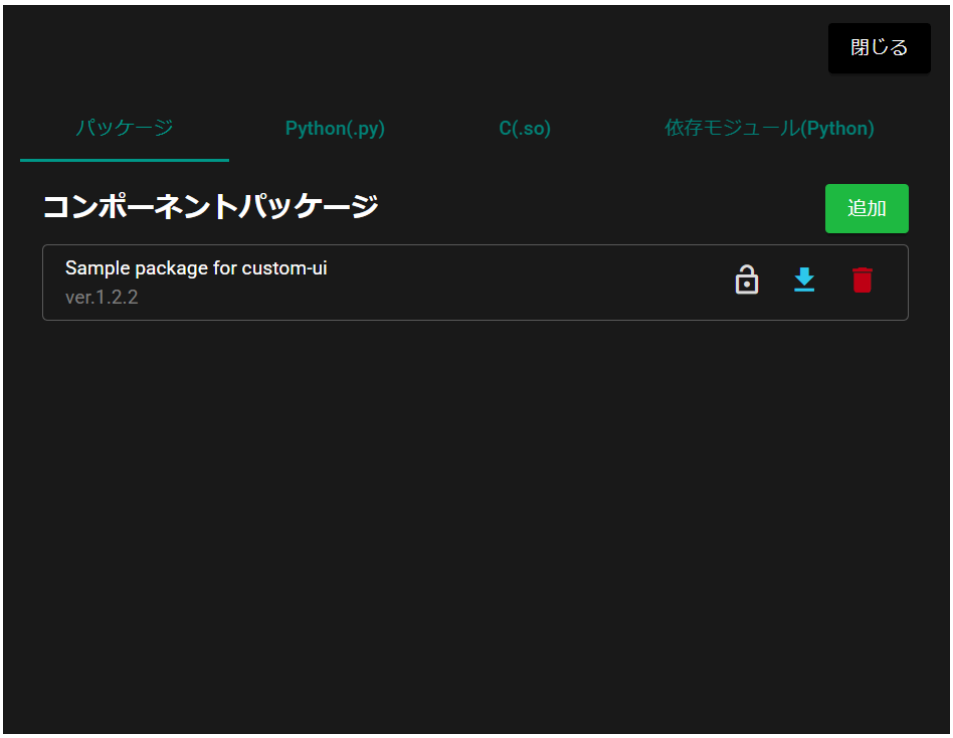
6.2.6.4.2 ダウンロード

次の手順で、登録したコンポーネントパッケージをダウンロードすることができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. コンポーネントパッケージの「ダウンロードアイコン（取得）」を押下します。



3. コンポーネントパッケージのファイルがダウンロードされます。

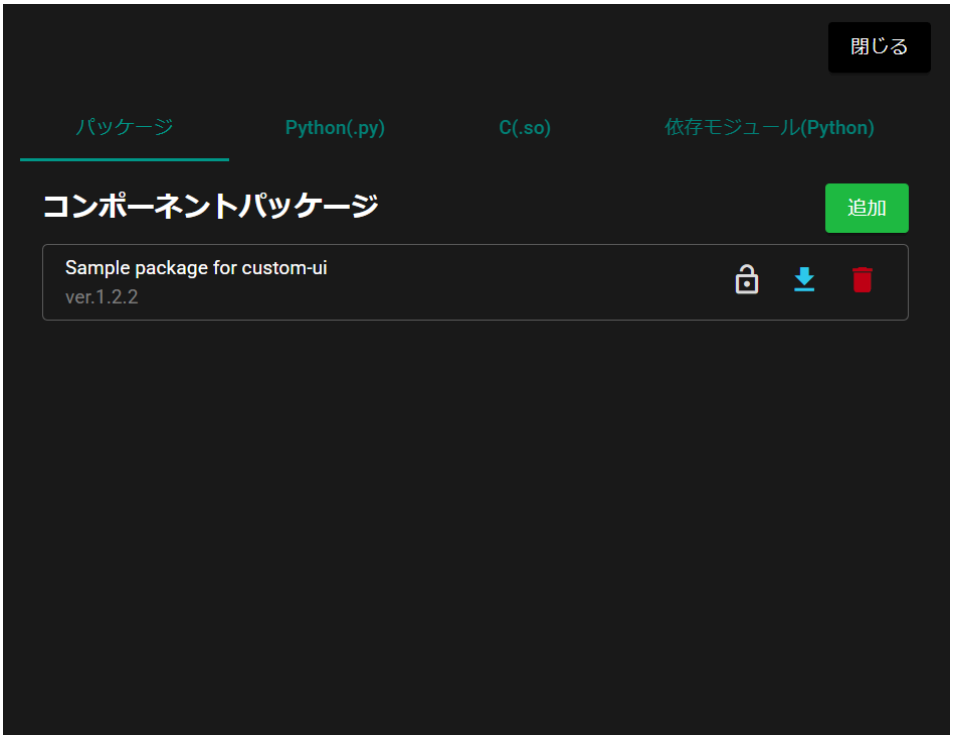
6.2.6.4.3 削除

次の手順で、登録したSCCPKGファイルをSpeeDBee Synapseから削除することができます。

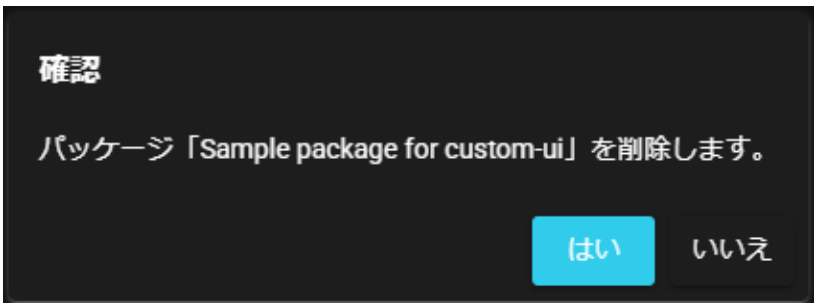
1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. 対象パッケージの右にある「ゴミ箱アイコン（削除）」を押下します。



3. 確認ダイアログで「はい」を選択します。



4. SCCPKGファイルが削除されます。

6.2.6.4.4 保護の付与

登録したパッケージには保護をかけることができます。
保護されたパッケージは次の操作が行えなくなるため、意図しない更新を防ぐことができます。

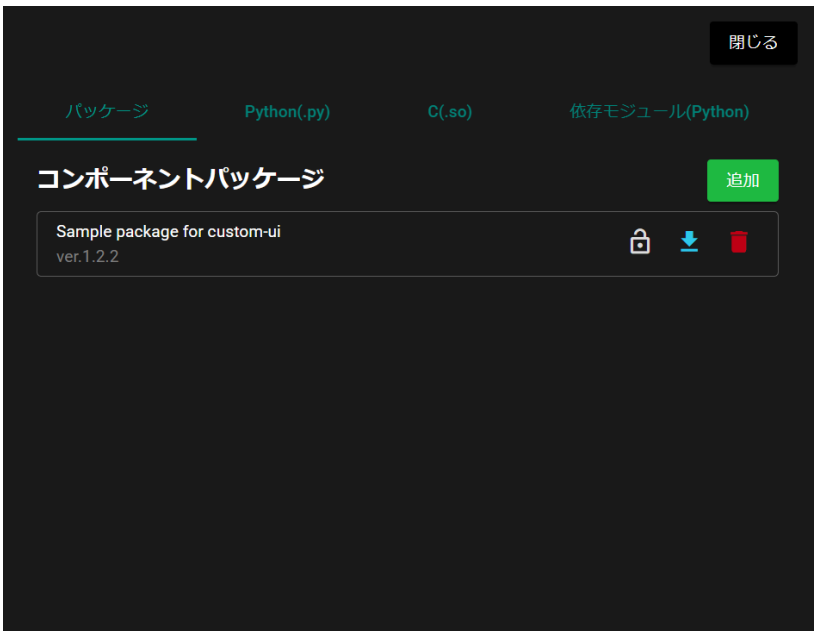
- 登録/更新
- 削除
- ダウンロード

次の手順で、パッケージに保護をかけることができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



2. 対象パッケージの右にある「鍵アイコン（保護）」を押下します。



3. 保護解除する際に必要となるパスワードを入力して、「保護」を押下します。



> **警告**
>
> - パスワードは復元できない形式で保存され、一度設定すると保護を解除するまで再設定できません。
> - 保護設定したパスワードを忘れた場合、該当のパッケージは操作不能となります。
>
> パスワードの取り扱いには十分ご注意ください。

4. コンポーネントパッケージが保護状態になります。



6.2.6.4.5 保護の解除

次の手順で、パッケージの保護を解除することができます。

1. 設定メニューアイコンを押下し、「カスタムコンポーネント」を選択します。



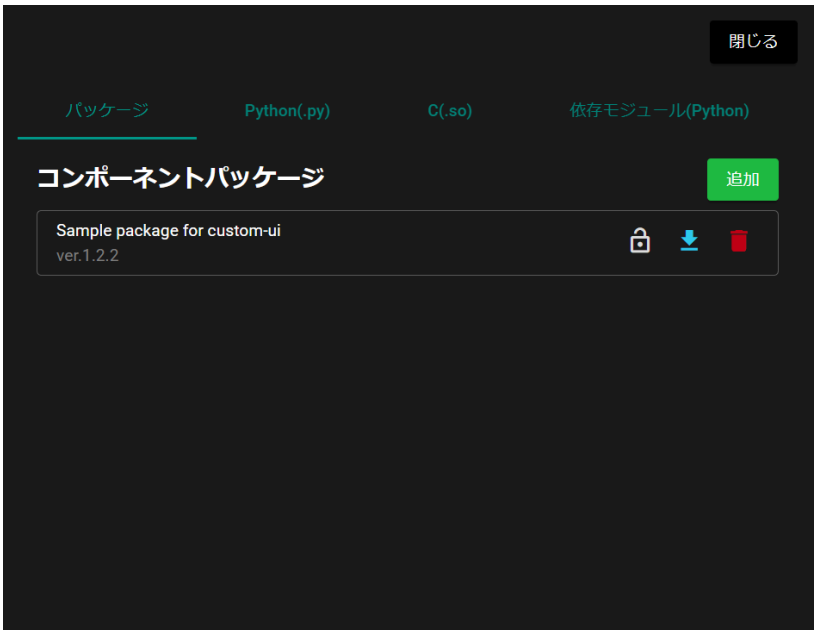
2. 対象パッケージの右にある「鍵アイコン（保護解除）」を押下します。



3. 保護をかけた際に設定したパスワードを入力して、「保護解除」を押下します。



4. コンポーネントパッケージの保護が解除されます。



6.3 制御プログラムとの連携（WebAPI）

6.3.1 WebAPIの概要

SpeeDBee Synapseの一部の操作は、APIによって外部から行うことができます。

6.3.2 WebAPIの実行方法

ここでは例として、curlコマンドによってAPIを実行する方法を記載します。
curlがインストールされていない場合、curlの公式サイトからcurlをダウンロードして、ご利用中のマシンへcurlをインストールして下さい。

1. curlコマンドを実行する環境を開きます。Linuxの場合はターミナルを、Windowsの場合はコマンドプロンプトを開きます。
2. 「API仕様」の各APIの「curlコマンドによるリクエスト」をコピーし、curlコマンドを実行する環境にペーストします。
   ※「curlコマンドによるリクエスト」の一部は、置き換える必要があります。詳細は、「共通事項」をご覧ください。
3. Enterキーを押下するとAPIが実行され、レスポンスが表示されます。レスポンスの詳細は、「API仕様」の各APIの「レスポンス」をご覧ください。

「API仕様」の各APIの「curlコマンドによるリクエスト」では、次の項目が指定されています。curlコマンド以外の方法でAPIを実行する場合は、同じ項目を指定する必要があります。

項目	説明
HTTPメソッド	-X "~~~"という形で指定されています。
リクエストURL	"http://"で始まるURLです。
HTTPヘッダ	-H "~~~"という形で指定されています。
POSTメソッドで送信するデータ	-d "~~~"という形で指定されています。

6.3.3 WebAPI仕様

APIによって行うことができる操作の一覧について、その概要・リクエスト方法・レスポンスを掲載します。 API仕様を理解し、各種プログラムから同一のリクエストを送信することで、各種プログラミング言語からの利用や制御プログラムとの連携が可能となります。

詳細なAPI仕様のドキュメントは、External API Interfaceを参照してください。

6.3.3.1 共通事項

次の一覧は、以降のリクエスト例に掲載されている内容について、共通となる事項です。

- は、SpeeDBee Synapseが稼働しているホスト・ポートに置き換えてください。
- <アクセストークン>は、「アクセストークン発行」で発行したアクセストークンに置き換えてください。
- <パネルID>は、取得したいパネルのIDを指定します。そのパネルに登録されている全てのコンポーネントの実行状態を取得します。パネルIDは、「全コンポーネントの実行状態取得」によって取得することができます。
- <コンポーネントID>は、実行状態を取得したいコンポーネントのIDを指定します。コンポーネントIDは、「全コンポーネントの実行状態取得」によって取得することができます。

6.3.3.2 全コンポーネントの実行状態取得

- 概要

  SpeeDBee Synapseに登録されている全てのコンポーネントの実行状態を取得します。
- リクエストURL

```
http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status
```

- curlコマンドによるリクエスト

  Linux

```
$ curl -X "GET" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status" -H
"accept: application/json" -H "X-hive-api-key: <アクセストークン>"
```

  Windows

```
$ curl.exe -X "GET" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status" -H
"accept: application/json" -H "X-hive-api-key: <アクセストークン>"
```

- レスポンス

コード	説明
200	正常

コード	説明
403	認証エラー
503	SpeeDBee Synapseがリクエストを処理できない状態

正常レスポンスは、以下の形式です。

```
{
  "status": "SpeeDBee Synapseのステータス",
  "components": [
    {
      "id": "コンポーネントID",
      "name": "コンポーネント名",
      "panel_id": "コンポーネントが登録されているパネルのID",
      "panel_name": "コンポーネントが登録されているパネルの名前",
      "status": "コンポーネントのステータス",
      "errors": [
        {
          "id": "エラーID",
          "parameter": ["エラーメッセージ"]
        }
      ]
    }
  ]
}
```

※1 "components"には、コンポーネントの実行状態が格納されます。コンポーネントが存在しない場合は、空配列となります。
※2 "errors"には、コンポーネントのエラー情報が格納されます。エラーが発生していない場合は、空配列となります。

6.3.3.3 個別コンポーネントの実行状態取得

- 概要

  SpeeDBee Synapseに登録されている個別のコンポーネントの実行状態を取得します。

- リクエストURL

```
http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status
```

- curlコマンドによるリクエスト
  - パネル毎

Linux

```
$ curl -X "POST" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status" -H "accept: application/json" -H "X-hive-api-key: <アクセストークン>" -H 'Content-Type: application/json' -d "{\"panel_ids\": [<パネルID>]}"
```

Windows

```
$ curl.exe -X "POST" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status" -H "accept: application/json" -H "X-hive-api-key: <アクセストークン>" -H "Content-Type: application/json" -d "{\"panel_ids\": [<パネルID>]}"
```

- コンポーネント毎

Linux

```
$ curl -X "POST" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status" -H "accept: application/json" -H "X-hive-api-key: <アクセストークン>" -H 'Content-Type: application/json' -d "{\"component_ids\": [\"<コンポーネントID>\"]}"
```

Windows

```
$ curl.exe -X "POST" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/status" -H "accept: application/json" -H "X-hive-api-key: <アクセストークン>" -H "Content-Type: application/json" -d "{\"component_ids\": [\"<コンポーネントID>\"]}"
```

- レスポンス

コード	説明
200	正常
403	認証エラー
422	バリデーションエラー
503	SpeeDBee Synapseがリクエストを処理できない状態

正常レスポンスは、以下の形式です。

```
{
  "status": "SpeeDBee Synapseのステータス",
  "components": [
    {
      "id": "コンポーネントID",
      "name": "コンポーネント名",
      "panel_id": "コンポーネントが登録されているパネルのID",
      "panel_name": "コンポーネントが登録されているパネルの名前",
      "status": "コンポーネントのステータス",
      "errors": [
        {
          "id": "エラーID",
          "parameter": ["エラーメッセージ"]
        }
      ]
    }
  ]
}
```

※1 "components"には、指定されたコンポーネントの実行状態が格納されます。指定されたコンポーネントが存在しない場合は、空配列となります。
※2 "errors"には、コンポーネントのエラー情報が格納されます。エラーが発生していない場合は、空配列となります。

6.3.3.4 コンポーネント一括実行開始

- 概要

  SpeeDBee Synapseに登録されているコンポーネントを開始します。

- リクエストURL

```
http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/start
```

- curlコマンドによるリクエスト
  - パネル毎

    Linux

```
$ curl -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/start" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"panel_ids\": [<パネルID>]}"
```

    Windows

```
$ curl.exe -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/start" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"panel_ids\": [<パネルID>]}"
```

- コンポーネント毎

  Linux

```
$ curl -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/start" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"component_ids\": [\"<コンポーネントID>\"]}"
```

  Windows

```
$ curl.exe -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/start" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"component_ids\": [\"<コンポーネントID>\"]}"
```

- レスポンス

コード	説明
200	正常
400	ポイント不足によりコンポーネントの実行を開始できない
403	認証エラー
422	バリデーションエラー
503	SpeeDBee Synapseがリクエストを処理できない状態

正常レスポンスは、以下の形式です。

```
[
  {
    "id": "コンポーネントID",
    "panel_id": "パネルID",
    "success": "コンポーネント開始の成否",
    "error": "エラー情報"
  }
]
```

※ 正常レスポンスの配列には、リクエストで指定されたコンポーネントの一覧が格納されます。指定されたコンポーネントが存在しない場合は、空配列となります。

6.3.3.5 コンポーネント一括実行停止

- 概要

SpeeDBee Synapseに登録されているコンポーネントを停止します。

- リクエストURL

```
http://<SpeeDBee Synapseのホスト・ポート>/ext_api/v1/components/stop
```

- curlコマンドによるリクエスト
  - パネル毎

Linux

```
$ curl -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/stop" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"panel_ids\": [<パネルID>]}"
```

Windows

```
$ curl.exe -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/stop" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"panel_ids\": [<パネルID>]}"
```

  - コンポーネント毎

Linux

```
$ curl -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/stop" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"component_ids\": [\"<コンポーネントID>\"]}"
```

Windows

```
$ curl.exe -X "PUT" "http://<SpeeDBee Synapseのホスト:ポート>/ext_api/v1/components/stop" -H
"accept: application/json" -H "x-hive-api-key: <アクセストークン>" -H "Content-Type:
application/json" -d "{\"component_ids\": [\"<コンポーネントID>\"]}"
```

- レスポンス

コード	説明
200	正常
403	認証エラー
422	バリデーションエラー
503	SpeeDBee Synapseがリクエストを処理できない状態

正常レスポンスは、以下の形式です。

```
[
  {
    "id": "コンポーネントID",
    "panel_id": "パネルID",
    "success": "コンポーネント停止の成否",
    "error": "エラー情報"
  }
]
```

※ 正常レスポンスの配列には、リクエストで指定されたコンポーネントの一覧が格納されます。指定されたコンポーネントが存在しない場合は、空配列となります。

6.4 Grafanaとの連携

6.4.1 Grafana連携の概要

Grafanaは、システムのデータを可視化するためのツールです。 Grafanaと連携することによって、SpeeDBee Synapseが収集したデータを用いてグラフを作成することができます。

Grafanaについての詳細は、公式サイトをご覧ください。

注意事項

本章の説明では、バージョン11.5.2のGrafanaを使用しています。バージョンの違いにより、操作の内容に違いがある場合がございます。

SpeeDBee SynapseのGrafana連携機能では、Grafanaのダッシュボードに「SpeeDBee Synapse」というフォルダを作成します。 このフォルダはSynapseが管理・更新するため、Grafana画面から手動で変更しないようご注意ください。変更・保存した場合、SynapseからのGrafana更新操作により、ユーザーによる変更は破棄されます。 手動でダッシュボードを管理したい場合は、「SpeeDBee Synapse」フォルダ以外の場所に任意のダッシュボードを追加してください。

6.4.2 Grafanaのインストール

- 本項目の手順は、Grafanaをインストールする環境で行ってください。
- 以降のLinuxの手順は、システム管理権限を必要とする操作には、すべてsudoを使用しています。
- Linuxで、root権限を持つユーザーで操作する場合は、sudoを除外したコマンドを入力してください。
- Windowsでは、Windows PowerShellを「管理者として実行」で起動しておきます。
- 管理者権限については、OSのシステム管理者へ相談して下さい。

Data Flow Controller版では、初期状態でGrafanaがインストールされているため、インストールは不要です。

1. Grafanaをインストールします。インストール手順は、公式サイトを参考にしてください。
2. 以下のコマンドを実行し、プラグインのインストール先ディレクトリを作成します。

Linux

```
$ sudo mkdir -p /var/lib/grafana/plugins/SpeeDBSynapse/
```

Windows

```
PS >New-Item -ItemType Directory -Path "C:\Program
Files\GrafanaLabs\grafana\data\plugins\SpeeDBeeSynapse" -Force
```

Standalone Linux Binaries

```
$ sudo mkdir -p <展開先ディレクトリ>/grafana-9.1.3/data/plugins/SpeeDBeeSynapse/
```

3. 任意のディレクトリに、「synapse_gplugin.tar.gz」を配置します。

> 「synapse_gplugin.tar.gz」は、GrafanaとSpeeDBee Synapseが連携するためのプラグインです。SpeeDBee Synapseのインストーラと合わせて配布しています。ご不明な場合はお問い合わせください。
>
> お問合せフォーム: https://www.saltyster.com/contact/

4. 以下のコマンドを実行し、「synapse_gplugin.tar.gz」を配置したディレクトリに移動します。

Linux

```
$ cd <synapse_gplugin.tar.gzを配置したディレクトリ>
```

Windows

```
PS >cd <synapse_gplugin.tar.gzを配置したディレクトリ>
```

Standalone Linux Binaries

```
$ cd <synapse_gplugin.tar.gzを配置したディレクトリ>
```

5. 以下のコマンドを実行し、プラグインをインストールします。

Linux

```
$ sudo tar xzf synapse_gplugin.tar.gz -C /var/lib/grafana/plugins/SpeeDBSynapse/
```

Windows

```
PS >tar xzf synapse_gplugin.tar.gz -C "C:\Program
Files\GrafanaLabs\grafana\data\plugins\SpeeDBeeSynapse"
```

Standalone Linux Binaries

```
$ sudo tar xzf synapse_gplugin.tar.gz -C <展開先ディレクトリ>/grafana-
9.1.3/data/plugins/SpeeDBeeSynapse/
```

6. 以下のコマンドを実行し、Grafanaを再起動します。

Linux

```
$ sudo systemctl restart grafana-server
```

Windows

```
PS >Restart-Service grafana
```

Standalone Linux Binaries

```
$ cd <展開先ディレクトリ>/grafana-9.1.3
$ sudo ./bin/grafana-server
```

systemdへのサービス登録

systemctlでサービスを起動したい場合は、次の手順に従ってsystemdにサービスを登録してください。

a. サービスユニットファイルを作成

/lib/systemd/system/grafana-server.service

```
[Unit]
Description=Grafana server

[Service]
Type=simple
WorkingDirectory=<展開先ディレクトリ>/grafana-9.1.3
ExecStart=<展開先ディレクトリ>/grafana-9.1.3/bin/grafana-server
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

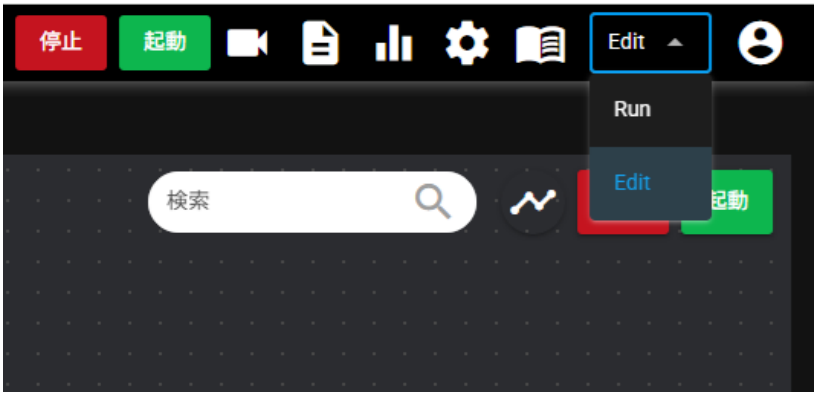
b. systemdにサービスを読み込ませ、サービスを起動。

```
$ sudo systemctl daemon-reload
$ sudo systemctl start grafana-server
```

6.4.3 Grafanaのセットアップ

本項目の手順の前に、「Grafanaのインストール」が完了している必要があります。

1. SpeeDBee SynapseのEditモードの画面を開きます。



2. 設定メニューアイコンを押下し、「Grafanaセットアップ」を選択します。



3. 次の項目を入力し、「自動セットアップ」を押下します。



項目	説明
ホスト	Grafanaをインストールした環境のホスト名またはIPアドレス
ポート	Grafanaが稼働しているポート ※初期設定では、Grafanaはポート3000で稼働します。

4. 次の項目を入力し、「セットアップ」を押下します。



項目	説明
ユーザー名	Grafanaへのログインに必要なユーザー名 ※初期設定値は"admin"です。
パスワード	Grafanaへのログインに必要なパスワード ※初期設定値は"admin"です。

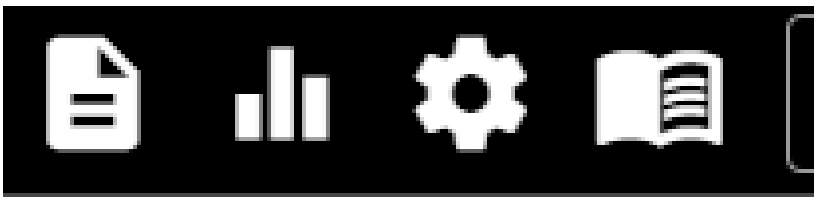
5. セットアップが完了した場合、APIキーが自動で入力されています。
   また「テスト」を押下すると、画面下部に「テスト接続に成功しました。」と表示されます。



6. 「保存」を押下します。



7. ヘッダのGrafanaアイコンを押下すると、Grafana画面に遷移できるようになっています。

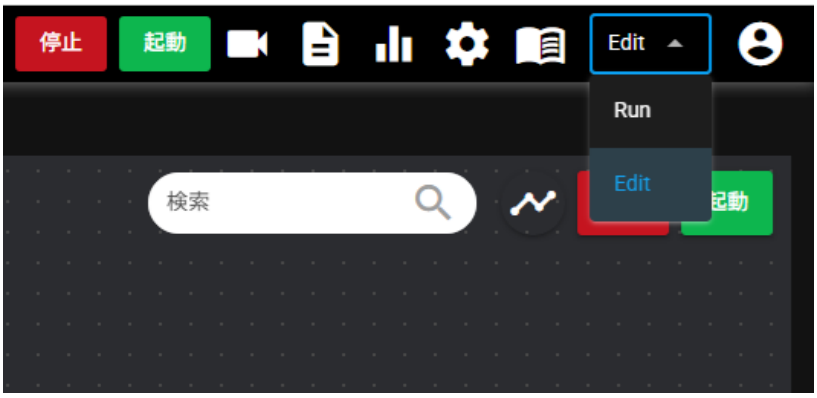


6.4.4 グラフの作成

本項目の手順の前に、「Grafanaのインストール」、「Grafanaのセットアップ」が完了している必要があります。

6.4.4.1 データの登録

1. SpeeDBee SynapseのEditモードの画面を開きます。



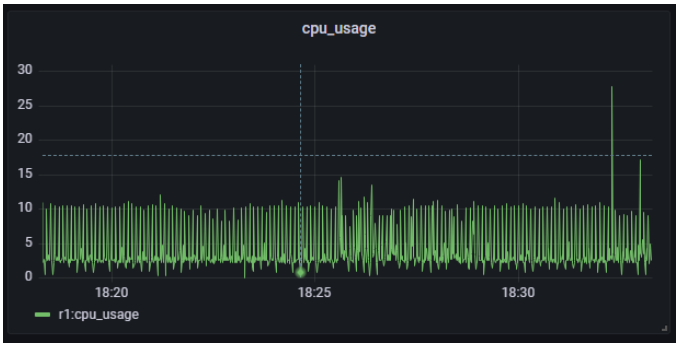
2. 設定メニューアイコンを押下し、「Grafanaセットアップ」を選択します。

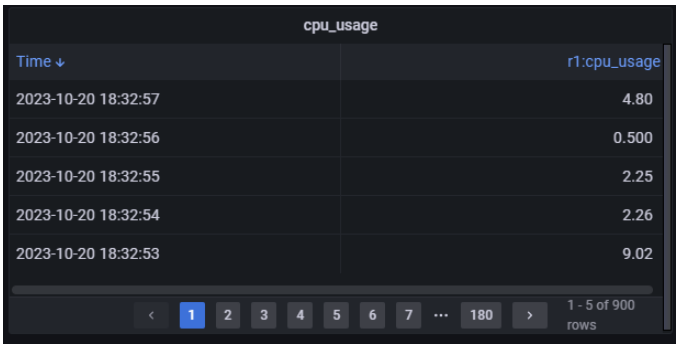
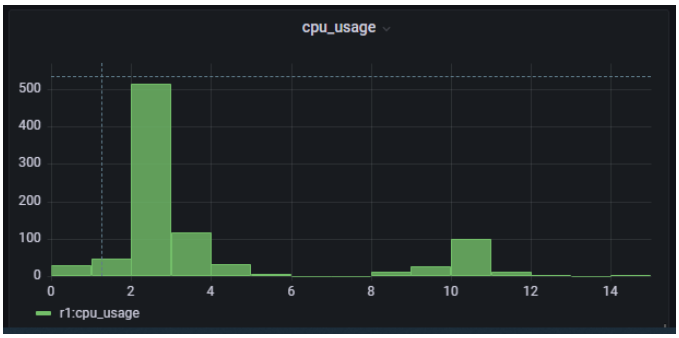


3. コンポーネントとグラフ種類を選択します。グラフ種類は個別で選択、もしくは「グラフ種類一括適用」を使用して一括で変更することも可能です。



グラフ種類は次の通りです。

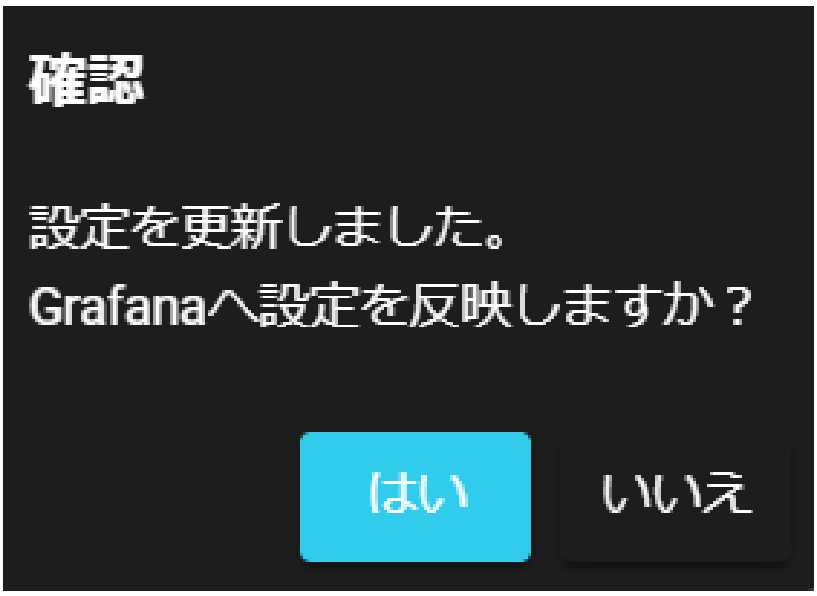
種類	説明	表示例
折れ線グラフ	値を時系列で表示します。	
現在値	折れ線グラフを背景にし、現在値を表示します。	

種類	説明	表示例
テーブル	値をテーブル形式で表示します。	
ヒストグラム	値の分布を計算し、棒グラフとして表示します。	
-	グラフを作成しません。	

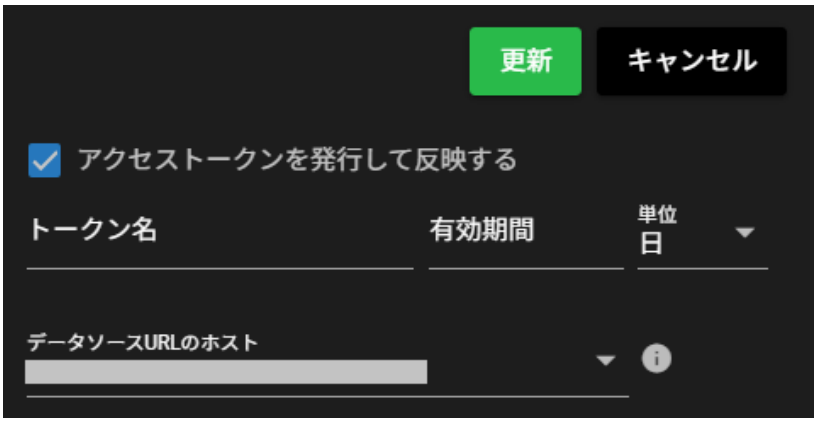
4. 「保存」を押下します。



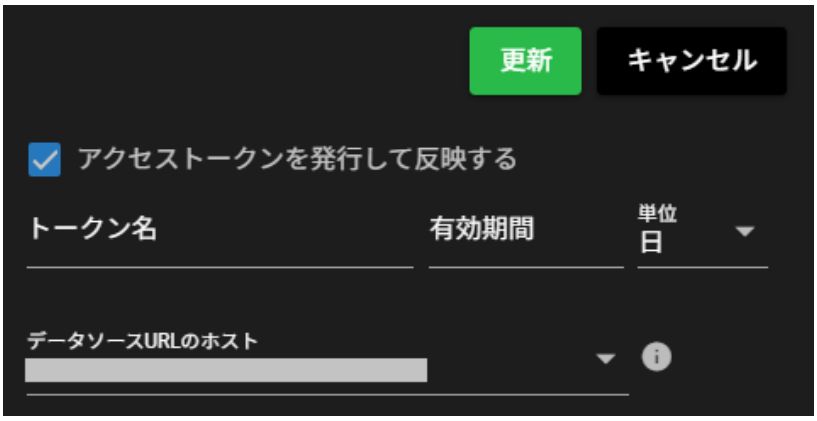
5. 「はい」を押下します。



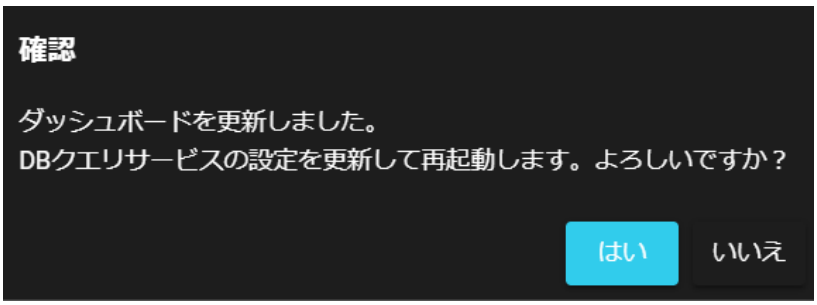
6. 「アクセストークンを発行して適用する」を選択し、トークンの名前と有効期限を入力します。また、「データソースURLのホスト」はGrafanaから見たSpeeDBee Synapseがインストールされたデバイスの IPアドレスまたはホスト名となります。



7. 「更新」を押下します。



8. 「はい」を押下します。
※アクセストークンを発行して更新した場合、DBクエリサービスの設定更新と再起動が必要になります。 手動で更新、起動する場合は、「DBクエリサービスの起動」を参照してください。

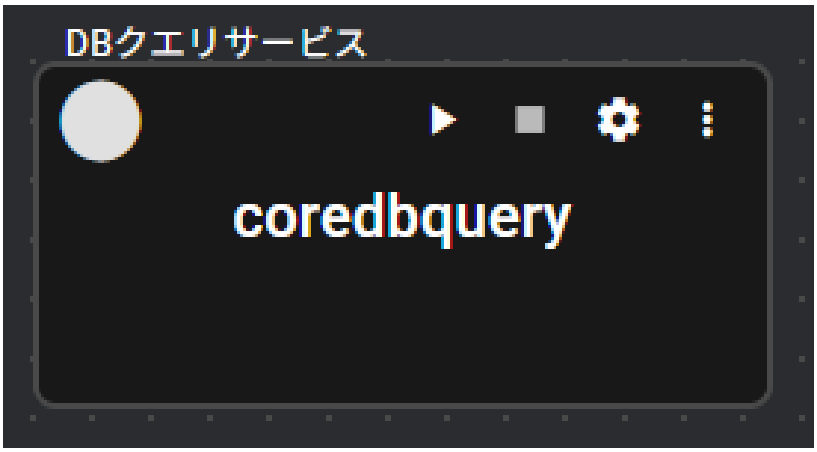


6.4.4.2 DBクエリサービスの起動

1. 「システム」のパネルタブを押下し、システムパネルを開きます。



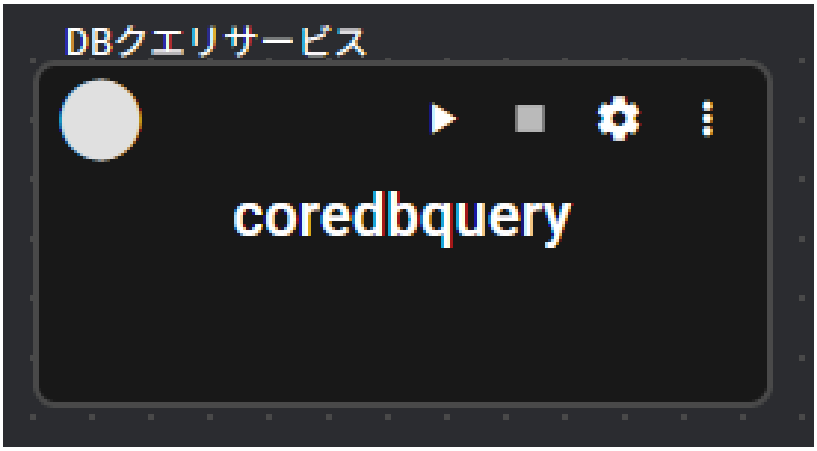
2. DBクエリサービスの設定アイコンを押下します。



3. 「トークン認証」をONにし、全てのトークンを選択して「保存」を押下します。

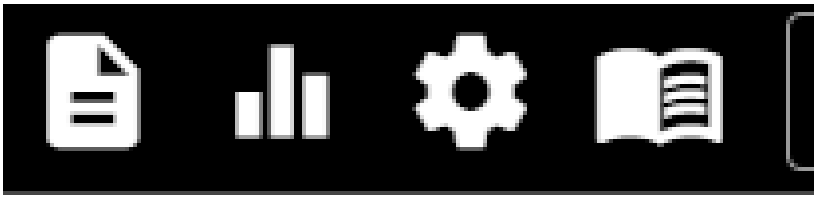


4. コアDBクエリの起動アイコンを押下します。既に起動している場合は、停止アイコンを押下した後に、起動アイコンを押下します。

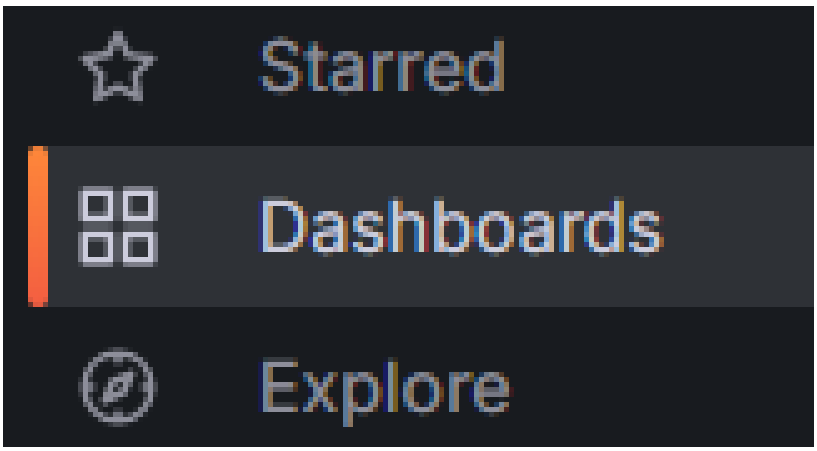


6.4.5 グラフの閲覧

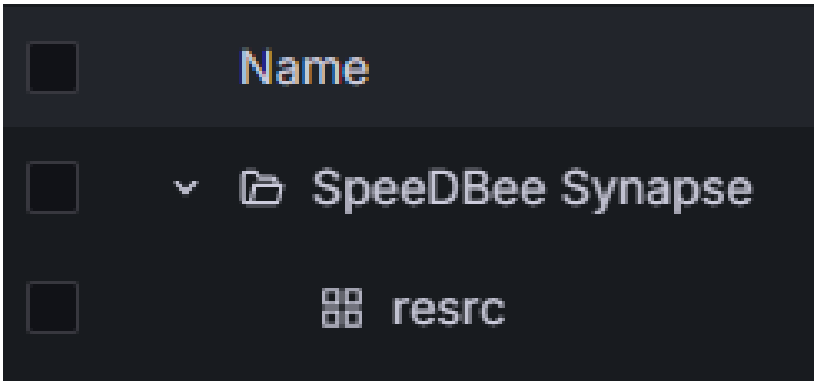
1. ヘッダのGrafanaアイコンを押下し、Grafana画面を開きます。



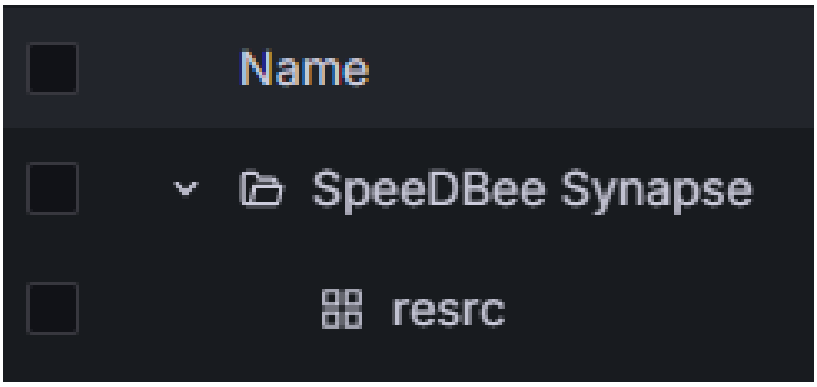
2. 左メニューの「Dashboards」アイコンを押下します。



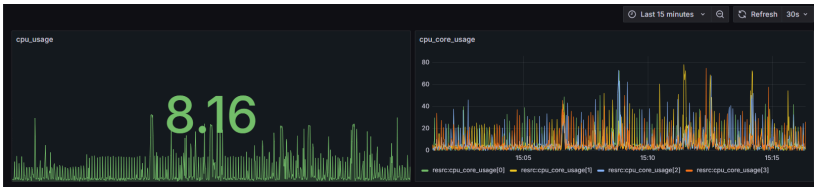
3. 「SpeeDBee Synapse」というフォルダの中に、「データの登録」で登録したコンポーネントが一覧で表示されます。



4. コンポーネントを選択します。



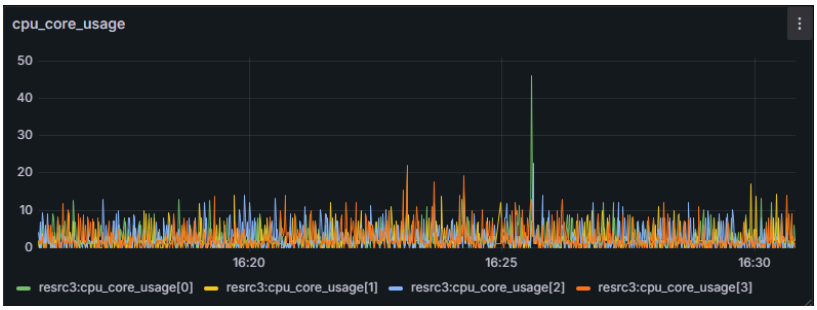
5. 選択したコンポーネントの各データのグラフが、一覧で表示されます。



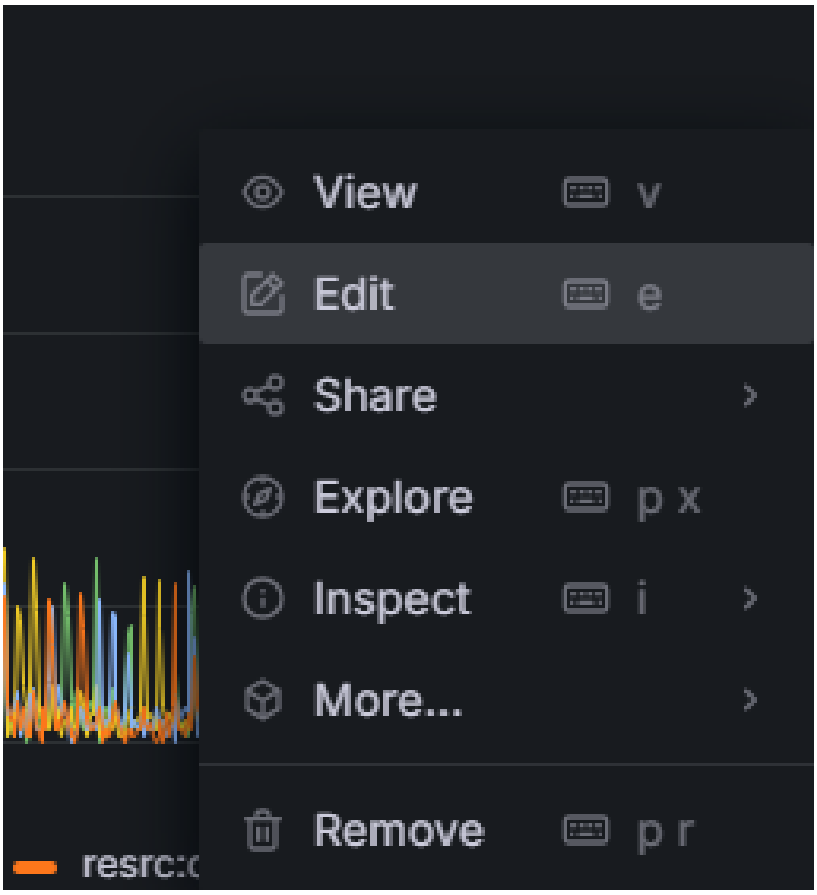
6.4.6 グラフのデータ件数上限の変更

データが一部分しか表示されていない場合、データの件数が上限を超えている場合があります。 下記の手順で、データ件数の上限を変更することができます。

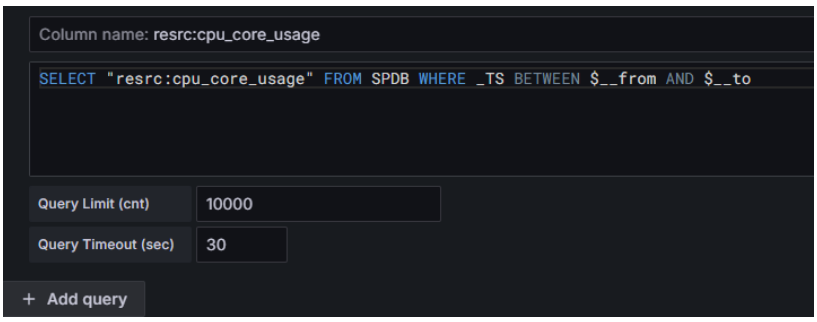
1. 「グラフの閲覧」の手順を実施し、グラフの一覧を表示します。
2. グラフ右上のメニューアイコンを押下します。



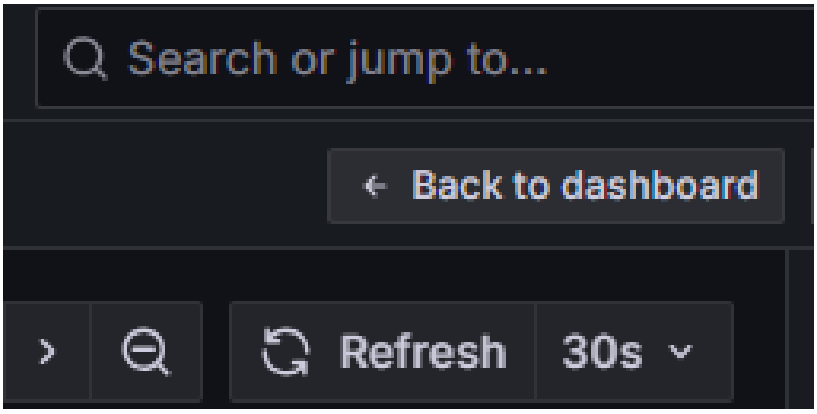
3. メニューが表示されます。その中から「Edit」を選択します。



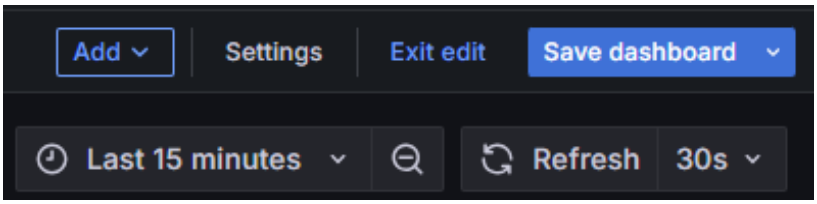
4. グラフの設定画面が表示されます。この画面の「Query Limit (cnt)」という設定項目で、データ件数の上限が設定されています。 値を変更すると、グラフに反映されます。
（「Query Limit (cnt)」のデフォルト値は10000です。）



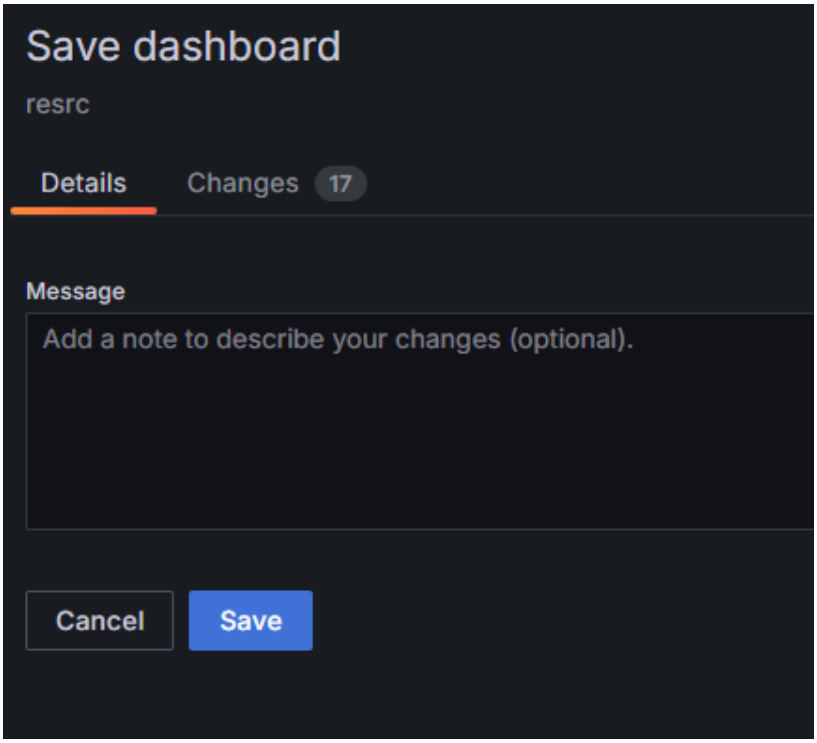
5. 値を変更した後は、ESCキーを押下するか、画面右上の「Back to dashboard」アイコンを押下します。



6. グラフ一覧の画面に戻ります。画面右上の「Save dashboard」アイコンを押下します。



7. 「Save」を押下すると、設定が保存されます。

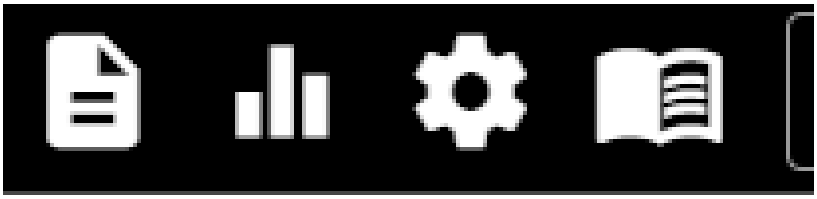


> **⚠ 設定がリセットされるタイミング**
>
> 設定を変更した後、「データの登録」の手順を実施すると、設定がリセットされるのでご注意ください。

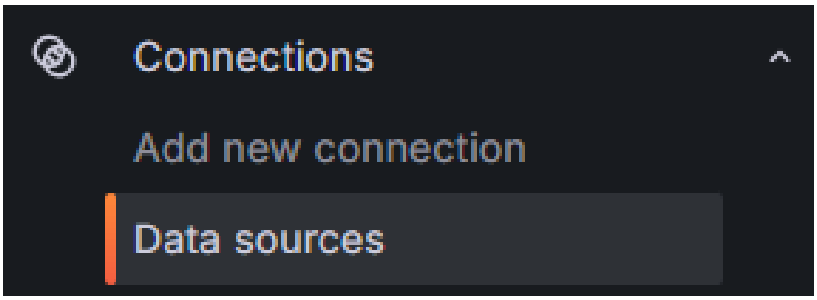
6.4.7 Grafanaデータソースの手動設定

Grafanaを手動で操作して、データ参照を行う方法について説明します。
この設定は、Grafanaセットアップにて設定済みであれば特に設定を行う必要はありません。

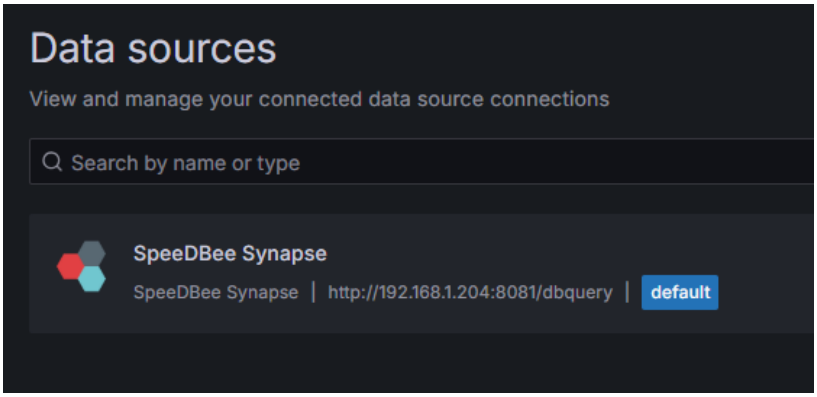
1. 「アクセストークン発行」を参考に、アクセストークンを発行します。
   ※発行したアクセストークンはのちに使うので、控えておきます。
2. 「DBクエリサービスの起動」の手順に従って、DBクエリサービスを再起動します。
3. ヘッダのGrafanaアイコンを押下し、Grafana画面を開きます。



4. 左メニューの「Connections」アイコンにカーソルを合わせ、表示されたメニューの中から「Data sources」を選択します。

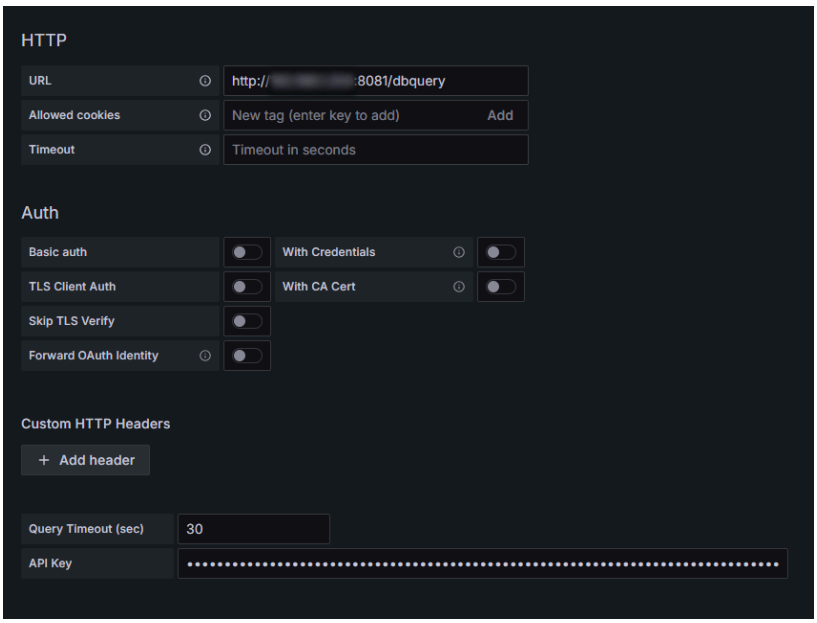


5. 「SpeeDBee Synapse」を選択します。

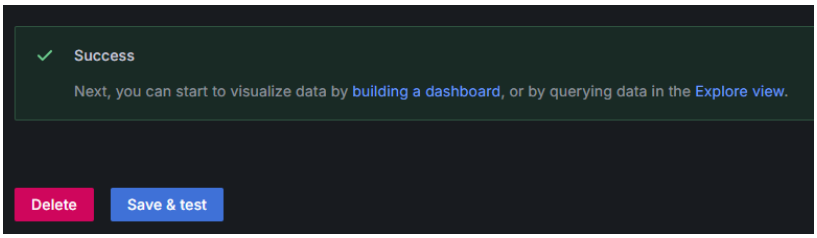


6. 以下の項目を入力します。

項目	説明
URL	「http://{ SpeeDBee Synapseが稼働している環境のIPアドレス }:{ DBクエリサービスが稼働しているポート }/dbquery」と入力してください。 ※DBクエリサービスが稼働しているポートは、DBクエリサービスコンポーネントの設定項目「ポート」で確認、変更することができます。詳細は「DBクエリサービス」をご覧ください。
API Key	上の手順で発行したアクセストークンを入力してください。



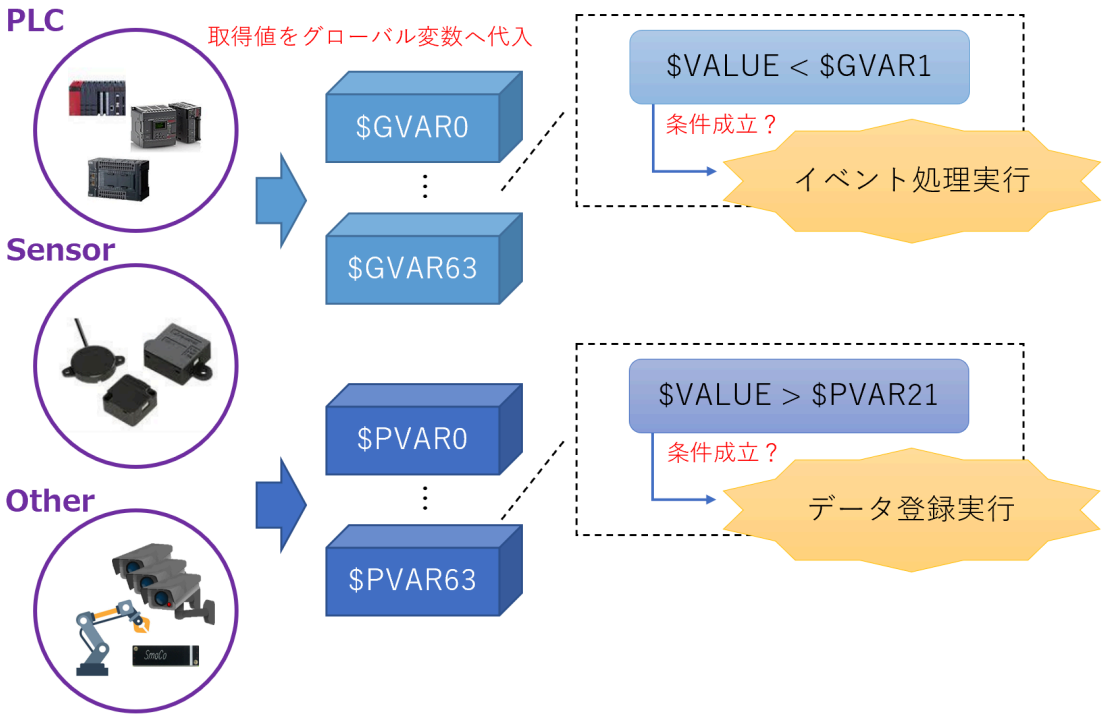
7. 「Save & test」を押下し、"Success"というメッセージが表示されれば、データソースの設定は完了です。



6.5 グローバル変数

6.5.1 グローバル変数の概要

グローバル変数は、頻繁に変化するデータをメモリ上の変数へ格納する機能です。この変数をSpeeDBee Synapseの各種機能（数値変換、登録条件、コンポーネントの演算式、条件式、SQLなど）で利用できます。
これにより、システム稼働中の条件式閾値の更新や特定のデータ値による他データ登録の抑制などが可能となります。



グローバル変数の利用できるのは、演算式、条件式、SQL内となります。 演算式、条件式については、こちらを参照してください。
SQLについては、こちらを参照してください。

6.5.2 グローバル変数の構文

グローバル変数には、通常のグローバル変数と永続化グローバル変数の2種類の指定方法があります。どちらも64個の変数を使用できます。
通常のグローバル変数は、SpeeDBee Synapseのコアを停止（稼働中のマシンを再起動するなど）すると、値は失われます。 永続化グローバル変数は、SpeeDBee Synapseのコアを停止しても、次の起動時に最終値が参照できます。

- $GVAR0 ～ $GVAR63（グローバル変数）

  GVAR（グローバル変数）は、SpeeDBee Synapseのコアを再起動する毎に、0で初期化されます。値の代入が行われるまで0のままとなります。

- $PVAR0 ～ $PVAR63（永続化グローバル変数）

PVAR（永続化グローバル変数）は、SpeeDBee Synapseのコアを初めて起動する時に、0で初期化されます。値の代入が行われると、コアを再起動しても値を保持します。

SpeeDBee Synapseの再起動時に、前回起動時に代入された値を使用したい場合は、永続化グローバル変数を使用してください。

1. 代入方法

```
GVAR0 := $VALUE
```

```
PVAR15 := ($VALUE * 20)
```

※代入するグローバル変数の先頭には「$」は必要ありません。

- 数値変換における代入時の注意点

  ```
  PVAR0 := $VALUE * 2
  ```

  上記のように代入を指定すると、登録される値は$VALUEに2を乗算した値となります。 登録される値が変化することに注意してください。

- 条件式における代入時の注意点

  ```
  GVAR0 := $VALUE
  ```

  条件式では、代入のみを指定することはできません。条件式内で、代入したい場合は 以下のように条件式としての指定が必要です。

  ```
  0 < (GVAR0 := $VALUE)
  ```

  この場合、$VALUEが0より大きい場合条件成立となります。GVAR0への代入は常に行われます。

2. 参照方法

```
$GVAR0 < $VALUE
```

```
$PVAR15 > ($VALUE + 20)
```

※参照するグローバル変数の先頭には「$」を付けます。

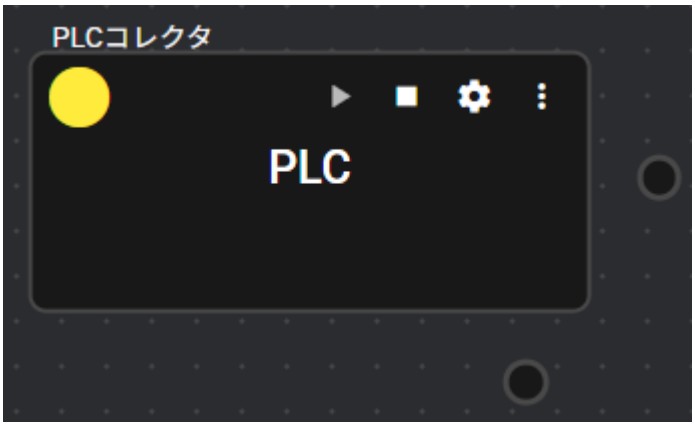
7 トラブルシュート

本章では、トラブル発生時の対処方法について説明します。

7.1 異常発生時共通の手順

本節では、SpeeDBee Synapseで問題が発生した際の、基本的な対処方法を説明します。

1. ステータスランプが赤・黄になっているコンポーネントを右クリックします。



2. 「ステータスタイムライン」を選択します。



3. タイムラインに表示されているエラーコードを確認します。



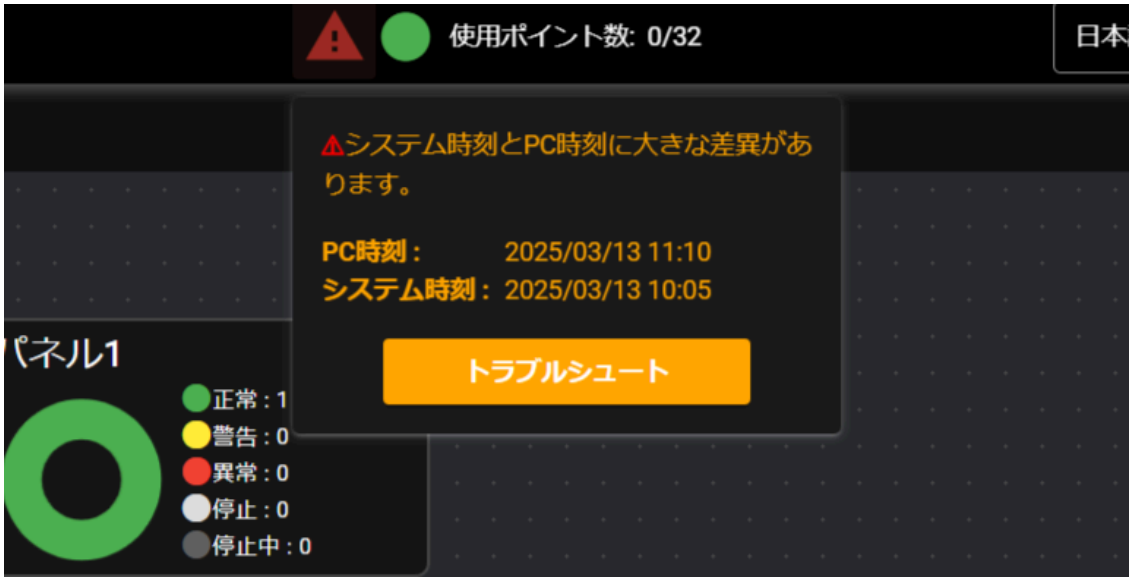
4. 本マニュアルの右上の検索ボックスから、タイムラインに表示されていたエラーコードを検索します。
5. エラーコードの概要を把握し、処置方法を実施します。
6. 処置方法を実施してもエラーが解消しない場合、「製造元に問い合わせ」を参考に、製造元に問い合わせください。

7.2 よくあるトラブルとその対処方法

ここでは、よくあるトラブルとその対処方法について解説します。

7.2.1 時刻設定に関する問題と対処方法

SpeeDBee Synapseの稼働しているマシンと使用ブラウザの時刻が大幅にずれている場合、画面上部に次の警告が表示されます。

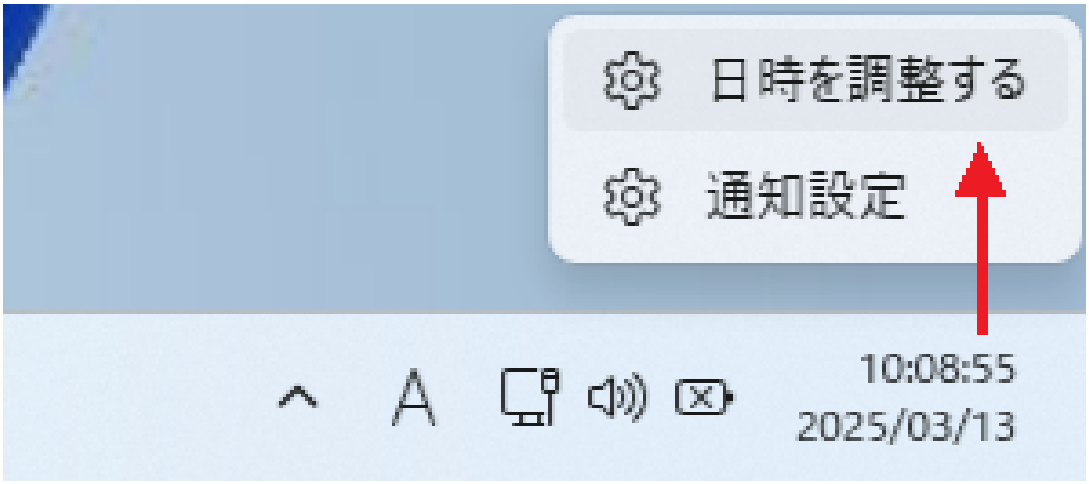


この警告が表示されている場合、SpeeDBee Synapseは、動作中にエラーが発生します。
これを回避するには、該当環境でシステム時刻を設定する方法がある場合は、設定内容をPC時刻の設定と合わせてください。詳細は当該環境のマニュアルをご確認ください。

7.2.1.1 SpeeDBee SynapseがWindowsで動作している場合

Windowsの時刻設定は、下記の手順で設定できます。

1. SpeeDBee Synapseの稼働しているマシンで、Synapseを停止します。停止方法については、ここを参照してください。
2. タスクバーの右端に表示されている日付と時刻を右クリックし、「日時を調整する」をクリックします。



PCがインターネットに接続している場合は次の手順に進み、インターネットに接続していない場合は手順4から行ってください。

3. 「日付と時刻」画面が表示されます。PCがインターネットに接続している場合、「時刻を自動的に設定する」をオンにして1分待ちます。その後手順6へ進んでください。



もし時刻の自動同期ができなかったり、自動同期を行っても問題が解消しない場合、PCがインターネットに接続されていない場合は、手動で時刻を設定する必要があります。次の手順に進んでください。

4. 「時刻を自動的に設定する」をオフにして「日付と時刻を手動で設定する」の変更ボタンを選択すると、時刻の設定メニューが開きます。



5. メニューで現在の時刻を設定し、「変更」ボタンを選択して1分待ちます。



6. SpeeDBee Synapseの稼働しているマシンで、Synapseを起動します。起動方法については、ここを参照してください。
7. 時刻が正しく設定されると、SpeeDBee Synapseの画面上部で警告が表示されなくなります。



7.2.1.2 SpeeDBee SynapseがLinuxで動作している場合

Linuxの時刻設定は、下記の手順で設定できます。

> Data Flow Controller版では、Data Flow ControllerのWebUIで設定できます。

1. SpeeDBee Synapseの稼働しているマシンで、Synapseを停止します。停止方法については、ここを参照してください。
2. Linuxの現在の時刻を確認します。下記は現在システムに設定されている時刻を確認するコマンドです。

   コマンド

   ```
   $ timedatectl
   ```

   時刻が表示された結果例は以下の通りです。「Local time」が現在設定されているシステム時刻になります。時刻がずれている場合は、設定を変更する必要があります。

   ```
         Local time: Wed 2025-03-12 05:44:33 UTC
     Universal time: Wed 2025-03-12 05:44:33 UTC
           RTC time: Wed 2025-03-12 05:44:33
          Time zone: Universal (UTC, +0000)
   Network time on: yes
   NTP synchronized: yes
    RTC in local TZ: no
   ```

3. 必要に応じて、タイムゾーンを正しい地域へ変更します。この設定は任意となるため、タイムゾーンが一致しているか、設定が不要な場合はこの手順を行わず、次の手順へ進んでください。
   例えばSpeeDBee Synapseが稼働しているマシンをJSTへ変更する場合、下記のコマンドを使用します。

   コマンド

   ```
   $ sudo timedatectl set-timezone Asia/Tokyo
   ```

   設定したいタイムゾーンがJST以外の場合、下記のコマンドで使用できるタイムゾーンを確認できます。

   コマンド

   ```
   $ timedatectl list-timezones
   ```

   タイムゾーンが変更された結果例は以下の通りです。手順1のコマンドで設定値を確認出来ます。

   ```
         Local time: Wed 2025-03-12 14:45:01 JST
     Universal time: Wed 2025-03-12 05:45:01 UTC
           RTC time: Wed 2025-03-12 05:45:01
          Time zone: Asia/Tokyo (JST, +0900)
   Network time on: yes
   NTP synchronized: yes
    RTC in local TZ: no
   ```

PCがインターネットに接続している場合は次の手順へ進み、インターネットに接続していない場合は手順5へ進んでください。

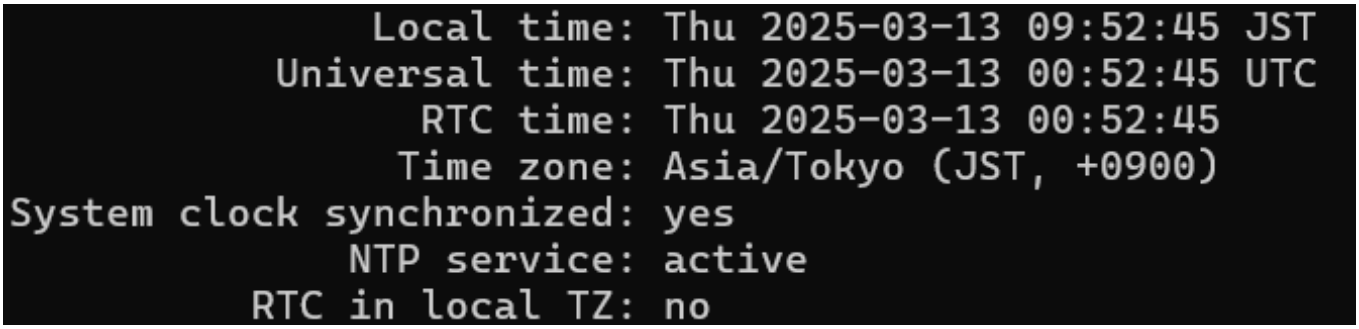
4. PCがインターネットに接続している場合、NTP Serverを有効にします。下記のコマンドを使用して設定後、1分待ち手順7へ進んでください。

コマンド

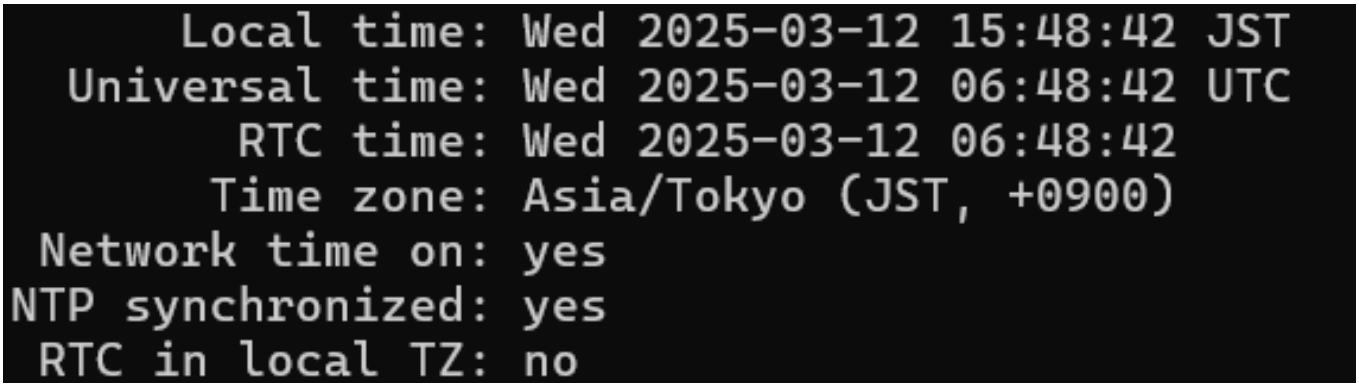
```
$  sudo timedatectl set-ntp yes
```

NTP Serverが有効化した結果例は以下の通りです。手順1のコマンドで設定値を確認出来ます。

OSがBlackBear以外の場合



OSがBlackBearの場合



もしもNTP serverの同期を行っても問題が解消しない場合や、PCがインターネットに接続されていない場合は、手動で時刻を設定する必要があります。次の手順に進んでください。

5. 時刻を手動で設定するために、NTP Serverを無効にします。下記のコマンドを使用して設定出来ます。

コマンド

```
$ sudo timedatectl set-ntp no
```

NTP Serverが無効化した結果例は以下の通りです。手順1のコマンドで設定値を確認出来ます。

OSがBlackBear以外の場合

```
               Local time: Wed 2025-03-12 16:03:22 JST
           Universal time: Wed 2025-03-12 07:03:22 UTC
                 RTC time: Thu 2025-03-13 00:57:15
                Time zone: Asia/Tokyo (JST, +0900)
System clock synchronized: no
              NTP service: inactive
          RTC in local TZ: no
```

OSがBlackBearの場合

```
      Local time: Wed 2025-03-12 14:42:04 JST
  Universal time: Wed 2025-03-12 05:42:04 UTC
        RTC time: Wed 2025-03-12 06:48:09
       Time zone: Asia/Tokyo (JST, +0900)
 Network time on: no
NTP synchronized: no
 RTC in local TZ: no
```

6. 次に日付と時刻を変更します。下記は2025年3月12日 16時03分20秒に設定する場合の例です。実際に使用する際は、日付と時刻を設定したい値に直して利用します。設定後1分待ちます。

   コマンド

   ```
   $ sudo date -s "2025-3-12 16:03:20"
   ```

   日付と時刻が変更された結果例は以下の通りです。手順1のコマンドで設定値を確認出来ます。

   ```
         Local time: Wed 2025-03-12 16:03:30 JST
     Universal time: Wed 2025-03-12 07:03:30 UTC
           RTC time: Wed 2025-03-12 07:03:30
          Time zone: Asia/Tokyo (JST, +0900)
    Network time on: no
   NTP synchronized: no
    RTC in local TZ: no
   ```

7. SpeeDBee Synapseの稼働しているマシンで、Synapseを起動します。起動方法については、ここを参照してください。
8. 時刻が正しく設定されると、SpeeDBee Synapseの画面上部で警告が表示されなくなります。



7.2.1.3 時刻設定に関する注意事項

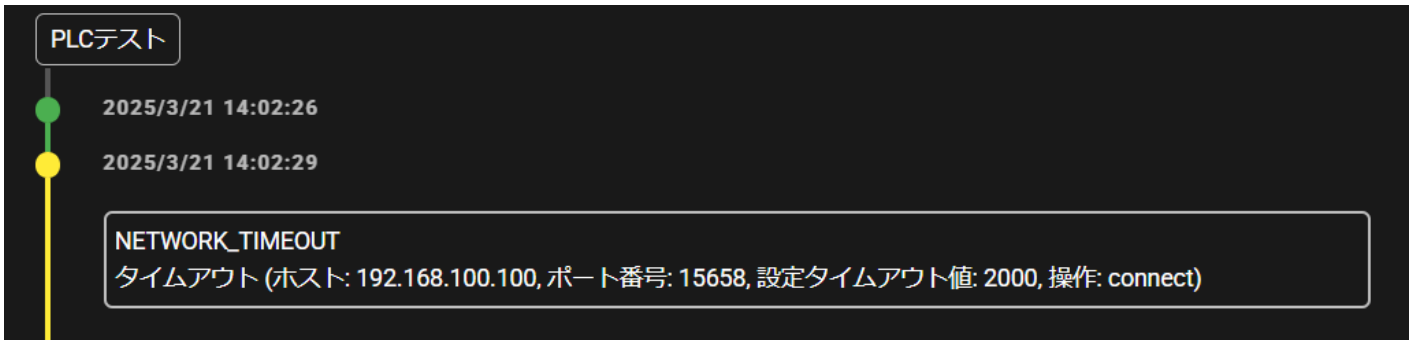
本項で示した対処に従って操作することで、時刻設定のトラブルは回避できます。 しかし、インターネットに接続されていないマシンやRTC（Real Time Clock）が搭載されていないマシンでは再起動後に同じ設定を行う必要があります。 そのため、次のいずれかの検討を推奨します。

- インターネットへ接続して時刻同期を正しく行う（セキュリティ上問題がない場合に限る）
- 稼働マシンのLAN内にNTP Serverを立ち上げ自動同期する
- RTCモジュールにより時刻ずれを防ぐ

7.2.2 PLCコレクタに接続できない場合の対処方法

PLCコレクタに接続できない場合、コンポーネントに黄色のステータスランプが表示され、ステータスタイムラインにNETWORK_TIMEOUTが表示されます。
ステータス確認方法は、ここを参照してください。





7.2.2.1 IPアドレスの確認

使用しているPLCのIPアドレスを確認します。

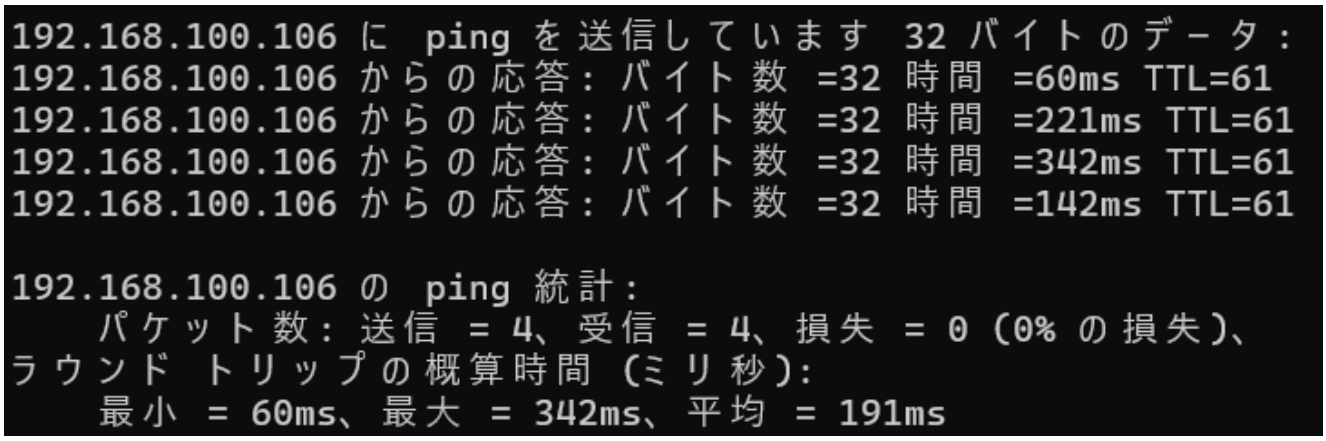
> **確認できる機器について**
>
> この章の内容は、SpeeDBee Synapseが稼働しているデバイスで行う必要はありません。お使いのSpeeDBee SynapseのWebUIを閲覧しているPCでも実行可能です。
> しかし、SpeeDBee Synapseが稼働している機器と同じネットワーク内のPCから実施したほうが確実になります。
> なお、Data Flow Controller版においてはData Flow ControllerのWebUIで設定できます。

1. ネットワークに接続しているPCでWindowsはPowerShell(コマンドプロンプトでも可)を開き、Linuxはターミナル上から次のコマンドを実行します。

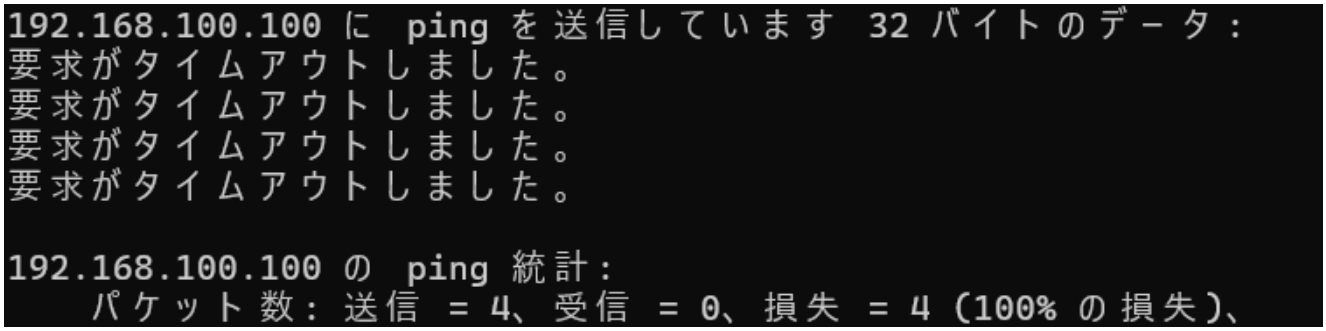
Windowsの場合

```
PS >ping 対象PLCのIPアドレス
```

- 成功画面



- 失敗画面

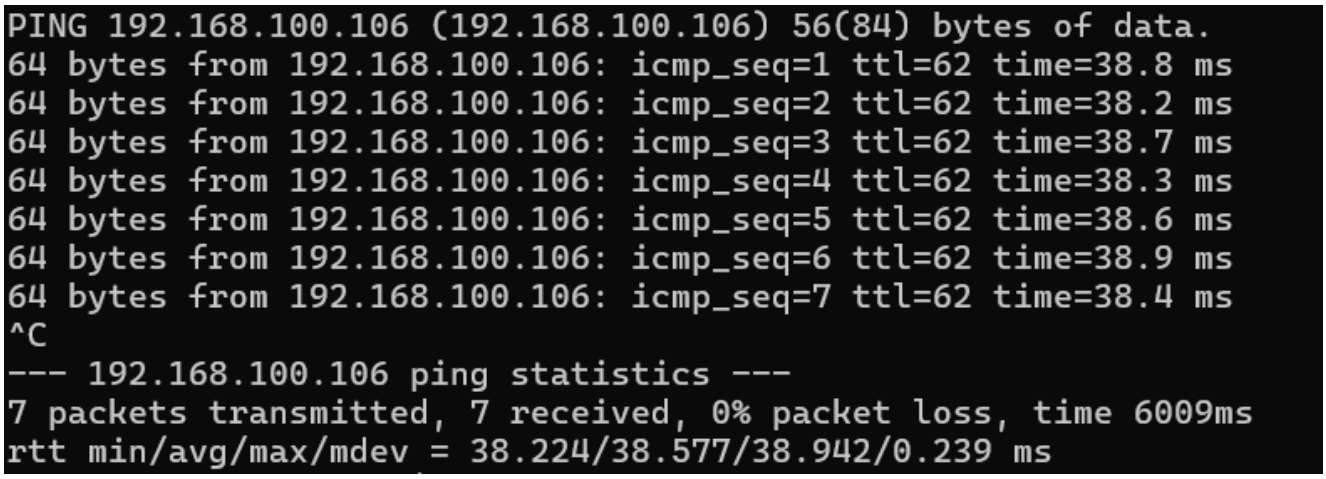


Linuxの場合

```
$ ping 対象PLCのIPアドレス
```

※Linuxは[ctrl]+[C]を入力するまで、pingを送り続けます。

- 成功画面



- 失敗画面

```
PING 192.168.100.100 (192.168.100.100) 56(84) bytes of data.

--- 192.168.100.100 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3006ms
```

pingの疎通が確認できなかった場合は、ご利用のPLC側の設定を確認してください。

7.2.2.2 ポートの確認

使用しているPLCのポート番号を確認します。

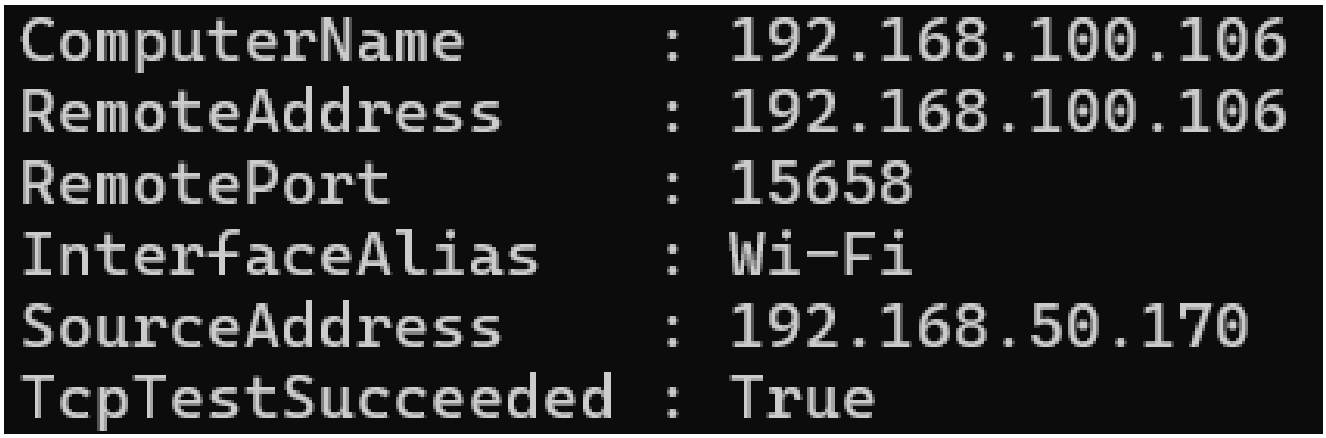
> **確認できる機器について**
>
> この章の内容は、SpeeDBee Synapseが稼働しているデバイスで行う必要はありません。お使いのSpeeDBee SynapseのWebUIを閲覧しているPCでも実行可能です。
> しかし、SpeeDBee Synapseが稼働している機器と同じネットワーク内のPCから実施したほうが確実になります。

1. ネットワークに接続しているPCでWindowsはPowerShell(コマンドプロンプトでも可)を開き、Linuxはターミナル上から次のコマンドを実行します。

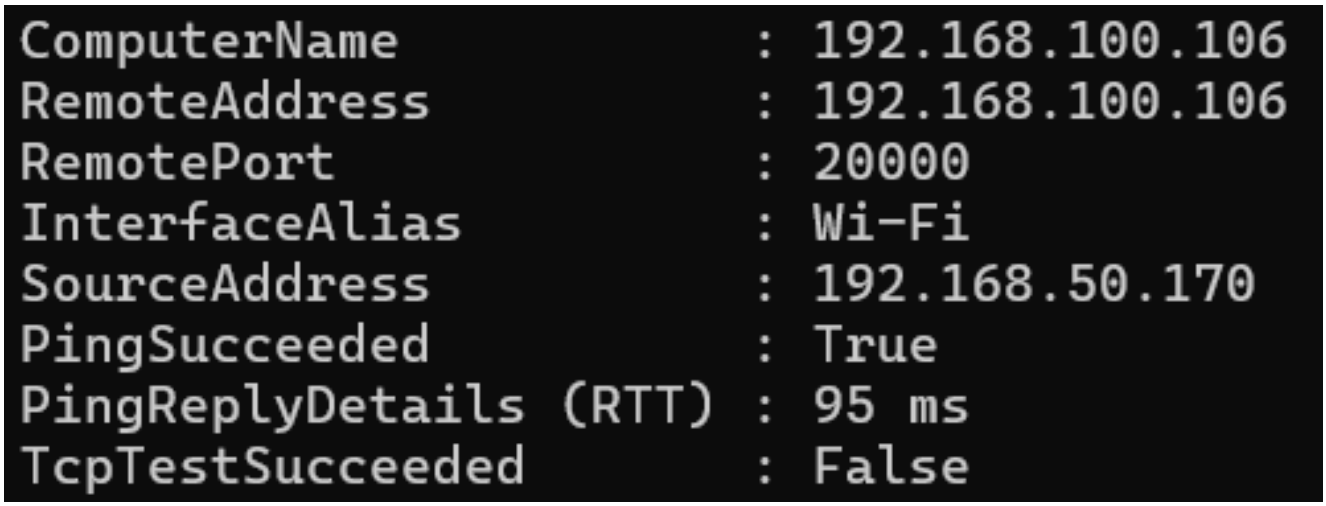
Windowsの場合

```
PS >Test-NetConnection 対象PLCのIPアドレス -Port 対象PLCのポート番号
```

- 該当のポートが開いている時の画面



- 該当のポートが開いていない時の画面



Linuxの場合

```
$ nc -zv 対象PLCのIPアドレス 対象PLCのポート番号
```

- 該当のポートが開いている時の画面



- 該当のポートが開いていない時の画面



ポートが開いていることを確認できなかった場合は、ご利用のPLC側の設定を確認してください。

7.2.2.3 PLC側の設定

PLCのIPアドレス、ポートの設定方法については、ご利用のPLCの取扱説明書などを確認してください。

7.3 異常が発生した場合に検知する方法

ここでは、コンポーネントで異常が発生した場合に検知を行う方法を解説します。

7.3.1 特定のコンポーネントで異常が発生した場合に検知する方法

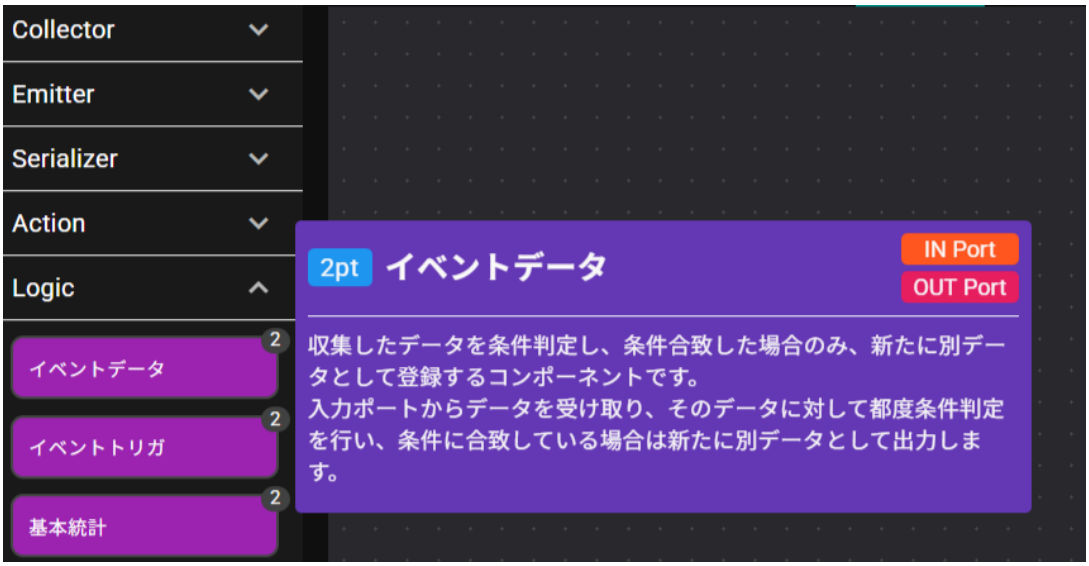
特定のコンポーネントで異常が発生したときに、メールを送信してお知らせする方法を解説します。



7.3.1.1 イベントデータの設定

検知対象コンポーネントから出力されるerrors_run_statusの値が、正常や停止以外の値の時、イベントデータが登録されるよう設定します。コンポーネントステータスが警告、エラーの場合にメール通知を行います。

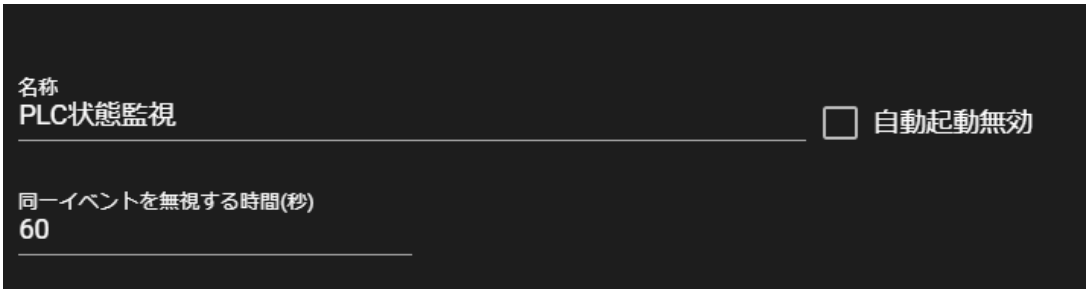
1. エラーを検知したいコンポーネントが稼働しているパネルを開き、Edit ModeでComponentBase -> Logic -> イベントデータをパネル上へドラッグアンドドロップします。



2. 入力ポートの接続先メニューが開きます。今回はシステムポートに接続するため、「閉じる」を選択します。



3. 任意の名称を入力します。今回は例として「PLC状態監視」という名前にします。



4. ゴミ箱のマークを選択して、データを削除します。



5. 「保存」を選択します。



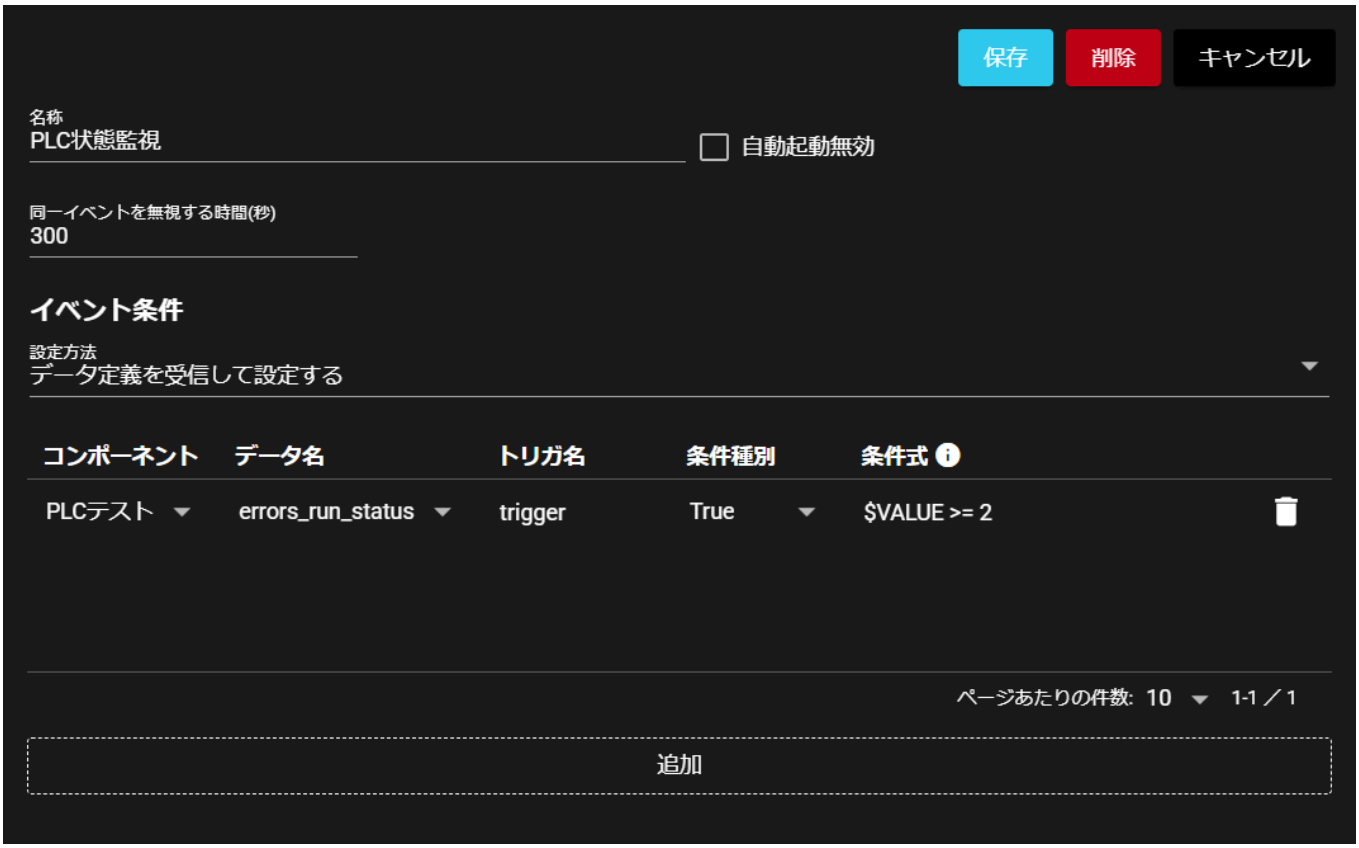
6. イベントデータを検知したいコンポーネントのシステムポートへ接続します。今回は例としてPLCコレクタにイベントデータを接続します。



7. イベントデータの設定メニューを開きます。



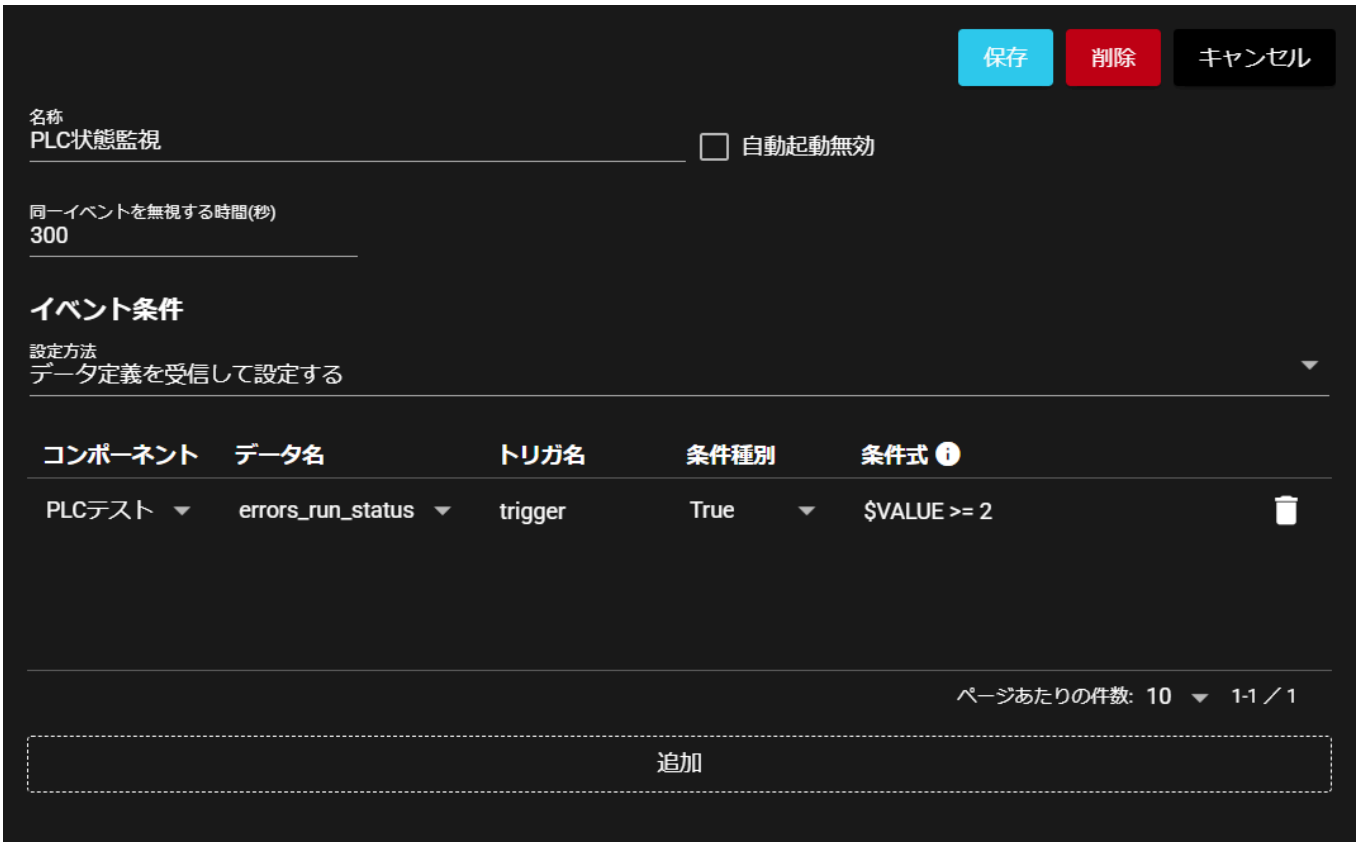
8. イベント条件を設定します。設定方法に「データ定義を受信して設定する」を選択します。データ名を「errors_run_status」、トリガ名を「trigger」、条件種別を「True」、条件式を「$VALUE >= 2」で設定します。そして、同一イベントを無視する時間(秒)を「300」に設定します。



メール送信に関する注意

この設定では、5分（300秒）経過しても、異常が続く限りメールが常に送信されます。 通知量が膨大になる事があるため注意してください。 異常が発生したタイミングで一度だけメール送信する場合は、条件種別を「Raise」に設定します。

9. 「保存」を選択します。



10. イベントデータを起動します。

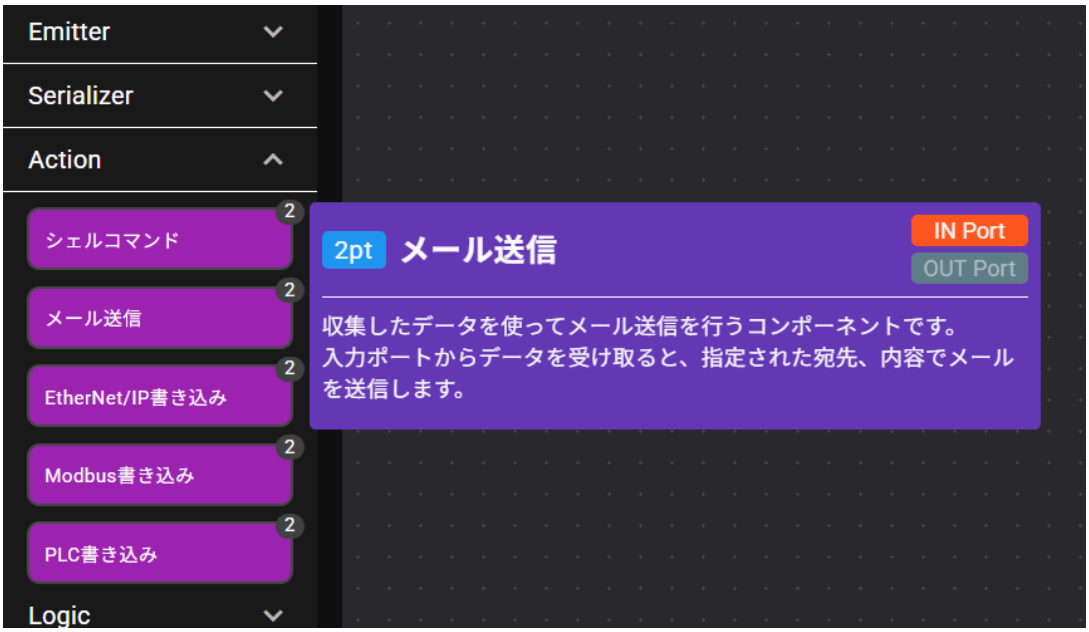


これでイベントデータの設定が完了しました。

7.3.1.2 メール送信の設定

イベントデータから受け取った値を、メールで送信するように設定します。
今回はgmailで送信する場合の設定を解説します。使用できるgmailアカウントを用意し、二段階認証を設定してアプリパスワードを作成しておいてください。

1. エラーを検知したいコンポーネントが稼働しているパネルを開き、Edit ModeでComponentBase -> Action -> メール送信をパネル上へドラッグアンドドロップします。



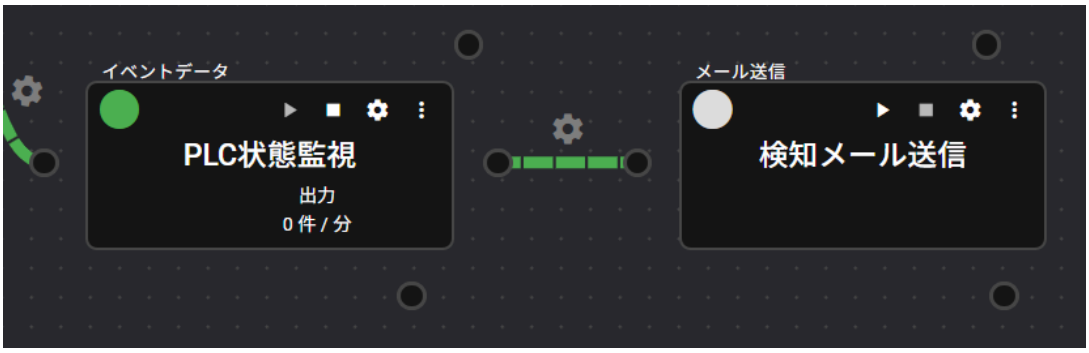
2. 必要な設定を行います。gmailの場合はSMTPサーバーが「smtp.gmail.com」、ポート (TLS)が「587」となります。SMTPパスワードには、作成したアプリパスワードを入力してください。


保存
削除
テスト
キャンセル
名称
検知メール送信
自動起動無効
SMTPホスト
smtp.gmail.com
SMTPポート
587
SSL/TLS
TLS
SMTPユーザー名
SMTPパスワード
タイムアウト(秒)
5
宛先
CC
BCC
FROM
TO
件名
PLCコレクタの異常を検知
本文
$HIVE_COMPONENTで異常を検知しました。
値：$HIVE_DATAVALUE
時刻：$HIVE_TS_DATE_LOCAL $HIVE_TS_TIME_LOCAL
値が2の場合、警告状態が検知されています。
値が3の場合、エラーが発生しています。

3. 「保存」を選択します。


保存
削除
テスト
キャンセル
名称
検知メール送信
自動起動無効
SMTPホスト
smtp.gmail.com
SMTPポート
587
SSL/TLS
TLS

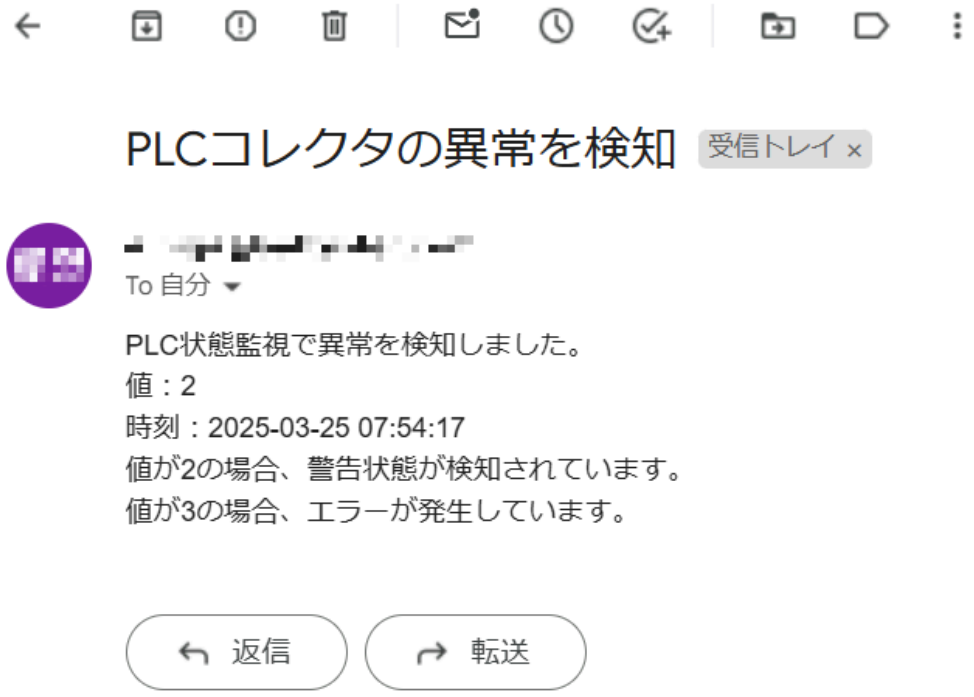
4. メール送信をイベントデータへ接続し、コンポーネントを起動します。



これでメール送信の設定が完了しました。



実際にコンポーネントで異常が発生すると、設定したメールアカウントにデータが送信されます。以下は受け取ったメールの例です。



7.3.2 Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法

いずれかのコンポーネントで異常が発生したときに、メールを送信してお知らせする方法を解説します。



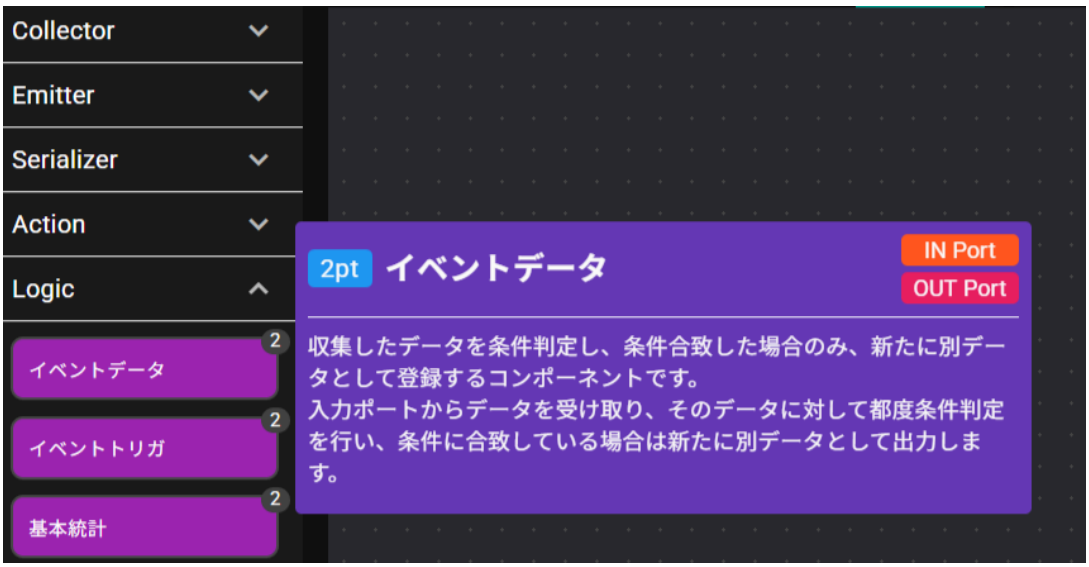
7.3.2.1 イベントデータの設定

エラーマネージャから出力される「latest_error_component_count」と「latest_warning_component_count」の値が増えた時、イベントデータが登録されるよう設定します。Synapse内でいずれかのコンポーネントステータスが警告、またはエラーの場合にメール通知を行います。

1. Edit Modeでシステムパネルを開き、エラーマネージャを起動します。



2. ComponentBase -> Logic -> イベントデータをパネル上へドラッグアンドドロップします。



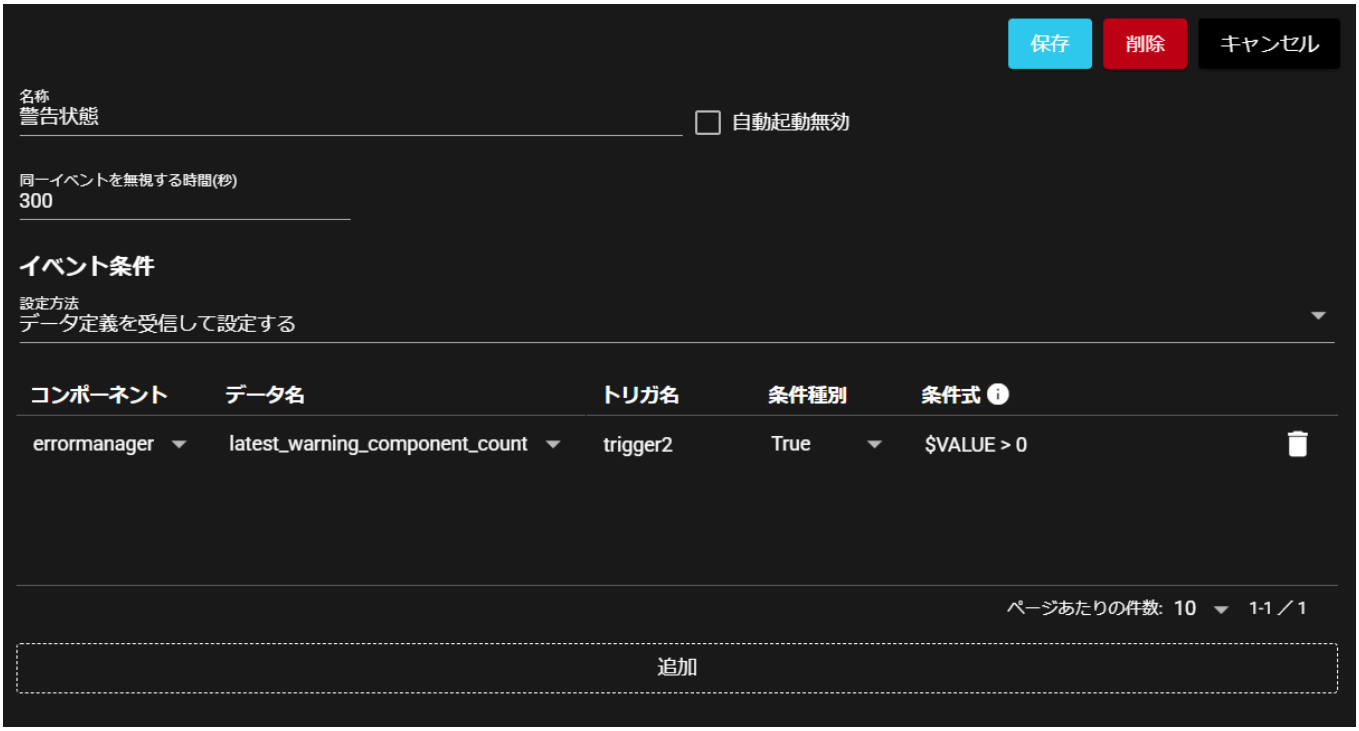
3. 入力ポートの接続先メニューが開きます。errormanagerにチェックを入れ、「接続」を選択します。



4. 任意の名称を入力した後、イベント条件を設定して「保存」を選択します。イベント条件は下記条件①②を設定したコンポーネントを、個々に作成します。

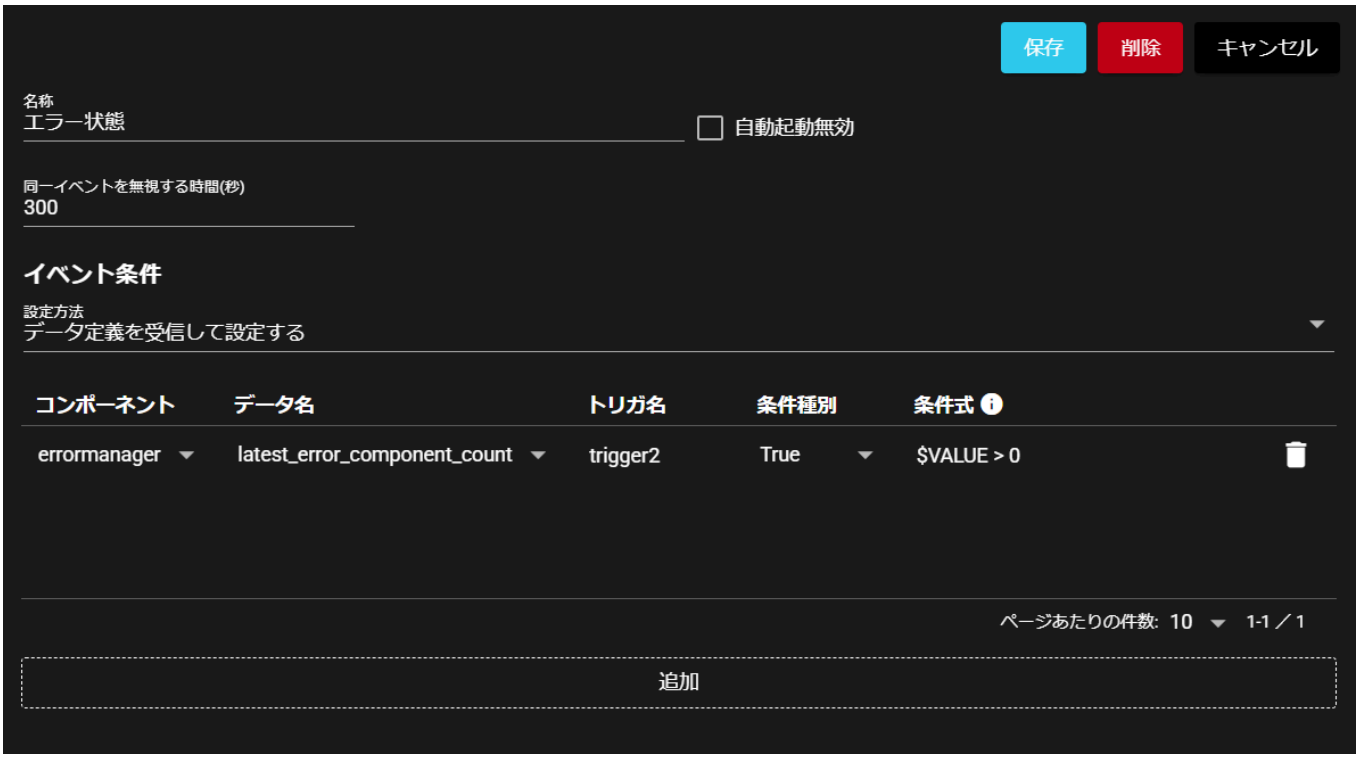
条件①

同一イベントを無視する時間(秒)「300」、データ名「latest_error_component_count」、トリガ名「trigger1」、条件種別「True」、条件式「$VALUE > 0」



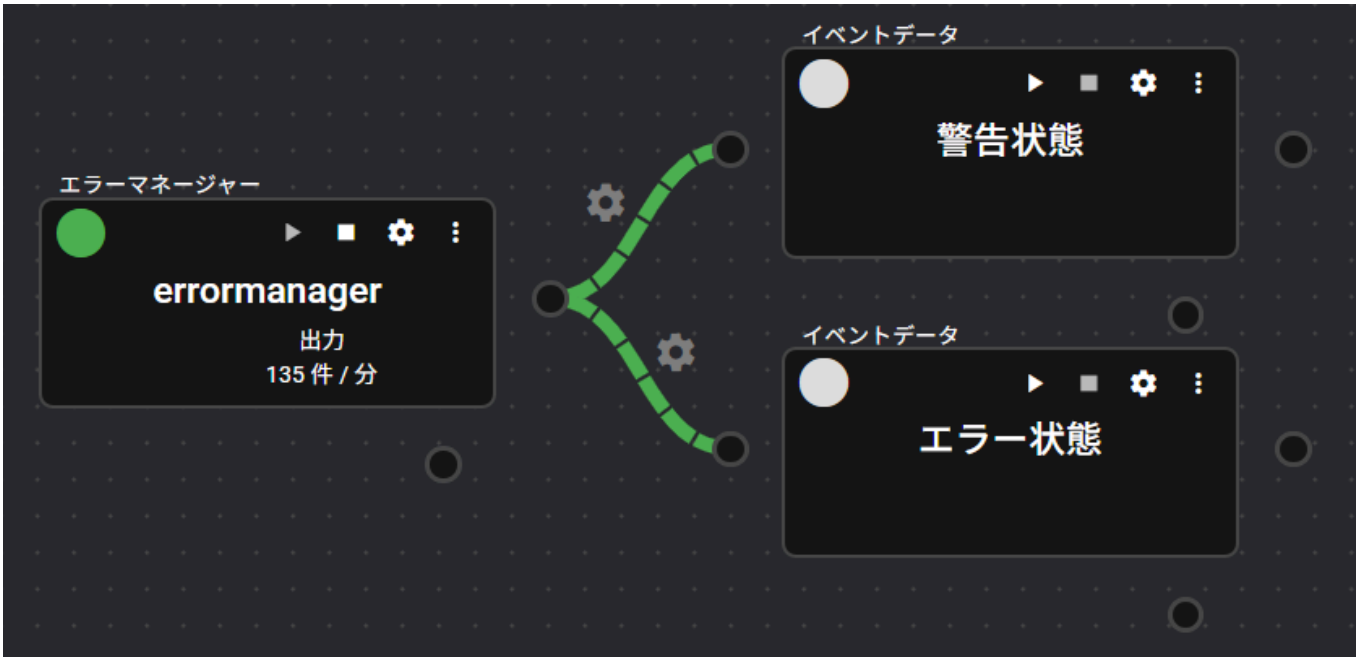
条件②

同一イベントを無視する時間(秒)「300」、データ名「latest_warning_component_count」、トリガ名「trigger2」、条件種別「True」、条件式「$VALUE > 0」



> **メール送信に関する注意**
>
> この設定では、5分（300秒）経過しても、異常が続く限りメールが常に送信されます。 通知量が膨大になる事があるため注意してください。 異常が発生したタイミングで一度だけメール送信する場合は、条件種別を「Raise」に設定します。

5. イベントデータを起動します。

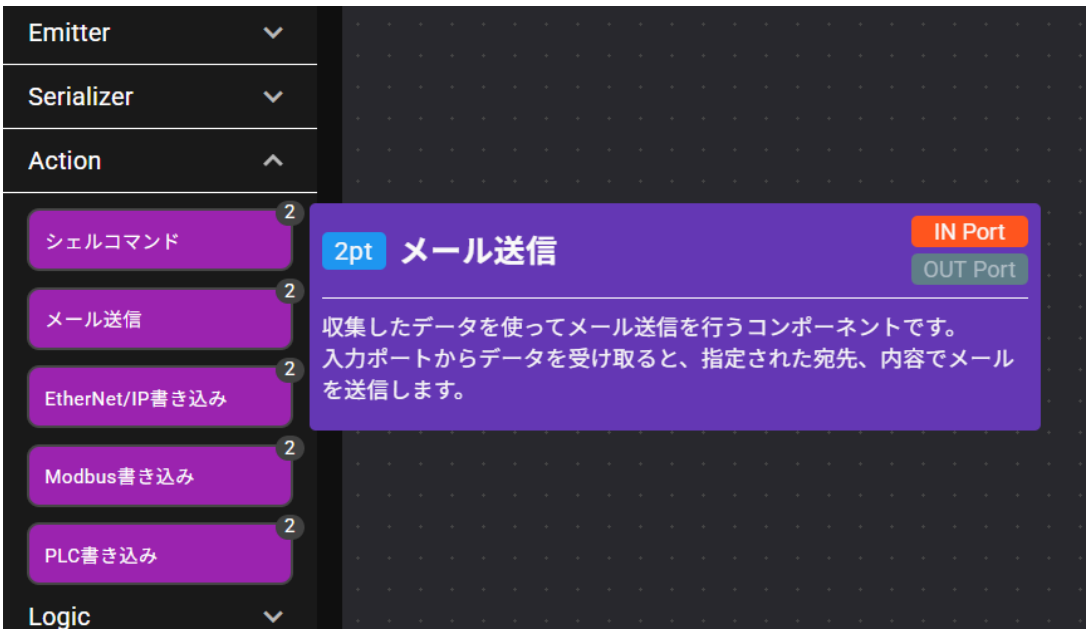


これでイベントデータの設定が完了しました。

7.3.2.2 メール送信の設定

イベントデータから受け取った値を、メールで送信するように設定します。
ここと同じく、gmailで送信する場合の設定を解説します。使用できるgmailアカウントを用意し、二段階認証を設定してアプリパスワードを作成しておいてください。

1. システムパネルを開き、Edit ModeでComponentBase -> Action -> メール送信をパネル上へドラッグアンドドロップします。



2. 必要な設定を行います。gmailの場合はSMTPサーバーが「smtp.gmail.com」、ポート (TLS)が「587」となります。SMTPパスワードには、作成したアプリパスワードを入力してください。


保存
削除
テスト
キャンセル
名称
検知メール送信_警告
自動起動無効
SMTPホスト
smtp.gmail.com
SMTPポート
587
SSL/TLS
TLS
SMTPユーザー名
SMTPパスワード
タイムアウト(秒)
5
宛先
CC
BCC
FROM
TO
件名
エラーマネージャで異常を検知
本文
$HIVE_COMPONENTのコンポーネントが$HIVE_DATAVALUE件あります。
検知時刻：$HIVE_TS_DATE_LOCAL $HIVE_TS_TIME_LOCAL

3. 「保存」を選択します。


保存
削除
テスト
キャンセル
名称
検知メール送信
自動起動無効
SMTPホスト
smtp.gmail.com
SMTPポート
587
SSL/TLS
TLS

4. 同じ手順でメール送信コンポーネントをもう一つ作成します。その後、メール送信コンポーネントをそれぞれのイベントデータへ接続し、コンポーネントを起動します。



これでメール送信の設定が完了しました。



実際にいずれかのコンポーネントで異常が発生すると、設定したメールアカウントにデータが送信されます。以下は受け取ったメールの例です。



7.3.3 電断などにより時系列データベースのデータ欠損を検知する方法

Synapseでは、稼働マシンの電断により内部で使用している時系列データベースに欠損が発生すると、コアインターナルは、次の起動時にその異常を警告状態として登録します。
システムポートでこの情報をとらえることで、データ欠損があった場合にメール通知することが可能です。
ここでは、その手順を解説します。



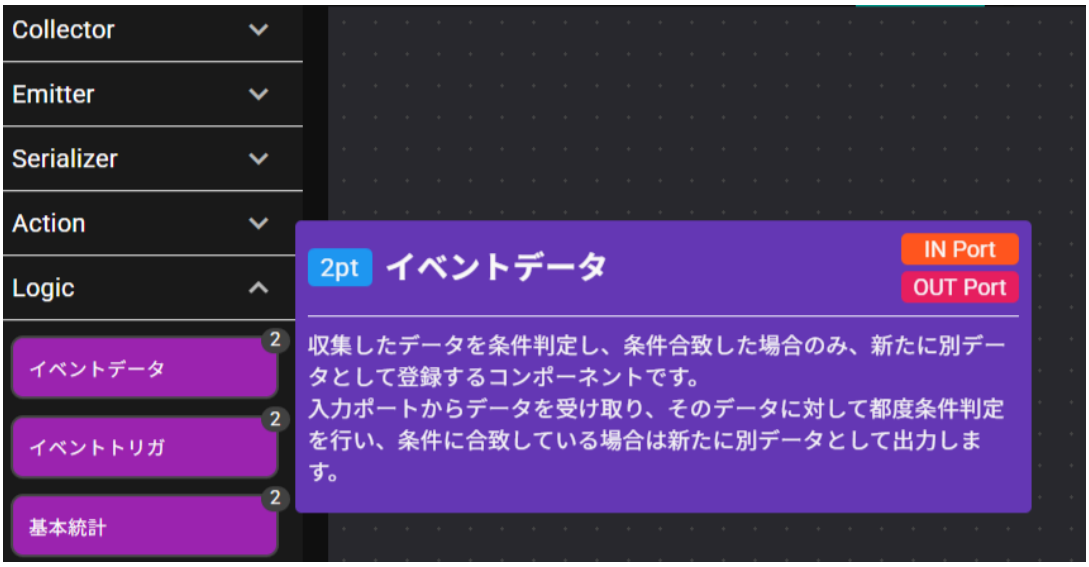
電断によるデータ欠損に関する注意

電断によるデータ欠損は検知可能ですが、データを回復することはできないため、ご注意ください。
あくまで、一部のデータ保存ができなかったことを検知する機能となります。

7.3.3.1 イベントデータの設定

コアインターナルから出力されるerrors_run_statusの値が、正常や停止以外の値の時、イベントデータが登録されるよう設定します。ステータスランプが警告またはエラーの場合にメール通知を行い、電断を検知できます。

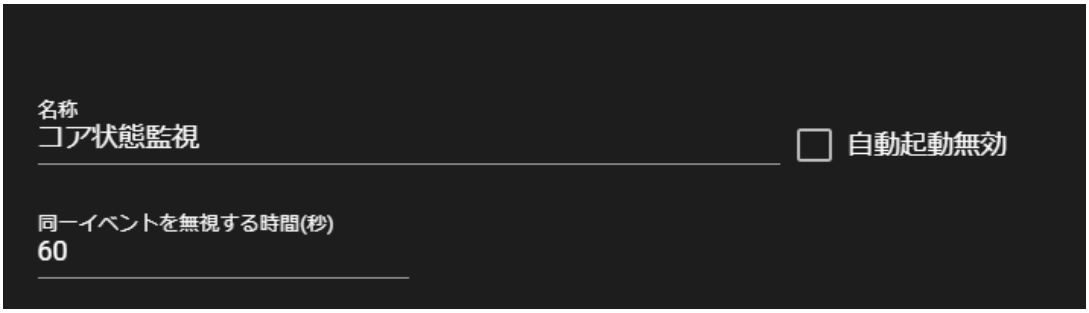
1. システムパネルを開き、Edit ModeでComponentBase -> Logic -> イベントデータをパネル上へドラッグアンドドロップします。



2. 入力ポートの接続先メニューが開きます。今回はシステムポートに接続するため、「閉じる」を選択します。



3. 任意の名称を入力します。今回は例として「コア状態監視」という名前にします。



4. ゴミ箱のマークを選択して、データを削除します。



5. 「保存」を選択します。



6. イベントデータをコアインターナルのシステムポートへ接続します。



7. イベントデータの設定メニューを開きます。



8. イベント条件を設定します。設定方法に「データ定義を受信して設定する」を選択します。データ名を「errors_run_status」、トリガ名を「trigger」、条件種別を「True」、条件式を「$VALUE >= 2」で設定します。そして、同一イベントを無視する時間(秒)を「300」に設定します。



9. 「保存」を選択します。



10. イベントデータを起動します。

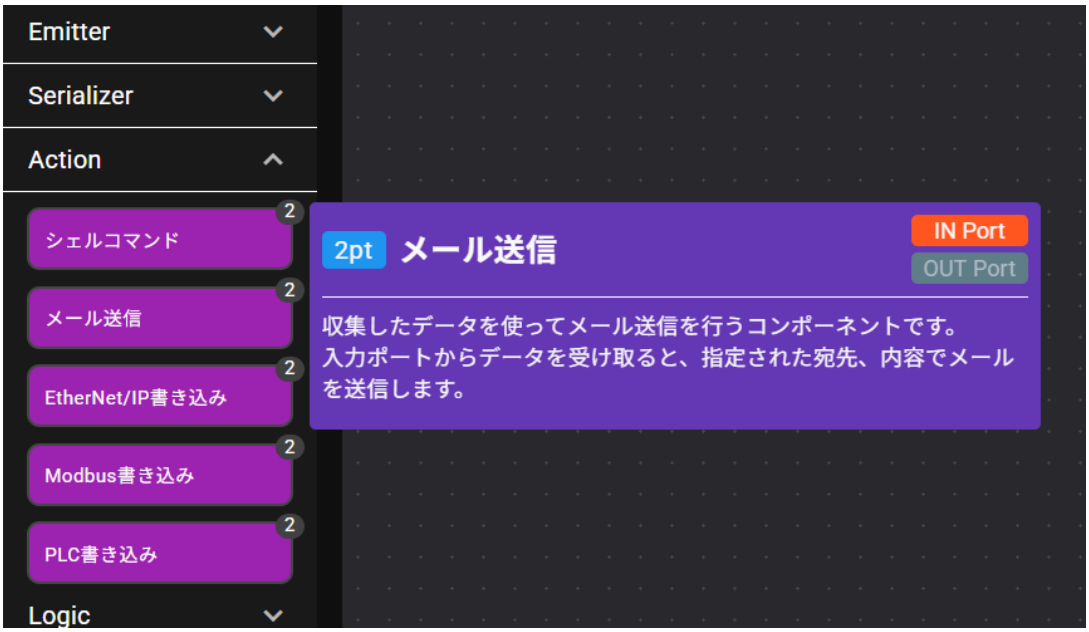


これでイベントデータの設定が完了しました。

7.3.3.2 メール送信の設定

イベントデータから受け取った値を、メールで送信するように設定します。

1. システムパネルを開き、Edit ModeでComponentBase -> Action -> メール送信をパネル上へドラッグアンドドロップします。



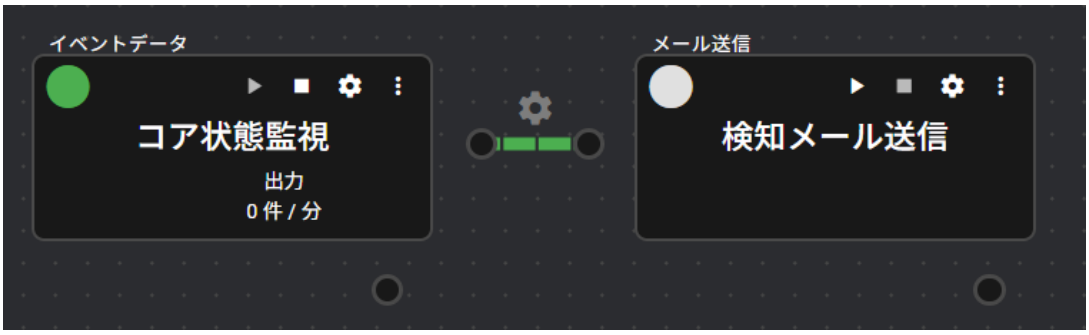
2. 必要な設定を行います。gmailの場合はSMTPサーバーが「smtp.gmail.com」、ポート (TLS)が「587」となります。SMTPパスワードには、作成したアプリパスワードを入力してください。


保存
削除
テスト
キャンセル
名称
検知メール送信
自動起動無効
SMTPホスト
smtp.gmail.com
SMTPポート
587
SSL/TLS
TLS
SMTPユーザー名
SMTPパスワード
タイムアウト(秒)
5
宛先
CC
BCC
FROM
TO
件名
電断を検知しました
本文
$HIVE_COMPONENTがコアインターナルの電断を検知しました。
データ欠損が発生しています。
時刻：$HIVE_TS_DATE_LOCAL $HIVE_TS_TIME_LOCAL

3. 「保存」を選択します。


保存
削除
テスト
キャンセル
名称
検知メール送信
自動起動無効
SMTPホスト
smtp.gmail.com
SMTPポート
587
SSL/TLS
TLS

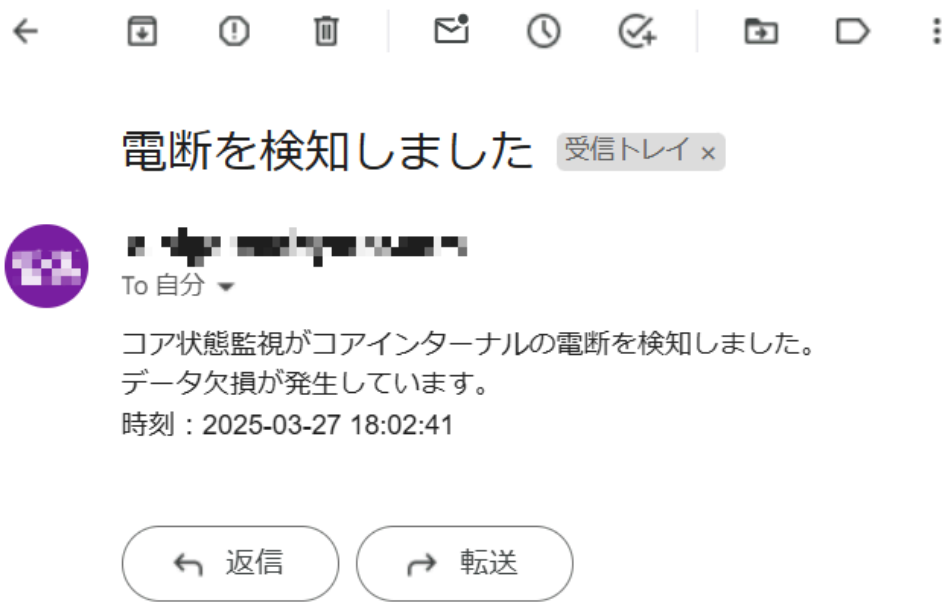
4. メール送信をイベントデータへ接続し、コンポーネントを起動します。



これでメール送信の設定が完了しました。



実際に電断が発生すると、設定したメールアカウントにデータが送信されます。以下は受け取ったメールの例です。



8 セキュア利用ガイド

本章では、SpeeDBee Synapseのセキュアな利用方法について説明します。

8.1 ユーザーの管理

セキュリティ水準を高めるための、ユーザー管理の運用方法について説明します。

adminユーザーの無効化

adminユーザーは、SpeeDBee Synapseに標準で登録されているユーザーです。 このユーザーにはすべての権限が与えられています。 不正なアクセスでの悪用を防ぐため、必要な場合を除いて無効化することを推奨します。 無効化する場合は、別でシステム管理者権限のユーザーを追加している必要があります。

ユーザーを無効にする方法については、「ユーザー管理」をご覧ください。

強固なパスワードポリシー

ユーザーに登録するパスワードは、強固なものにするのが望ましいです。 以下は、強固なパスワードポリシーの例となります。 パスワードを自動生成するツールを使用する際は、以下の条件に当てはまるように生成することを推奨します。

- 長さが8文字以上であること
- 英字の大文字と小文字、数字、記号がそれぞれ1つ以上使用されていること
- 他のアカウントのパスワードとは異なること
- 名前などの個人情報からは推測できないこと
- 英単語などをそのまま使用していないこと
- 類推しやすい並び方やその安易な組合せにしないこと

8.2 HTTPSの利用

SpeeDBee Synapseを操作する際に、HTTPSを利用して通信を暗号化することで、 セキュリティ水準を高めることができます。

HTTPSの利用方法については、「HTTP/HTTPS通信」をご覧ください。

使用する証明書について

証明書を使った暗号化通信を行う場合、公的認証局の発行した正規の証明書をご利用下さい。
もし検証目的など一時的に自己証明書などを利用する場合は、SAN(Subject Alternative Name)を含めて証明書を作成して下さい。各コンポーネントの暗号化に利用する証明書についても、同様にSANを含めて下さい。

8.3 トークンの厳重管理

アクセストークンは、外部からSpeeDBee Synapseにアクセスする際の、リソースへのアクセス権を示すものです。不正な攻撃者によってアクセストークンが悪用されると、SpeeDBee Synapseのリソースに不正にアクセスされる恐れがあります。 アクセストークンが第三者の手に渡ることがないよう、厳重に管理しておくことを推奨します。

8.4 暗号化通信の利用

一部のコンポーネントでは、暗号化通信を利用することで、セキュリティ水準を高めることができます。

項目	説明
MQTTコレクタ	CA証明書を準備することで、暗号化通信が利用できます。
Cloudエミッタ(Azure用)	このエミッタでは、通信は暗号化されています。
Cloudエミッタ(AWS用)	このエミッタでは、通信は暗号化されています。
FTP/FTPS/SFTPエミッタ	FTPSまたはSFTPを利用することで通信は暗号化されます。
メール送信	TLSを利用することでよりセキュアに通信は暗号化されます。

8.5 操作ログの確認

操作ログを定期的に確認することで、不審なアクティビティを早期に発見することができます。 操作ログの確認方法については、「各種ログ」をご覧ください。

9 後方互換に関して

本章では、後方互換に関して説明します。

9.1 後方互換の対象

SpeeDBee Synapseの後方互換の考え方について説明します。 後方互換の対象は、初回2024年度発売の「SpeeDBee Synapse」を対象とします。それ以前の他製品、類似製品は後方互換の対象外となります。（製品互換はありません）

対象	説明
設定情報	ユーザーの設定した各種情報。（パネル、コンポーネント設定等、ユーザー管理情報を含む）
コンポーネント	SpeeDBee Synapseが提供するコンポーネント。 ユーザーが追加したカスタムコンポーネントを含む。

9.2 後方互換

後方互換の基本的な考え方としては、原則は後方互換を意識した製品提供を行います。

- リビジョン、マイナーバージョンアップデート

既存の設定のままアップデート可能で、後方互換を維持します。 （内部設定の構造変更などはアップデート処理にて行う場合もありますが、ユーザーが意識する必要はありません）

- メジャーアップデート

大規模な機能拡張などを行った際のアップデートです。この場合、原則は後方互換の維持に努めますが 後方互換をサポートしない場合があります。その際のアップデート手順については、アップデート時に詳細と合わせて 個別の情報提供を行います。

10 実行ライセンス

本章は、SpeeDBee Synapseの実行ライセンスについて説明します。

10.1 ライセンスの種類

ライセンスの種類について説明します。
現在のSpeeDBee Synapseには、3種類のライセンスが存在します。

項目	説明
開発ライセンス	SpeeDBee Synapseの全機能、操作が可能なライセンスです。 システム構築は、通常このライセンスを利用します。本マニュアルは本章を除き、開発ライセンスを前提に説明しています。
実行ライセンス	SpeeDBee Synapseの一部機能が使用できる実行環境向けライセンスです。 開発ライセンスにより構築された設定を引継ぎ、複数環境で動作させることができます。
評価ライセンス	SpeeDBee Synapseを評価目的で動作させるライセンスです。 コンポーネントが起動してから、連続稼働できるのが1時間という制約があります。 SpeeDBee Synapseの購入前に、評価目的で使用するライセンスです。

実行ライセンスについて

実行ライセンスは、実行ライセンス単体では利用できません。最低でも１つ以上の開発ライセンスを使用して運用する必要があります。

10.2 実行ライセンス利用環境の動作仕様

本節では、実行ライセンスを適用したSpeeDBee Synapseの動作仕様について説明します。

開発ライセンスと実行ライセンスについて、代表的な機能の可否を下表に示します。

項目	開発 ライセンス	実行 ライセンス
ユーザー管理/認証	○	○ ※開発ライセンスのユーザー情報は引き継がれません

項目	開発 ライセンス	実行 ライセンス
Grafana連携	○	○ ※開発ライセンスの設定は引き継がれません
カスタムコンポーネント （Python、C）	○	○ ※手動でカスタムコンポーネントや外部モジュールを設定する必要があります
外部制御Web-API	○	○
アクセストークン発行	○	○ ※開発ライセンスの設定は引き継がれません
暗号化通信	○	○ ※開発ライセンスの設定は引き継がれません
DBクエリサービス	○	○ ※設定は引き継がれますが、トークン認証は実行ライセンス側で手動で再設定する必要があります
Run Mode	○	○
Edit Mode	○	△ ※制限つきで利用できます
パネルの追加、更新、削除	○	×
新しいコンポーネントの登録	○	×
設定済みコンポーネントの設定更新	○	○
設定済みコンポーネントの削除	○	×
フローリンクの追加、削除	○	×
各種ポートの設定	○	○

項目	開発 ライセンス	実行 ライセンス
フィルタの設定	○	×
データモニタ	○	○
ステータスタイムライン	○	○
ログ	○	○
実行ライセンス設定 ダウンロード	○	×
プロジェクト設定	○	×

異種OS間の実行ライセンス向け設定の取込について

開発ライセンス稼働環境と実行ライセンス稼働環境のOSが異なる場合、実行ライセンス向け設定の取込時に、一部の設定が自動で変換されます。ただし、下記以外は変換されないため、手動で再設定が必要になる場合があります。

- 開発ライセンス側Linuxー＞実行ライセンス側Windowsの場合

  設定ファイル中のパスを次のように変換します。

  ```
  /var/speedbeesynapse -> C:\ProgramData\SALTYSTER\SpeeDBeeSynapse
  ```

- 開発ライセンス側Windowsー＞実行ライセンス側Linuxの場合

  設定ファイル中のパスを次のように変換します。

  ```
  C:\ProgramData\SALTYSTER\SpeeDBeeSynapse -> /var/speedbeesynapse
  ```

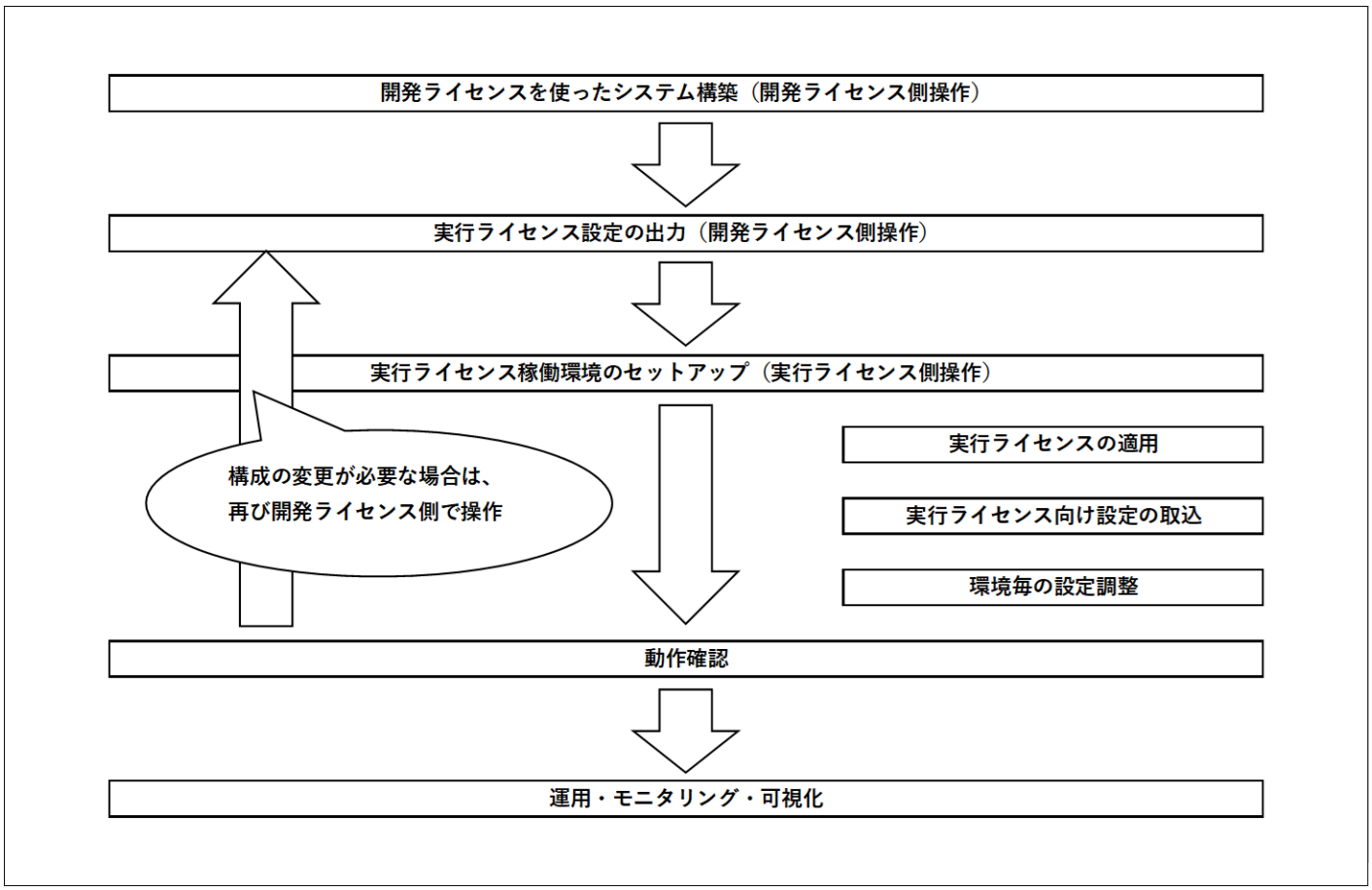
※設定の取込後、異常がある場合は、手動にて再設定してください。

10.3 実行ライセンスの運用方法

本節では、実行ライセンスの運用方法について説明します。

10.3.1 運用までの流れ

実行ライセンスの運用の流れを下図に示します。



①開発ライセンスを使ったシステム構築（開発ライセンス側操作）

最初に、開発ライセンスを適用したSpeeDBee Synapseを使ってシステムを構築します。 データ収集、出力フォーマット、データ送信・連携などのデータフローの設定を行います。 具体的な設定内容は、システムの目的毎に異なりますが、参考として、導入フローを参照してください。

②実行ライセンス設定の出力（開発ライセンス側操作）

実行ライセンスで利用予定のデータフローの設定が済んだら、実行ライセンス設定の出力を行います。この操作は、開発ライセンスを適用したSpeeDBee Synapseでのみ利用できます。

③実行ライセンス稼働環境のセットアップ（実行ライセンス側操作）

実行ライセンス向けの稼働マシンのセットアップおよびSpeeDBee Synapseのインストール等を行います。インストール後に、購入した実行ライセンスの適用を行います。これにより実行ライセンス向けの設定が適用されますが、実行ライセンス稼働環境毎の設定を必要とする場合があります。例えば、外部機器のIPアドレスの設定などは利用環境に合わせて設定を必要とする場合があります。

④動作確認

実行ライセンス稼働環境の設定が済んだら、実際にシステムを動作させます。外部機器の設定項目などの不備がないか確認します。なお、この操作では、新しいコンポーネントを追加して、新たなデータフローを設定することはできません。再設定が必要な場合、①開発ライセンスを使ったシステム構築、ならびに②実行ライセンス設定の出力を再度行い、実行ライセンス向け設定を取込む必要があります。

⑤運用・モニタリング・可視化

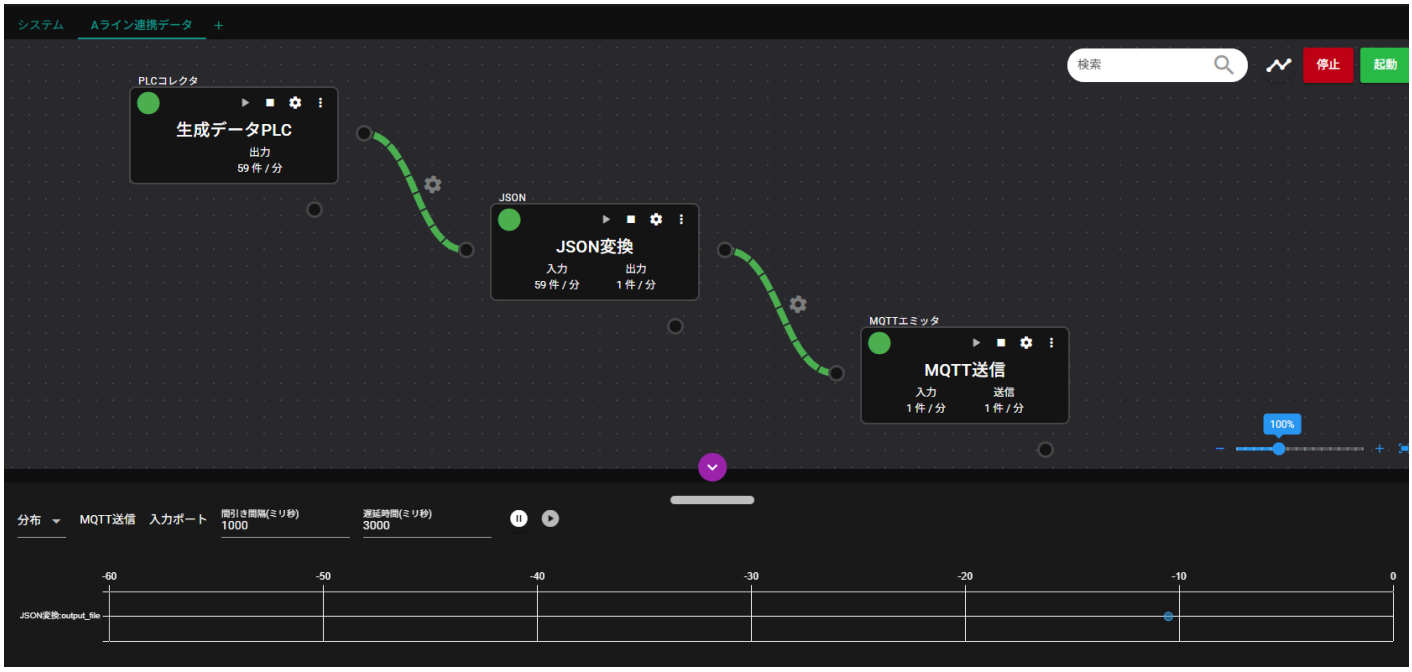
実行ライセンス稼働環境での運用を開始します。データフローの変更など必要になった場合は、再び開発ライセンス側での操作を行い、本手順にそって設定を変更してください。

10.3.2 簡易チュートリアル

ここでは、実際に実行ライセンス稼働環境を構築するまでの流れを解説します。 なお、本チュートリアルでは、開発ライセンス稼働マシンと実行ライセンス稼働マシンは、同一のネットワーク内に配置されていることを前提とします。また、SpeeDBee Synapseは、それぞれインストールは事前に行っているものとします。

本チュートリアルで、利用するデータフローは次のとおりです。

項目	説明
Aライン連携データ	PLCコレクタで集めたデータをJSON形式で、MQTTにより外部へデータ送信



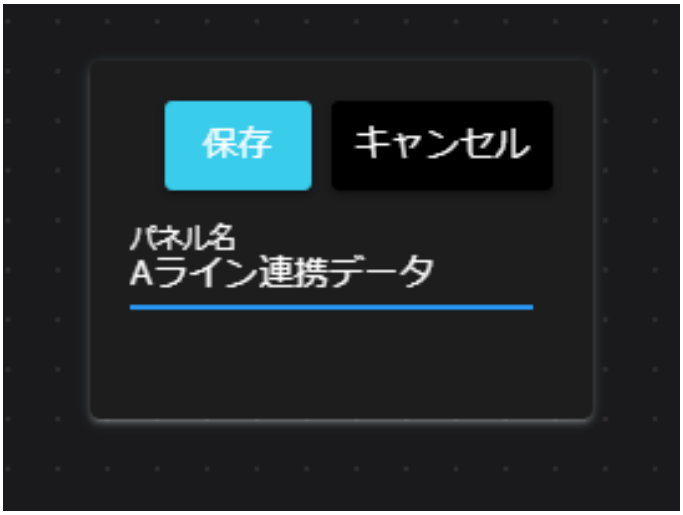
10.3.2.1 開発ライセンスを使ったシステム構築

ここでの操作は、開発ライセンスを使ったSpeeDBee Synapseで行います。 以降を参考に、PLCコレクタ、JSONシリアライザ、MQTTエミッタを設定してください。

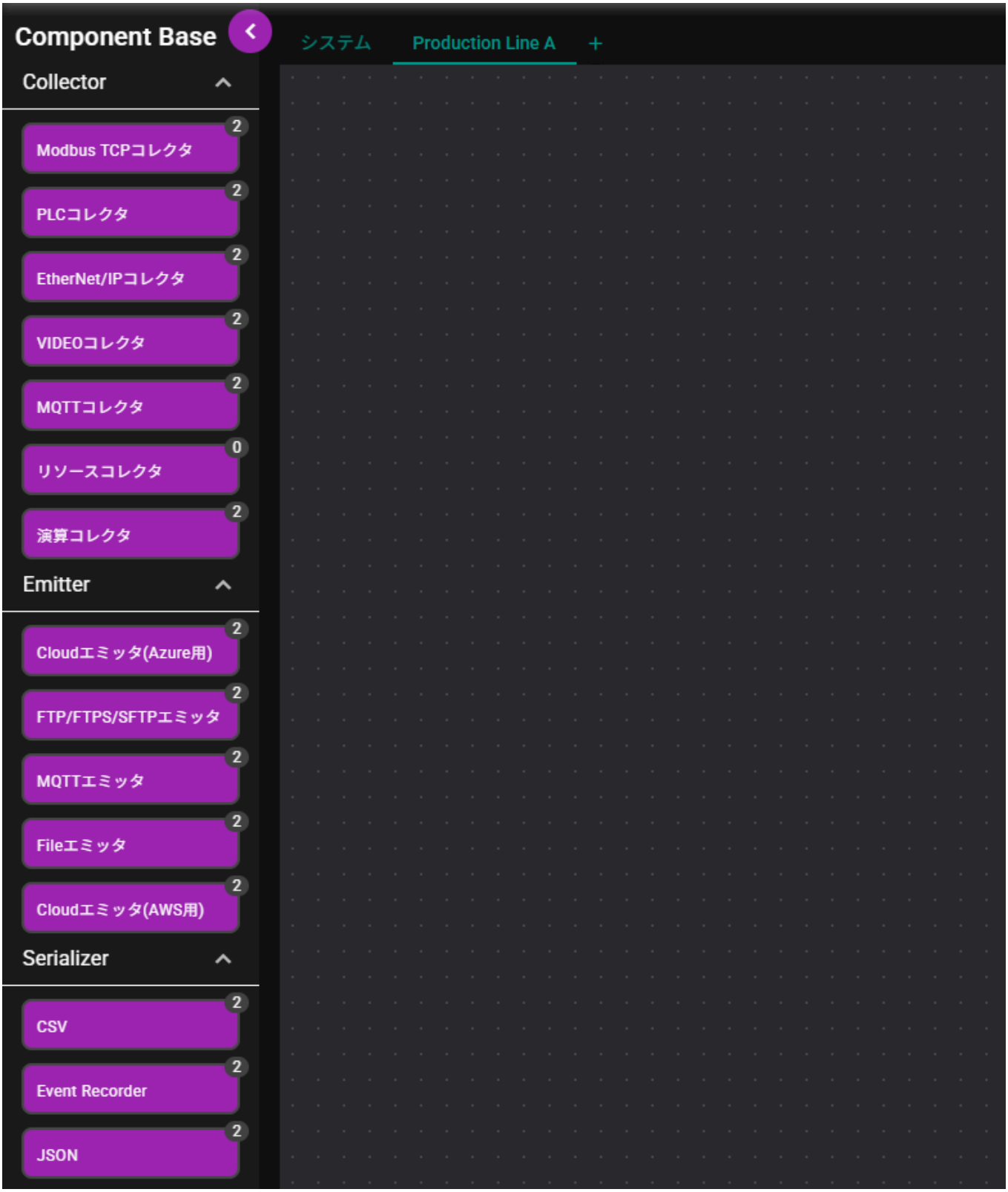
1. 「パネル1」を選択して、右クリックメニューから「パネル設定」をクリックします。



2. 「パネル1」を「Aライン連携データ」へ変更して保存します。



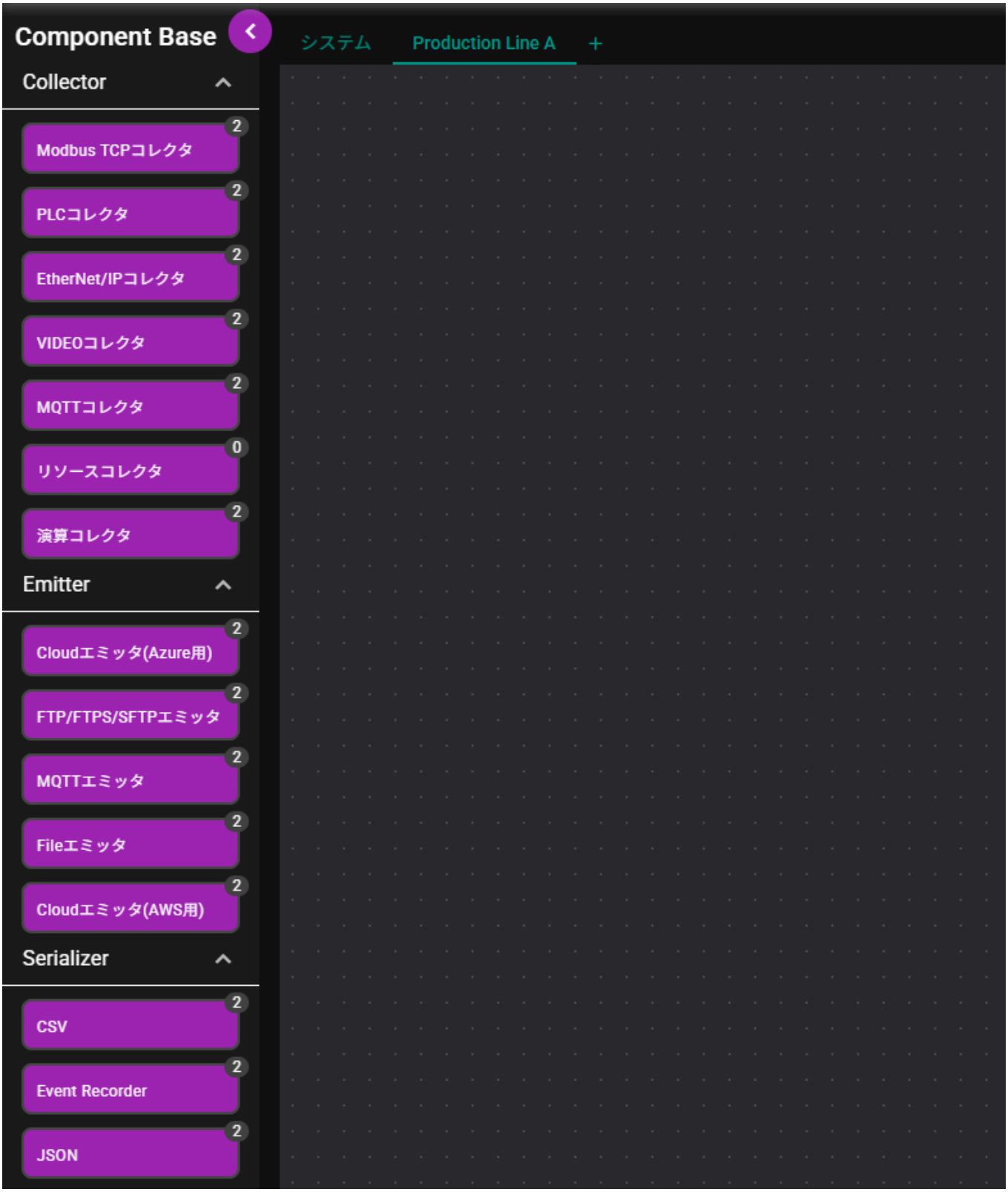
3. PLCコレクタの設定を行います。フローエディタ左部より「PLCコレクタ」をフローエディタ中央へドラッグ＆ドロップします。



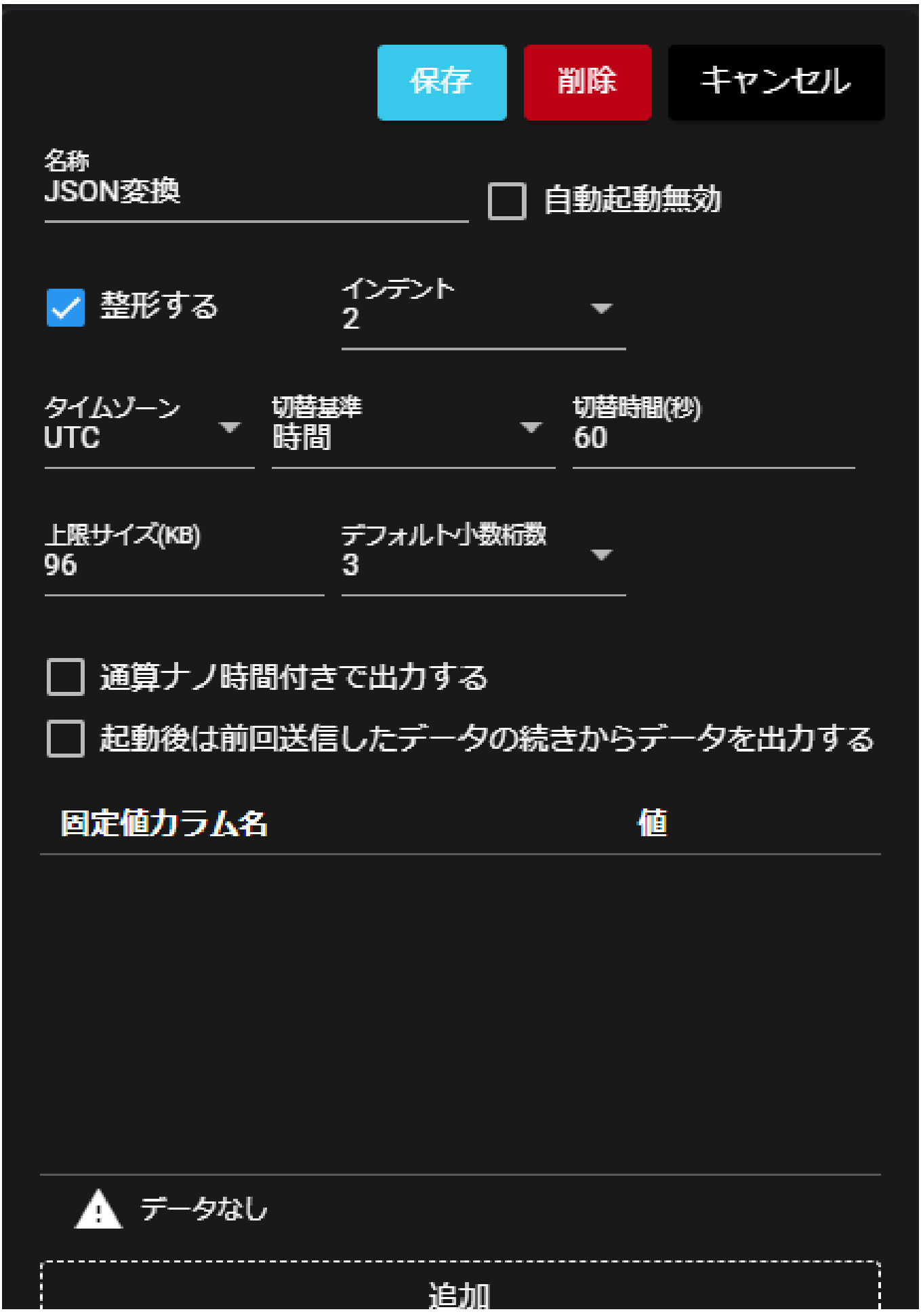
4. PLCコレクタの設定値を入力して、「保存」をクリックします。IPアドレスやポート、データなどはご利用の環境に合わせて下さい。
（※もしPLCコレクタが利用できない場合、リソースコレクタなどへ置き換えて設定してください。）



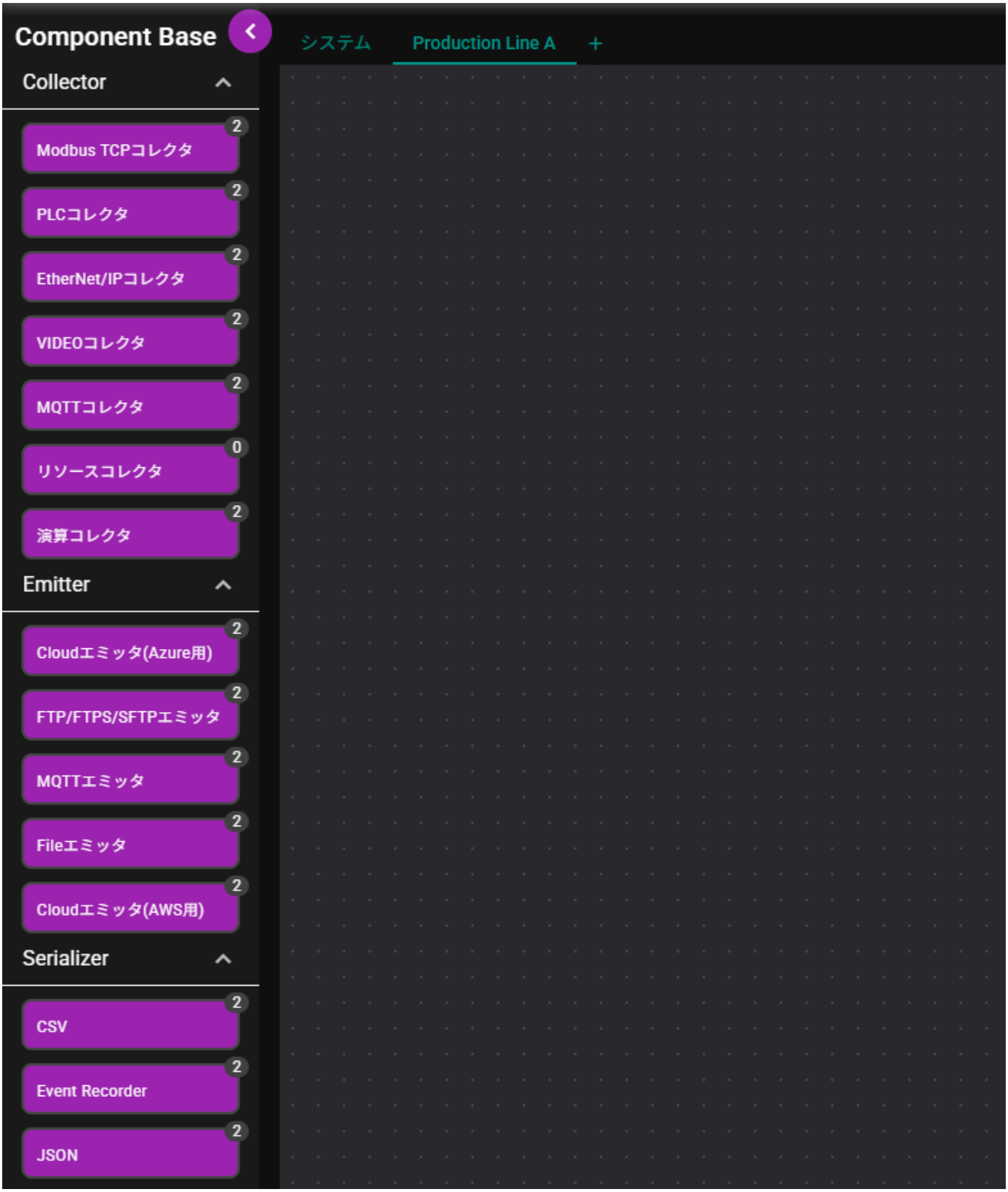
5. シリアライザの設定を行います。フローエディタ左部より「JSON」をフローエディタ中央へドラッグ＆ドロップします。



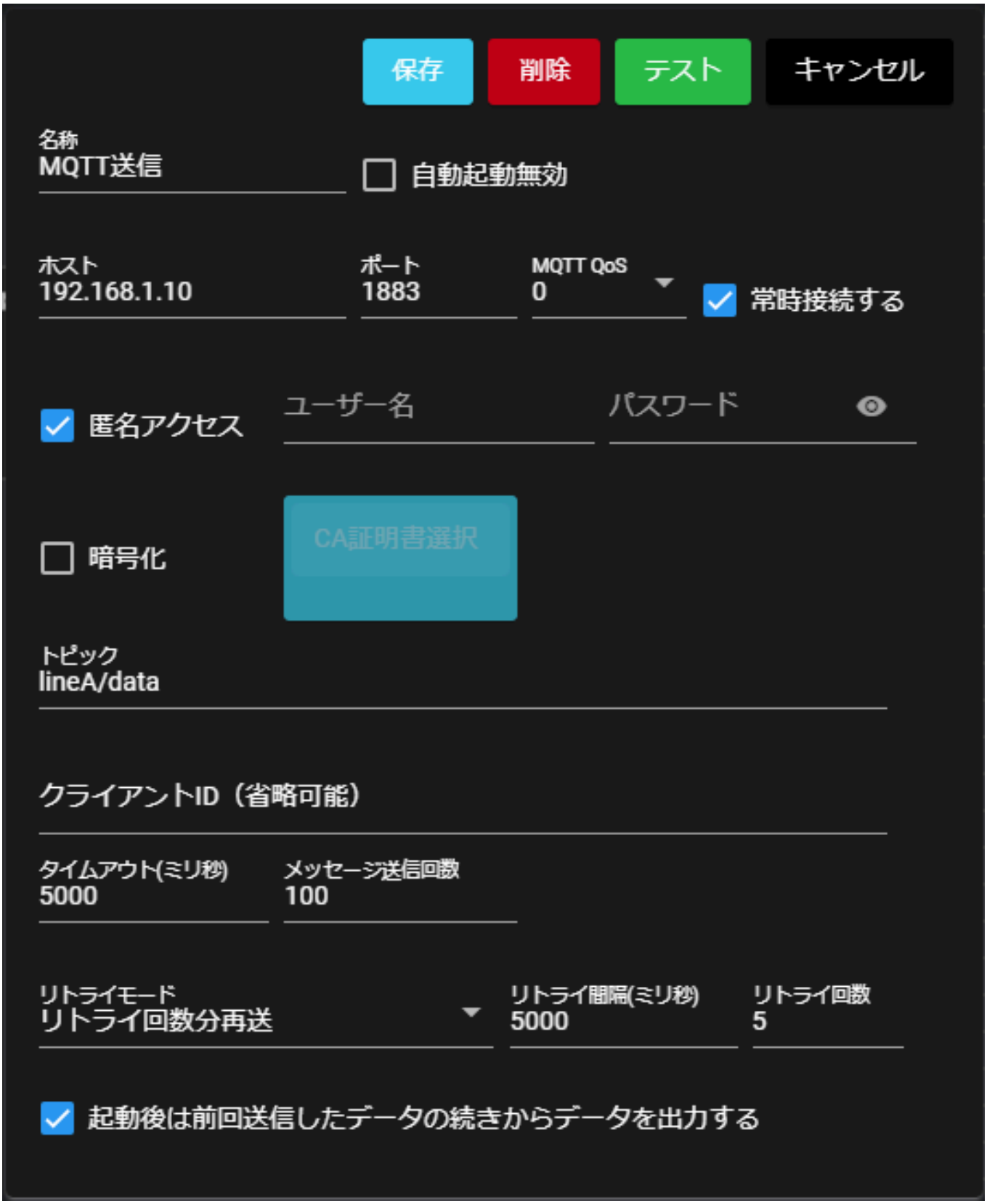
6. JSONの設定値を入力して、「保存」をクリックします。


保存
削除
キャンセル
名称
JSON変換
自動起動無効
整形する
インデント
2
タイムゾーン
UTC
切替基準
時間
切替時間(秒)
60
上限サイズ(KB)
96
デフォルト小数桁数
3
通算ナノ時間付きで出力する
起動後は前回送信したデータの続きからデータを出力する
固定値カラム名
値
データなし
追加

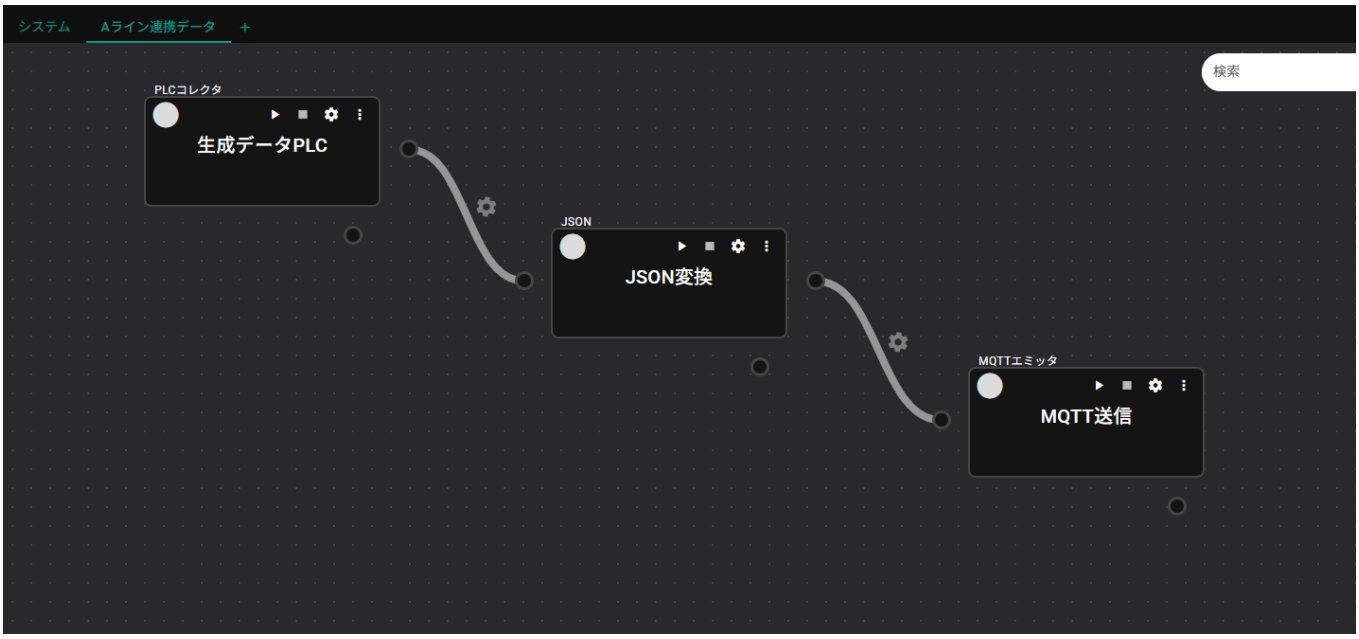
7. MQTTエミッタを設定します。フローエディタ左部より「MQTTエミッタ」をフローエディタ中央へドラッグ＆ドロップします。



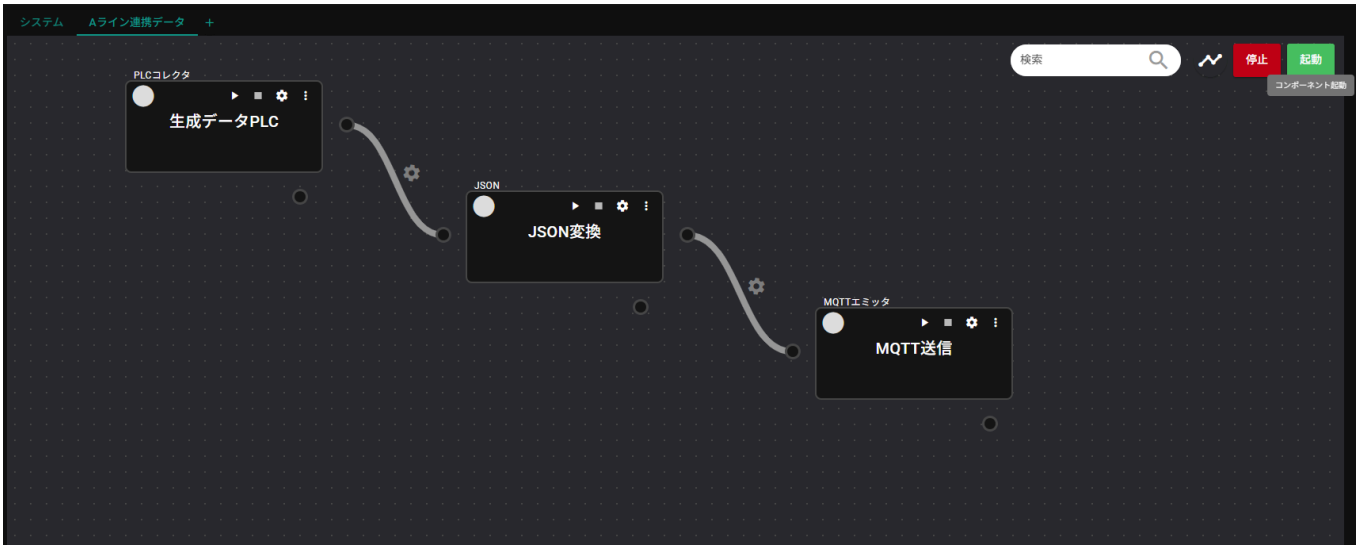
8. MQTTエミッタの設定値を入力して、「保存」をクリックします。IPアドレスやポートなどはご利用の環境に合わせて下さい。
（※もしMQTTエミッタが利用できない場合、Fileエミッタなどへ置き換えて設定してください。）



9. フローリンクを繋ぎます。

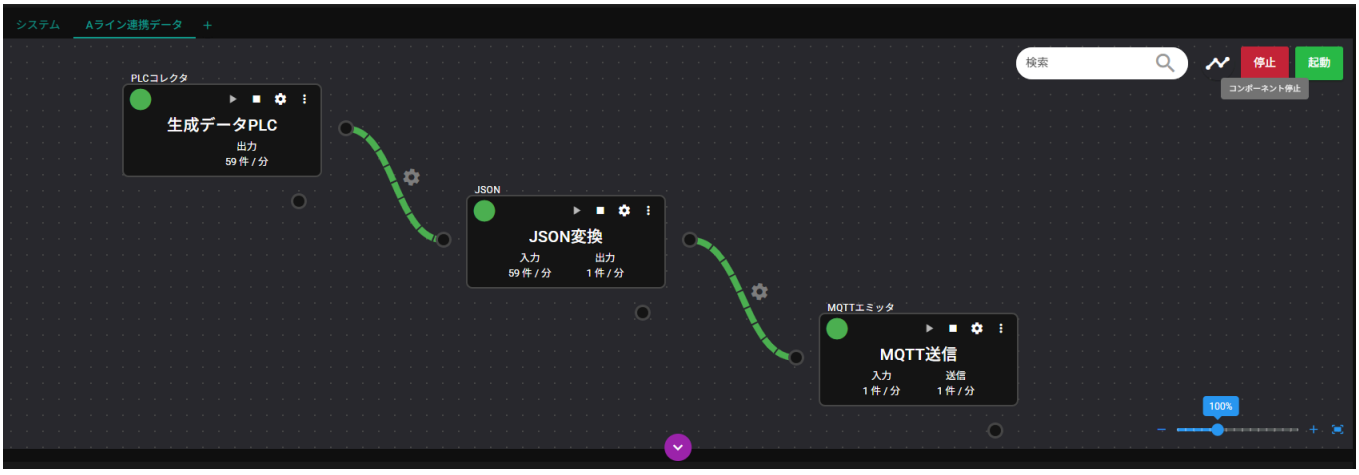


10. コンポーネントを動作させます。それぞれのコンポーネントが正常に動作することを確認してください。一定時間動作させ、異常が発生しないかなどを確認します。





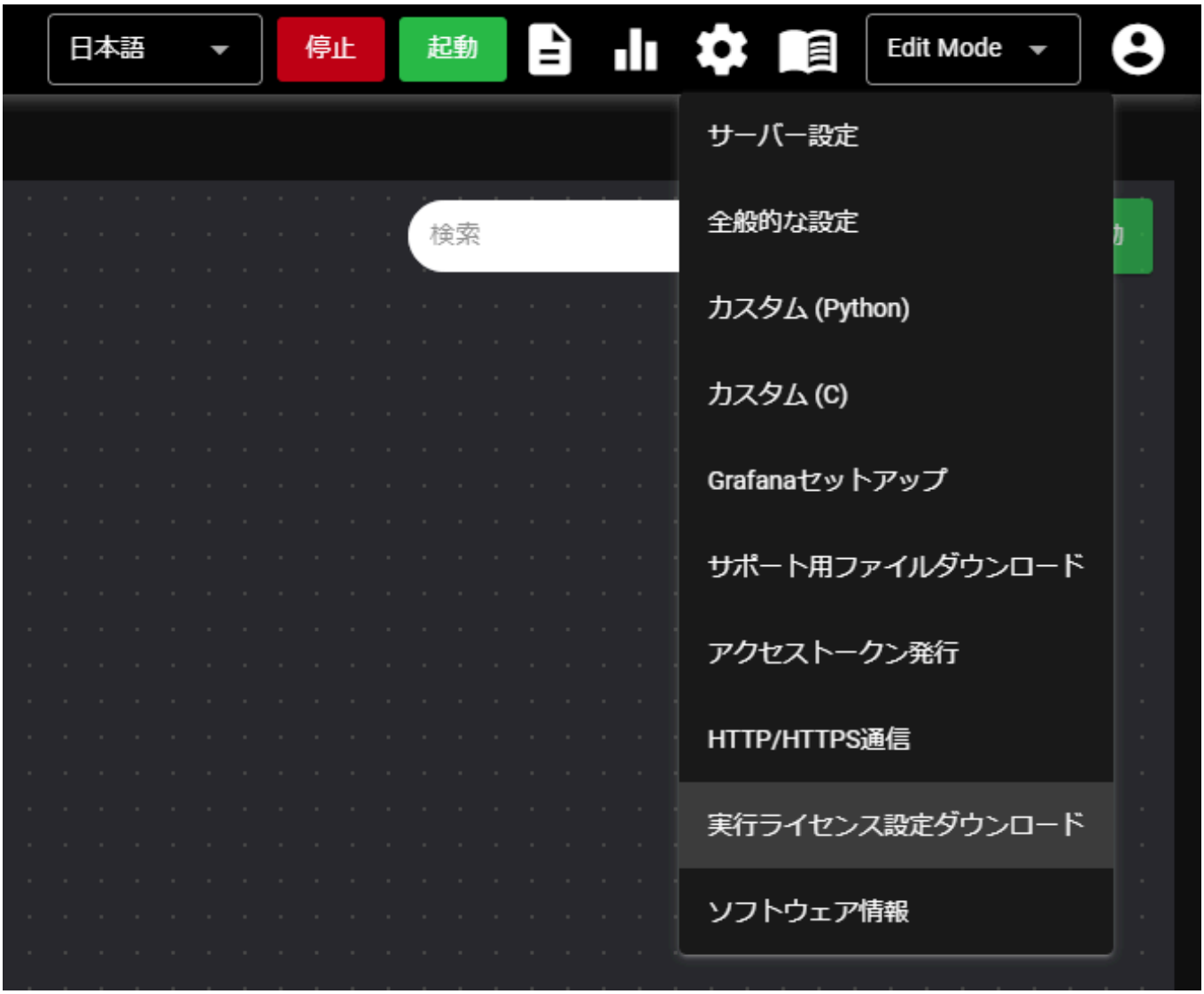
11. すべて正常に動作することを確認したら、コンポーネントを停止してください。



10.3.2.2 実行ライセンス設定の出力

開発ライセンス稼働環境での設定構築が済んだら、実行ライセンス向け設定の出力を行います。

1. 設定メニューの「実行ライセンス設定ダウンロード」をクリックします。 このファイルを実行ライセンス稼働環境で使用します。

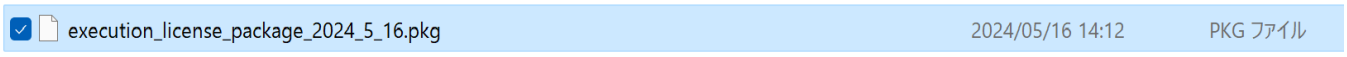


2. 確認ダイアログで、「ダウンロード」をクリックします。秘匿情報を含める場合は、「ダウンロード」のクリック前に、「秘匿情報を含める」をチェックしてください。選択したパネルのみ含める場合は、含めるパネルをチェックしてください。また、初期の自動起動設定を指定できます。▶（自動起動）■（自動起動無効）



> **秘匿情報について**
>
> 秘匿情報（例えば、Azureの接続文字列やMQTT等のユーザ名、パスワードなど）を含めて、実行ライセンス設定ダウンロードを行うと、出力ファイルに秘匿情報が含まれていることに注意してください。
> また、各コンポーネントに設定した証明書のファイルおよびカスタムコンポーネントの添付ファイルも秘匿情報の扱いとなります。秘匿情報を含めない場合は、これらが出力されないため、実行ライセンス稼働マシン側で、秘匿情報を個別に設定する必要があります。

3. 次のようなファイルがダウンロードされます。ファイル名の形式は、「execution_license_package_西暦_月_日.pkg」になります。

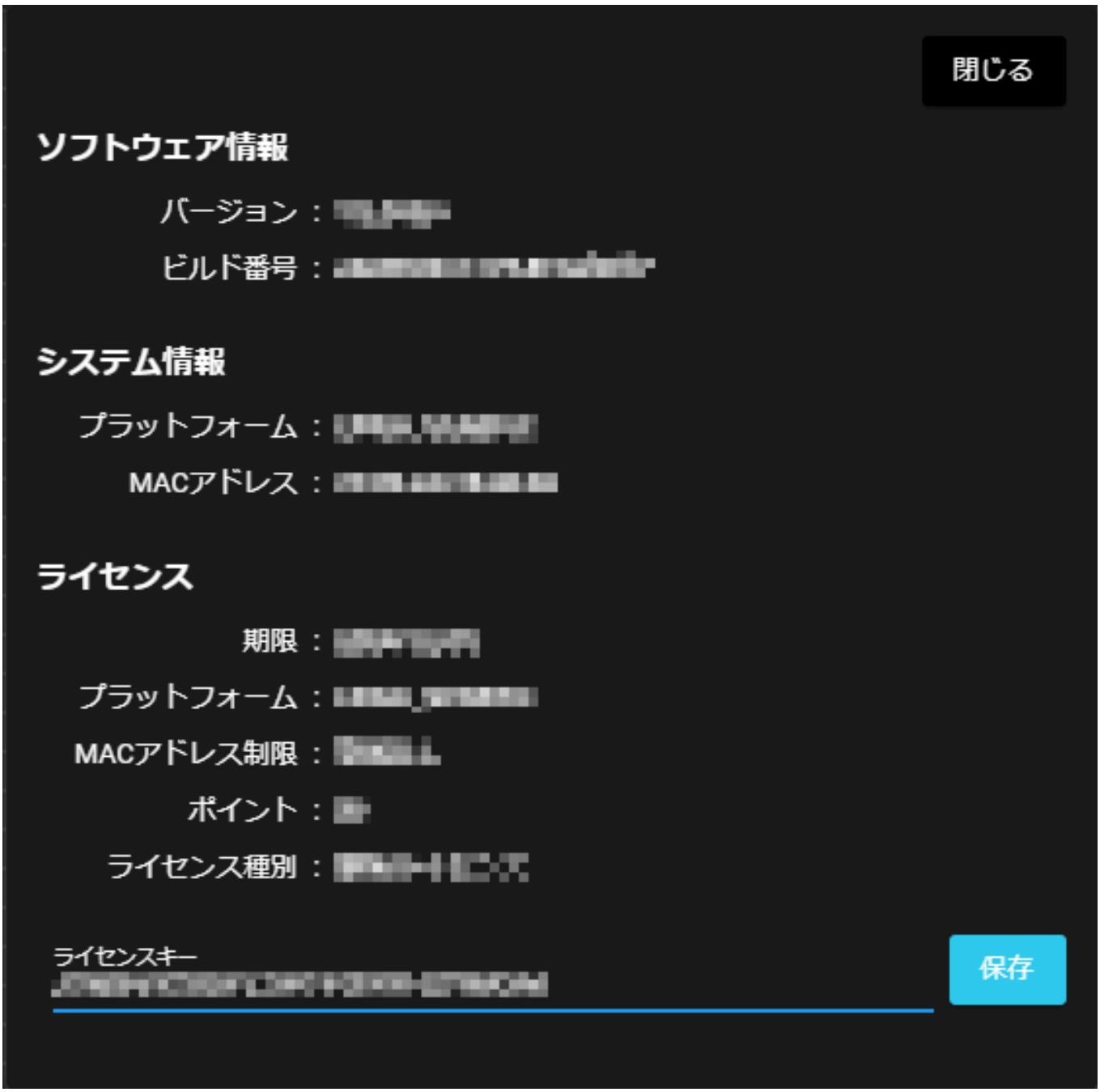


10.3.2.3 実行ライセンス稼働環境のセットアップ

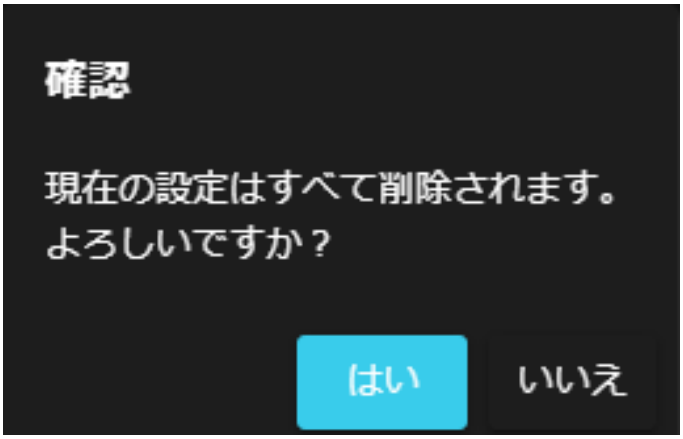
実行ライセンス稼働環境での設定を行います。SpeeDBee Synapseの「Edit Mode」にて操作を行います。

1. 設定メニューの「ソフトウェア情報」からライセンスキーを入力して保存します。





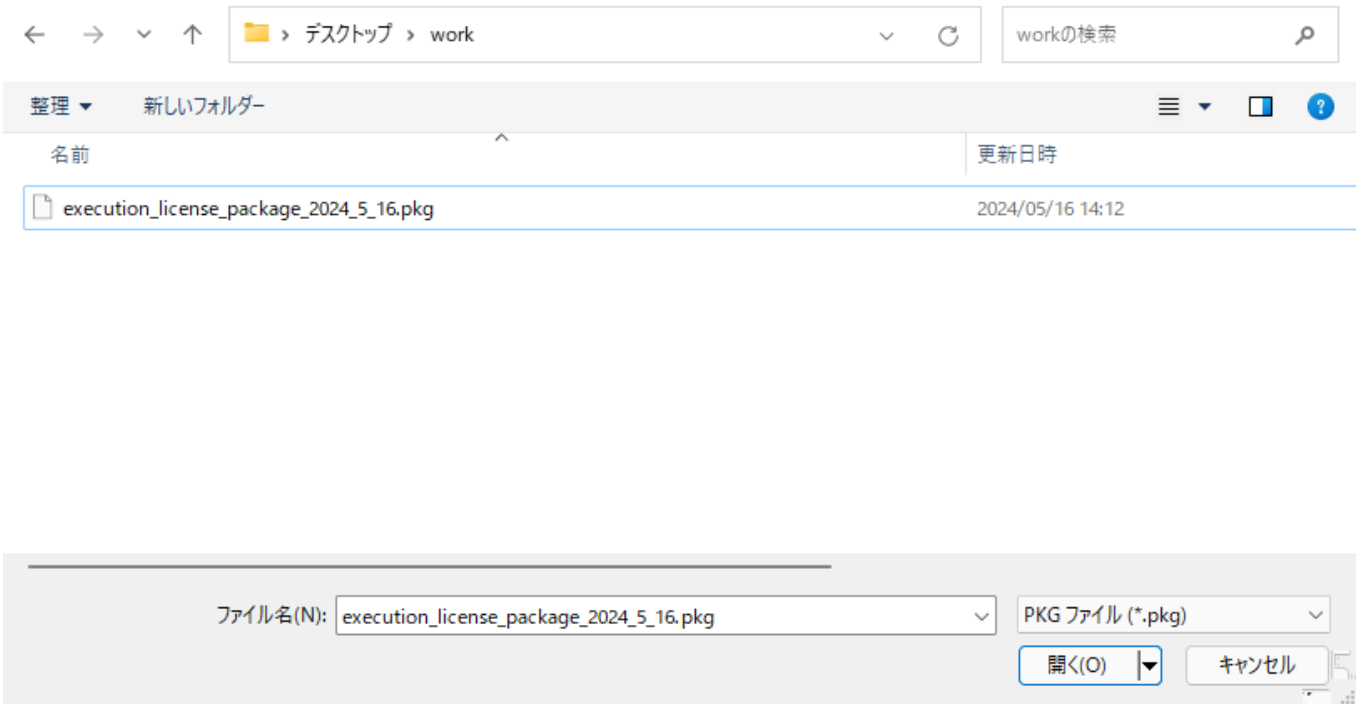
2. メッセージを確認して問題なければ、「はい」をクリックします。



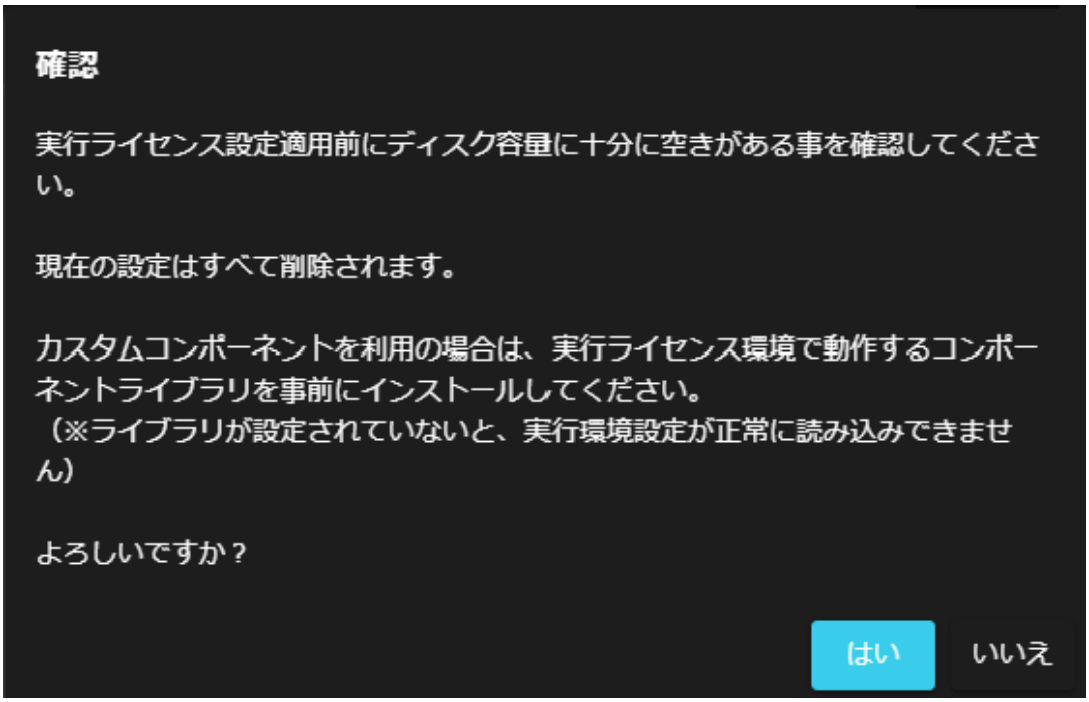
3. 実行ライセンス用設定ファイルをインポートします。「インポート」をクリックします。



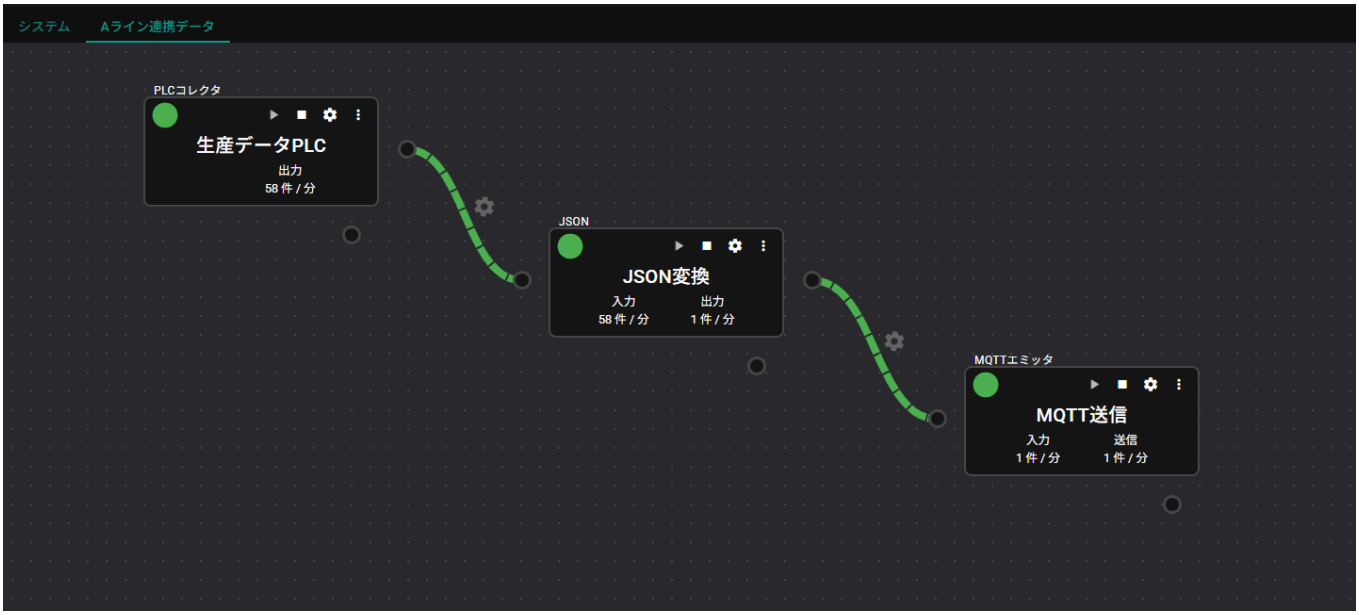
4. 実行ライセンス向け設定ファイルを選択します。



5. メッセージを確認して問題なければ、「はい」をクリックします。



6. 一定時間経過後に、コンポーネントが起動します。
（※表示の確認は、「ソフトウェア情報」を閉じて確認します。）



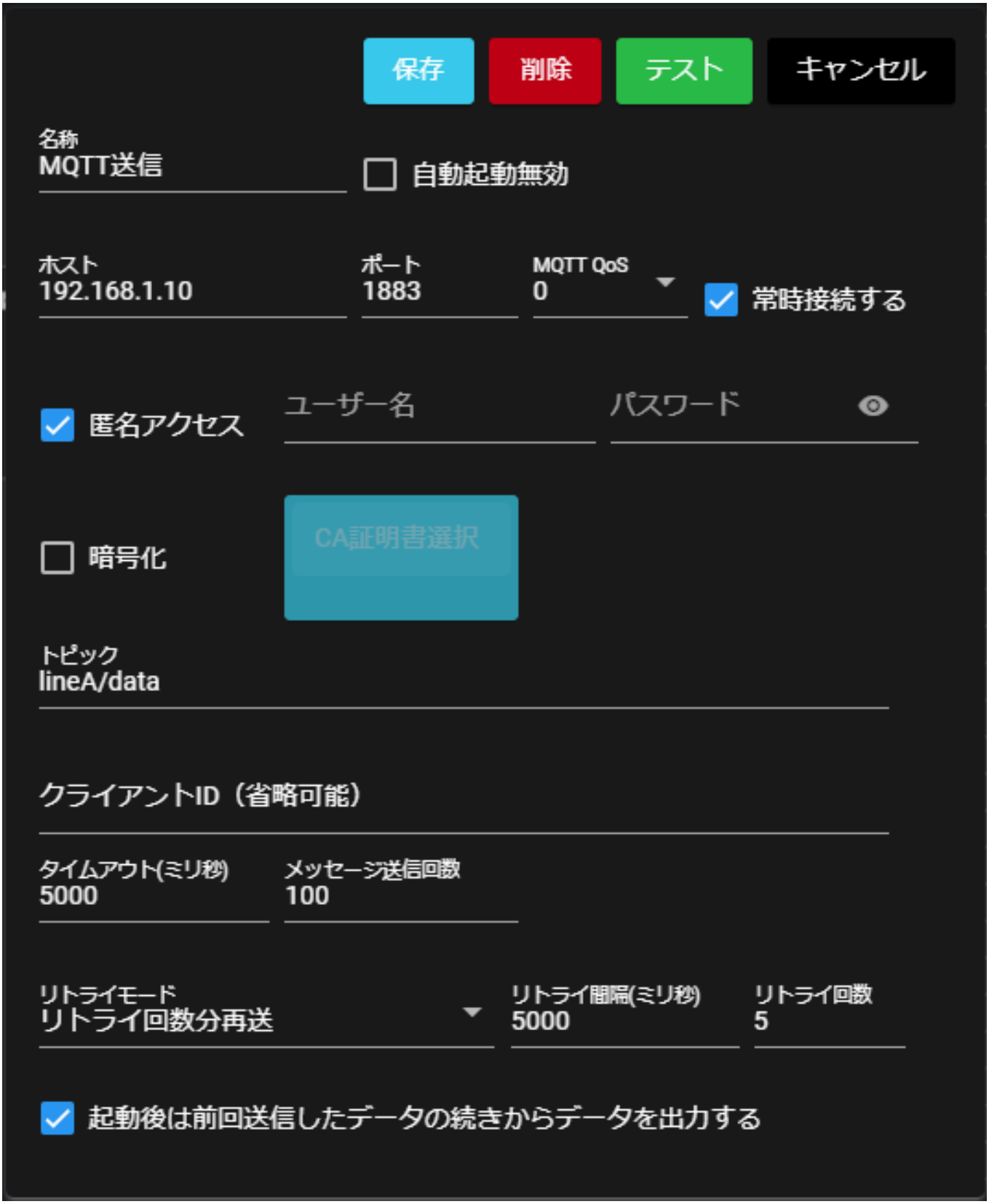
秘匿情報を持つコンポーネントについて

秘匿情報を含めていない実行ライセンス向け設定をインポートした場合、それらの情報が含まれていないため、秘匿情報の設定を必要とするコンポーネントは、自動起動が無効になります。秘匿情報や添付ファイルについては、個別に再設定してください。

7. PLCコレクタの設定を必要に応じて変更します。開発ライセンス稼働マシンと実行ライセンス稼働マシンでは、接続先が異なる場合など設定を行います。
（※チュートリアルでは、同じネットワーク環境を前提としているため、開発ライセンス側を停止していれば変更の必要はありません。）



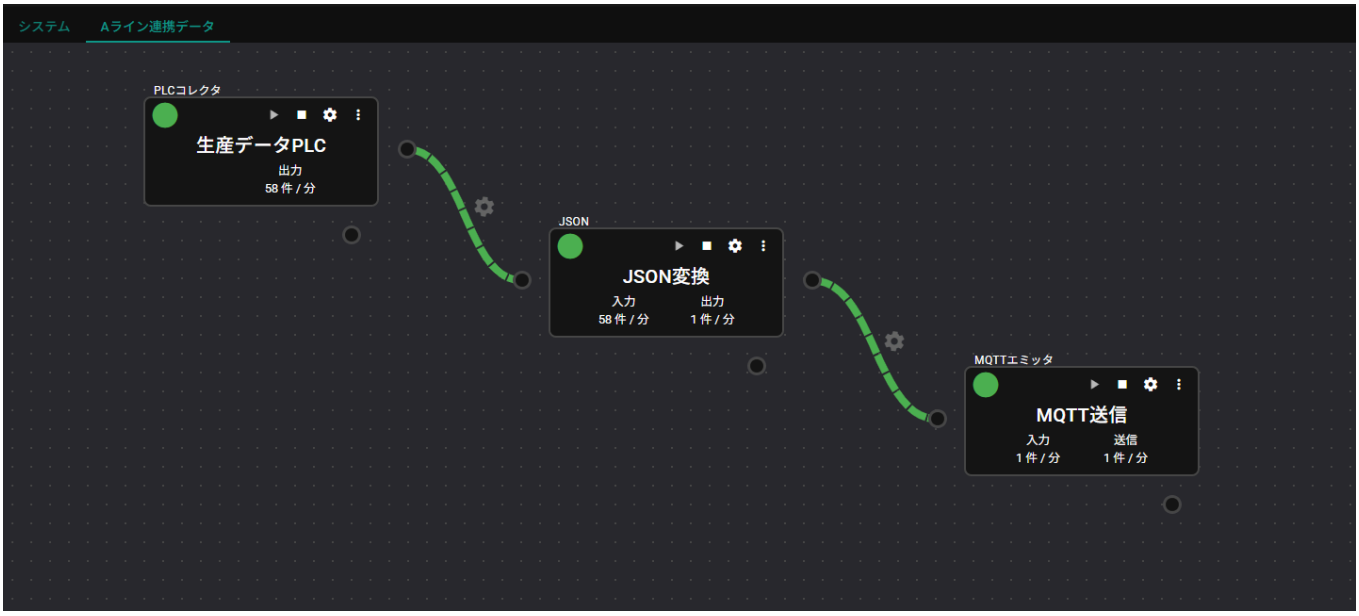
8. MQTTコレクタの設定を必要に応じて変更します。開発ライセンス稼働マシンと実行ライセンス稼働マシンでは、MQTTブローカーへの接続情報が異なっている場合など設定を行います。
（※チュートリアルでは、同じネットワーク環境を前提としているため、開発ライセンス側を停止していれば変更の必要はありません。）



10.3.2.4 動作確認

前項までの設定で、正しく動作することを確認します。

1. コンポーネントのランプを確認します。警告やエラーがないことを確認します。問題があれば設定の見直しを行います。



10.3.2.5 運用・モニタリング・可視化

前項までの設定で、正常に稼働できることが確認できれば運用に入ります。 モニタリング・可視化については、導入フローを参照してください。
実行ライセンスの運用チュートリアルは以上となります。

10.4 実行ライセンス稼働環境に関する注意事項

実行ライセンス稼働環境に関する注意事項を次に示します。

注意事項

- 実行ライセンスを適用すると、実行ライセンスに従った動作モードとなり、使用できる機能が制限されます。（参考：実行ライセンス利用環境の動作仕様）
- 実行ライセンス向け設定の取込を行うと、選択中のプロジェクトのそれ以前の設定は削除されます。（カスタムコンポーネントや外部モジュールは削除されません）
- 実行ライセンスに切り替わる際、選択されていないプロジェクトは全て削除されます。残しておきたいプロジェクトは事前にエクスポートしておいてください。（参考：エクスポート）
- アクセストークンについては、開発ライセンス側の設定を引き継ぎません。また、実行ライセンス側で生成したアクセストークンは、実行ライセンス向け設定の取込を行っても削除されません。
- DBクエリサービスの設定は、実行ライセンス側へ引き継がれます。しかし、トークン認証に使用するアクセストークンは引き継がれないため、実行ライセンス側で設定を行う必要があります。
- 実行ライセンス向け設定で、カスタムコンポーネントやPythonの外部モジュールを使用している場合は、実行ライセンス向け設定取込前に、それらを事前に設定している必要があります。
- 実行ライセンス向け設定の取込前に、稼働マシンのストレージ容量の空きが十分であるかなど確認してください。ストレージ容量が不足した場合は、実行ライセンス向け設定の取込に失敗します。
- 実行ライセンス稼働環境において、ユーザー管理を行う場合は、それぞれの稼働環境毎にユーザー管理を行う必要があります。開発ライセンス側の設定は引き継がれません。

11 i-Trigger（拡張パッケージ）

11.1 i-Trigger（拡張パッケージ）

ここでは、SpeeDBee Synapseの拡張パッケージであるi-Triggerについて説明します。

記載内容は次のとおりです。

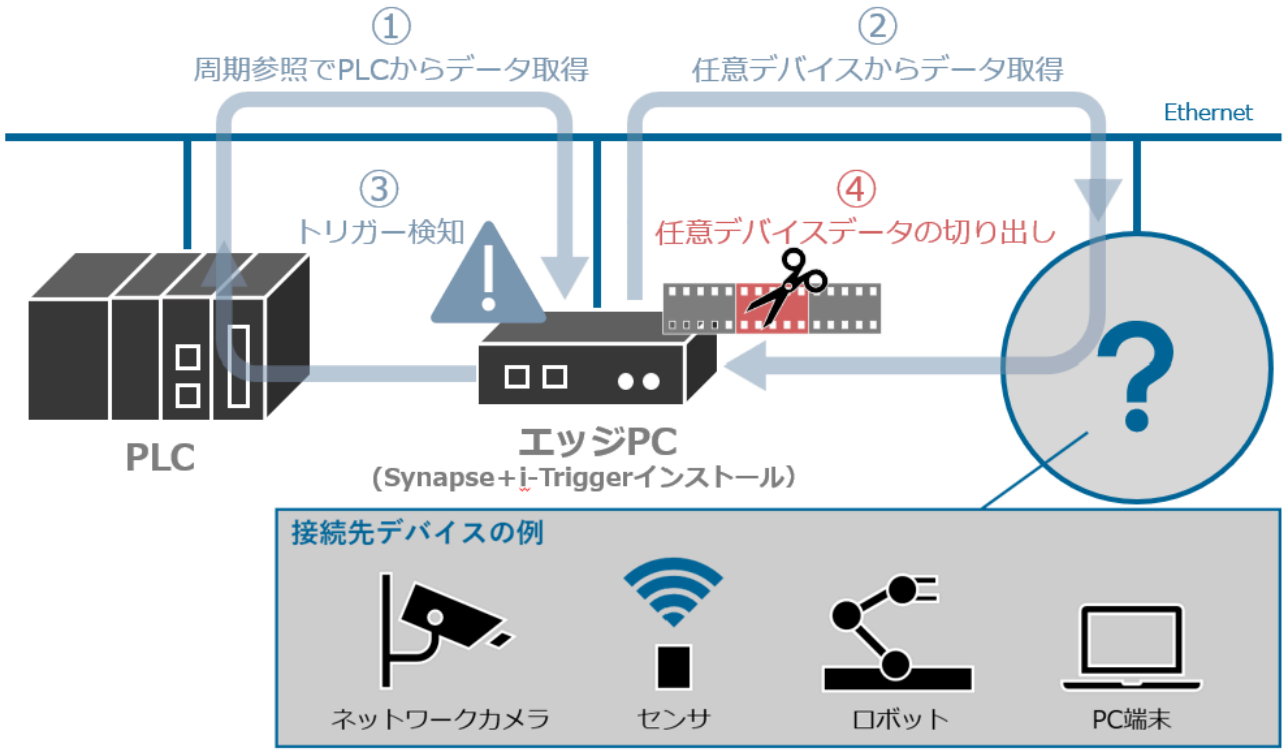
1. i-Triggerの概要
2. 利用するカメラについて
3. インストール
4. 動画再生用ビューワ
5. コンポーネント
6. チュートリアル
7. i-Triggerの記録仕様

11.2 概要

ここではi-Triggerの概要について説明します。

11.2.1 i-Triggerの特長

i-Trigger（アイトリガー）は、製造業におけるチョコ停を監視するSpeeDBee Synapseの拡張パッケージです。
設備におけるドライブレコーダーのような機能を提供します。製造ラインに異常が発生した際に、各種コンポーネントが収集したデータから、異常発生前後の数値データ、カメラで撮影した動画データを自動的に切出し指定のファイル書式で保存します。



処理概要

① 周期参照でPLCからデータ収集

トリガー検知および記録対象となるデータをPLCから取得します。

②任意デバイスからデータ取得

PLCだけでなく、各コンポーネントから収集したデータもPLC同様にトリガー検知等に利用できます。

③トリガー検知

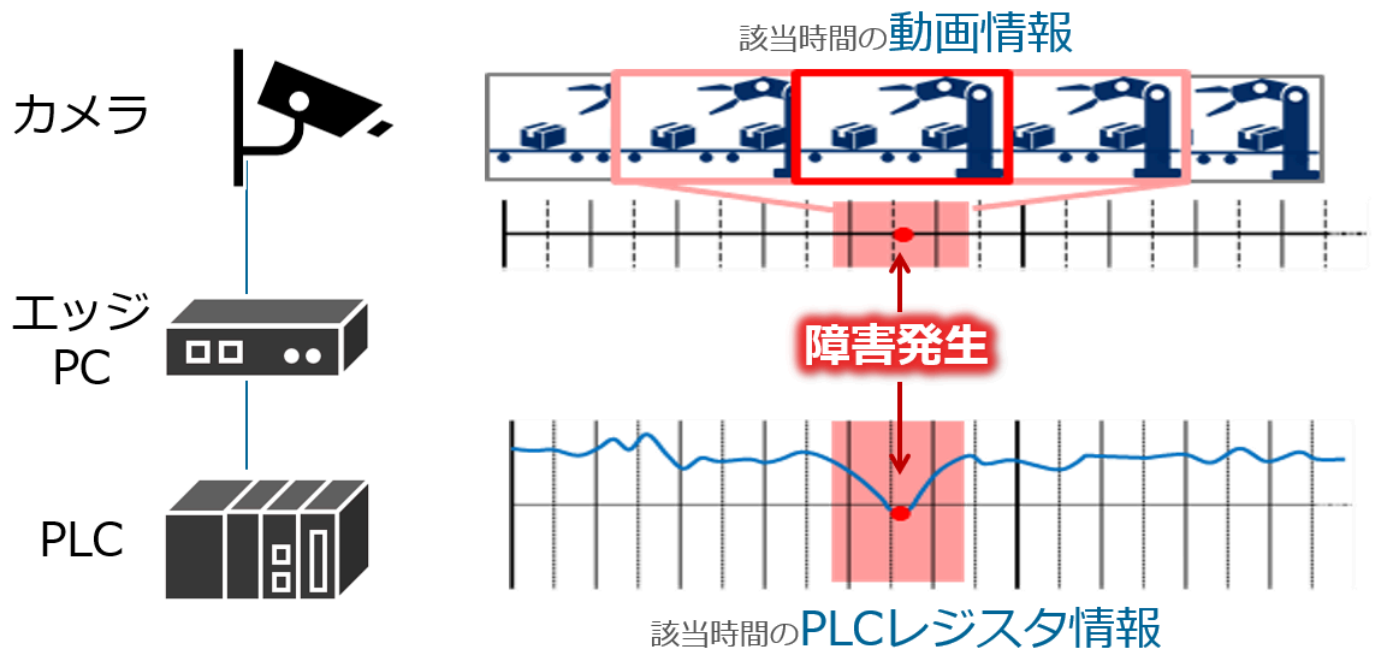
収集したデータが、事前設定した条件に合致するとトリガー検知として切り出し処理が行われます。

④任意デバイスデータの切り出し

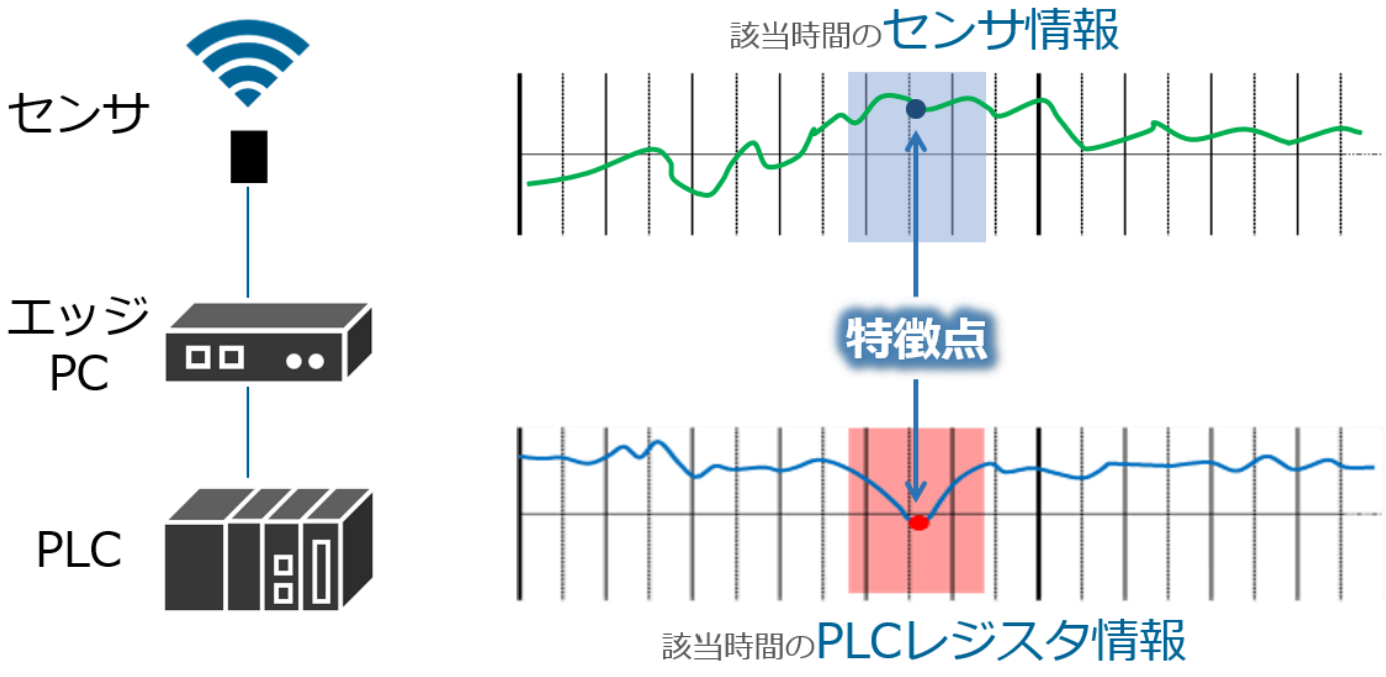
トリガー検知後に、事前設定した記録範囲に従って、対象のデバイス（ネットワークカメラやPLCの収集データ）のデータを切り出しします。切り出ししたデータは専用のビューワで参照できます。

専用ビューワ

i-Triggerには、切り出したデータを閲覧するための専用ビューワが搭載されています。下図で示すように、障害発生時の動画と収集データを同時再生できる特長があります。



動画と収集した様々なデータを同時刻のものを可視化することによって、データと事象の関連性の解析に役立ちます。



11.2.2 i-Triggerのご利用にあたって

i-Triggerは、SpeeDBee Synapseの拡張パッケージであるため、利用にはSpeeDBee Synapseの本体が必要となります。そのため、SpeeDBee Synapseと同様の環境で動作します。詳細は、実行環境を参照してください。

11.3 使用できるカメラについて

本節では、i-Triggerで使用できるカメラについて説明します。

11.3.1 使用できるカメラ仕様・対応コーデック

i-Triggerで使用できるカメラは、次の通りです。

形態	対応しているビデオコーデック	動作確認済みプロファイル
RTSP（Real-Time Streaming Protocol）対応のネットワークカメラ	H.264	• Baseline Profile • Main Profile • High Profile

11.3.2 カメラの利用台数

i-Triggerで、同時に利用できるカメラには台数の制限があります。SpeeDBee Synapseのライセンスに、カメラ台数が付与されており、その台数を超えてカメラを同時に録画することはできません。

項目	台数
標準で使用できる台数	4台
最大で利用できる台数	30台 ※使用するには、台数に応じたライセンスが必要

必要となるカメラ台数の付与されたライセンスを購入してください。

Data Flow Controller版、OMRON-SBS for Windows版では、初期状態のライセンスで接続可能なカメラ台数が30台に固定されています。

11.4 インストール

本節では、インストール手順およびアンインストール手順について説明します。

11.4.1 インストールおよびアップデート

下記の手順で、i-Triggerをインストールできます。なお、SpeeDBee Synapseとi-Triggerは、必ず同じバージョンを利用してください。アップデート時も同様の手順となります。
Windowsでは、SpeeDBee Synapseのインストール時に、同時にインストールします。「インストール（Windows）」を参照してください。

> Data Flow Controller版では、初期状態でi-Triggerがインストールされているため、インストールは不要です。

ファイルを配置

1. i-Triggerのファイルを、インストール対象のデバイス内（/tmpディレクトリ）へ配置します。下記は、Windows PowerShellからscpコマンドで配置する例です。

デバイスのOSがUbuntu20(amd64)の場合

```
PS >scp speedbeesynapse_ubuntu20_x.x.x-r0_amd64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
PS >scp speedbeesynapse-itrigger_ubuntu20_x.x.x-r0_amd64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
```

※同一バージョンのSpeeDBee Synapseのインストールが済んでいる場合は、SpeeDBee Synapseの転送は不要です。
※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(amd64)の場合

```
PS >scp speedbeesynapse_ubuntu22_x.x.x-r0_amd64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
PS >scp speedbeesynapse-itrigger_ubuntu22_x.x.x-r0_amd64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
```

※同一バージョンのSpeeDBee Synapseのインストールが済んでいる場合は、SpeeDBee Synapseの転送は不要です。
※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(arm64)の場合

```
PS >scp speedbeesynapse_ubuntu22_x.x.x-r0_arm64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
PS >scp speedbeesynapse-itrigger_ubuntu22_x.x.x-r0_arm64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
```

※同一バージョンのSpeeDBee Synapseのインストールが済んでいる場合は、SpeeDBee Synapseの転送は不要です。
※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがRaspberry Pi OS(bookworm 64-bit)の場合

```
PS >scp speedbeesynapse_raspi_bookworm_x.x.x-r0_arm64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
PS >scp speedbeesynapse-itrigger_raspi_bookworm_x.x.x-r0_arm64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
```

※同一バージョンのSpeeDBee Synapseのインストールが済んでいる場合は、SpeeDBee Synapseの転送は不要です。
※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがdebian12(bookworm arm64)の場合

```
PS >scp speedbeesynapse_debian12_x.x.x-r0_arm64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
PS >scp speedbeesynapse-itrigger_debian12_x.x.x-r0_arm64.deb （アカウント名)@(デバイスのIPアドレス):/tmp
```

※同一バージョンのSpeeDBee Synapseのインストールが済んでいる場合は、SpeeDBee Synapseの転送は不要です。
※x.x.xの部分には、バージョン番号が入ります。

インストール及び起動

1. sshを使用してデバイスへログインします。下記はWindows PowerShellからsshコマンドでログインする例です。

```
PS >ssh (アカウント名)@(デバイスのIPアドレス)
```

2. 下記のコマンドを実行し、i-Triggerをインストールします。

デバイスのOSがUbuntu20(amd64)の場合

- SpeeDBee Synapseを同時インストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse_x.x.x-r0_amd64.deb /tmp/speedbeesynapse-itrigger_ubuntu20_x.x.x-r0_amd64.deb
```

- i-Triggerのみインストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse-itrigger_ubuntu20_x.x.x-r0_amd64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(amd64)の場合

- SpeeDBee Synapseを同時インストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse_ubuntu22_x.x.x-r0_amd64.deb /tmp/speedbeesynapse-itrigger_ubuntu22_x.x.x-r0_amd64.deb
```

- i-Triggerのみインストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse-itrigger_ubuntu22_x.x.x-r0_amd64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがUbuntu22(arm64)の場合

- SpeeDBee Synapseを同時インストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse_ubuntu22_x.x.x-r0_arm64.deb /tmp/speedbeesynapse-itrigger_ubuntu22_x.x.x-r0_arm64.deb
```

- i-Triggerのみインストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse-itrigger_ubuntu22_x.x.x-r0_arm64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがRaspberry Pi OS(bookworm 64-bit)の場合

- SpeeDBee Synapseを同時インストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse_raspi_bookworm_x.x.x-r0_arm64.deb /tmp/speedbeesynapse-itrigger_raspi_bookworm_x.x.x-r0_arm64.deb
```

- i-Triggerのみインストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse-itrigger_raspi_bookworm_x.x.x-r0_arm64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

デバイスのOSがdebian12(bookworm arm64)の場合

- SpeeDBee Synapseを同時インストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse_debian12_x.x.x-r0_arm64.deb /tmp/speedbeesynapse-itrigger_debian12_x.x.x-r0_arm64.deb
```

- i-Triggerのみインストール（アップデート）する場合

```
$ sudo apt install /tmp/speedbeesynapse-itrigger_debian12_x.x.x-r0_arm64.deb
```

※x.x.xの部分には、バージョン番号が入ります。

上記のコマンドによりSpeeDBee Synapse本体のサービスが再起動します。

11.4.2 アンインストール

下記の手順で、デバイスからi-Triggerをアンインストールすることができます。Windowsについては、「アンインストール（Windows）」を参照してください。

1. sshを使用してデバイスへログインします。下記はWindows PowerShellからsshコマンドでログインする例です。

```
PS >ssh（アカウント名）@（デバイスのIPアドレス）
```

2. 下記のコマンドを実行し、i-Triggerをアンインストールします。

```
$ sudo apt remove speedbeesynapse-itrigger
```

または

```
$ sudo apt purge speedbeesynapse-itrigger
```

上記いずれのコマンドでも、アンインストール時の動作に違いはありません。

上記のコマンドによりSpeeDBee Synapse本体のサービスが停止します。 再び起動する場合は、SpeeDBee Synapseの制御を参考にサービスを起動してください。

アンインストール時の注意

i-Triggerをアンインストールした状態で、SpeeDBee Synapseのサービスを起動すると、i-Trigger関連（VIDEOコレクタ）などの設定が残っていた場合、それらの設定は削除されます。

再度i-Triggerをインストールする場合の手順は、「インストール」に記載の手順と同じです。

11.5 動画再生用ビューワ

本節では、記録した動画を再生するビューワについて説明します。

11.5.1 ビューワへのアクセス

動画再生用ビューワを開くには、次の方法があります。

1. URL直打ち

   ブラウザから下記のURLにアクセスすることで、動画再生用ビューワを開くことができます。

   - `http://{ホスト名orデバイスのIPアドレス}:8120/video-viewer`

     例） `http://192.168.1.3:8120/video-viewer`

2. SpeeDBee Synapseの画面から遷移する

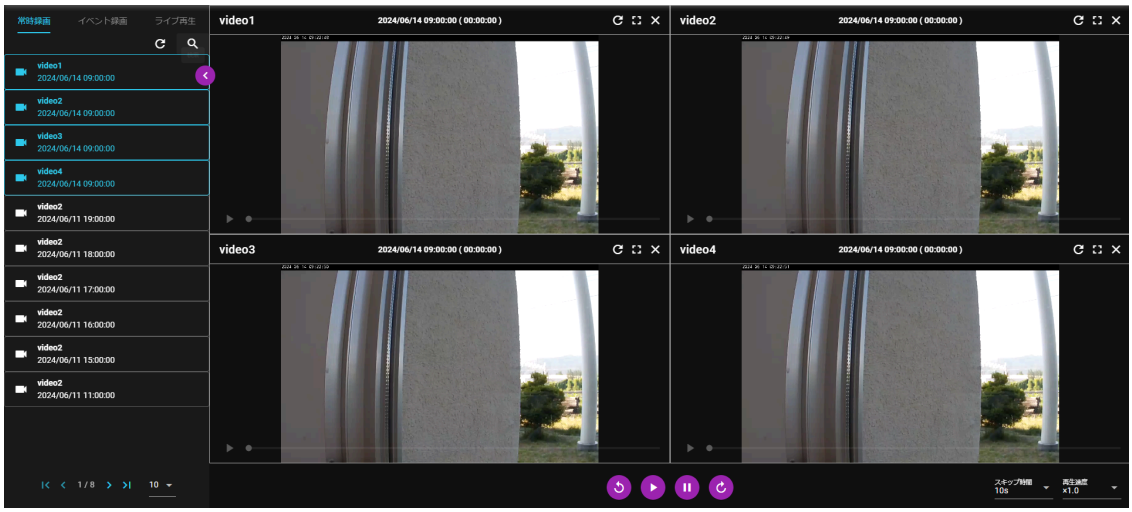
   SpeeDBee Synapseのヘッダにある、動画再生用ビューワへの遷移アイコンを押下することで、動画再生用ビューワを開くことができます。このアイコンは、i-Triggerが有効の場合のみ表示されます。



画面を開くと、最初に常時録画再生の画面が表示されています。 左メニューのタブを選択することで、表示する画面を切り替えることができます。

11.5.2 常時録画再生

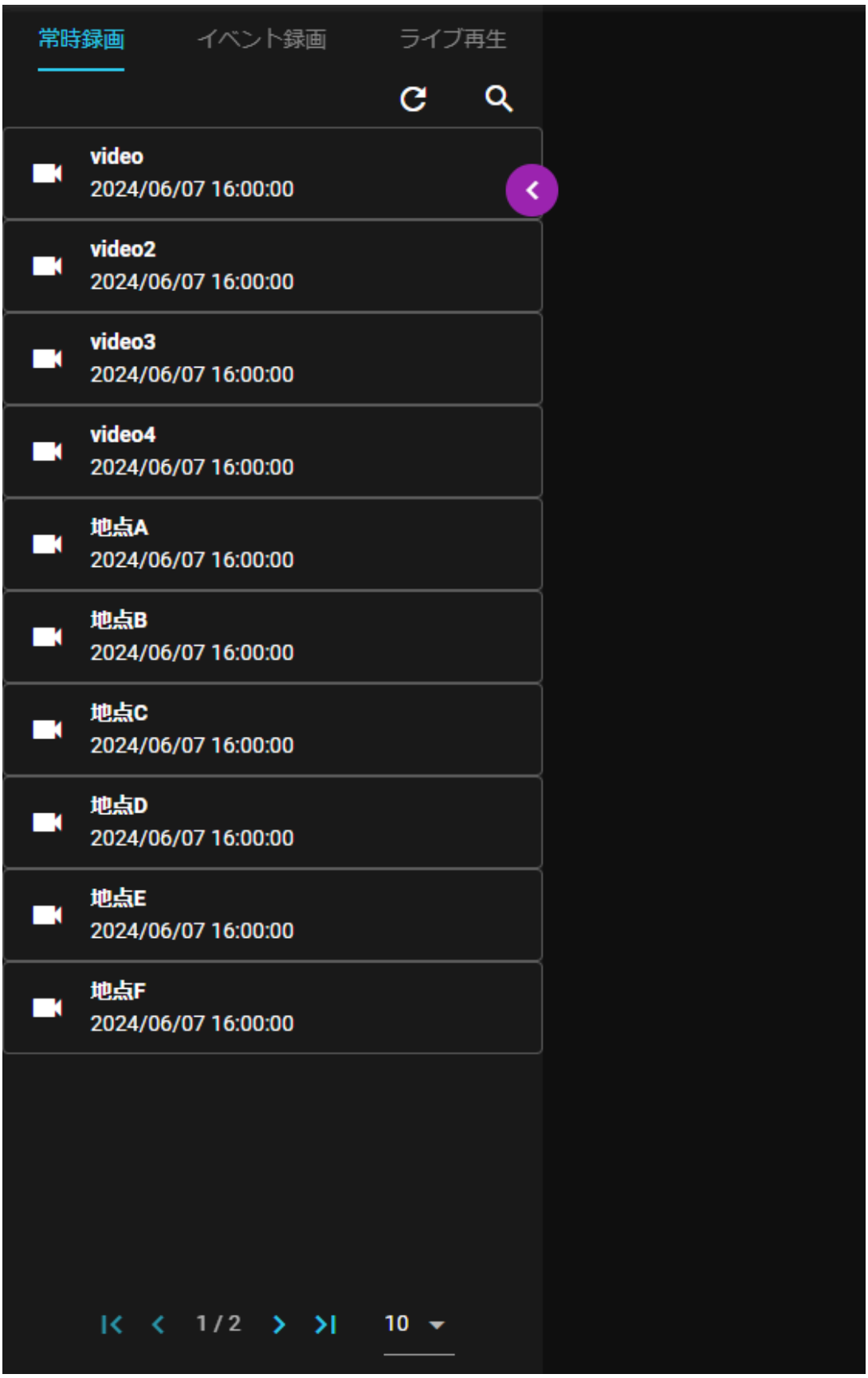
常時録画とは、VIDEOコレクタが稼働している間に常に録画している映像を指します。 常時録画の映像を再生することができます。



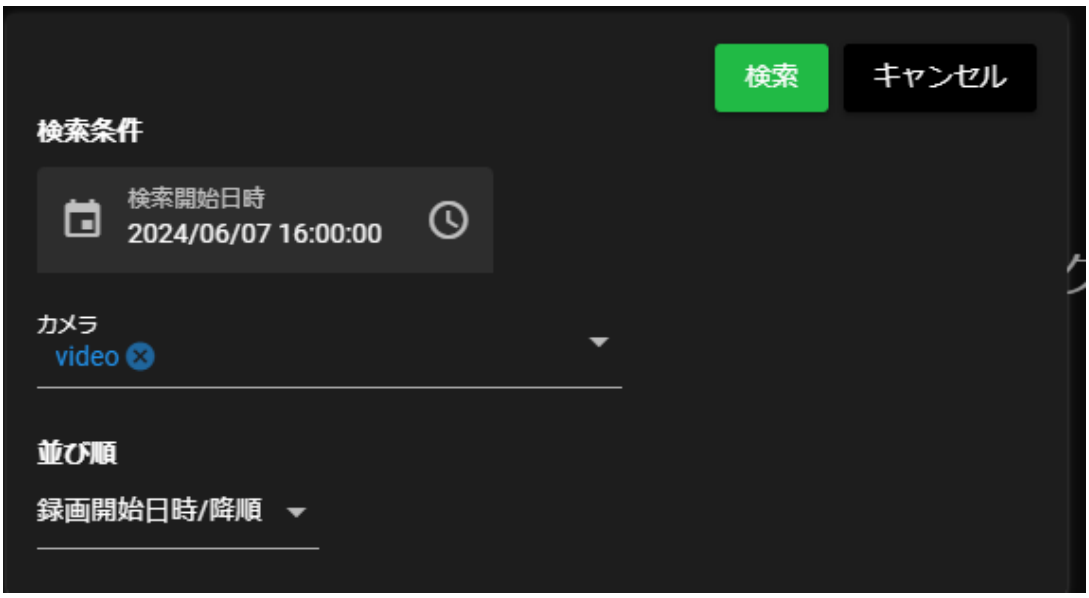
11.5.2.1 一覧

画面の左側に、常時録画の一覧が表示されます。初期状態では、画面を開いた日時以前に録画された映像が全て表示されています。また、次の手順で検索を行うことができます。

1. 虫眼鏡アイコンを押下します。

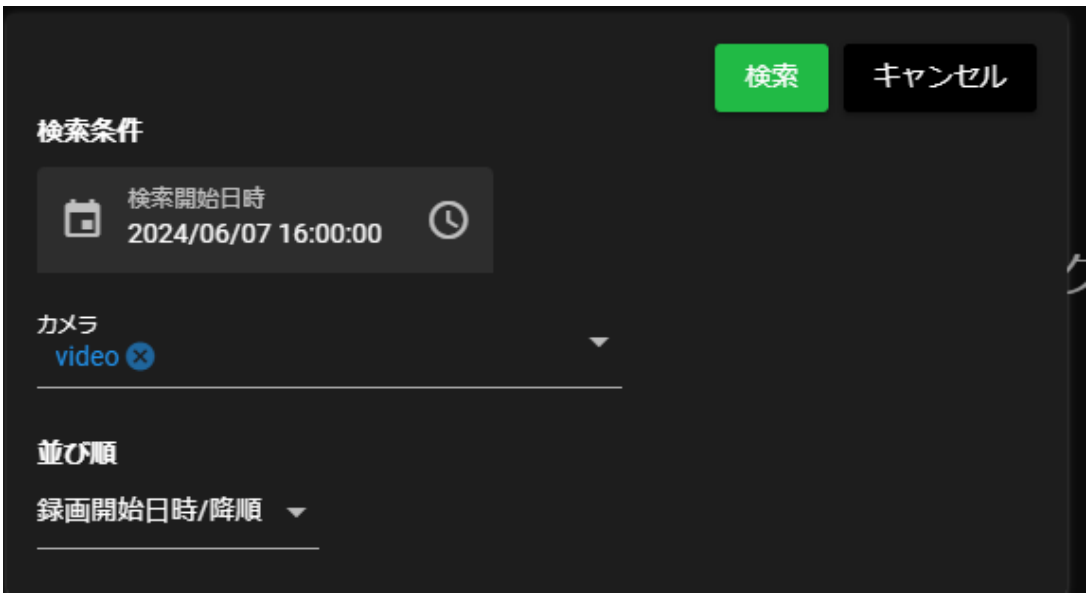


2. 以下の項目を入力します。

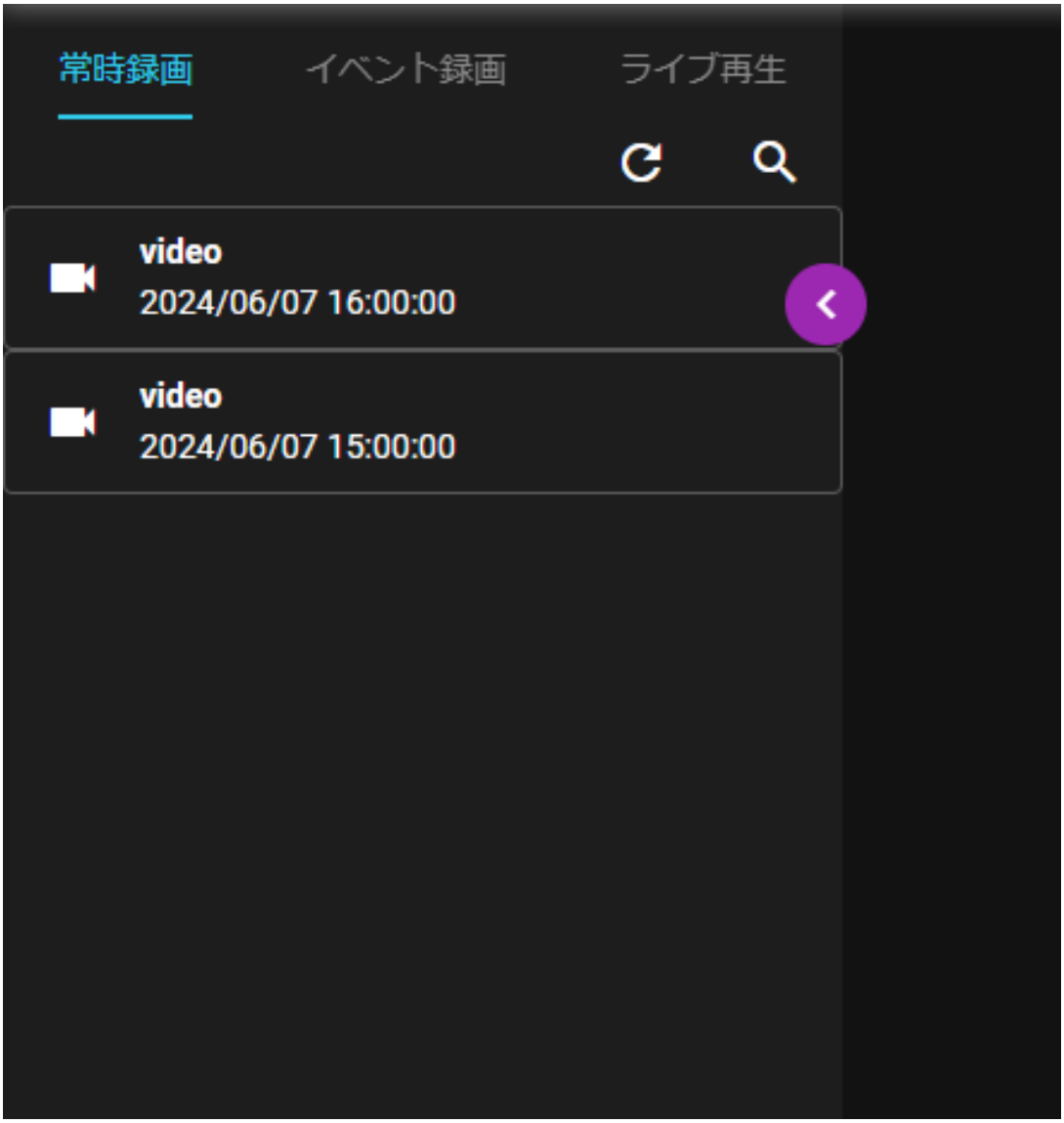


項目	説明
検索開始日時	検索対象とする時間範囲。 ※入力した時間までに録画された映像を検索します。
カメラ	検索対象とするVIDEOコレクタの名称
並び順	並び替えのキーとする項目と順序の組み合わせ ※1並び替えのキーは次の中から選択します。 ・録画開始日時 ・カメラ名 ※2並び替えの順序は次の中から選択します。 ・昇順 ・降順

3. 「検索」を押下します。



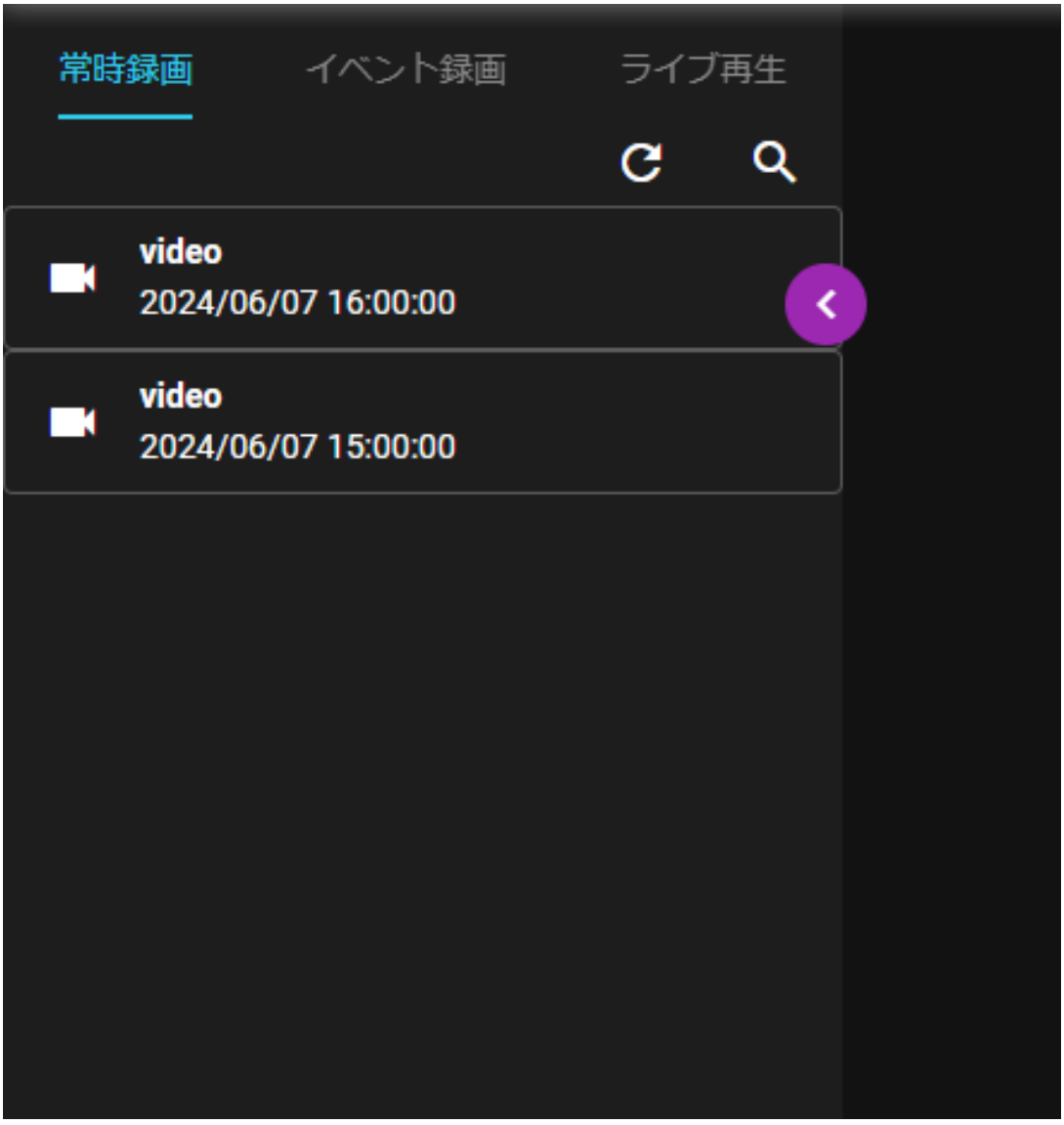
4. 検索結果が表示されます。



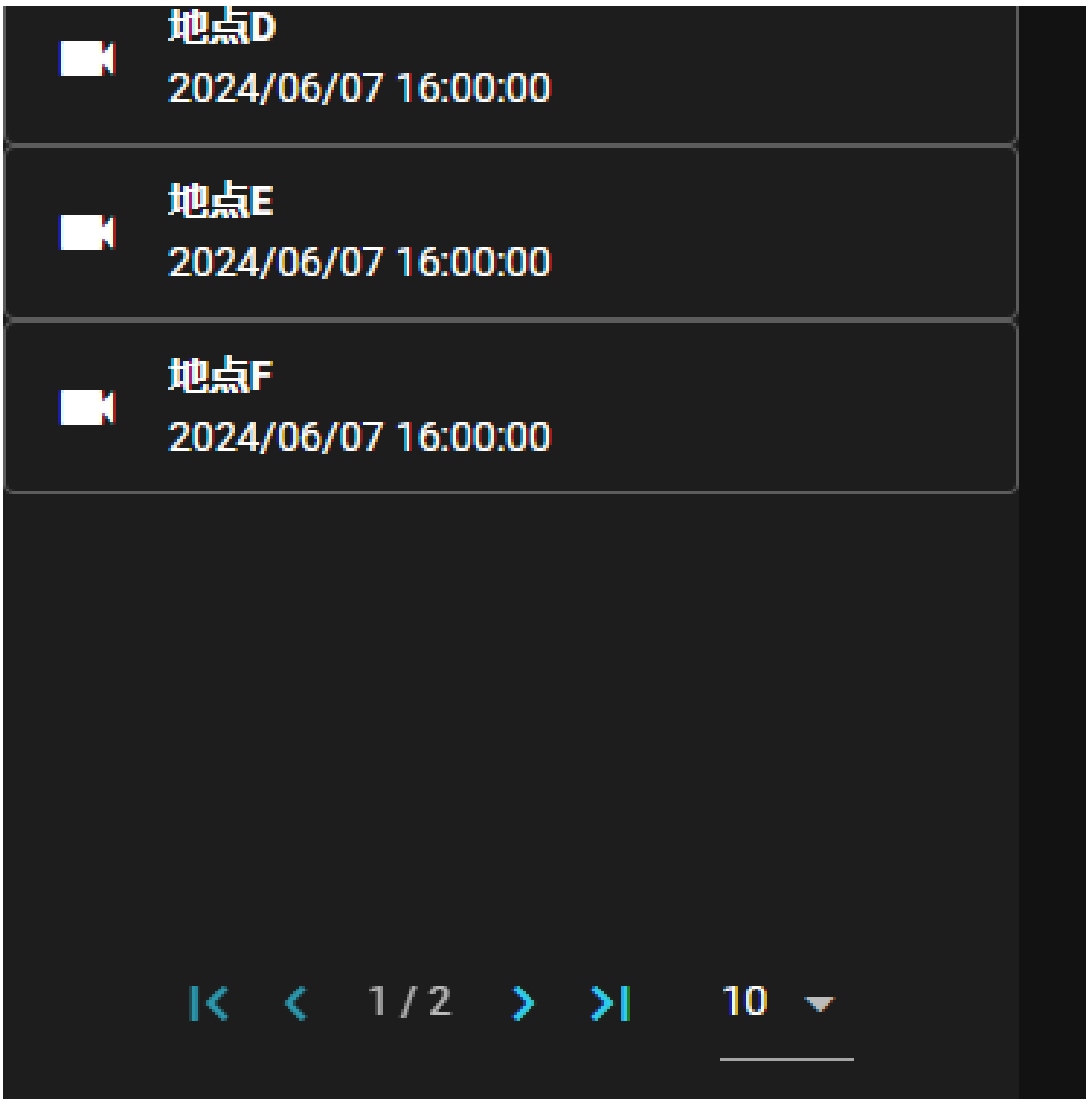
表示される項目は次の通りです。

- VIDEOコレクタの名称
- 録画開始日時

5. リフレッシュアイコンを押下すると、同じ条件で再検索を行います。また、リフレッシュは自動でも30秒に1回の間隔で行われます。



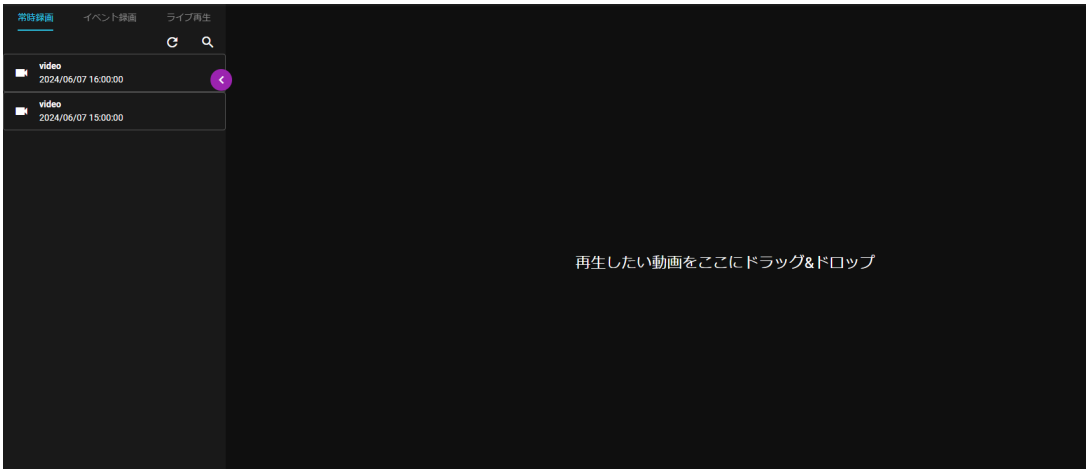
6. 表示件数が多い場合は、複数のページに分けて表示されます。1ページ当たりの表示件数を10、100、1000の中から選択することが可能です。



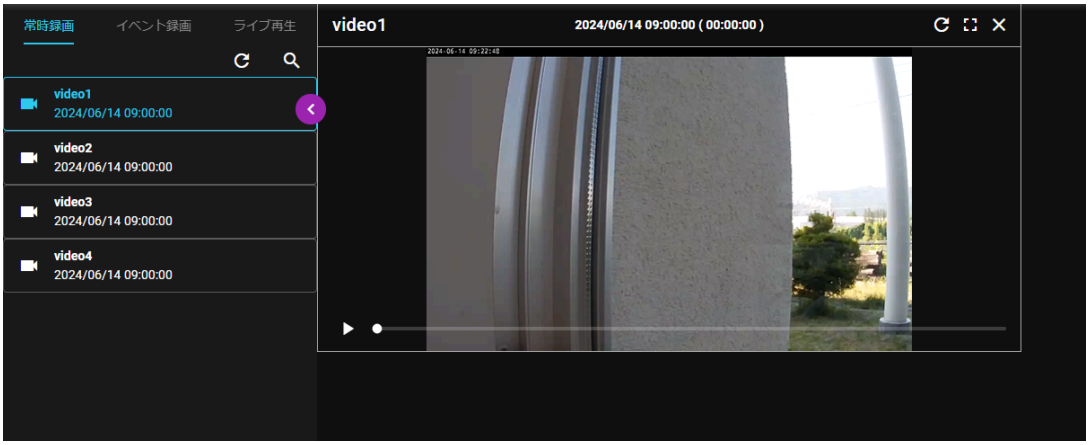
11.5.2.2 再生

次の手順で、常時録画の映像を再生することができます。同時に10個の映像を再生することができます。

1. 一覧の常時録画をドラッグし、右側にドロップします。



2. 常時録画の映像が表示されます。読み込みが完了した後、再生アイコンを押下することで再生することができます。



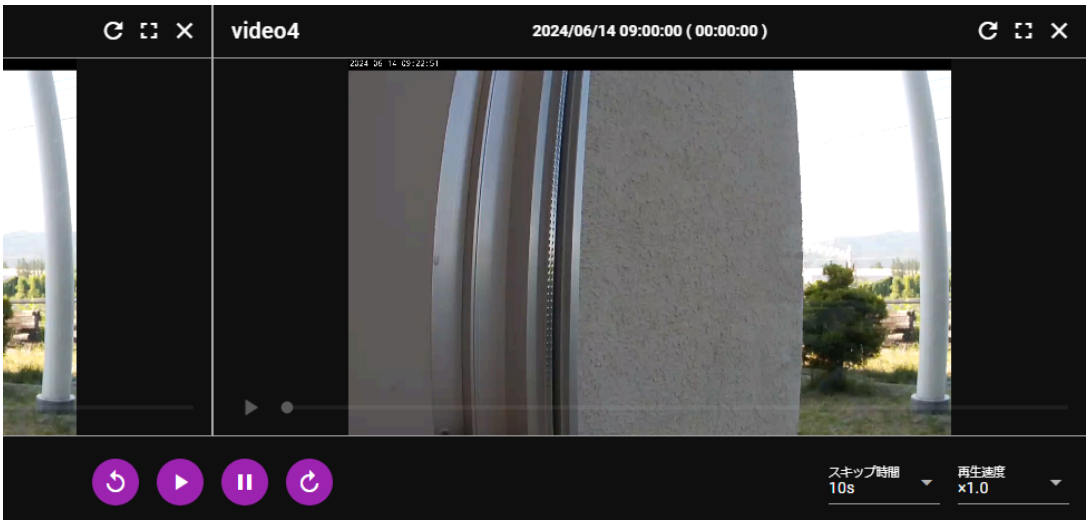
画面項目は次の通りです。



番号	説明
①	VIDEOコレクタの名称を表示します。
②	録画を開始した日時を表示します。
③	映像を再読み込みします。
④	アイコンを押下した映像を全画面で表示します。
⑤	アイコンを押下した映像を非表示とします。
⑥	常時録画の映像を表示します。

番号	説明
⑦	押下すると、映像の再生・停止を切り替えることができます。
⑧	現在の再生位置が表示されます。また、現在位置のアイコンをドラッグアンドドロップすることで、再生位置を変更することができます。

3. 画面右側の下部のエリアを操作することで、複数の動画を同時に制御することができます。操作項目は次の通りです。



項目	説明
①	押下すると、決められたスキップ時間の分だけ再生位置を前に戻します。
②	押下すると、表示されているすべての映像を再生します。
③	押下すると、表示されているすべての映像を停止します。
④	押下すると、決められたスキップ時間の分だけ再生位置を先に進めます。
⑤	スキップ送り、スキップ戻しの秒数を、次の中から選択します。 ・10秒 ・30秒 ・60秒
⑥	動画の再生速度を、次の中から選択します。 ・0.5倍速 ・0.75倍速 ・1倍速 ・1.25倍速

項目	説明
	・1.5倍速 ・2倍速

常時録画再生について

常時録画再生では、1時間単位で再生候補を表示します。複数の動画を同時に再生できることで、以下のような使い方が可能です。

- 異なるカメラの同時間帯の映像を再生（同じ時間帯の映像を確度の違うカメラで同時再生するなど）
- 同一のカメラで、違う時間帯の映像を再生（例えば正常時、異常時を一緒に見比べることが可能）

11.5.3 イベント録画再生

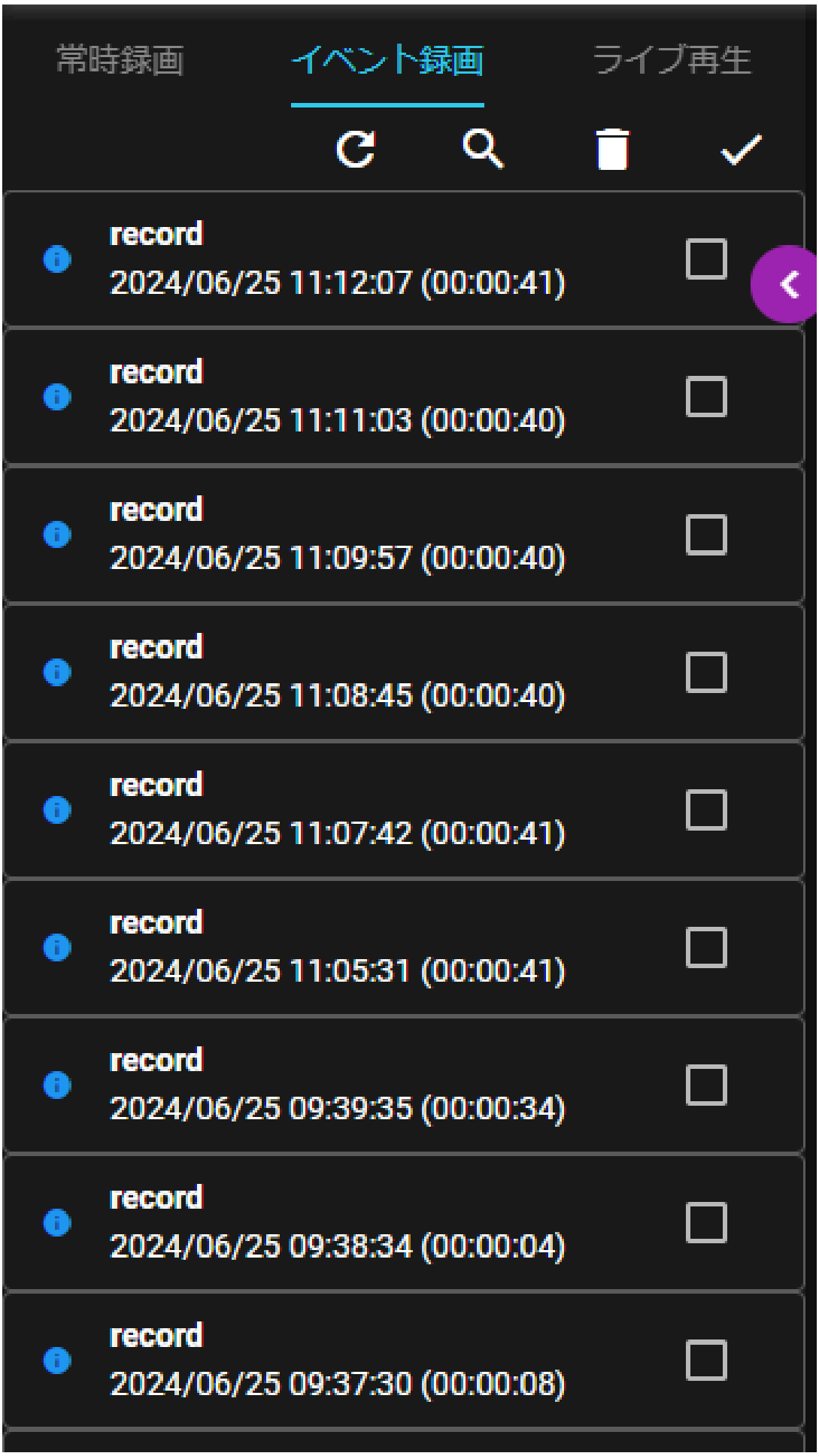
SpeeDBee SynapseのEvent Recorderで生成したイベントを確認することができます。 イベント発生時のカメラ映像、データを確認することができます。

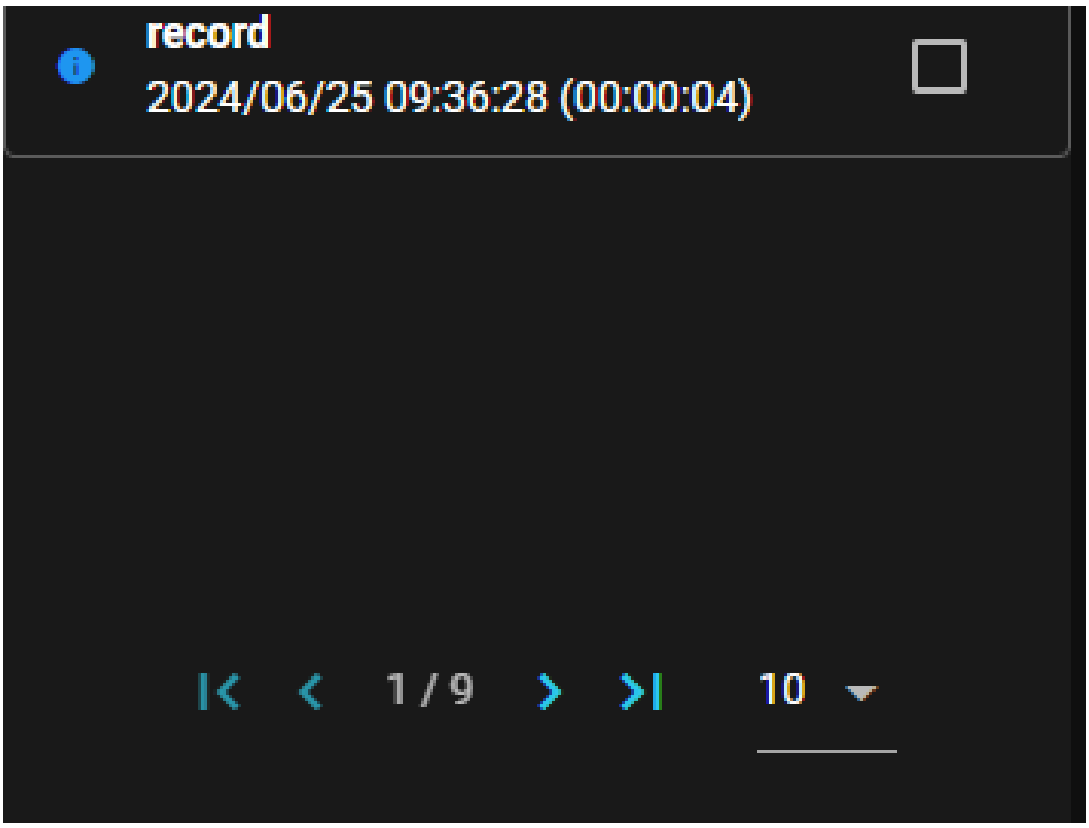


11.5.3.1 一覧

画面の左側に、イベントの一覧が表示されます。初期状態では、画面を開いた日の00:00:00~24:00:00に発生した全てのイベントが表示されています。また、次の手順でイベントを検索することができます。

1. 虫眼鏡アイコンを押下します。


常時録画
イベント録画
ライブ再生
record
2024/06/25 11:12:07 (00:00:41)
record
2024/06/25 11:11:03 (00:00:40)
record
2024/06/25 11:09:57 (00:00:40)
record
2024/06/25 11:08:45 (00:00:40)
record
2024/06/25 11:07:42 (00:00:41)
record
2024/06/25 11:05:31 (00:00:41)
record
2024/06/25 09:39:35 (00:00:34)
record
2024/06/25 09:38:34 (00:00:04)
record
2024/06/25 09:37:30 (00:00:08)



2. 以下の項目を入力します。

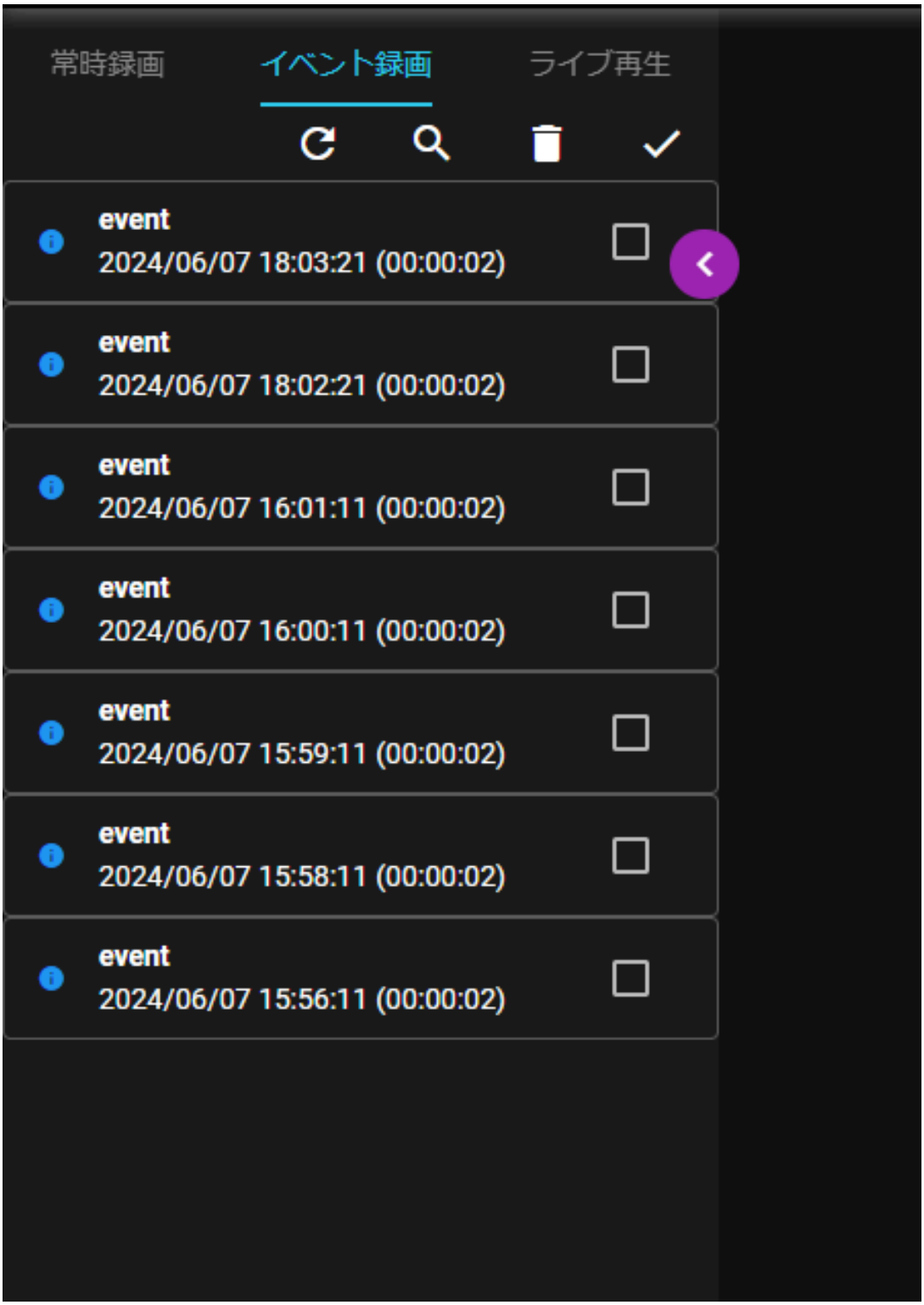


項目	説明
検索開始日時・検索終了日時	検索対象とする時間範囲。 ※入力した時間範囲の間に、開始イベントが発生したイベントを検索します。
イベント名	検索対象のイベントの名前
イベント分類	検索対象のイベントの分類
カメラ	検索対象のイベントに紐づいているカメラの名前
並び順	並び替えのキーとする項目と順序の組み合わせ ※1並び替えのキーは次の中から選択します。 ・記録開始日時 ・イベント名 ・イベント種別 ※2並び替えの順序は次の中から選択します。 ・昇順 ・降順

3. 「検索」を押下します。



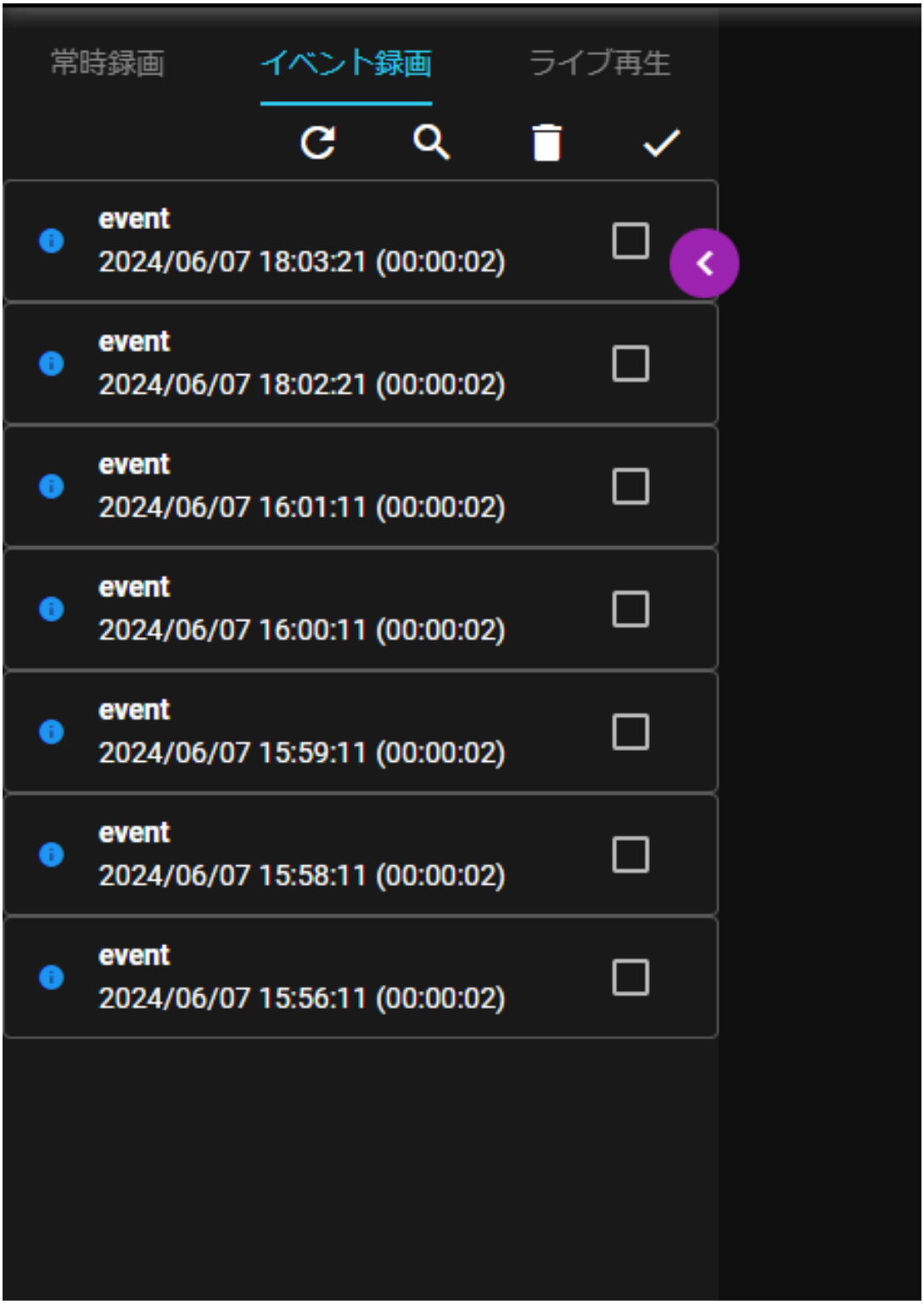
4. 検索結果が表示されます。



表示される項目は次の通りです。

- イベント分類を表すアイコン
- イベント名
- 開始イベントの発生日時
- イベントの記録時間

5. リフレッシュアイコンを押下すると、同じ条件で再検索を行います。また、リフレッシュは自動でも30秒に1回の間隔で行われます。



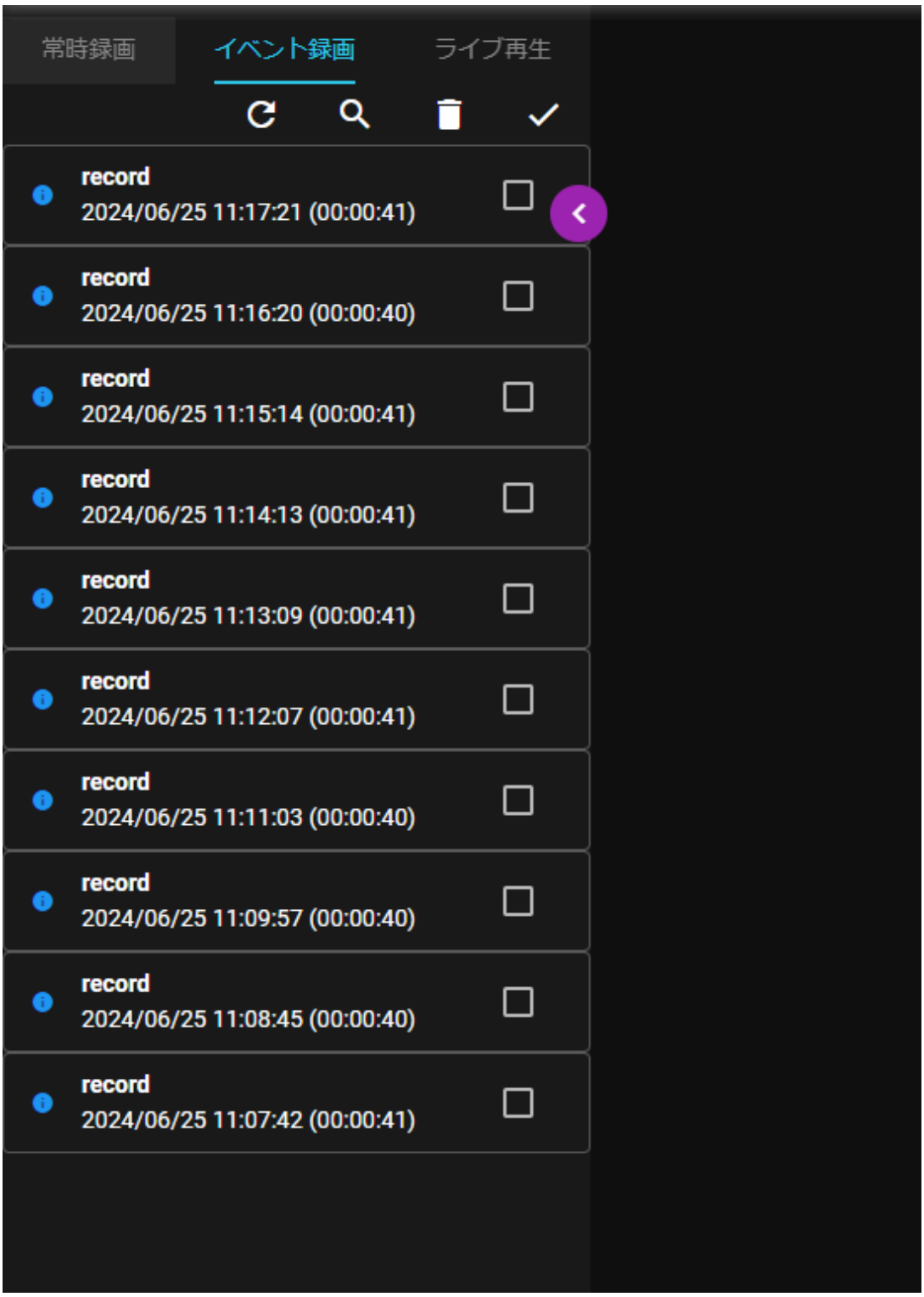
6. 表示件数が多い場合は、複数のページに分けて表示されます。1ページ当たりの表示件数を10、100、1000の中から選択することが可能です。



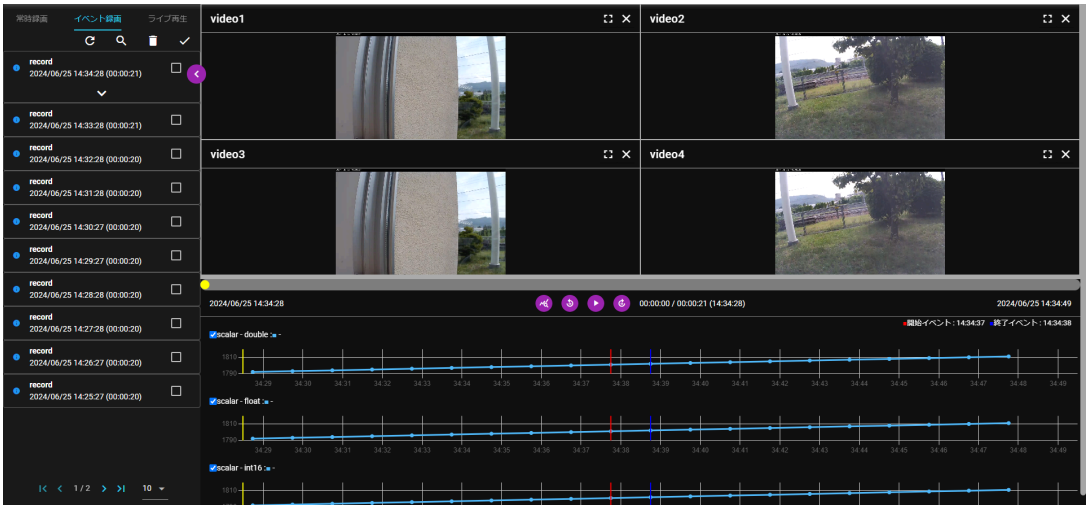
11.5.3.2 再生

次の手順で、イベント発生時のコンポーネントのデータのグラフと、カメラの映像を再生することができます。同時に再生できるカメラの台数は4台です。

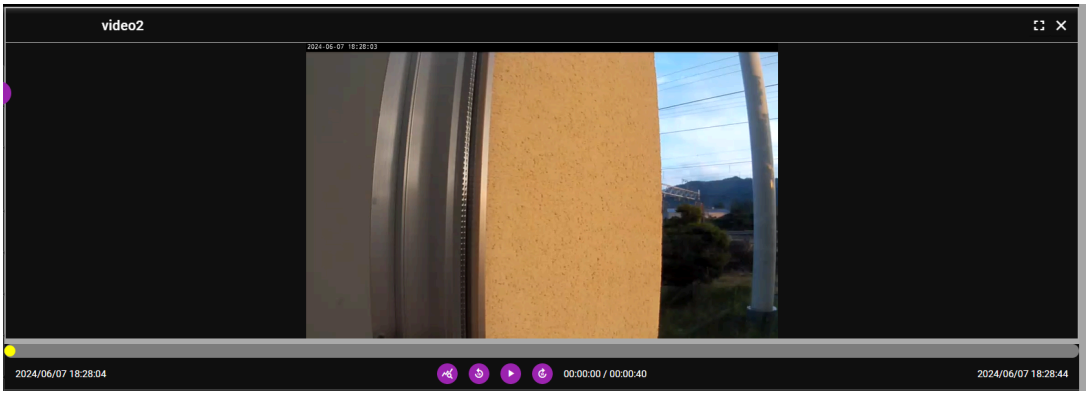
1. イベントを選択します。



2. 選択したイベントの動画とデータが、右側の画面に表示されます。



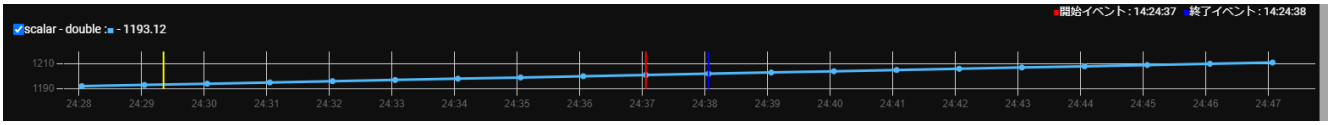
3. 再生ボタンまたは動画を選択すると、イベントに紐づく動画とグラフが再生されます。画面項目は次の通りです。
   - 動画部分

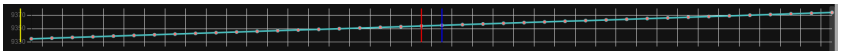


番号	説明
①	VIDEOコレクタの名称を表示します。
②	アイコンを押下した動画を全画面で表示します。
③	アイコンを押下した動画を非表示とします。
④	イベント発生時の、イベントに登録されたカメラの映像を表示します。押下することで、イベントの再生・停止を切り替えることができます。
⑤	動画の現在の再生位置を表示します。現在位置の点をドラッグアンドドロップすることで、再生位置を変更することも可能です。 また、動画との間にある領域をドラッグした状態で、上下に移動させることで、動画の表示領域を調整することができます。
⑥	イベントの記録開始日時を表示します。

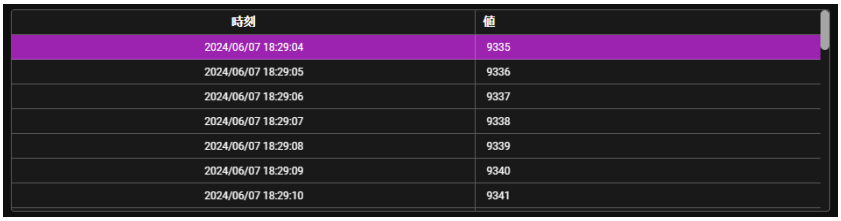
番号	説明
⑦	押下すると、グラフの表示状態を次の中から切り替えます。 ・全体グラフ...グラフのすべてのデータを表示します。 ・詳細グラフ...現在の再生位置の30秒前後のデータを表示します。
⑧	押下すると、再生位置を10秒前にスキップします。
⑨	押下すると、動画とグラフの再生を開始します。
⑩	押下すると、再生位置を10秒後にスキップします。
⑪	イベントの記録時間と、再生経過時間を表示します。
⑫	イベントの記録終了日時を表示します。

- グラフ部分



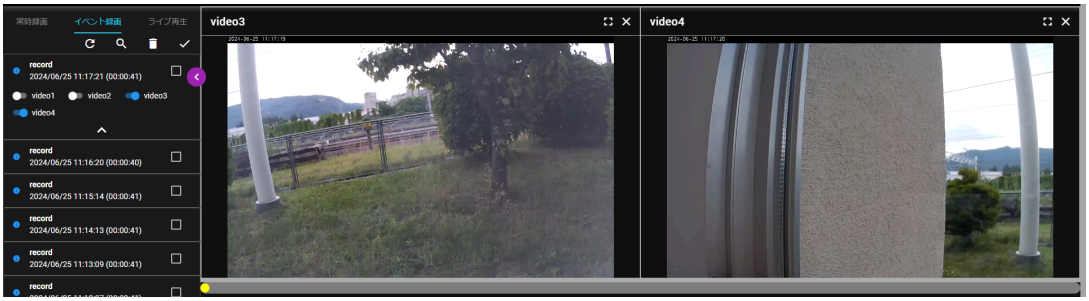
項目	説明
①	チェックを入れたグラフを表示し、チェックを外したグラフを非表示とします。
②	データの送信元であるコンポーネントの名前とデータ名を表示します。
③	データの線の色と、現在の再生位置の値を表示します。
④	開始イベント、終了イベントの時間と線の色を表示します。
⑤(数値データの場合)	イベント発生時の、イベントに登録されたコンポーネントのデータが、折れ線グラフで表示されます。X軸が時間を、Y軸が値を表します。また、以下の線が表示されます。 ・黄色い線...動画の再生と連動して移動し、現在の再生位置を示します。 ・赤い線...開始イベントの時間を示します。 ・青い線...終了イベントの時間を示します。終了イベントが発生した場合のみ表示されます。 
⑤(文字列データの場合)	イベント発生時の、イベントに登録されたコンポーネントのデータが、時刻・値の2列からなるテーブルで表示されます。また、以下の表示がなされます。

項目	説明
	・紫色の行...動画の再生と連動して移動し、現在の再生位置を示します。 ・赤い印...時間の横に表示され、開始イベントを示します。 ・青い印...時間の横に表示され、終了イベントを示します。終了イベントが発生した場合のみ表示されます。



時刻	値
2024/06/07 18:29:04	9335
2024/06/07 18:29:05	9336
2024/06/07 18:29:06	9337
2024/06/07 18:29:07	9338
2024/06/07 18:29:08	9339
2024/06/07 18:29:09	9340
2024/06/07 18:29:10	9341

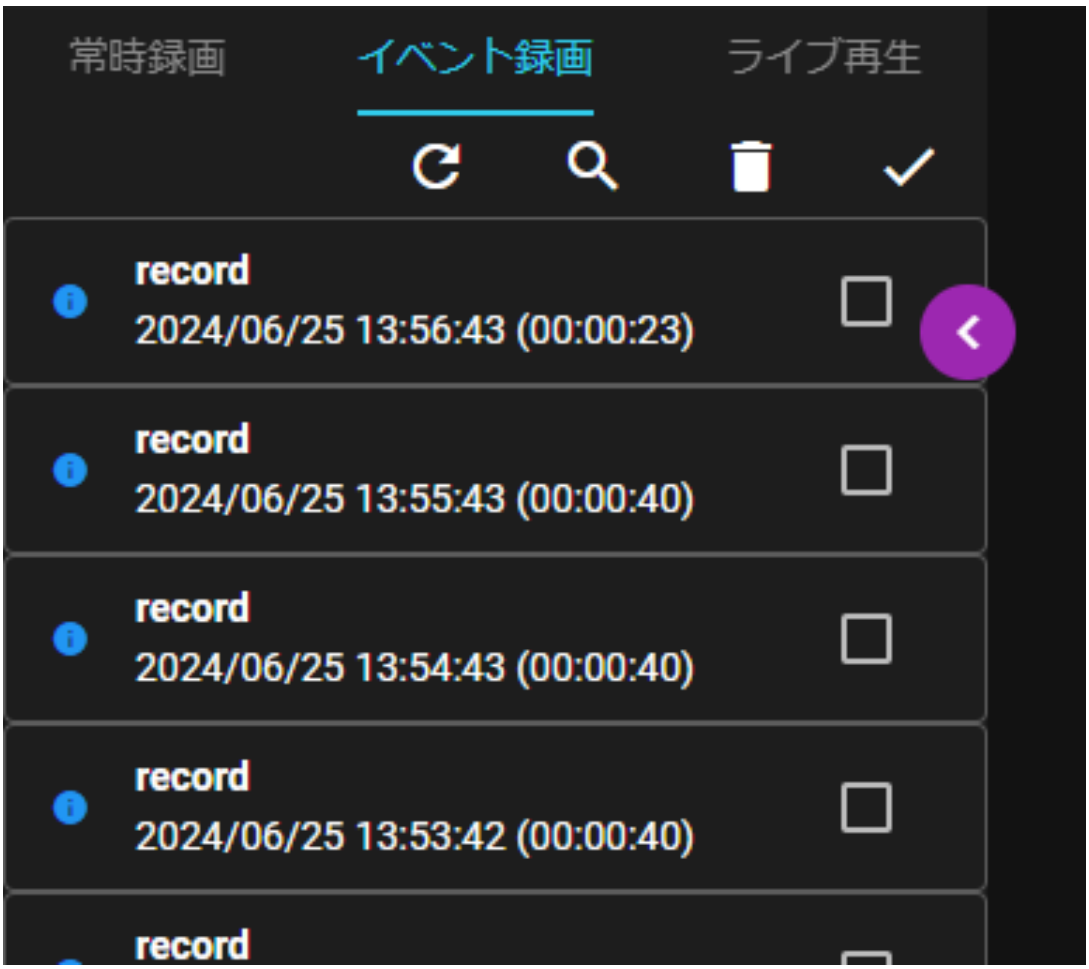
4. 一覧内の選択したイベントに表示されている、カメラ名の横のトグルスイッチを切り替えることで、動画の表示・非表示を切り替えることができます。



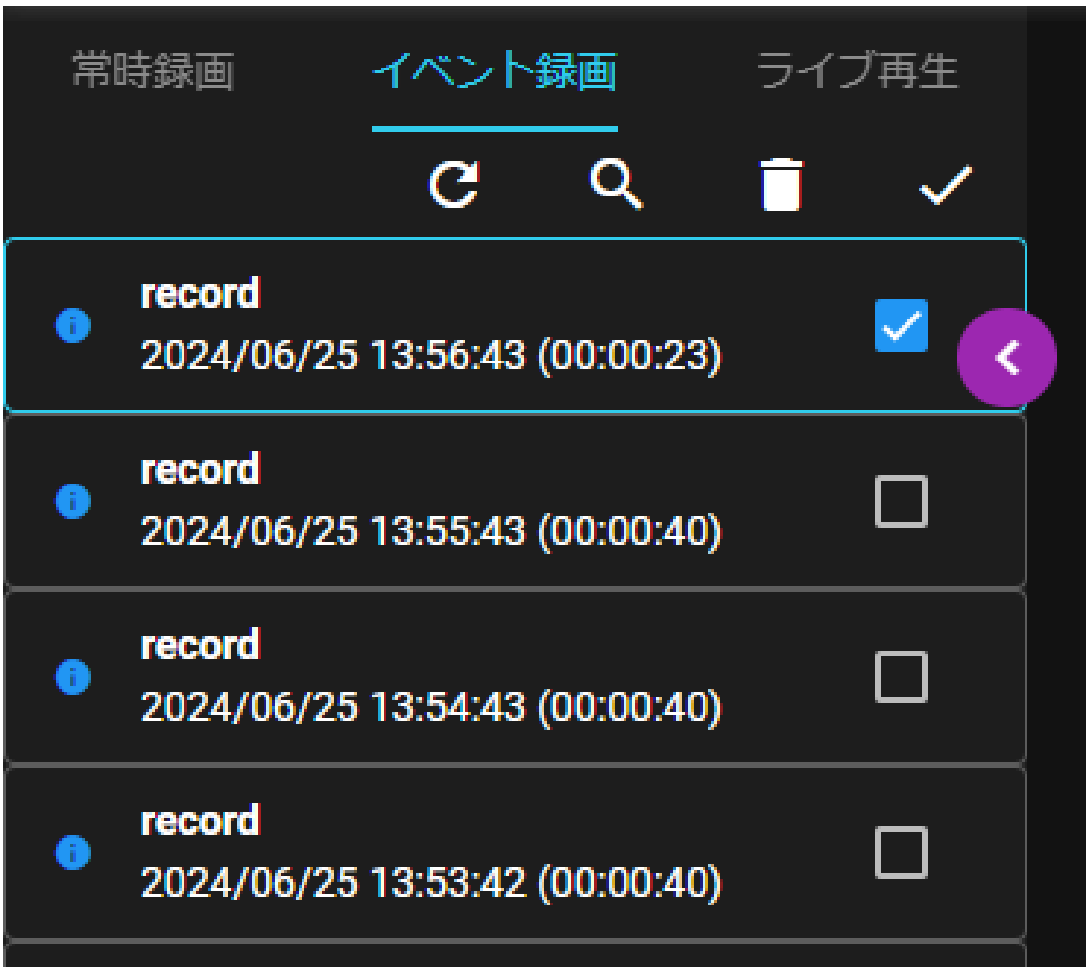
11.5.3.3 削除

次の手順で、イベントを削除することができます。

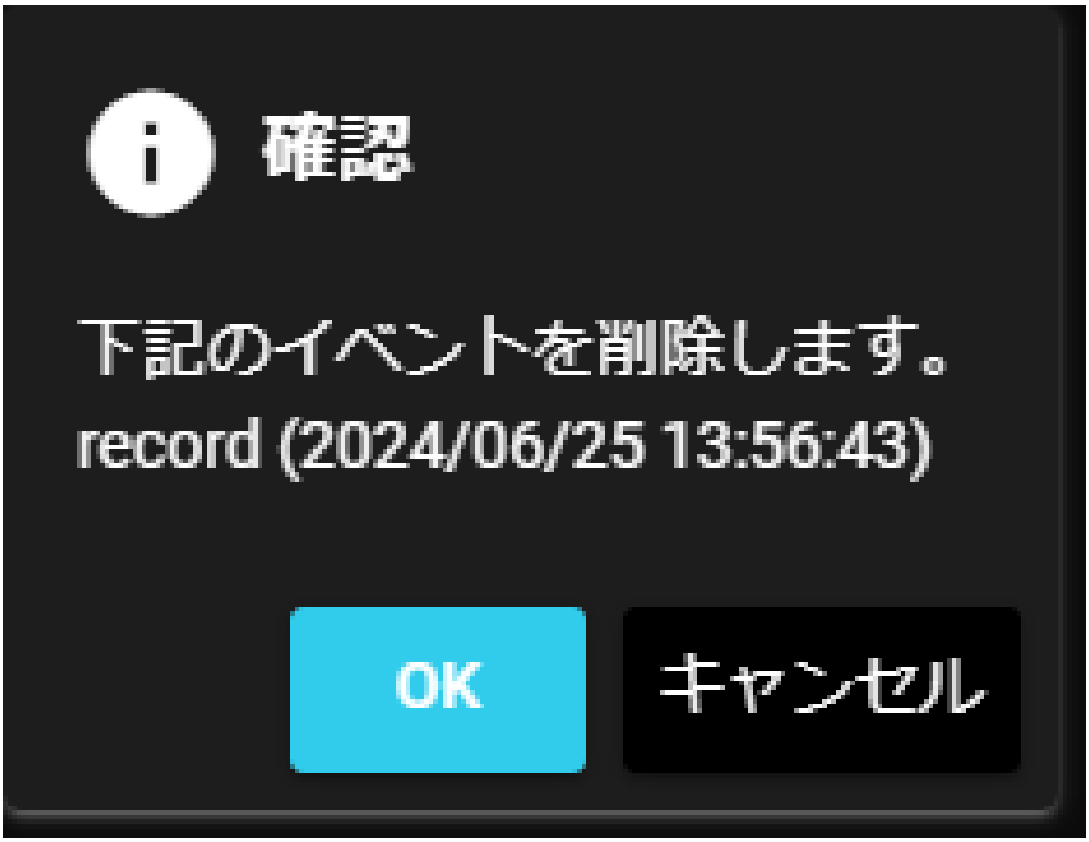
1. イベントのチェックボックスにチェックを入れます。ヘッダのチェックアイコンを押下すると、リストに表示されているすべてのイベントにチェックを入れることができます。



2. ヘッダの削除アイコンを押下します。


常時録画
イベント録画
ライブ再生
record
2024/06/25 13:56:43 (00:00:23)
record
2024/06/25 13:55:43 (00:00:40)
record
2024/06/25 13:54:43 (00:00:40)
record
2024/06/25 13:53:42 (00:00:40)

3. 「OK」を選択します。


確認
下記のイベントを削除します。
record (2024/06/25 13:56:43)
OK
キャンセル

4. チェックを入れたイベントが削除されます。

11.5.3.4 URL指定によるイベント表示

SpeeDBee SynapseのEvent Recorderで生成されたイベントには、録画開始日時 (YYYY/MM/DD hh:mm:ss) を含む下記のURLを指定することで直接アクセスできます。同じ時刻に複数のイベントがある場合は、URLの末尾にIDが付番されます。

- `http://{ホスト名orデバイスのIPアドレス}:8120/video-viewer/event_record/{YYYY-MM-DDThhmmss}`

  例1）

  - `http://192.168.1.3:8120/video-viewer/event_record/2025-03-28T165209`

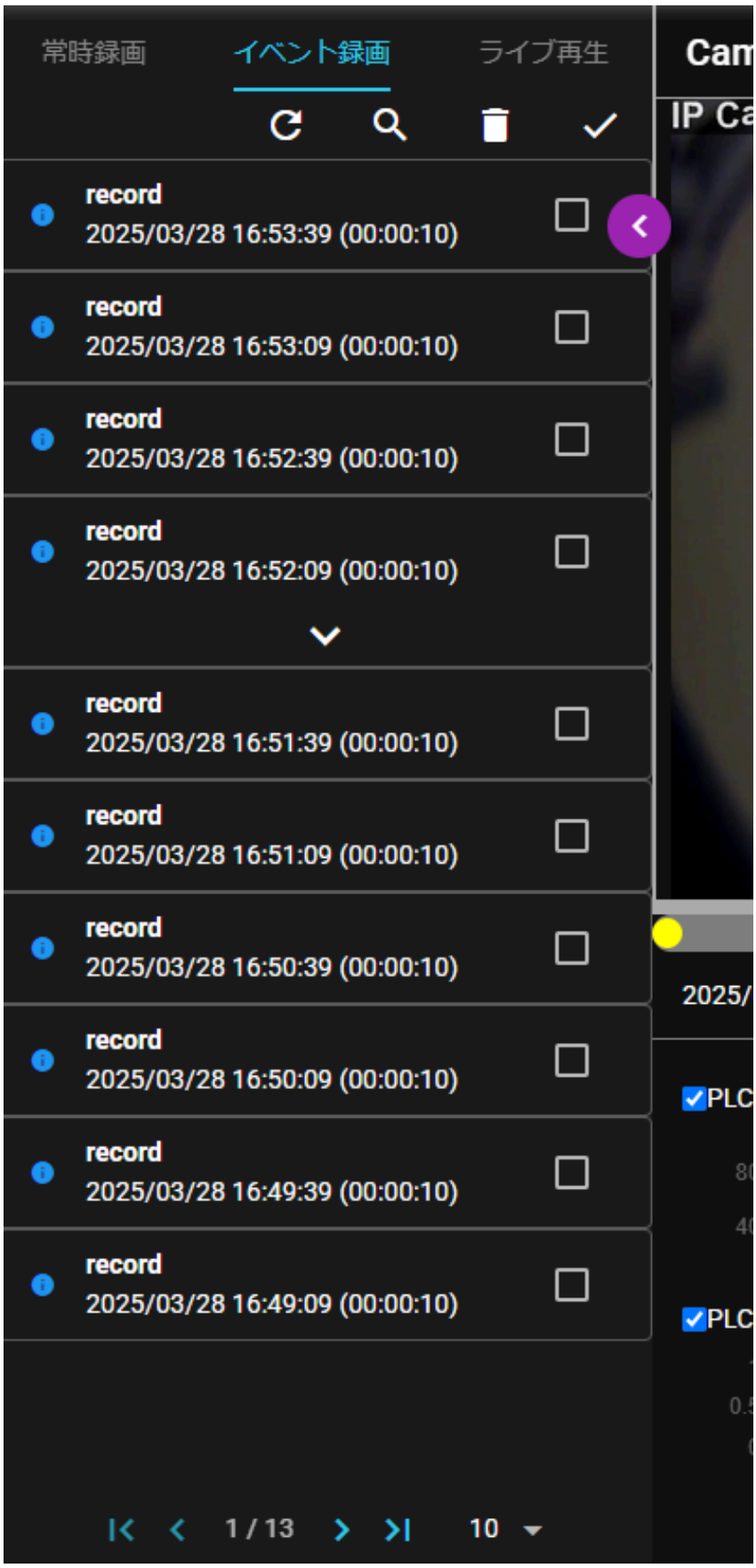
  例2)※時刻が重複するイベントがある場合

  - `http://192.168.1.3:8120/video-viewer/event_record/2025-03-28T165210/1`
  - `http://192.168.1.3:8120/video-viewer/event_record/2025-03-28T165210/2`

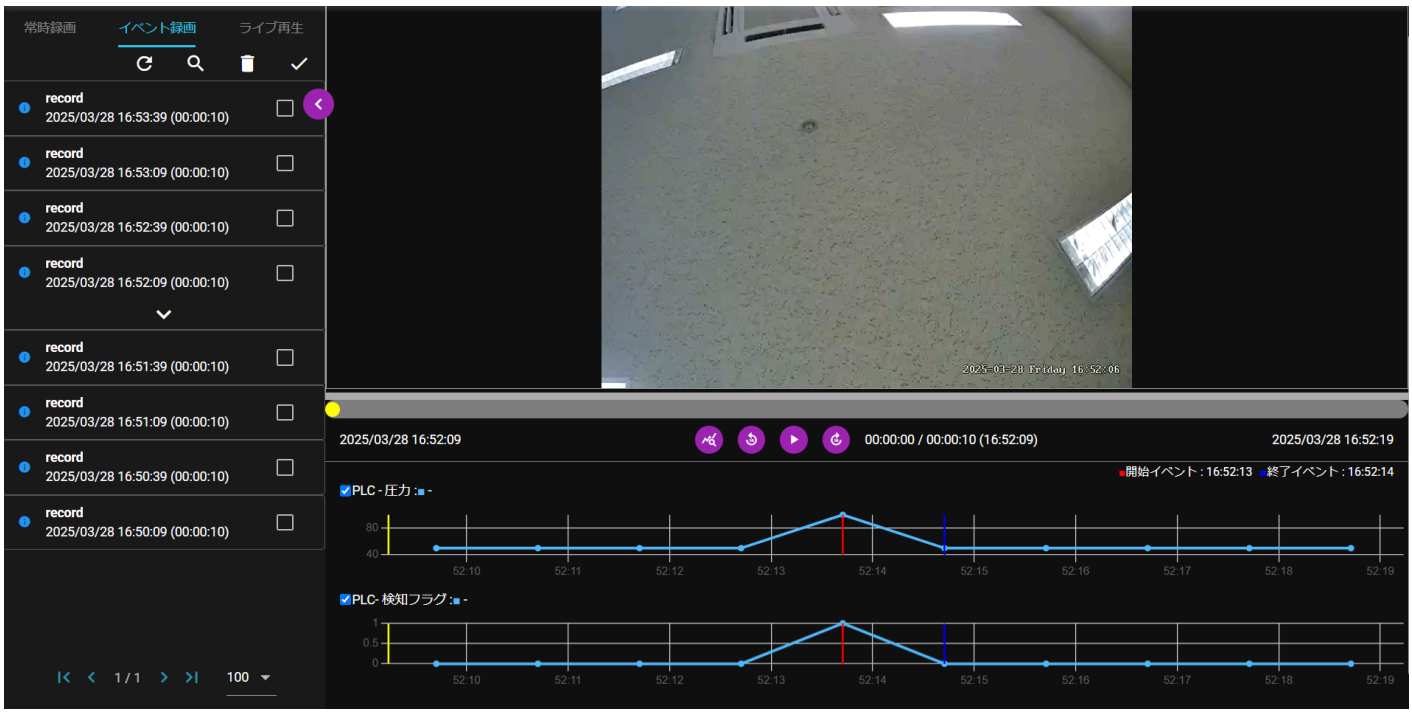
URLはイベントの録画開始日時のフォーマットを「YYYY-MM-DDThhmmss」に変換して生成しています。



各イベントを押下することでURLも動的に更新されます。

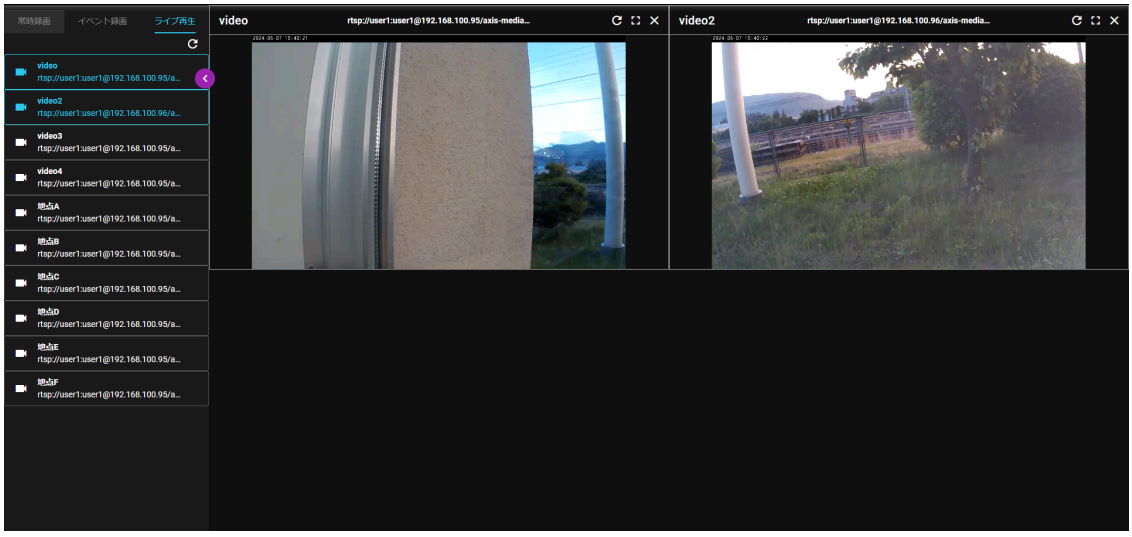


URLを指定してイベントにアクセスすると、対象のイベントの録画開始時刻の前後2分以内に発生したイベントが一覧に表示されます。アクセス時、イベント一覧の1ページ当たりの最大表示件数は100件となり、対象のイベントが選択された状態になります。



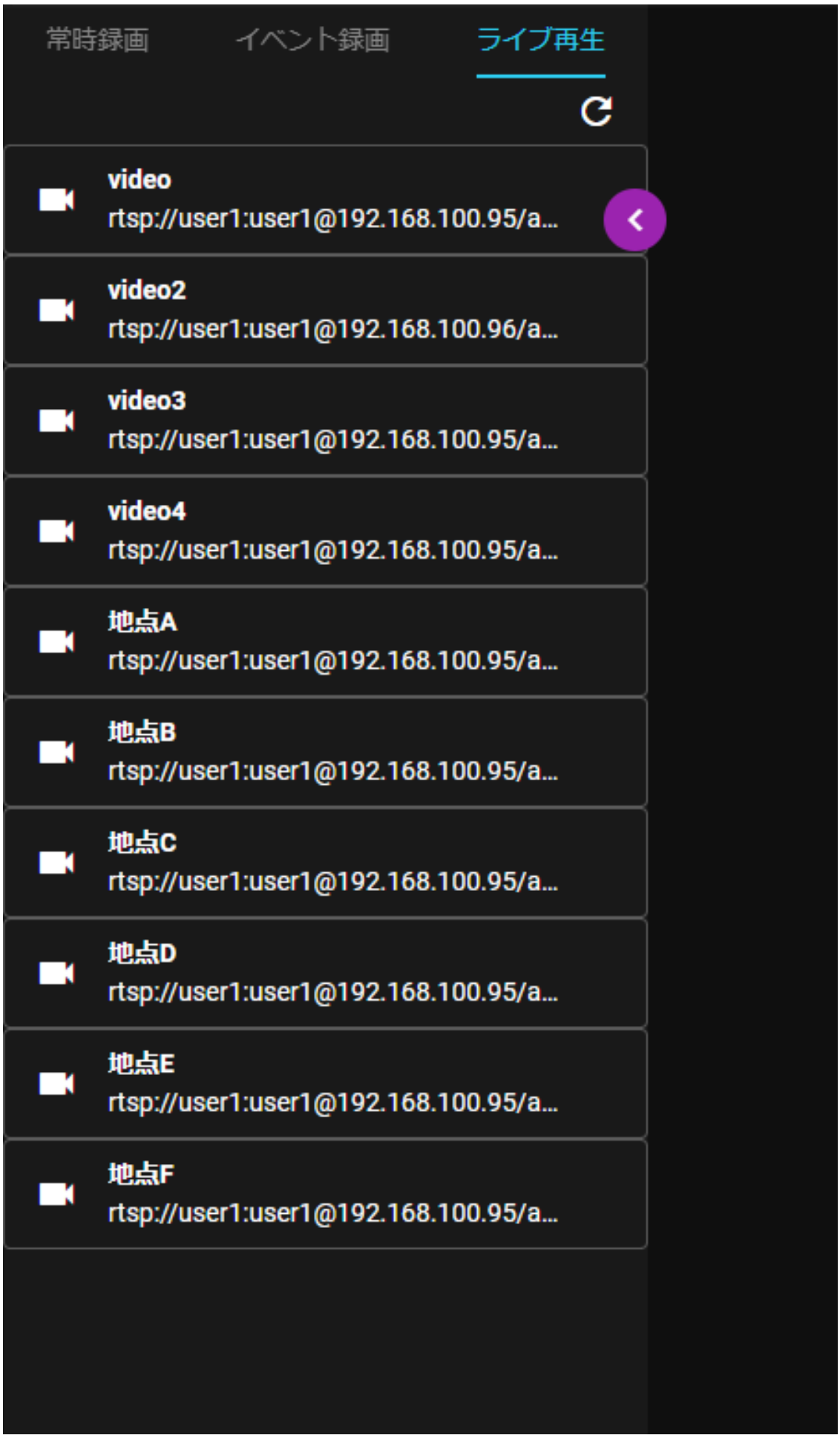
11.5.4 ライブ再生

SpeeDBee Synapseで稼働しているVIDEOコレクタの映像（現在のカメラ映像）を再生することができます。



11.5.4.1 一覧

画面の左側に、VIDEOコレクタの一覧が表示されます。

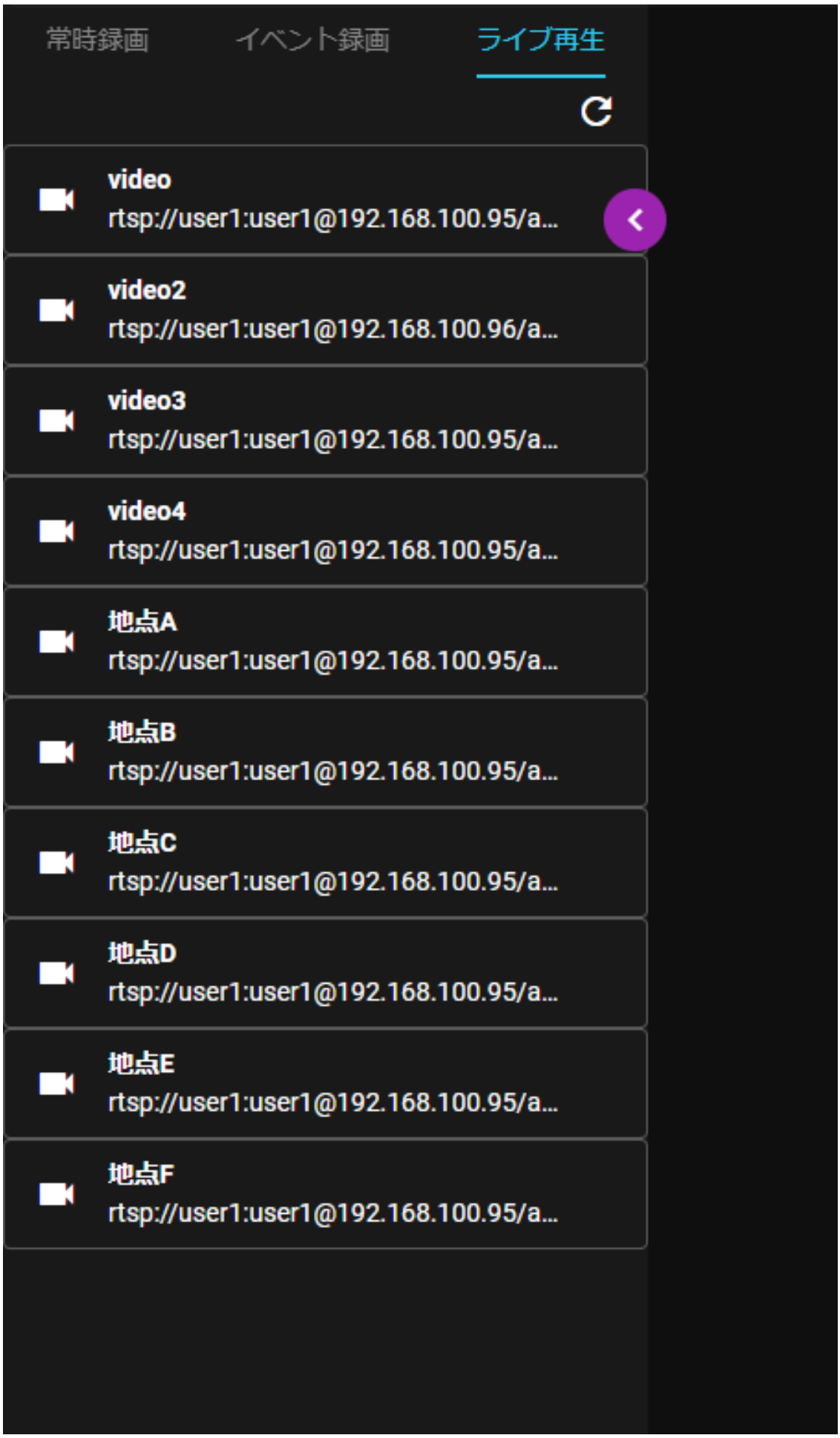

常時録画
イベント録画
ライブ再生
video
rtsp://user1:user1@192.168.100.95/a...
video2
rtsp://user1:user1@192.168.100.96/a...
video3
rtsp://user1:user1@192.168.100.95/a...
video4
rtsp://user1:user1@192.168.100.95/a...
地点A
rtsp://user1:user1@192.168.100.95/a...
地点B
rtsp://user1:user1@192.168.100.95/a...
地点C
rtsp://user1:user1@192.168.100.95/a...
地点D
rtsp://user1:user1@192.168.100.95/a...
地点E
rtsp://user1:user1@192.168.100.95/a...
地点F
rtsp://user1:user1@192.168.100.95/a...

[redacted]

表示される項目は次の通りです。

- VIDEOコレクタの名称
- VIDEOコレクタに登録されている、カメラのRTSP URL

また、リフレッシュアイコンを押下すると、VIDEOコレクタの最新の状態に合わせて、一覧の表示内容が更新されます。（登録、更新、削除されたものが適用）


常時録画
イベント録画
ライブ再生
video
rtsp://user1:user1@192.168.100.95/a...
video2
rtsp://user1:user1@192.168.100.96/a...
video3
rtsp://user1:user1@192.168.100.95/a...
video4
rtsp://user1:user1@192.168.100.95/a...
地点A
rtsp://user1:user1@192.168.100.95/a...
地点B
rtsp://user1:user1@192.168.100.95/a...
地点C
rtsp://user1:user1@192.168.100.95/a...
地点D
rtsp://user1:user1@192.168.100.95/a...
地点E
rtsp://user1:user1@192.168.100.95/a...
地点F
rtsp://user1:user1@192.168.100.95/a...

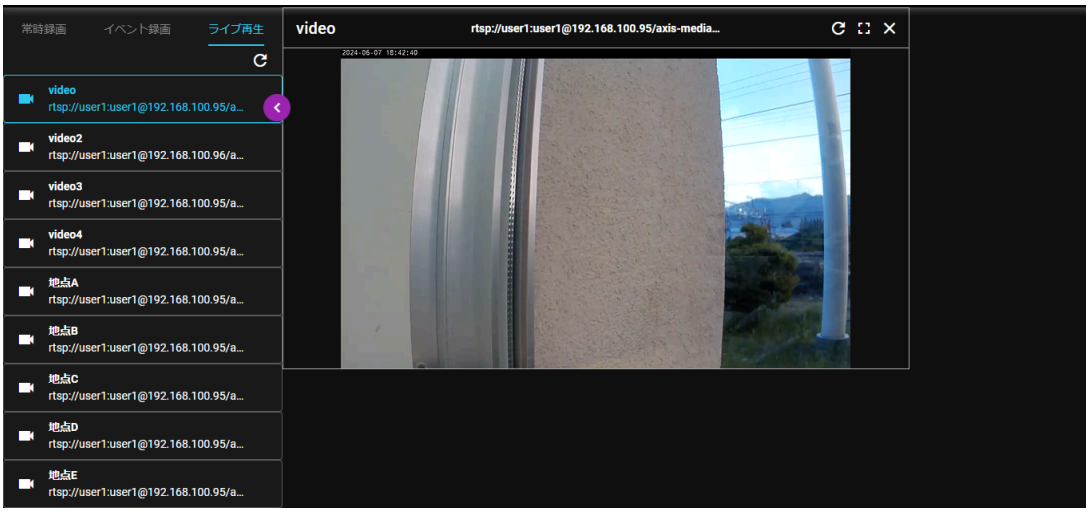
11.5.4.2 再生

次の手順で、カメラの現在の映像を再生することができます。同時に10台のカメラを再生することができます。

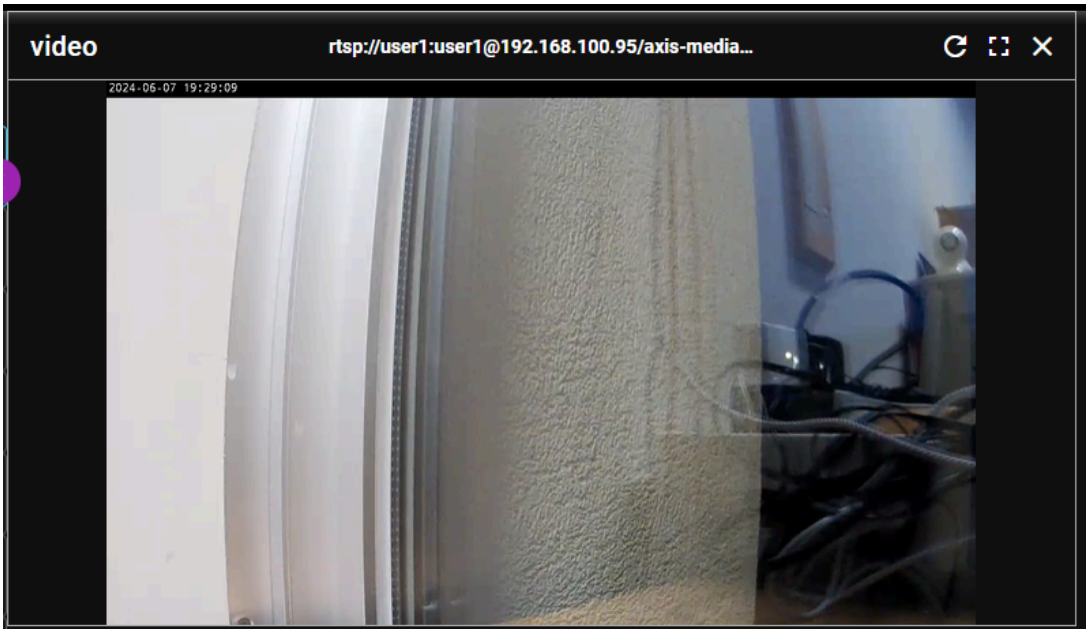
1. 一覧のカメラをドラッグし、右側にドロップします。



2. カメラの現在の映像が再生されます。



画面項目は次の通りです。



番号	説明
①	VIDEOコレクタの名称を表示します。
②	VIDEOコレクタに登録されている、カメラのRTSP URLを表示します。
③	アイコンを押下したカメラに再接続します。
④	アイコンを押下したカメラの映像を全画面で表示します。
⑤	アイコンを押下したカメラを非表示とします。
⑥	カメラのライブ映像を再生します。

ストレージの空き容量がない場合

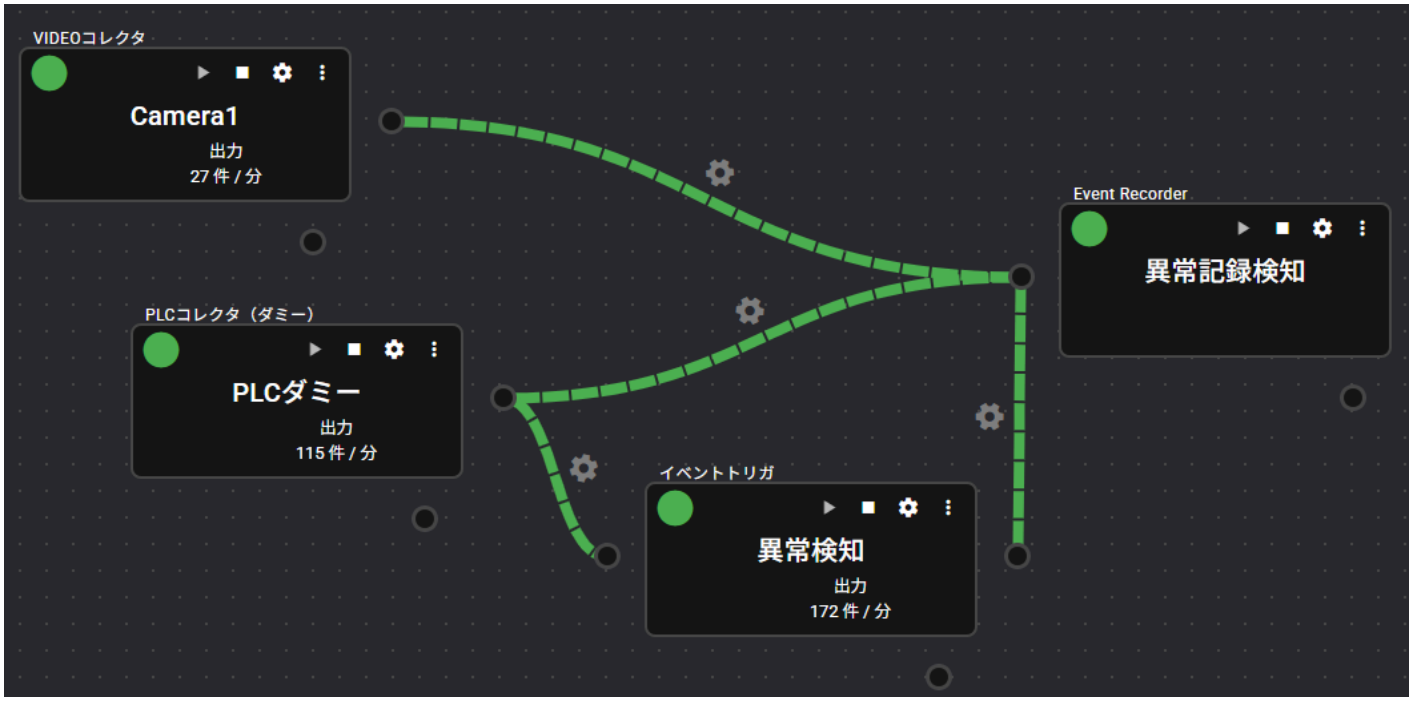
i-Triggerが稼働しているマシンのストレージに空き容量がない場合、ライブ再生はできません。

11.6 関連コンポーネント

本節では、i-Triggerで使用するコンポーネントについて説明します。

11.6.1 VIDEOコレクタ

RTSPで配信されているネットワークカメラの映像を受信、録画ファイルを保存するコンポーネントです。
録画ファイルは指定ディレクトリ内にtsファイルとして保存されるほか、オプションでmp4形式での出力も可能です。
保存された動画ファイルは一定期間保持されており、その間はビューワ画面で再生することができます。
このコンポーネントは、Proxy Serverを経由した通信には対応していません。



11.6.1.1 基本情報

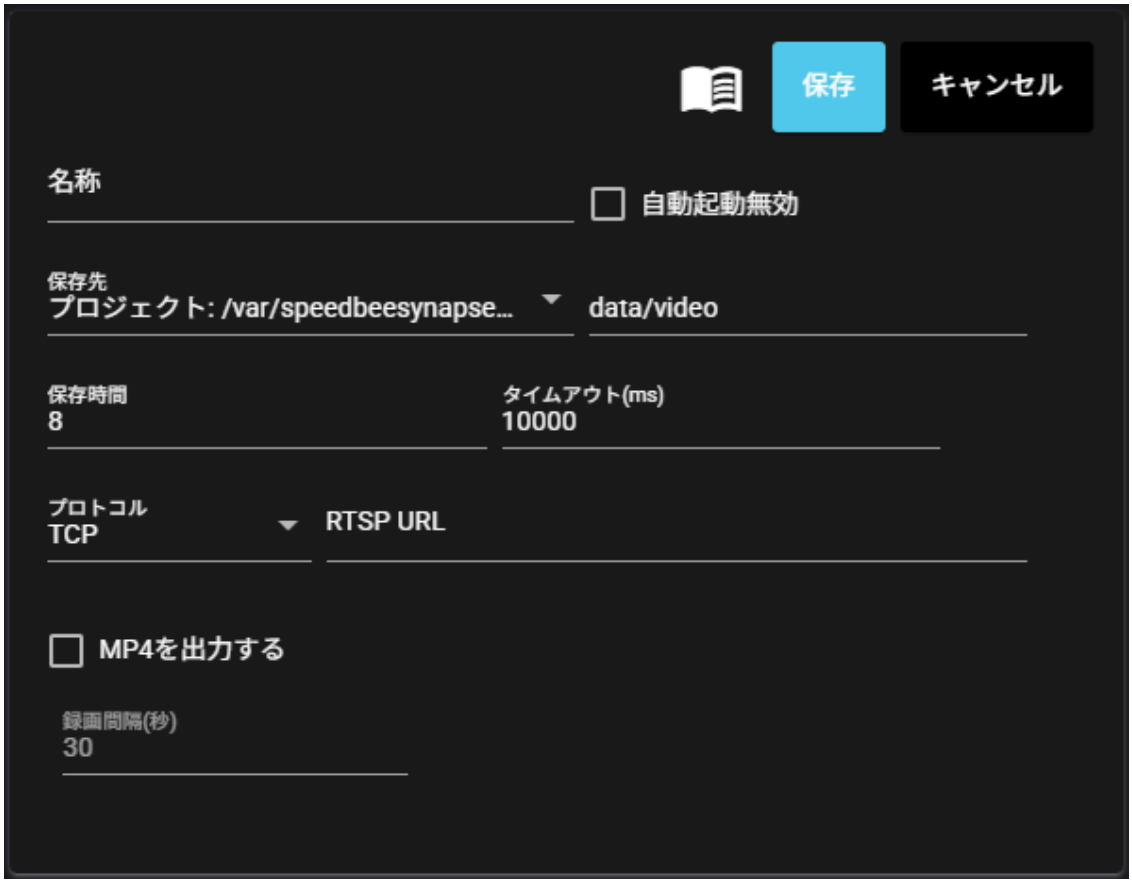
項目	説明
ベース名	VIDEOコレクタ
説明	ネットワークカメラの映像を受信して、指定ディレクトリへTSファイル（MPEG2-TS）を保存します。
消費ポイント	2

項目	説明
ポート	入力：0 出力：1 受信した映像を録画データとしてTSファイルへ出力します。このファイルをビューワで再生できます。 出力するデータの詳細は、「出力データ項目」をご覧ください。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。 出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

11.6.1.2 設定項目

VIDEOコレクタの設定項目は、次の通りです。

- プロトコル「TCP」選択時



- プロトコル「UDP」選択時



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合
保存先	動画ファイルを保存するディレクトリ。ベースディレクトリと、そこからの相対パスを指定します。ベースディレクトリは以下から選択可能です。 • プロジェクトパス（プロジェクト配下へ保存されます。プロジェクトを複製などした場合、新しいプロジェクトの配下が保存先となります。） ※一部の環境では、この設定を変更/編集できません。 • 任意のパス（通常はベースディレクトリに対する相対パスを指定しますが、任意の絶対パスを直接指定します。プロジェクトを複製などしても指定したパスが保存先となります。）
保存時間	動画ファイルを保持する期間。時間単位で指定 ※保存時間を過ぎると動画は自動削除されます

項目	説明
タイムアウト(ミリ秒)	通信がタイムアウトするまでの時間をミリ秒単位で入力
プロトコル	トランスポートに使用するプロトコル • TCP • UDP ※一部の環境では、この設定をUDPへ変更/編集できません。
RTSP URL	ネットワークカメラのライブストリームURL
MP4を出力する	有効にすると、一定間隔でMP4の動画ファイルを生成します。 ※無効の場合は、TSファイルのみが出力されます
録画間隔	「MP4を出力する」が有効な場合に、 MP4の1ファイルあたりの録画時間を秒単位で指定 ※動画の長さは指定した設定と完全に一致しないことがあります
ポート範囲を指定する	「UDP」が選択されている場合に、データ受信用のUDPポートを範囲で指定

11.6.1.3 出力データ項目

VIDEOコレクタは、次のデータを出力します。

項目	カラム名	データ型	説明
TSファイル	$ss_movie_file	FILE	TSファイル（MPEG2-TS）を出力するカラム。メディアタイプは"video/MP2T"固定
コンテナファイル	container_file	FILE	「MP4を出力する」が有効な場合に、MP4ファイルを出力するカラム。メディアタイプは"video/mp4"固定

11.6.1.4 テスト接続

次の手順で、設定したカメラに対してテスト接続を行うことができます。

1. 登録したVIDEOコレクタの設定画面を開きます。
2. 「テスト」を押下します。



3. 設定したカメラへのテスト接続が行われ、正常に接続できる場合に、カメラの映像がライブ再生されます。



カメラのテスト接続に関する注意点

- カメラへ接続できない場合

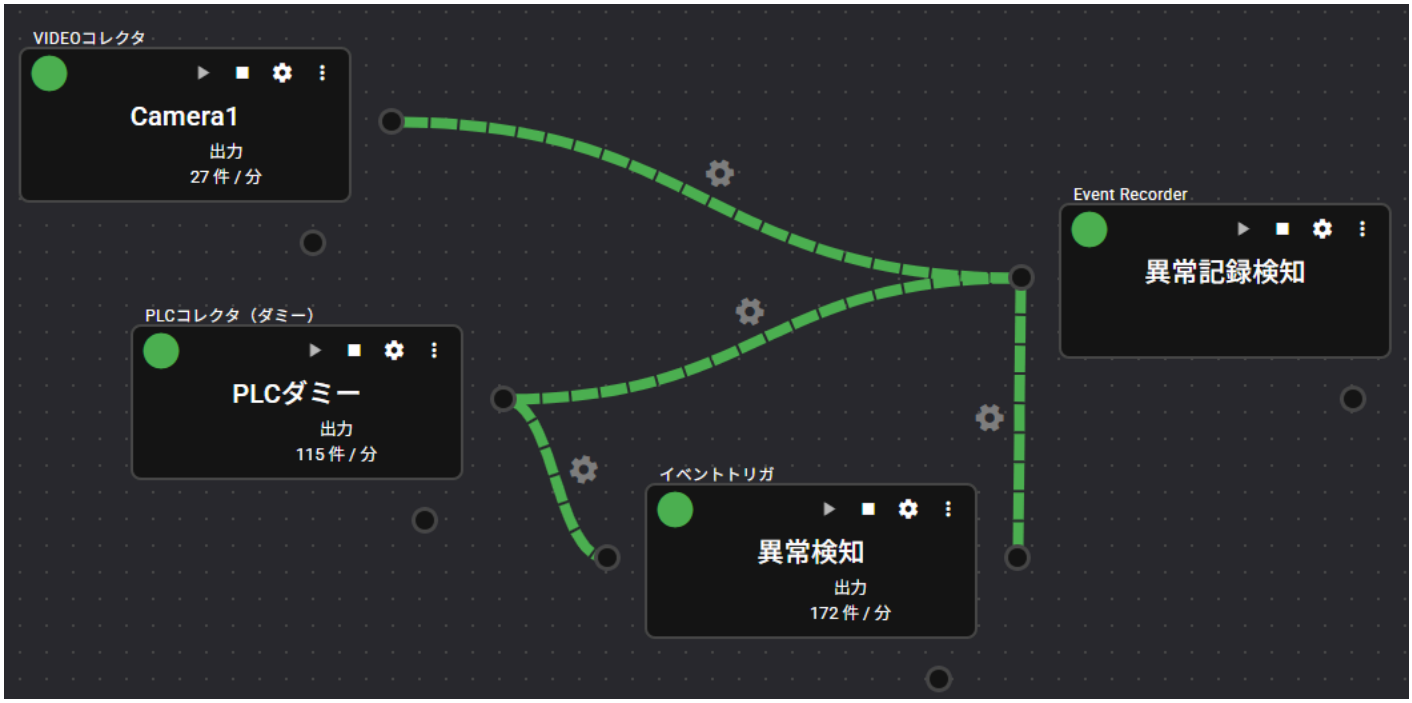
RTSP URLなどが正しいか確認してください。また、ご利用のネットワークカメラが使用できるカメラであるか確認してください。対応しているカメラでも、カメラ側の設定が対応しているビデオコーデックとなっていない場合は、カメラ側の設定を変更する必要があります。
参考：使用できるカメラについて
また、タイムアウトの設定を伸ばすと、解消することがあります。

- 同時接続数について

カメラによって、同一のストリーミングに対して、同時に接続できる接続数は異なります。低スペックのカメラでは、1接続のみしか同時再生できない場合もあるため、カメラの取扱説明書iやカメラの製品ホームページ等をご確認ください。

11.6.2 Event Recorder

イベント区間のデータを集めて、ビューワ画面のイベント録画機能で参照可能なJSON形式で出力するコンポーネントです。
VIDEOコレクタが入力ポートに接続されている場合は、区間内の動画ファイルも収集されます。
イベント区間の判定はイベントトリガコンポーネントで行うため、入力ポートにイベントトリガを接続する必要があります。



11.6.2.1 基本情報

項目	説明
ベース名	Event Recorder
説明	イベント検出後のデータを集めて、収集したデータをJSONで出力します。同区間に、動画ファイルがあれば同じく取り纏めします。指定ディレクトリへ収集したデータを保存することで、ビューワで再生できます。
消費ポイント	2
ポート	入力：1 他のコンポーネントからデータを受け取ります。 出力：1 記録対象となる区間のデータをJSONファイルへ出力します。 出力するデータの詳細は、「出力データ項目」をご覧ください。 システム：1 コンポーネントの稼働状況やエラー情報を出力します。 出力するエラー情報の詳細は、「エラーコード一覧」をご覧ください。

11.6.2.2 設定項目

Event Recorderの設定項目は、次の通りです。



項目	説明
名称	コンポーネントの名前を入力 ※他のコンポーネント名と重複する事はできません。
自動起動無効	コンポーネントの自動起動を無効にする場合
分類	イベントに対して設定できる分類。イベントの内容に合わせて、任意で指定します。 • インフォメーション • 警告 • エラー
保存時間	記録したデータおよび動画を保持する期間。時間単位で指定 ※保存時間を過ぎると動画はイベントデータマネージャによって削除されます
保存先	イベント検知した区間のデータおよび動画を保存するディレクトリ。ベースディレクトリと、そこからの相対パスを指定します。ベースディレクトリは以下から選択可能です。 • プロジェクトパス（プロジェクト配下へ保存されます。プロジェクトを複製などした場合、新しいプロジェクトの配下が保存先となります。） ※一部の環境では、この設定を変更/編集できません。 • 任意のパス（通常はベースディレクトリに対する相対パスを指定しますが、任意の絶対パスを直接指定します。プロジェクトを複製などしても指定したパスが保存先となります。）

11.6.2.3 出力データ項目

Event Recorderは、次のデータを出力します。

項目	カラム名	データ型	説明
イベントファイル	event_file	FILE	イベント記録のJSONファイルを出力するカラム。メディアタイプは"application/json"固定

11.6.3 イベントトリガ

SpeeDBee Synapse本体に含まれるコンポーネントです。詳しくは、「イベントトリガ」を参照してください。

11.6.4 イベントデータマネージャー

SpeeDBee Synapse本体に含まれるコンポーネントです。詳しくは、「イベントデータマネージャー」を参照してください。

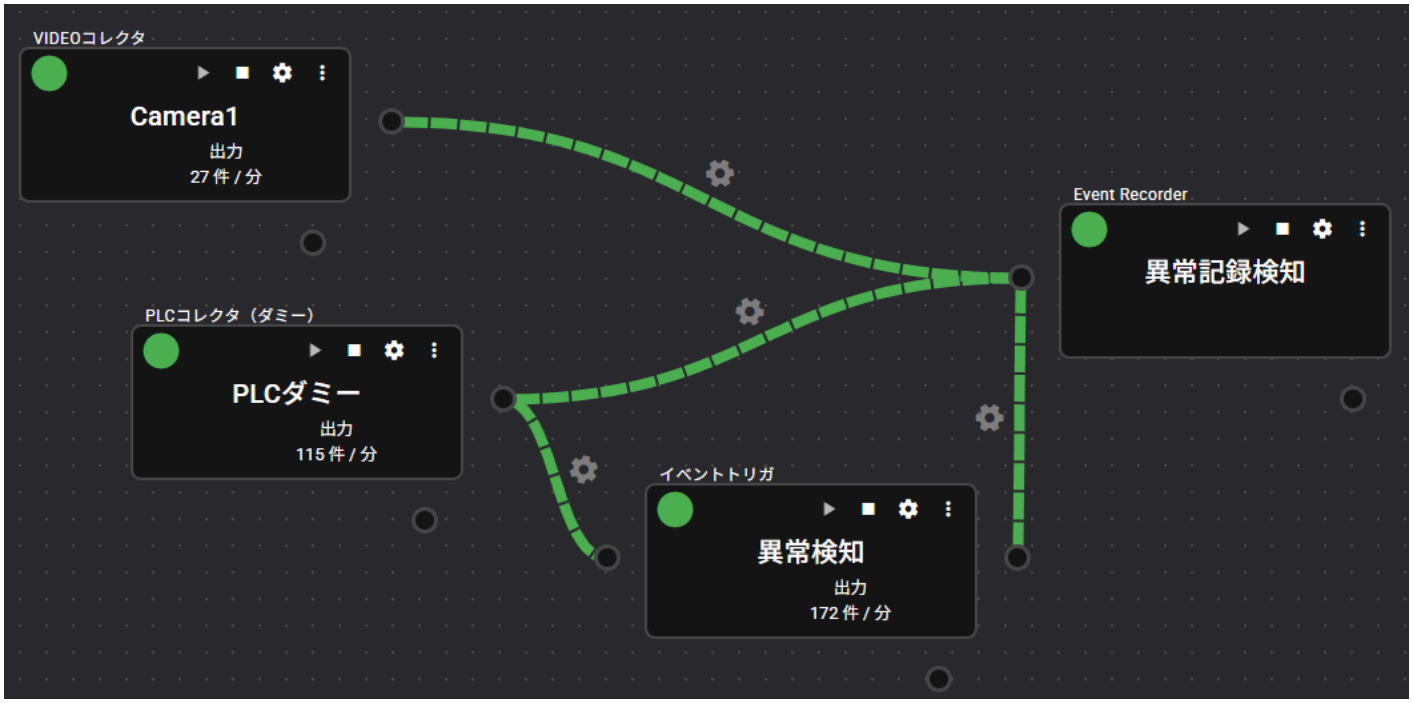
11.7 チュートリアル

本節では、i-Triggerの簡単な利用手順について、チュートリアル形式で解説します。

11.7.1 想定するユースケース

本チュートリアルでは、次のようなユースケースで、異常検知を行います。

PLCからの収集データで異常を検知したら、その前後の動画および収集したデータをビューワで確認する（※ここでは、PLCコレクタに見立てたカスタムコンポーネントを使用します）



1. PLCから次のデータを収集する
   - 異常検知フラグ
   - 圧力
2. 1台のネットワークカメラで映像を記録する
3. 異常検知フラグがONになったらイベントの記録を開始
   - 異常検知フラグ検知前5秒、異常検知フラグOFFから後ろ5秒を記録する
   - ビューワで表示するデータは、PLCから取得した異常検知フラグ、圧力とその区間の映像

11.7.2 カスタムコンポーネントの設定（疑似PLCコレクタ）

最初に、異常検知および圧力のデータを収集するためのカスタムコンポーネントを設定します。

11.7.2.1 カスタムコンポーネントのコード

次のコードをコピーして、メモ帳などのテキストエディタへ貼り付け、PC内に「plc_dummy.py」というファイル名で保存してください。文字コードは「UTF-8」を指定します。

```
from speedbeesynapse.component.base import HiveComponentBase, HiveComponentInfo, DataType
import time
import json


@HiveComponentInfo(uuid='12318c73-fe45-fded-150a-b95c894f3692', name='PLCコレクタ（ダミー）',
inports=0, outports=1)
class HiveComponent(HiveComponentBase):
    def __init__(self):
        # called at creating component instance
        pass
    def __del__(self):
        # called at deleting component instance
        pass

    def premain(self, param):
        # called before main()
        # 疑似データを登録するカラムを作成する
        self.data = self.out_port1.Column('圧力', DataType.INT16)
        self.flag  = self.out_port1.Column('異常検知フラグ', DataType.BOOLEAN)

    def postmain(self, param):
        # called after main() finished
        pass

    def main(self, param):
        value = False
        count = 0
        while self.is_runnable():
            ts = self.get_timestamp()
            pressure = 50
            count += 1
            # count30回毎に異常データを登録する
            if (count % 30) == 0:
                value = True
                pressure = 100
            else:
                value = False
            # 疑似データを登録する
            self.flag.insert(value, ts)
            self.data.insert(pressure, ts)

            time.sleep(1.0)

    def notify_stop(self):
        # called before the component received stop request from system
        pass
```

上記のコードは、次の処理を行います。

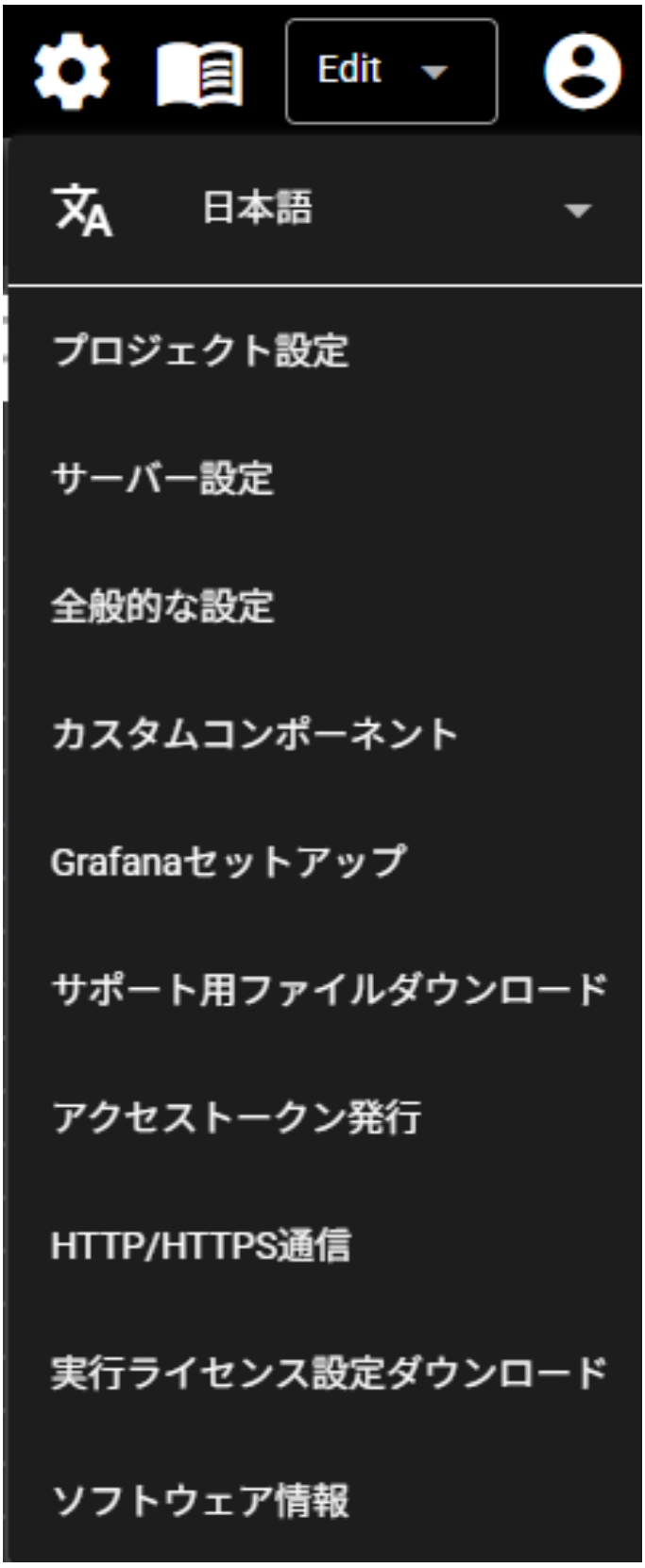
- 約30秒間隔で、異常検知のフラグをTrueにしてデータ登録

- 圧力は正常時は50で登録
- 異常時は圧力の値を100で登録

11.7.2.2 カスタムコンポーネントの登録

保存したPythonのソースコードをSpeeDBee Synapseで使うための設定を行います。

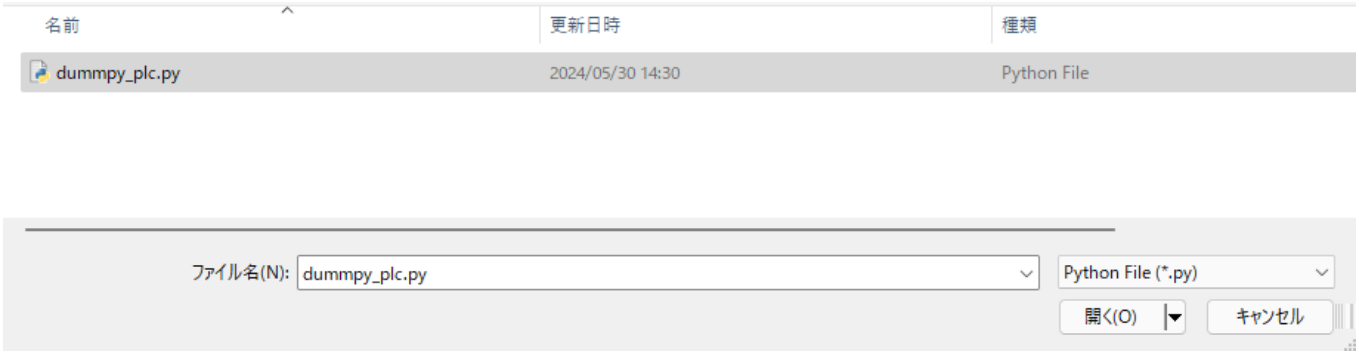
1. 「サーバー設定アイコン」の「カスタムコンポーネント」をクリックします。



2. カスタムコンポーネントの「追加」をクリックします。



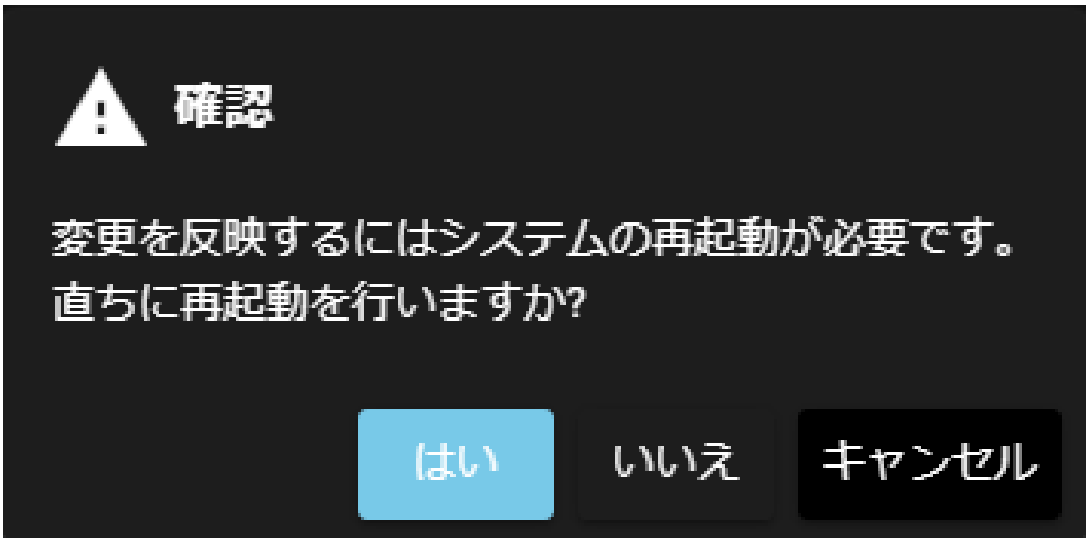
3. 先ほど保存した「dummy_plc.py」を選択します。



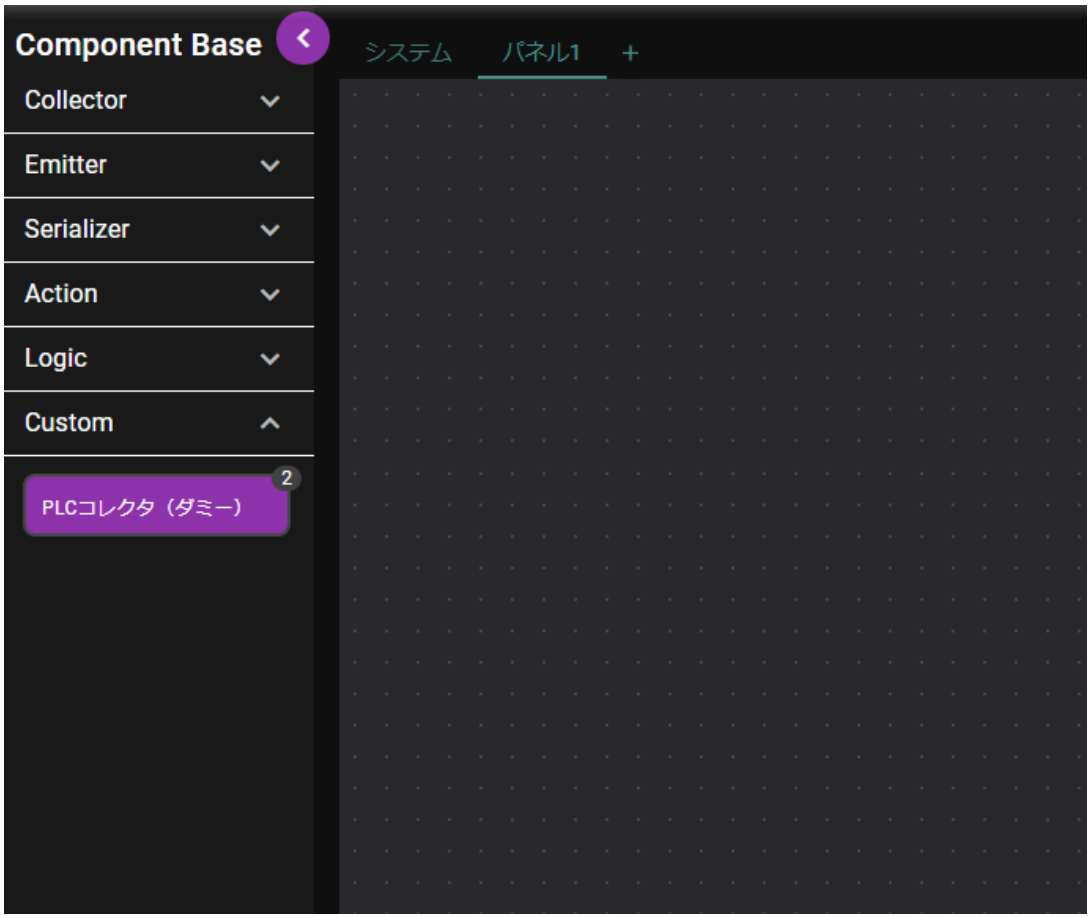
4. カスタムコンポーネントが表示されることを確認します。
「dummpy_plc.py」が表示されていたら、ダイアログの「閉じる」をクリックします。



5. 再起動を促されるので、「はい」をクリックします。



6. フローエディタの左部の「Custom」の「PLCコレクタ（ダミー）」をエディタ中央部へドラッグ＆ドロップします。



7. 名称を「PLCダミー」へ設定して、「保存」をクリックします。



8. カスタムコンポーネントの準備は完了です。



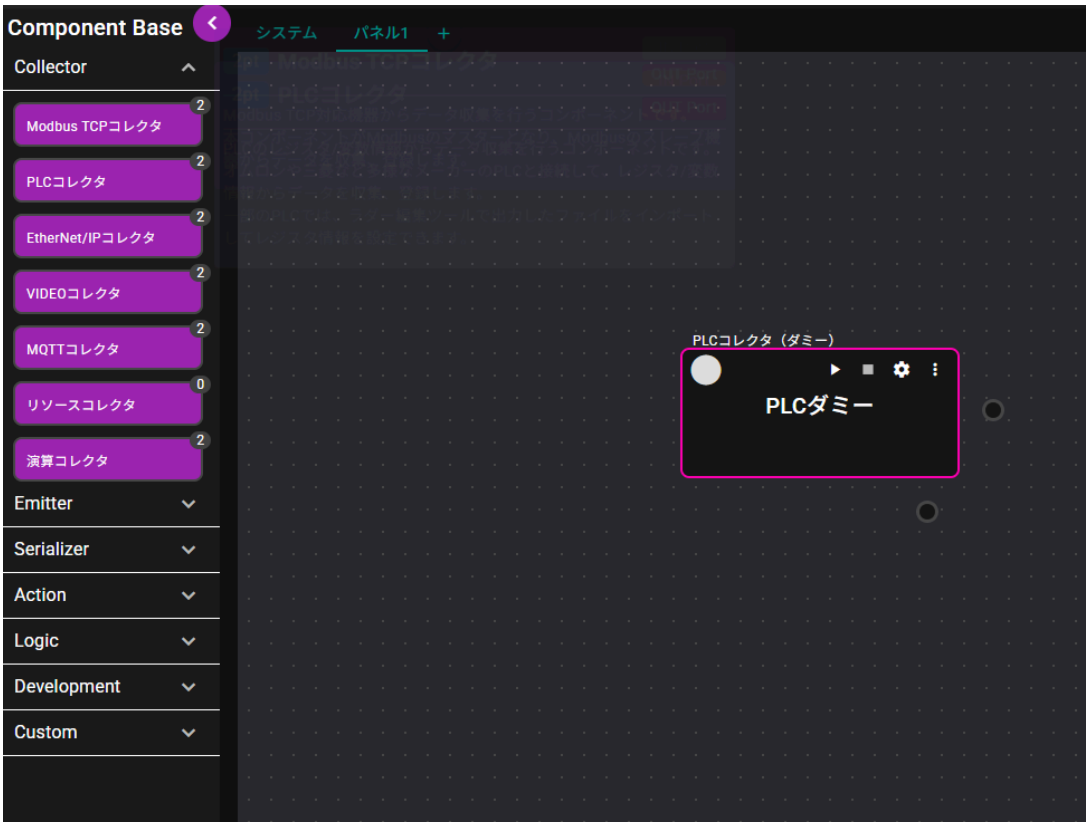
11.7.3 コンポーネントの配置・設定

i-Triggerを使うための各種コンポーネントを設定していきます。

11.7.3.1 VIDEOコレクタの設定

動画を記録するため、VIDEOコレクタを設定します。
利用できるネットワークカメラがない場合は、この設定は省略しても構いません。

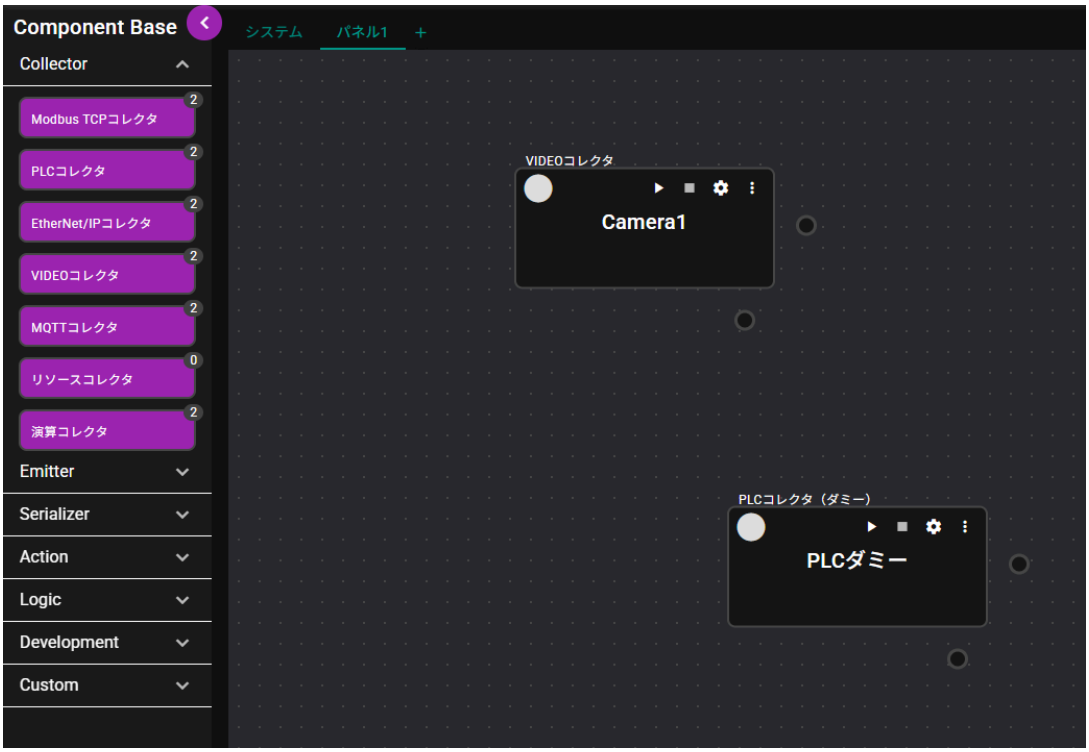
1. フローエディタの左部の「Collector」の「Camera1」をエディタ中央部へドラッグ＆ドロップします。



2. 名称を「Camera1」へ設定して、「RTSP URL」は、カメラの指定URLを入力します。そして、「保存」をクリックします。



3. Videoコレクタ「Camera1」の準備は完了です。



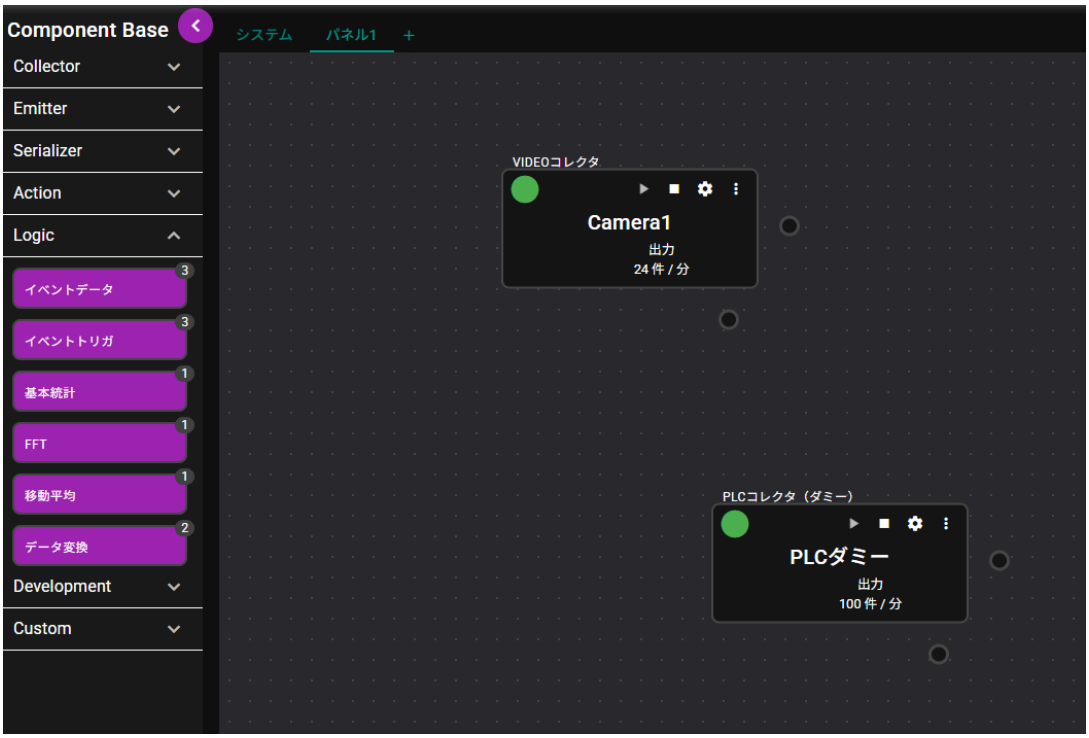
11.7.3.2 イベントトリガの設定

イベントトリガで、異常検知時の条件を行います。

1. イベントトリガの設定前に、「PLCダミー」と「Camera1」は起動します。パネル右上の「起動」をクリックします。



2. フローエディタの左部の「Logic」の「イベントトリガ」をエディタ中央部へドラッグ＆ドロップします。



3. 入力ポートへ接続するコンポーネントは、「PLCダミー」にチェックして「接続」をクリックします。



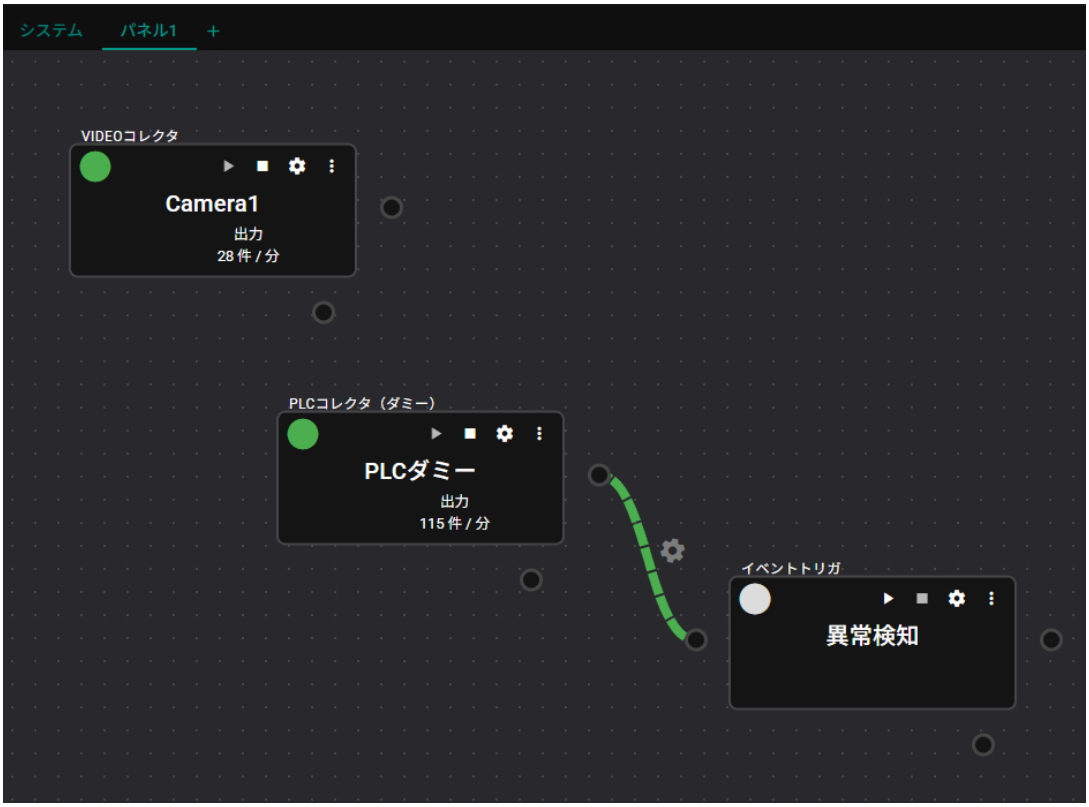
4. 名称を「異常検知」とします。その他項目は、下記を参考に入力して、「保存」をクリックします。



項目	値	補足
同一イベントを無視する時間（秒）	0	今回は無視する設定不要のため0とする
基本記録時間（分）	1	
開始イベント	-	-
コンポーネント	PLCダミー	

項目	値	補足
データ名	異常検知フラグ	
トリガ名	ON	
条件種別	Raise	
条件式	$VALUE = 1	フラグがtrueとなる立ち上がり条件とする
終了イベント	-	-
コンポーネント	PLCダミー	
データ名	異常検知フラグ	
トリガ名	OFF	
条件種別	Raise	
条件式	$VALUE = 0	フラグがfalseとなる立ち下がり条件とする

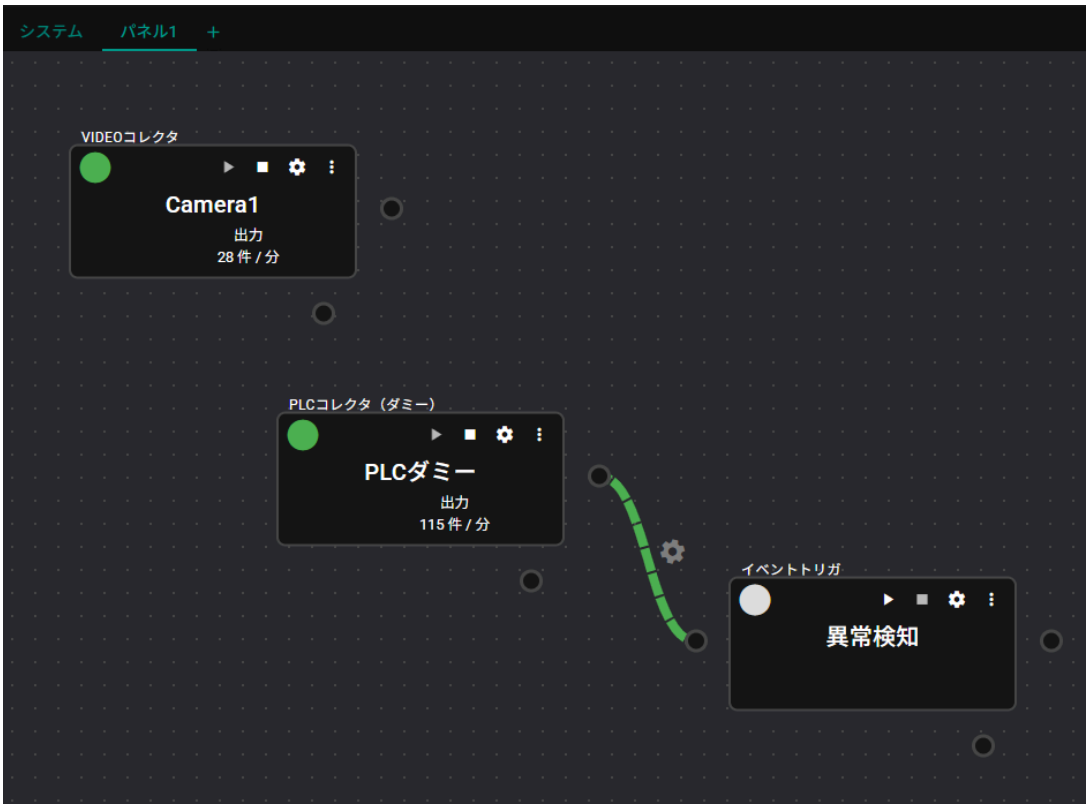
5. イベントトリガ「異常検知」の設定は完了です。



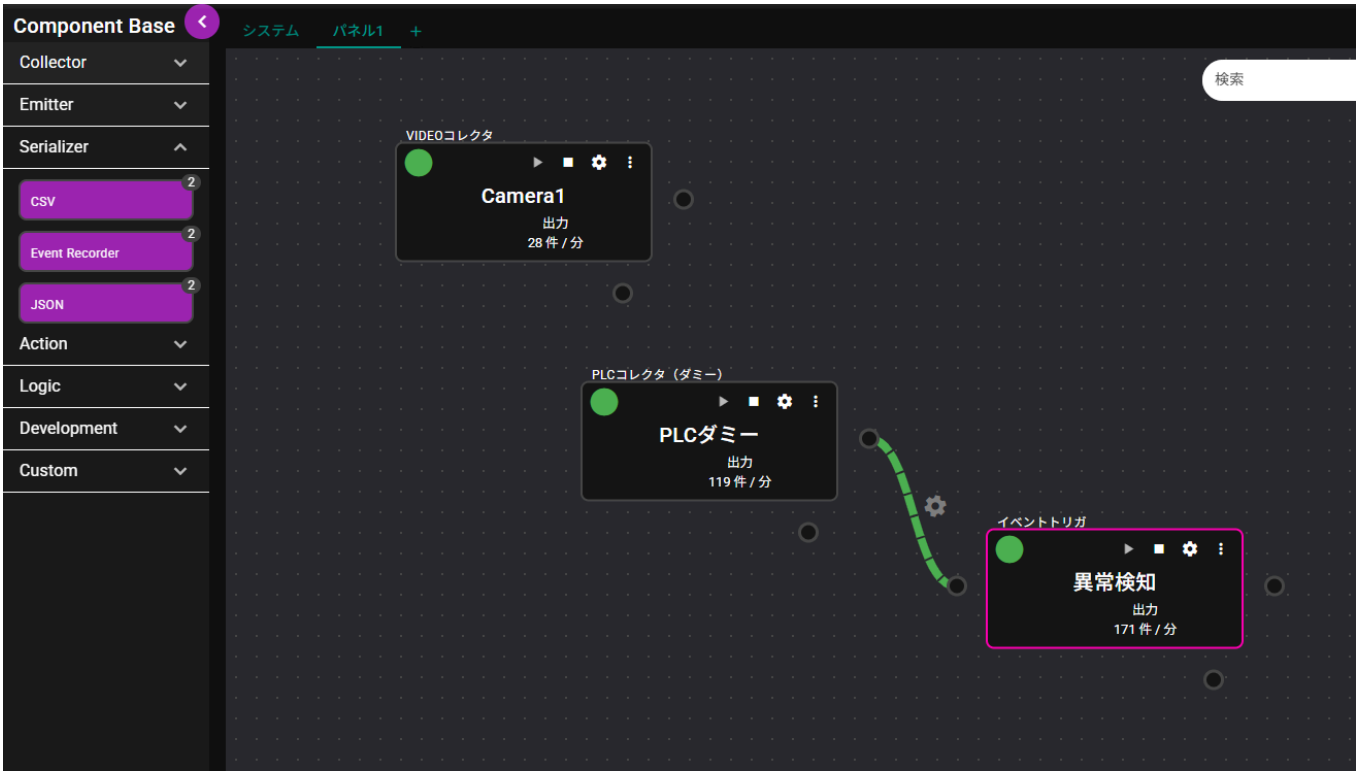
11.7.3.3 Event Recorderの設定

最後にEvent Recorderの設定を行います。異常発生時の記録区間などの設定もここで行います。

1. イベントトリガ「異常検知」を起動しておきます。「起動アイコン」をクリックします。



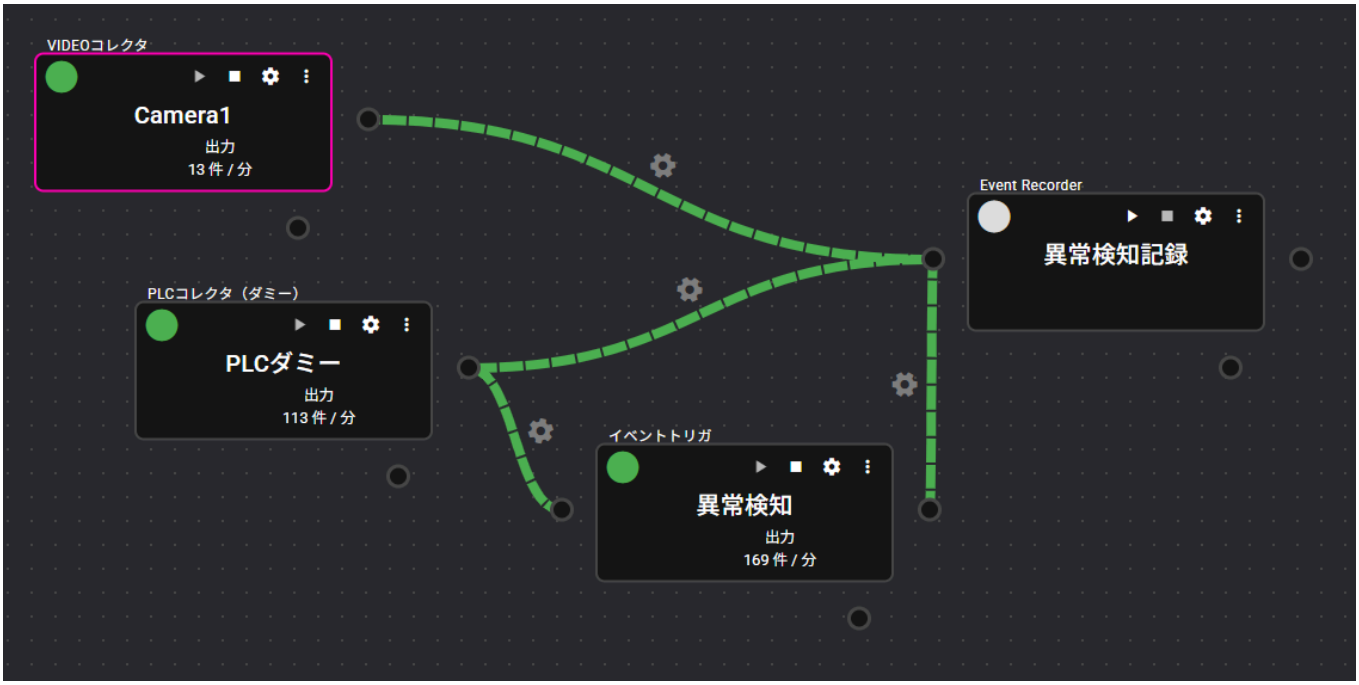
2. フローエディタの左部の「Serializer」の「Event Recorder」をエディタ中央部へドラッグ＆ドロップします。



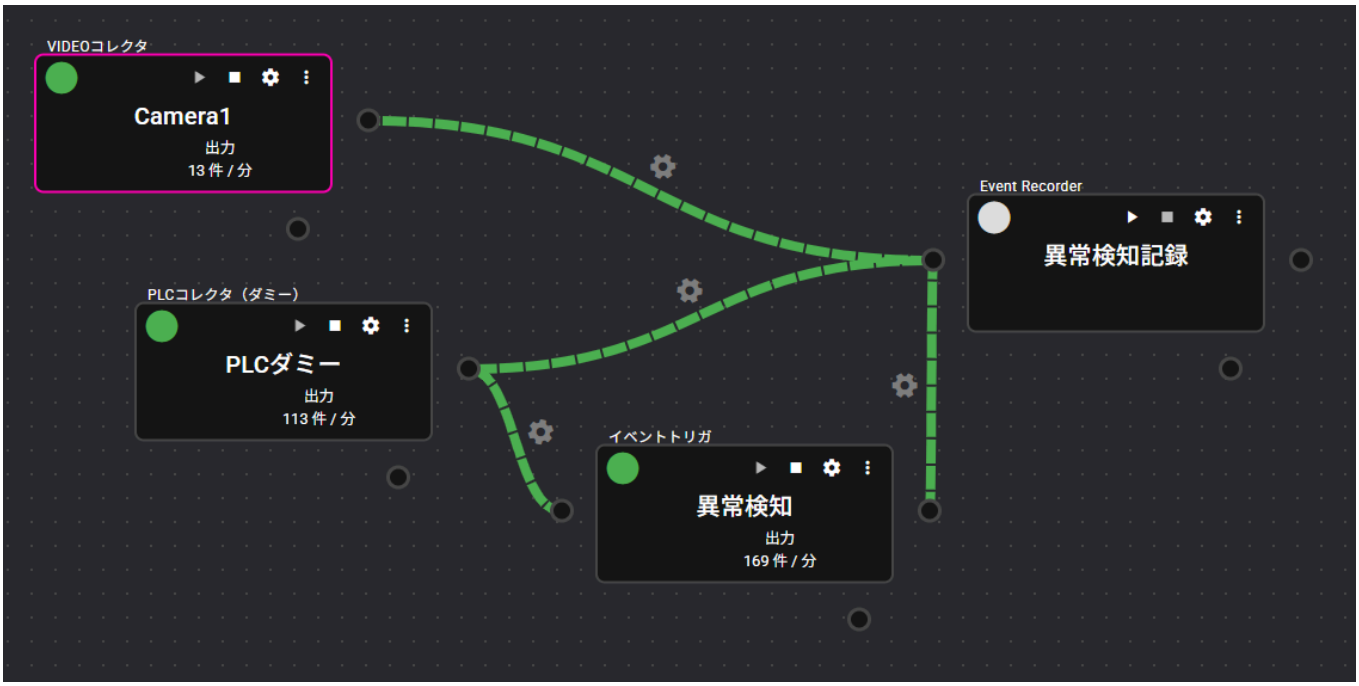
3. 名称を「異常検知記録」へ設定とします。「分類」は、「警告」を選択します。そして、「保存」をクリックします。



4. 「異常検知記録」へ、「PLCダミー」、「Camera1」、「異常検知」のフローリンクを接続します。



5. 「異常検知記録」の入力ポートの設定を開きます。



6. 入力ポートを設定して、「保存」をクリックします。入力値は下記を参考にしてください。



項目	値	補足
イベントを使用する	チェック ON	この設定がないと、Event Recorderでイベント発生範囲が記録されない。
イベント	異常検知	この設定がないと、Event Recorderでイベント発生範囲が記録されない。
イベント発生前の取得時間（秒）	5	

項目	値	補足
イベント終了後の取得時間（秒）	5	

7. イベントの記録のすべての設定は完了です。

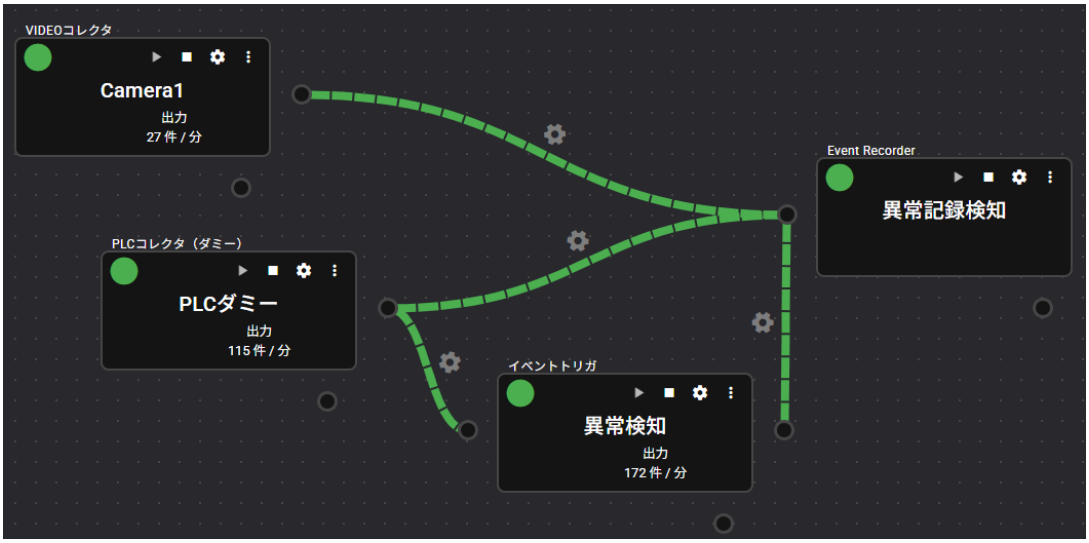
11.7.4 動作確認

ここまでに設定したデータフローを動作させます。

1. 設定したコンポーネントを起動します。



2. 配置したコンポーネントが正常に起動することを確認します。コンポーネントのランプに異常があれば設定を見直してください。



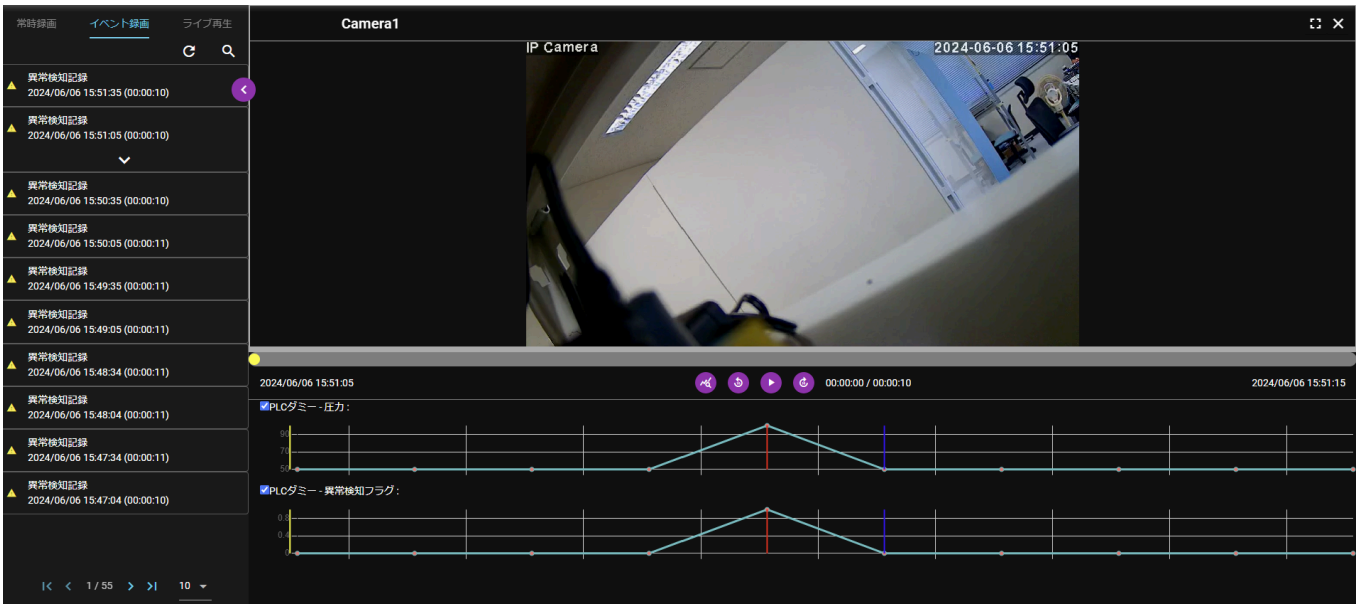
3. 数分以上稼働させたら、ビューワを確認します。「ビューワアイコン」をクリックします。



4. ビューワ左部で、「イベント録画」をクリックします。「異常検知記録」の一覧が表示されているか確認します。


常時録画
イベント録画
ライブ再生
異常検知記録
2024/06/04 20:00:29 (00:00:11)
異常検知記録
2024/06/04 20:00:00 (00:00:10)
異常検知記録
2024/06/04 19:59:29 (00:00:11)
異常検知記録
2024/06/04 19:58:59 (00:00:11)
異常検知記録
2024/06/04 19:58:29 (00:00:11)
異常検知記録
2024/06/04 19:57:59 (00:00:11)
異常検知記録
2024/06/04 19:57:29 (00:00:11)
異常検知記録
2024/06/04 19:56:59 (00:00:10)
異常検知記録
2024/06/04 19:56:29 (00:00:10)
異常検知記録
2024/06/04 19:55:59 (00:00:10)
1 / 11
10

5. 任意の「異常検知記録」をクリックして、ビューワ全体の表示を確認します。



6. 「再生アイコン」をクリックして、イベント録画を再生します。約10〜11秒の記録と合わせて、異常検知が1になったタイミングで、圧力も100となっていることが確認できれば正常に動作しています。



7. チュートリアルは以上となります。

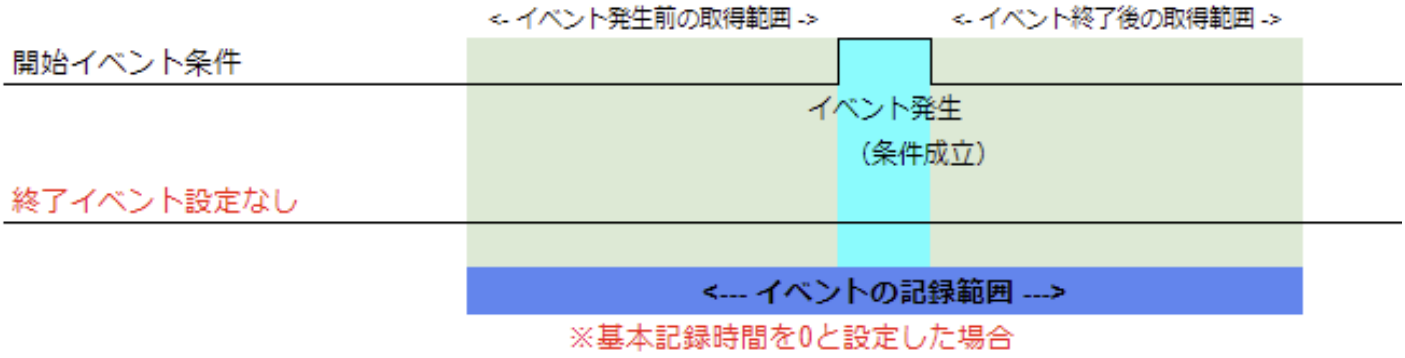
11.8 i-Triggerの記録仕様

ここでは、i-Triggerの記録仕様について説明します。 記録仕様について、少々複雑なため、詳しく確認したい場合に参照してください。

11.8.1 「イベントトリガ」で「開始イベント」のみを設定している場合

イベントトリガで、開始イベントのみを設定している場合、記録区間の考えは次のとおりです。

パターン1



- 基本記録時間を0分としています
- イベント発生前の取得範囲、イベント終了後の取得範囲にそれぞれ5秒を指定
- 開始イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。イベント発生前後の記録を5秒としているため、約10秒間イベントが記録されます。

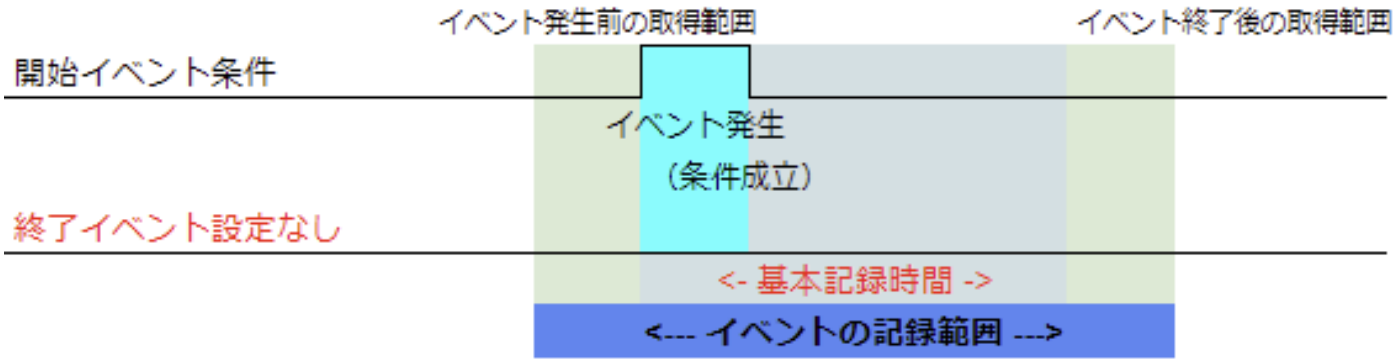
パターン2



- 基本記録時間を1分としています
- イベント発生前の取得範囲、イベント終了後の取得範囲はそれぞれ0秒を指定
- 開始イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。イベント発生してから、約1分が記録される範囲となります。

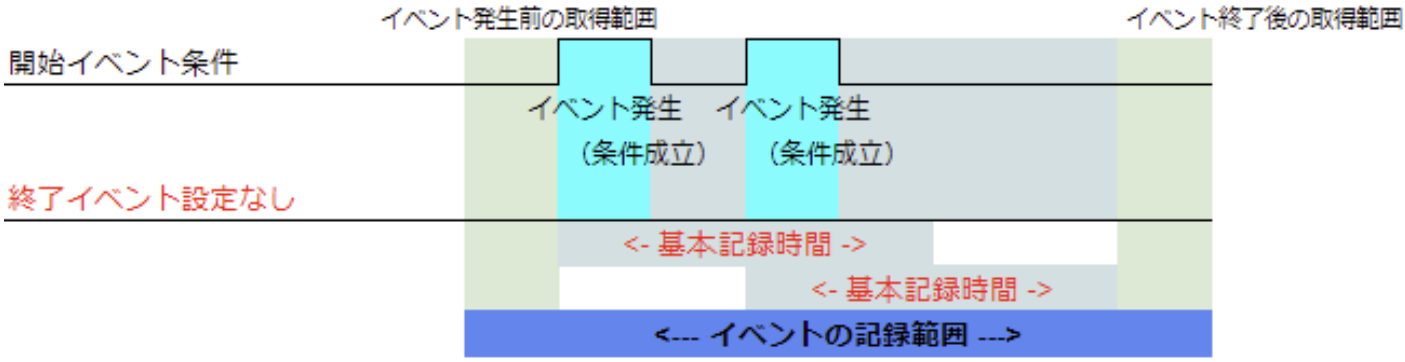
パターン3



- 基本記録時間を1分としています
- イベント発生前の取得範囲、イベント終了後の取得範囲にそれぞれ5秒を指定
- 開始イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。この場合は、基本記録時間＋イベント前後の範囲が記録されるため、約70秒間イベントが記録されます。

パターン4（イベントの延長）



- 基本記録時間を1分としています
- イベント発生前の取得範囲、イベント終了後の取得範囲にそれぞれ5秒を指定
- 開始イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない
- 初回のイベント成立から、30秒後にイベントが再び成立

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。基本記録時間の区間内で、再びイベントが成立したため、イベントが延長されます。この場合、最初のイベント成立区間30秒+2回目のイベント成立からの基本記録時間（60秒）+発生前後10秒となるため、約100秒間イベントが記録されます。

イベントの延長について

イベントの延長については、最大15分の上限があります。15分以上イベントが延長された場合は、その続きは新しいイベントとして記録されます。

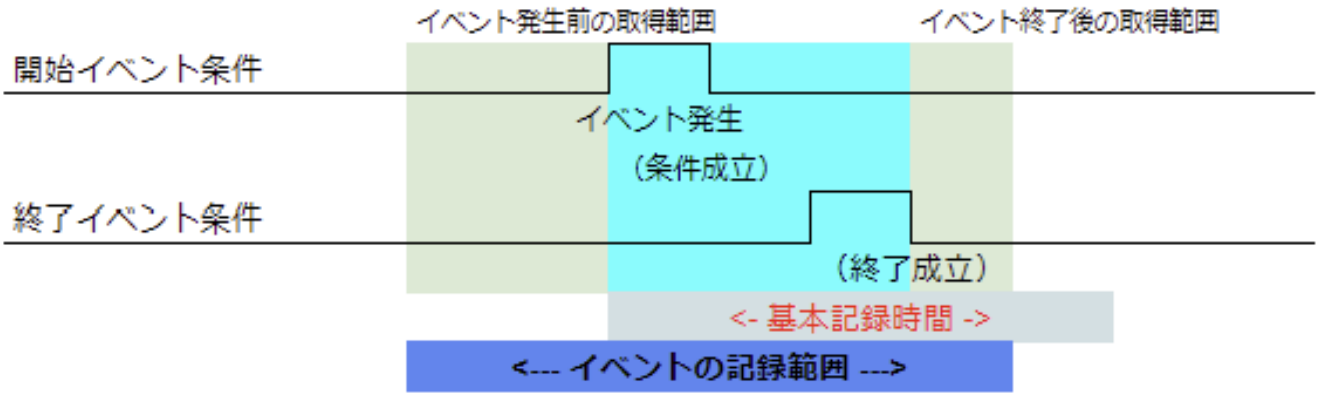
イベントの条件種別について

ここまでの説明はすべて、イベントの条件種別を「Raise」としています。この場合、値の立ち上がりが条件となるため、記載したパターンの動作となります。条件種別を「True」などにしている場合は、条件成立中は常にイベントが延長されることに注意してください。

11.8.2 「イベントトリガ」で「開始イベント」、「終了イベント」それぞれを設定している場合

イベントトリガで、開始イベント、終了イベントそれぞれを設定している場合、記録区間の考えは次のとおりです。基本的な考え方は、開始イベントのみ設定した場合と大きくは変わりません。

パターン5



- 基本記録時間を1分としています
- イベント発生前の取得範囲を10秒、イベント終了後の取得範囲を5秒に指定
- 開始イベント条件、終了イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない
- 開始イベント成立から、30秒後に終了イベントが成立

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。イベント発生前10秒+終了後5秒＋イベント開始から終了までが30秒となり、約45秒間のイベントが記録されます。基本記録時間に達する前に、終了イベントが成立したため、記録時間は1分より短くなります。

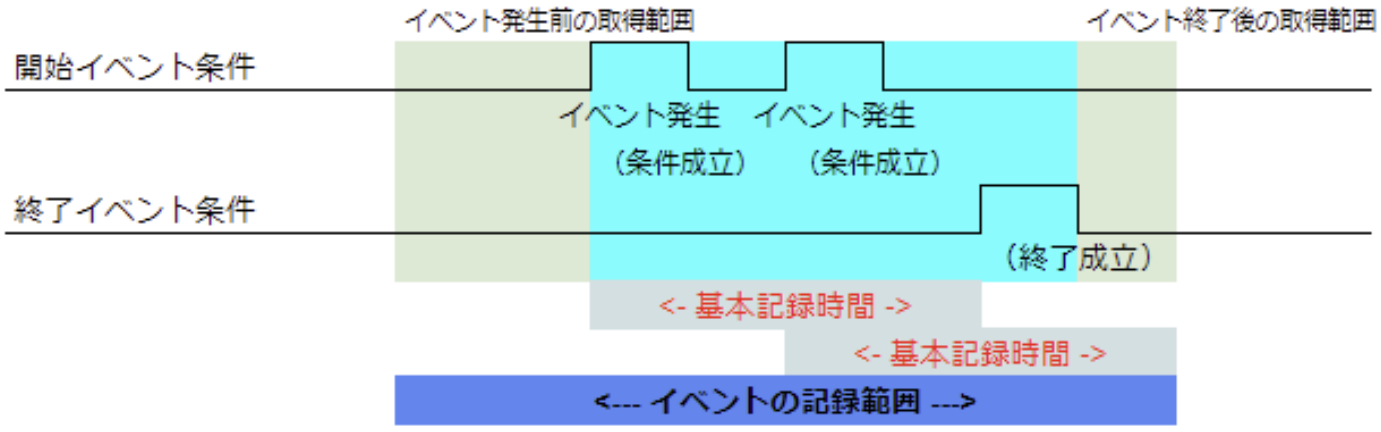
パターン6



- 基本記録時間を1分としています
- イベント発生前の取得範囲を10秒、イベント終了後の取得範囲を5秒に指定
- 開始イベント条件、終了イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない
- 開始イベント成立から、１分経過しても終了イベントが成立しなかった

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。イベント発生前10秒+終了後5秒＋イベント開始から基本記録時間の到達までとなり、約75秒間のイベントが記録されます。基本記録時間以内に、終了イベントが成立しなかったため、記録は打ち切られます。この場合、終了イベントが記録区間を超えて成立しても無視されます。

パターン7（イベントの延長）



- 基本記録時間を1分としています
- イベント発生前の取得範囲を10秒、イベント終了後の取得範囲を5秒に指定

- 開始イベント条件、終了イベント条件はRaiseとして一度成立した後、条件が未成立にならないと連続では条件成立しない
- 開始イベント成立から、30秒後に開始イベントが再び成立、そこから35秒経過して終了イベントが成立

このパターンで記録される範囲は、「イベントの記録範囲（青背景色部分）」となります。イベント発生前10秒+発生後5秒＋最初のイベント成立区間30秒＋次のイベント成立からさらに35秒までとなり、約80秒間のイベントが記録されます。基本記録時間以内に、開始イベントが再び成立したため、基本記録時間は延長されます。

イベントの延長について

イベントの延長については、最大15分の上限があります。15分以上イベントが延長された場合は、その続きは新しいイベントとして記録されます。

イベントの条件種別について

ここまでの説明はすべて、イベントの条件種別を「Raise」としています。この場合、値の立ち上がりが条件となるため、記載したパターンの動作となります。条件種別を「True」などにしている場合は、条件成立中は常にイベントが延長されることに注意してください。

12 リリースノート

本章は、SpeeDBee Synapseのリリースノートです。

12.1 v.4.9.9 (2025/9/12)

- BlackBear OS(v3.0.10)対応
- 下記の不具合を修正
  - 出力ポートの未使用カラムが無効化されず余分にメモリを確保し続けることがある
  - ファイル型カラムの実ファイルの削除処理が動作しないことがある

12.2 v.4.9.8 (2025/9/10)

- 仕様変更、改善
  - DBクエリにおけるレスポンスデータ件数の上限の１万件を撤廃、および、メモリ使用効率の改善
- 下記の不具合を修正
  - 実行ライセンス用設定ファイルを読み込んだ際に一部のプロジェクト情報が引き継げず多数のコンポーネントが起動できなくなることがある

12.3 v.4.9.5 (2025/9/2)

- 仕様変更、改善
  - httpsのリバースプロキシ経由でWeb画面にアクセスした場合の挙動改善
  - 全般的な設定でtmpfsをデータディレクトリに設定した場合でもWeb画面上のディスク使用量情報に表示する改善
  - マニュアルにData Flow Controller及びOMRON-SBS for Windowsに関する補足事項を追記
  - SCCDEパッケージのカスタムHTMLで作成されたパラメータ画面をオペレータ権限でも表示可能に変更
- 下記の不具合を修正
  - サポートファイルダウンロード時にイベント録画のDBファイルが含まれない場合がある
  - Event Recorder実行時にディスクフルとなった場合にイベント録画データに不整合が起こる
  - HTMLによるカスタムコンポーネント用カスタムUI機能でパラメータ情報が多い場合にカスタムHTMLへのデータ伝達に失敗することがある
  - グループ単位の保存期間設定がOFFの場合、シリアライザで出力したファイル型データの保存時間が極端に短くなる

- SCCDEパッケージでカスタムHTMLを使用している場合、パッケージ保護を適用すると設定が表示できなくなる

12.4 v.4.9.4 (2025/8/28)

- 下記の不具合を修正
  - DBクエリに改行コードなどの特殊文字を含んだSQLを送信するとコアが停止/再起動される場合がある
  - 全般的な設定でデータディレクトリを変更している場合にデータディレクトリがディスクフルになると、イベント録画のデータ一覧に不整合が生じる

12.5 v.4.9.3 (2025/8/25)

- 下記の不具合を修正
  - イベントトリガ、イベントデータ、基本統計、移動平均、FFTコンポーネントにて、同一のカラムに対して複数の条件設定を行うとコンポーネント開始時にデットロックが発生することがある
- マニュアルの改善
  - インストールマニュアル、およびトラブルシューティングの説明文を修正

12.6 v.4.9.2 (2025/8/18)

- DBクエリの実行で大量のデータを取得する場合にメモリ枯渇する可能性がある問題の追加対応

12.7 v.4.9.1 (2025/8/8)

- 仕様変更、改善
  - マニュアルのキャプチャ画像に最新のUIのデザインを反映
  - Linux版インストール時のセットアップスクリプトからbash依存構文を排除
- 下記の不具合を修正
  - DBクエリの実行で大量のデータを取得する場合にメモリ枯渇する可能性がある

12.8 v.4.9.0 (2025/7/31)

- 仕様変更、改善
  - Cloudエミッタ(AWS用,Azure用), MQTTコレクタ、MQTTエミッタにてプロキシを利用した通信に対応
  - Pythonカスタムコンポーネントの依存モジュールインストール時にプロキシを利用した通信に対応
  - 数値変換機能においてBCD形式データを通常の数値に変換する計算式の入力に対応

    - 出力ポート設定における永続化、数値変換、登録条件の設定有無の表示改善
    - 入力ポート設定、出力ポート設定、システムポート設定、フローリンク設定を右クリックなしでも設定可能とするUI改善
    - PLCコレクタ、PLC書き込みの機種選択に 三菱電機 MELSEC iQ-R シリーズを追加
- 下記の不具合を修正
    - サポートファイルのサイズが巨大な場合ダウンロード時にメモリ枯渇する可能性がある
    - Windows環境でv4.7.2から上書きインストールした場合に一部のコンポーネントのデータが引き継がれない
    - v4.7.2から上書きインストールした後に発行したアクセストークンがDBクエリで使用できない
    - DBクエリコンポーネントへのリクエストヘッダのAPIキーに余分な文字列を追加するとエラーとなる

12.9 v.4.8.0 (2025/6/30)

- 仕様変更、改善
    - プロジェクト機能を追加
    - 英語版マニュアルを追加
    - 時系列データベース（Synapseが収集したデータの保存先）のサイズ上限設定を追加
    - コンポーネント個別のログ表示・ログレベル切り替えに対応
    - データ変換コンポーネントを追加
    - VIDEOコレクタ/Fileエミッタ/Event Recorderの出力先設定の改善
    - ログおよび一時ファイルなどの削除機能を追加
    - EtherNet/IPコレクタにて `SHORT STRING` 型データの取得に対応
    - 各コンポーネントの設定画面にマニュアルへのリンクアイコンを追加
    - PLCコレクタとModbusTCPの収集間隔の単位を秒からミリ秒へ変更
    - HTMLによるカスタムコンポーネント用カスタムUI機能を追加
    - DBクエリコンポーネントにてSQLのINSERT構文に対応
    - フローエディタにおけるタッチパネル関連操作を改善（iPad + Safari）
- 下記の不具合を修正
    - カスタムコンポーネントのパラメータ設定で添付ファイルを利用できない（Windows版のみ）
    - PLC/MODBUS/EtherNetIPコレクタで収集間隔を100ms未満にするとサーバー設定のメモリ保持期間より短い期間しか保持されない
    - 入力ポート設定、出力ポート設定、フィルタ設定にて対象データの型が変更された場合に対象外とされてしまう
    - Windows版のEtherNet/IPコレクタ実行中にネットワークが切断されると復旧後も接続が再開されない
    - メール送信コンポーネントで認証エラーの場合にもコンポーネントが停止せず動き続けてしまう

    - MQTTエミッタのテスト接続時に設定項目のタイムアウト値が反映されていない
- 内部変更
    - プロジェクト機能追加に伴う内部データのディレクトリ構成変更
    - 大量のカラムを作成するコンポーネント開始時の性能改善
    - 多数のコンポーネントを作成したときのWeb画面の動作性能改善

12.10 v.4.7.6 (2025/3/19)

- 仕様変更、改善
    - EtherNet/IP書き込みコンポーネントにおいてCIPObjectへの書き込みに対応
    - カスタムコンポーネント保護機能の追加
    - Synapseカスタムコンポーネント開発環境の追加
    - Synapseが稼働するシステム時刻に異常を検出した場合に画面上に通知する機能を追加
    - シミュレーションモードの追加
- 下記の不具合を修正
    - EtherNet/IPコレクタにてCIPObjectのSTRING型を取得できない場合がある
    - EtherNet/IPコレクタにて同一変数名を複数設定した場合に一部が取得できない
    - MQTTコレクタにてnull値を受信した場合にエラーとなる
- マニュアルの章構成の見直し
- 内部変更
    - 過去バージョンからのデータ移行処理を改善
    - 内部データのディレクトリ構成変更

12.11 v.4.7.5 (2025/3/13)

- 下記の不具合を修正
    - BlackBear版にて特定の条件によりインストールが失敗する

12.12 v.4.7.3 (2025/2/1)

- BlackBear対応
- 仕様変更、改善
    - ライセンス照合に使用するMACアドレスから、仮想インタフェースを除外

12.13 v.4.7.2 (2025/1/31)

- v.4.7.0における変更内容を他の動作環境へ反映
- 下記の不具合を修正
  - フローエディタのコンポーネント検索後に、検索欄から選択したコンポーネントが画面の中央に表示されない

12.14 v.4.7.0 (2024/12/20)

対象動作環境：Ubuntu22(arm64)

- 仕様変更、改善
  - コンポーネントベース一覧のツールチップ表示改善
  - フローエディタ内のコンポーネントの複数選択に対応
  - フローエディタ内のコンポーネントのコンテキストメニューを改善
  - 全般的な設定ダイアログを追加（タイムゾーン設定のみ）
  - パネルの並び替え、複製に対応
  - メール送信コンポーネントにて使用可能な変数を追加
  - PLCコレクタ,EtherNet/IPコレクタ, Modbus TCPコレクタにおけるタイムスタンプの付与方法に関する設定を追加
  - カスタムコレクタの読み込みに失敗した場合のコンポーネント動作を改善
- 下記の不具合を修正
  - FTPエミッタにて特定のFTPサーバーへのファイル保存に失敗する
  - 演算コレクタにて入力されたデータがBOOL型の場合に不正な演算結果が出力される

12.15 v.4.6.2 (2024/10/17)

対象動作環境：Ubuntu20(amd64), Ubuntu22(amd64,arm64), RaspberryPiOS(bookworm 64bit), Windows10(64bit 22H2以降), Windows11(64bit 23H2以降), 特殊環境（Arm32 bit）

- 仕様変更、改善
  - メール送信コンポーネントにおける送信失敗時のエラー通知改善
- 下記の不具合を修正
  - Cloudエミッタ(Azure用)、Cloudエミッタ(AWS用)、FTP/SFTPエミッタが特定の条件で起動できなくなる
  - EtherNet/IP書き込みコンポーネントにおいて一部の受信データが欠落していた場合にエラーとなってしまう
  - 入力ポート設定のタンブリングウィンドウにおいて、RAW_DATAをOFFにして集約すると、指定幅による集約結果が得られない
  - EtherNet/IP書き込みで一部の構造体の書き込みに失敗する

12.16 v.4.6.0 (2024/9/30)

対象動作環境：Windows10(64bit 22H2以降), Windows11(64bit 23H2以降)

- Windows対応
- 仕様変更、改善
  - VIDEOコレクタにおいて、保存先に入力できる文字数上限を200文字へ変更
  - Cカスタムコンポーネントサンプルのディレクトリ構成を変更
  - Grafanaのデータソースに設定するURLをSynapseの画面から指定できるように改善
  - EtherNet/IPコレクタにおいて、BOOL型配列を変数名のみで指定した場合、配列数が0または1ならタプルの1要素目を登録するように対応
- 下記の不具合を修正
  - 演算コレクタにおいて、複数の入力データをトリガとした計算式を設定した場合、コレクタが異常停止する
  - 演算コレクタにおいて、複数の入力データを演算する際にトリガを一つのみ設定すると、演算結果が不正となる
  - SFTPエミッタにおいて、ファイルが意図したものとは異なるディレクトリに配置されることがある
  - SFTPエミッタ稼働中にSSH接続が増え続ける
  - MQTTエミッタにおいて、QoSが1または2の場合であっても、ネットワーク切断->再開時にメッセージが欠損することがある

12.17 v.4.5.1 (2024/9/20)

対象動作環境：特殊環境（Arm32 bit）

- EtherNet/IPコレクタ配列要素を指定しての取得を追加
- 下記の不具合を修正
  - EtherNet/IP書き込みでBOOL配列の書き込みができない
  - EtherNet/IPコレクタでBOOL配列の読み込みができない
  - イベントトリガの設定を変更した際に場合によってはトリガが実行されない
  - EtherNet/IPコレクタの長時間データ取得で周期がずれる

12.18 v.4.5.0 (2024/9/6)

対象動作環境：特殊環境（Arm32 bit）

- PLCコレクタ、PLC書き込みにてOMRON Sysmac NJ/NXを追加
- EtherNet/IP書き込みのプロトタイプを追加
- 下記の不具合を修正

    - pythonコンポーネントの入力ポートに配列データが入ってきた場合、全て0番目の要素と同じ値になる
- 内部変更
    - WebAPIのアドレス設定を127.0.0.1に変更
    - コアのポート変更時の内部挙動を変更
    - system_configの不要になった設定項目を削除
    - イベントトリガの挙動を変更
    - ポート番号のチェックを追加

12.19 v.4.4.0 (2024/7/31)

対象動作環境：Ubuntu20(amd64), Ubuntu22(amd64,arm64), RaspberryPiOS(bookworm 64bit)

- 実行ライセンス設定の出力のエクスポート時に対象パネルと自動起動の有無を選択する機能を追加
- カスタムコンポーネント用Python-APIに最新値取得関数を追加
- HTTP/HTTPS通信の設定画面にWebUIが使用するアドレス/ポートの設定項目を追加
- ソフトウェア情報画面にライセンス判定用MACアドレスの情報を表示
- ソフトウェア情報画面にて不正なライセンスキーを入力したときのエラー表示を改善
- 下記の不具合を修正
    - ステータスタイムラインで再起動前のデータを取得するをONにしたときにコアが異常終了する
    - 特定のコンポーネントをCSV/JSONシリアライザに繋ぐとシリアライザがエラーで停止する
- ドキュメント改善
    - Pythonカスタムコンポーネントのファイル名についての補足
    - Modbusのアドレス範囲についての補足
    - 文章校正（誤字脱字の修正、解りづらい文章表現の改善等）
- 内部変更
    - コアのライブラリ構成変更
    - インストールスクリプトのログとエラー判定の改善
    - 永続化データの出力間隔のデフォルト値変更（１秒→２０秒）

12.20 v.4.3.0 (2024/8/2)

対象動作環境：特殊環境（Arm32 bit）

- 4.2以降の一部の修正を反映した上で、特殊環境へ移植

12.21 v.4.2.2(2024/6/28)

対象動作環境：Ubuntu20(amd64), Ubuntu22(amd64,arm64), RaspberryPiOS(bookworm 64bit)

- Synapse動作環境の追加：RaspberryPiOS(bookworm 64bit)
- i-Trigger関連機能の追加
  - VIDEOコレクタ
  - Event Recorder
  - 動画再生用ビューワ
- PLCコレクタのデータ取得最短周期を100msから10msに短縮
- MQTTコレクタにAWS IoT Core（カスタム認証）の設定項目を追加
- MQTTコレクタによるクラウドサービスからのデータ受信の手順をマニュアルに追加
- 不具合の修正
  - ディスク空き容量が不足した場合の動作改善
  - Synapse再インストール時にカスタムコンポーネント用のPython外部モジュールが削除される問題を修正

12.22 v.4.1.0 (2024/6/17)

対象動作環境：Ubuntu22(arm64)

- 実行ライセンス機能の追加
- Grafana連携においてGrafanaのサービスアカウント機能に対応
- Pythonカスタムコンポーネントの実行用に、カスタムコンポーネント設定画面からのwhlファイルの登録・インストールに対応
- カスタムUIにおいて入れ子になったデータ構造の定義に対応
- カスタムUIにおいて設定画面のサイズ指定に対応
- SpeeDBee Synapseドキュメント校正（誤字脱字の修正、解りづらい文章表現の改善等）
- 不具合の修正
  - ディスク空き容量が不足した場合の動作改善
  - ステータスタイムラインにおけるエラー情報の表示改善
  - 永続化されたデータベースが特定の条件で保存期間を過ぎても削除されなくなる不具合を修正
  - PLCコレクタMELSECプロトコルにおいてbitレジスタに0を書き込めなくなる不具合を修正
  - adminユーザーを無効化できない不具合を修正
  - PLCコレクタFINSプロトコル利用時にシステムが再起動する不具合を修正
  - PLCコレクタMELSEC(3E)プロトコルのZRレジスタ利用時に指定アドレスと異なるアドレスにアクセスしてしまう不具合を修正

- コアプロセス起動時の処理改善

12.23 v.4.0.2 (2024/4/1)

対象動作環境：Ubuntu20(amd64), Ubuntu22(amd64,arm64)

- SpeeDBee Hiveを進化させた新製品「SpeeDBee Synapse」をリリースしました。

SpeeDBeeおよびSpeeDBee Synapseは株式会社ソルティスターの登録商標です。
「SpeeDBee Synapse」のロゴは、商標登録出願中です。

Microsoft Azure、Microsoft Defender、Windowsは米国Microsoft Corporationの米国、日本およびその他の国における登録商標または商標です。

その他、記載されている会社名と製品名などにつきましては、各社の登録商標または商標です。
または、
その他、本文中に掲載している会社名および製品名は、それぞれ各社の商標または登録商標です。

本製品（SpeeDBee Synapse）を購入した販売元へご確認ください。

15 付録

15.1 付録

1. エラーコード一覧
2. カスタムコンポーネントPython-APIリファレンス
3. カスタムコンポーネントC-APIリファレンス
4. カスタムUI JS-APIリファレンス
5. 利用しているオープンソース
6. 索引
7. DBクエリサービスWeb-API仕様

15.2 エラーコード一覧

ステータスタイムラインに表示されるエラー識別子の一覧および、識別子ごとに次の項目を掲載します。

項目	説明
メッセージ	エラー識別子に割り当てられたメッセージ
内容	エラーの詳しい内容
処置方法	エラーを解消するための方法

15.2.1 コンポーネント共通

15.2.1.1 NO_MEMORY

項目	説明
メッセージ	メモリ不足
内容	SpeeDBee Synapseが稼働しているマシンのメモリが不足しています。
処置方法	使用していないコンポーネントは停止して下さい。また、無駄なデータ収集など設定されていなかなどを確認して下さい。 それでも解消しない場合は、システム管理者に相談し、メモリの増設または他アプリケーションとの調整を行ってください。

15.2.1.2 PARAMETER_PARSE_ERROR

項目	説明
メッセージ	パラメータ パースエラー
内容	設定ファイルが正しい形式で保存されていません。設定内容が破損している可能性があります。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.1.3 PARAMETER_FORMAT_ERROR

項目	説明
メッセージ	パラメータエラー
内容	設定ファイルに不正なパラメータが存在します。
処置方法	演算コレクタでは変数、式が何も設定されていない場合に発生します。その他のコンポーネント（Azureの接続文字列不正など）でも、設定が不十分な場合に発生することがあります。該当のコンポーネント設定を確認して下さい。エラーが解消しない場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.1.4 FRAMEWORK_ERROR

項目	説明
メッセージ	フレームワークAPIエラー
内容	該当のコンポーネントが使用している内部フレームワークでエラーが発生しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.1.5 DISK_FULL

項目	説明
メッセージ	ディスクフル
内容	SpeeDBee Synapseが稼働しているマシンのストレージ空き容量が不足しています。
処置方法	不要なファイルを削除するなどの処置が必要です。システム管理者に相談し、ストレージ空き容量を確保してください。

15.2.1.6 IO_ERROR

項目	説明
メッセージ	IOエラー
内容	SpeeDBee Synapseが稼働しているマシンで、I/O処理に失敗しました。(ストレージの読み書き失敗など)
処置方法	システム管理者に相談し、ストレージの状態などを確認してください。(故障の可能性など)

15.2.1.7 NETWORK_ACCESS_ERROR

項目	説明
メッセージ	ネットワークアクセスエラー
内容	該当のコンポーネントで、通信の異常が発生しました。
処置方法	(処置方法1) SpeeDBee Synapseが稼働しているマシンが、ネットワークに接続されていることを確認してください。 (処置方法2) 該当のコンポーネントの通信先の機器やサービスが、通信可能な状態であることを確認してください。

15.2.1.8 PROCESSING_DELAY

項目	説明
メッセージ	処理遅延
内容	該当のコンポーネントで、処理が遅延しています。
処置方法	(処置方法1) 該当のコンポーネントが送受信しているデータ件数を減らしてください。 (処置方法2) SpeeDBee Synapseで稼働中のコンポーネントのうち、不要なものを停止または削除してください。 (処置方法3) SpeeDBee Synapseが稼働しているマシンのネットワークに、遅延が発生していないか確認してください。

15.2.1.9 INVALID_REGEX

項目	説明
メッセージ	正規表現エラー
内容	該当のコンポーネントの設定項目に、不正なフォーマットの正規表現が入力されています。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 不正なフォーマットの正規表現が入力されている設定項目を確認してください。 3. 手順2で確認した設定項目に、正しいフォーマットの正規表現を入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.1.10 COMPONENT_DATA_MISMATCH

項目	説明
メッセージ	指定したパラメータに一致するコンポーネントのデータがない ※次の情報を含みます。 ・コンポーネント ・データ
内容	設定項目で指定した内容に該当するデータが存在しません。 この状態では、該当の設定による処理が行われません。
処置方法	1. メッセージに含まれているコンポーネント、データを確認してください。 2. 該当のコンポーネントの入力ポートを右クリックしてください。 3. 設定項目「設定方法」に「データ定義を受信して設定する」を指定し、データ一覧を確認してください。 4. 該当のコンポーネントの設定画面を開いてください。 5. 手順1で確認したデータが指定されている設定項目に、手順3で確認したデータを入力して保存してください。 ※正規表現でデータを指定する場合は、手順3で確認したデータを1件以上含む正規表現を入力してください。 6. 該当のコンポーネントを再起動してください。

15.2.1.11 STATISTICS_INIT_ERROR

項目	説明
メッセージ	統計処理 初期化エラー
内容	統計処理の初期化に失敗しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.1.12 STATISTICS_INSERT_ERROR

項目	説明
メッセージ	統計値登録エラー
内容	統計値の登録に失敗しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.1.13 UNAVAILABLE_DATA_TYPE

項目	説明
メッセージ	条件式などに使用できない型が指定された ※次の情報を含みます。 ・コンポーネント ・データ
内容	条件式などのデータに使用できない型が指定されています。文字列や配列などは条件式で使用できません。
処置方法	1. メッセージに含まれているコンポーネント、データを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したデータの型を指定している設定項目に、データに適した型を指定して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.1.14 UNKNOWN_ERROR

項目	説明
メッセージ	不明なエラー
内容	原因不明のエラーが発生しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.1.15 INTERNAL_ERROR

項目	説明
メッセージ	予期しないエラー
内容	他のエラー識別子に該当しないエラーが発生しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2 リソースコレクタ

15.2.2.1 CPUUSAGE_OPENREAD_ERROR

項目	説明
メッセージ	CPU使用率情報 オープンエラー
内容	CPU使用率情報を含む以下のファイルにアクセスできません。 /proc/stat
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.2 CPUUSAGE_FORMAT_ERROR

項目	説明
メッセージ	CPU使用率情報 取得エラー
内容	以下のファイルから、CPU使用率情報を取得できませんでした。 /proc/stat
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.3 CPUFREQ_OPENREAD_ERROR

項目	説明
メッセージ	CPU周波数情報 オープンエラー
内容	CPU周波数情報を含む以下のファイルにアクセスができません。 /sys/devices/system/cpu/cpu*/cpufreq/cpuinfo_cur_freq
処置方法	お使いのマシン、プラットフォームによっては、CPU周波数を取得することはできません。上記のファイルが存在し、このエラーが発生する場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.4 CPUFREQ_FORMAT_ERROR

項目	説明
メッセージ	CPU周波数情報 取得エラー
内容	以下のファイルから、CPU周波数情報を取得できませんでした。 /sys/devices/system/cpu/cpu*/cpufreq/cpuinfo_cur_freq
処置方法	お使いのマシン、プラットフォームによっては、CPU周波数を取得することはできません。上記のファイルが存在し、このエラーが発生する場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.5 CPUTEMP_OPENREAD_ERROR

項目	説明
メッセージ	CPU温度情報 オープンエラー
内容	CPU温度情報を含む以下のファイルにアクセスができません。 /sys/class/thermal/thermal_zone0/temp
処置方法	お使いのマシン、プラットフォームによっては、CPU温度を取得することはできません。上記のファイルが存在し、このエラーが発生する場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.6 CPUTEMP_FORMAT_ERROR

項目	説明
メッセージ	CPU温度情報 取得エラー
内容	以下のファイルから、CPU温度情報を取得できませんでした。 /sys/class/thermal/thermal_zone0/temp
処置方法	お使いのマシン、プラットフォームによっては、CPU温度を取得することはできません。上記のファイルが存在し、このエラーが発生する場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.7 IOSTAT_NOT_FOUND

項目	説明
メッセージ	IO情報 デバイスエラー
内容	IO情報を含む以下のファイルが存在しません。 /dev/{設定項目の「I/Oデバイス」}
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.8 IOSTAT_OPENREAD_ERROR

項目	説明
メッセージ	IO情報 オープンエラー
内容	IO情報を含む以下のファイルにアクセスができません。 /dev/{設定項目の「I/Oデバイス」}
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.9 IOSTAT_FORMAT_ERROR

項目	説明
メッセージ	IO情報 取得エラー
内容	以下のファイルから、IO情報を取得できませんでした。 /dev/{設定項目の「I/Oデバイス」}
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.10 IOSTAT_STORAGE_MOUNT_POINT_ERROR

項目	説明
メッセージ	IO情報取得不可(マウントポイントが取得できない) ※次の情報を含みます。 ・デバイス名
内容	マウントポイントに関する情報が、設定ファイルに正しく定義されていません。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.11 IOSTAT_STORAGE_INFO_ERROR

項目	説明
メッセージ	IO情報取得不可(statvfsで情報が取得できない) ※次の情報を含みます。 ・デバイス名 ・マウントポイント ・ステータス
内容	設定ファイルに定義されているマウントポイントに対して、statvfsコマンドを使用してIO情報を取得できませんでした。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.12 MEM_OPENREAD_ERROR

項目	説明
メッセージ	メモリ情報 オープンエラー
内容	メモリ情報を含む以下のファイルにアクセスができません。 /proc/meminfo
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.13 MEM_FORMAT_ERROR

項目	説明
メッセージ	メモリ情報 取得エラー
内容	以下のファイルから、メモリ情報を取得できませんでした。 /proc/meminfo
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.14 NETWORK_OPENREAD_ERROR

項目	説明
メッセージ	ネットワーク情報 オープンエラー
内容	ネットワーク情報を含む以下のファイルにアクセスができません。 /proc/net/dev
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.2.15 NETWORK_FORMAT_ERROR

項目	説明
メッセージ	ネットワーク情報 取得エラー
内容	以下のファイルから、ネットワーク情報を取得できませんでした。 /proc/net/dev
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.3 PLCコレクタ

15.2.3.1 CONNECTION_REFUSED

項目	説明
メッセージ	接続が切断
内容	PLCとの接続が切断されました。

項目	説明
処置方法	(処置方法1) コンポーネントを再起動してください。 (処置方法2) 1. PLC側の設定で、通信可能なホスト・ポートを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「ホスト」「ポート」に、手順1で確認したホスト・ポートを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.3.2 NETWORK_TIMEOUT

項目	説明
メッセージ	タイムアウト ※次の情報を含みます。 ・アドレス ・ポート番号 ・設定されているタイムアウト値 ・操作種別
内容	(操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」の時間内にPLCへ接続できませんでした。 (操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」の時間内にPLCからデータを受信できませんでした。
処置方法	1. メッセージに含まれている操作種別を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. (手順1で確認した操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」を大きくしてください。 (手順1で確認した操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」を大きくしてください。 4. 該当のコンポーネントを再起動してください。

15.2.3.3 REGISTER_ADDRESS_ERROR

項目	説明
メッセージ	指定レジスタが不正 ※次の情報を含みます。

項目	説明
	・レジスタ名 ・レジスタアドレス ・操作種別
内容	ビット指定できないレジスタアドレスが、ビット指定されて設定項目「アドレス」に入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、ビット指定が可能なレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.3.4 REGISTER_ADDRESS_RANGE_ERROR

項目	説明
メッセージ	不正なレジスタアドレス ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「アドレス」に、PLCのプロトコルでは扱えないレジスタアドレスが入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、PLCプロトコルで扱えるレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。 ※利用できるアドレス、その範囲については、該当機器のマニュアル等を参照して下さい。

15.2.3.5 REGISTER_ADDRESS_DATA_TYPE_ERROR

項目	説明
メッセージ	データ型が不正 ※次の情報を含みます。 ・レジスタ名

項目	説明
	・レジスタアドレス ・操作種別
内容	設定項目「データ型」に、PLC側の設定とは異なるデータ型が指定されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. PLCの設定で、手順1で確認したレジスタのデータ型を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「データ型」に、手順2で確認したデータ型を指定して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.3.6 REGISTER_ADDRESS_SIZE_ERROR

項目	説明
メッセージ	連続領域数が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「連続領域数」に、PLC側の設定とは異なる連続領域数が入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. PLCの設定で、手順1で確認したレジスタの連続領域数を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「連続領域数」に、手順2で確認した連続領域数を入力して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.3.7 REGISTER_READ_ERROR

項目	説明
メッセージ	レジスタ取得エラー ※次の情報を含みます。 ・レジスタ名

項目	説明
	・レジスタアドレス ・プロトコルエラー番号
内容	レジスタの取得に失敗しました。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレス、プロトコルエラー番号を確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したプロトコルエラー番号の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合) システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.3.8 GROUP_REGISTER_READ_ERROR

項目	説明
メッセージ	レジスター括取得エラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・プロトコルエラー番号
内容	レジスタの一括取得に失敗しました。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレス、プロトコルエラー番号を確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したプロトコルエラー番号の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合) システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.4 Modbus TCPコレクタ

15.2.4.1 CONNECTION_REFUSED

項目	説明
メッセージ	接続が切断
内容	機器との接続が切断されました。
処置方法	(処置方法1) コンポーネントを再起動してください。 (処置方法2) 1. 機器側の設定で、通信可能なホスト・ポートを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「ホスト」「ポート」に、手順1で確認したホスト・ポートを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.4.2 NETWORK_TIMEOUT

項目	説明
メッセージ	タイムアウト ※次の情報を含みます。 ・アドレス ・ポート番号 ・設定されているタイムアウト値 ・操作種別
内容	(操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」の時間内に機器へ接続できませんでした。 (操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」の時間内に機器からデータを受信できませんでした。
処置方法	1. メッセージに含まれている操作種別を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. (手順1で確認した操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」を大きくしてください。 (手順1で確認した操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」を大きくしてください。 4. 該当のコンポーネントを再起動してください。

15.2.4.3 REGISTER_ADDRESS_ERROR

項目	説明
メッセージ	指定レジスタが不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	ビット指定できないレジスタアドレスが、ビット指定されて設定項目「アドレス」に入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、ビット指定可能なレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.4.4 REGISTER_ADDRESS_RANGE_ERROR

項目	説明
メッセージ	不正なレジスタアドレス ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「アドレス」に、機器のプロトコルでは扱えないレジスタアドレスが入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、機器プロトコルで扱えるレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。 ※利用できるアドレス、その範囲については、該当機器のマニュアル等を参照して下さい。

15.2.4.5 REGISTER_ADDRESS_DATA_TYPE_ERROR

項目	説明
メッセージ	データ型が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「データ型」に、機器側の設定とは異なるデータ型が指定されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 機器の設定で、手順1で確認したレジスタのデータ型を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「データ型」に、手順2で確認したデータ型を指定して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.4.6 REGISTER_ADDRESS_SIZE_ERROR

項目	説明
メッセージ	連続領域数が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「連続領域数」に、機器側の設定とは異なる連続領域数が入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 機器の設定で、手順1で確認したレジスタの連続領域数を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「連続領域数」に、手順2で確認した連続領域数を入力して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.4.7 REGISTER_READ_ERROR

項目	説明
メッセージ	レジスタ取得エラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・プロトコルエラー番号
内容	レジスタの取得に失敗しました。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレス、プロトコルエラー番号を確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したプロトコルエラー番号の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合) システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.4.8 GROUP_REGISTER_READ_ERROR

項目	説明
メッセージ	レジスター括取得エラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・プロトコルエラー番号
内容	レジスタの一括取得に失敗しました。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレス、プロトコルエラー番号を確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したプロトコルエラー番号の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合) システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.5 EtherNet/IPコレクタ

15.2.5.1 TYPE_MISMATCH

項目	説明
メッセージ	型不一致 ※次の情報を含みます。 ・タグ名 ・設定の型 ・読み出した型
内容	設定項目「データ型」に、機器側の設定とは異なるデータ型が指定されています。
処置方法	1. メッセージに含まれているタグ名、設定の型、読み出した型を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したタグの設定項目「データ型」に、手順1で確認した読みだしたタグの型を指定して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.5.2 UNKNOWN_TAG

項目	説明
メッセージ	指定タグ読み取り不可 ※次の情報を含みます。 ・タグ名
内容	指定されたタグのデータが取得できません。 ※設定項目「設定方法」が「変数」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているタグ名を確認してください。 2. 機器側の設定で、正しいタグ名を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したタグの設定項目「変数名」に、手順2で確認したタグ名を入力して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.5.3 CIP_SETTING_ERROR

項目	説明
メッセージ	CIPオブジェクト設定エラー ※次の情報を含みます。 ・クラスID ・インスタンスID ・アトリビュートID
内容	指定されたCIPオブジェクトのデータが取得できません。 ※設定項目「設定方法」が「CIPオブジェクト」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているクラスID、インスタンスID、アトリビュートIDを確認してください。 2. 機器側の設定で、正しいクラスID、インスタンスID、アトリビュートIDを確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したCIPオブジェクトの設定項目「クラスID」「インスタンスID」「アトリビュートID」に、手順2で確認した情報を入力して保存してください。 5. 該当のコンポーネントを再起動してください。 ※アクセスできるCIPオブジェクトについては、該当機器のマニュアル等を参照して下さい。

15.2.5.4 ARRAYSIZE_MISMATCH

項目	説明
メッセージ	配列サイズエラー ※次の情報を含みます。 ・カラム名
内容	受信したデータの配列数が、設定項目「配列数」に入力した配列数を下回っています。
処置方法	1. メッセージに含まれているカラム名を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したカラムの設定項目「配列数」に、より小さい数を入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.5.5 READ_TIME_OVER

項目	説明
メッセージ	読み込み時間オーバー ※次の情報を含みます。 ・読み込みにかかった時間
内容	設定項目「取得間隔」に入力した間隔よりも、データの読み込みに時間がかかっています。
処置方法	1. メッセージに含まれている、読み込みにかかった時間を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「取得間隔」に、手順1で確認した時間より大きい時間を入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.6 演算コレクタ

15.2.6.1 EXPRESSION_FORMAT_ERROR

項目	説明
メッセージ	計算式のフォーマットが不正 ※次の情報を含みます。 ・出力名
内容	設定項目「演算式」に、不正なフォーマットの演算式が入力されています。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.2 SOME_VARIABLE_NOT_SPECIFIED

項目	説明
メッセージ	計算式内にある一部の変数が指定されていない ※次の情報を含みます。 ・出力名 ・変数名
内容	設定項目「演算式」に、未定義の変数が含まれています。

項目	説明
処置方法	(対処方法1) 1. メッセージに含まれている出力名、変数名を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「変数名」が、手順1で確認した変数名と同じである入力データを追加してください。 4. 手順1で確認した出力データの設定項目「トリガ変数」に、手順3で追加した変数を指定して保存してください。 5. 該当のコンポーネントを再起動してください。 (対処方法2) 1. メッセージに含まれている出力名、変数名を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認した出力データの設定項目「演算式」から、手順1で確認した変数を削除して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.6.3 NO_TRIGGER_SPECIFIED

項目	説明
メッセージ	いずれの変数にもトリガが設定されていない ※次の情報を含みます。 ・出力名
内容	設定項目「トリガ変数」に、変数が登録されていません。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.4 EVALUATION_ERROR

項目	説明
メッセージ	計算式の評価エラー ※次の情報を含みます。 ・出力名 ・エラーステータス
内容	設定項目「演算式」を評価できませんでした。

項目	説明
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.5 NO_VARIABLE_SPECIFIED

項目	説明
メッセージ	計算式の中に変数がひとつも指定されていない ※次の情報を含みます。 ・出力名
内容	設定項目「演算式」に、設定項目「トリガ変数」で指定された変数が含まれていません。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.6 COMPUTE_ERROR

項目	説明
メッセージ	演算処理でエラー ※次の情報を含みます。 ・出力名
内容	設定項目「演算式」の実行時にエラーが発生しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.7 OUTCOLUMN_CREATE_ERROR

項目	説明
メッセージ	出力カラムの作成に失敗 ※次の情報を含みます。 ・出力名

項目	説明
内容	設定項目「演算式」の実行結果を登録する出力カラムの生成に失敗しました。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.8 OUTCOLUMN_INSERT_ERROR

項目	説明
メッセージ	出力カラムへのデータ登録に失敗 ※次の情報を含みます。 ・出力名
内容	設定項目「演算式」の実行結果を出力カラムに登録できませんでした。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.6.9 VARIABLE_NOT_UNIQUE_ERROR

項目	説明
メッセージ	変数設定が重複していて変数名が特定できない ※次の情報を含みます。 ・コンポーネント名 ・データ名 ・統計タイプ
内容	設定項目「入力データ」に、定義が重複している変数が存在します。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.7 MQTTコレクタ

15.2.7.1 AUTHENTICATION_FAILURE

項目	説明
メッセージ	認証エラー
内容	MQTTブローカーへの認証が失敗しました。
処置方法	(設定項目「匿名アクセス」をOFFにする場合) 1. MQTTブローカー側の設定で、正しいユーザー情報(ユーザー名・パスワード)を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「匿名アクセス」をOFFにしてください。 4. 設定項目「ユーザー名」「パスワード」に、手順1で確認したユーザー名・パスワードを入力して保存してください。 5. 該当のコンポーネントを再起動してください。 (設定項目「匿名アクセス」をONにする場合) 1. MQTTブローカー側の設定で、匿名アクセスが許可されていることを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「匿名アクセス」をONにして保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.7.2 MESSAGE_PARSE_ERROR

項目	説明
メッセージ	受信メッセージ解析エラー
内容	想定とは異なるフォーマットのデータを受信しました。
処置方法	Publishしている機器側の設定で、次のフォーマットのデータを送信するようにしてください。 { 'records' : [{ 'timestamp': 'タイムスタンプ(ISO 8601形式)' },{ 'データ名': '値' }]} ※1 JSON形式で、recordsというキーに対して配列形式でデータを定義してください。 ※2 配列の中に、{ 'データ名': '値' }という形式でデータを定義してください。 ※3 タイムスタンプの設定は任意です。 設定する場合は、{ 'timestamp': 'タイムスタンプ(ISO 8601形式)' }でタイムスタンプを定義してください。 設定しない場合は、自動的にタイムスタンプが付与されます。この場合、SpeeDBee Synapseの稼働するマシンのシステム時刻が使用されます。

15.2.7.3 UNSUPPORTED_DATA_TYPE

項目	説明
メッセージ	未対応のデータ型
内容	受信したデータに、MQTTコレクタが対応していない型の値が含まれていました。
処置方法	Publishしている機器側の設定で、次の種類のデータを送信するようにしてください。 ・整数値 ・小数値 ・文字 ・真偽値

15.2.7.4 TIMESTAMP_ERROR

項目	説明
メッセージ	タイムスタンプエラー ※次の情報を含みます。 ・前回登録時間 ・エラー登録時間
内容	前回受信したデータのタイムスタンプよりも、今回受信したデータのタイムスタンプの方が古い時刻です。
処置方法	1. メッセージに含まれている前回登録時間を確認してください。 2. Publishしている機器側の設定で、手順1で確認した前回登録時間よりも新しいタイムスタンプのデータを送信するようにしてください。

15.2.7.5 TIMESTAMP_RANGE_ERROR

項目	説明
メッセージ	タイムスタンプ許容範囲エラー ※次の情報を含みます。 ・タイムスタンプ ・許容最小時刻 ・許容最大時刻を含む
内容	受信したデータ内で定義されているタイムスタンプと実タイムスタンプの誤差が、設定項目「タイムスタンプ誤差許容範囲」の設定値より大きいです。

項目	説明
処置方法	(処置方法1) Publishしている機器側の設定で、タイムスタンプを含めないデータを送信するようにしてください。 (処置方法2) 1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「タイムスタンプ誤差許容範囲」を確認してください。 3. Publishしている機器側の設定で、実際の時刻との誤差が手順2で確認した許容範囲内に収まるタイムスタンプを持つデータを送信してください。 (処置方法3) 1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「タイムスタンプ誤差許容範囲」に、より大きい値を入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.7.6 TIMESTAMP_PARSE_ERROR

項目	説明
メッセージ	タイムスタンプパースエラー
内容	受信したデータ内で定義されているタイムスタンプが、想定するフォーマットとは異なりました。
処置方法	Publishしている機器側の設定で、ISO 8601形式のタイムスタンプを持つデータを送信するようにしてください。
備考	ISO 8601形式のタイムスタンプの例は次の通りです。 2023-10-30T10:00:00.012345678Z

15.2.7.7 RECEIVED_PARAMETER_ERROR

項目	説明
メッセージ	受信データエラー ※次の情報を含みます。 ・カラム名(またはキー名称) ・エラー要因
内容	受信したデータ内で定義されているパラメータに問題がありました。 エラー要因の種類は次の通りです。 ・data_type：作成されたカラムの型と受信したデータの型が異なる

項目	説明
	・array_size：作成されたカラムの配列数と受信したデータの配列数が異なる ・value：受信したデータの値がカラムの型の範囲外 ・key_name：設定項目「動的カラム」がONで、コレクタが受信したJSONデータをもとにカラムを生成する際、キーがカラム名に使用できない(不正な文字、文字数上限を超えているなど)
処置方法	(エラー要因がdata_typeの場合) 1. メッセージに含まれているカラム名を確認してください。 2. Publishしている機器側の設定で、手順1で確認したカラムのデータ型を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したカラムの設定項目「データ型」に、手順2で確認したデータ型を指定して保存してください。 5. 該当のコンポーネントを再起動してください。 (エラー要因がarray_sizeの場合) 1. メッセージに含まれているカラム名を確認してください。 2. Publishしている機器側の設定で、手順1で確認したカラムの配列数を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したカラムの設定項目「データ数」に、手順2で確認した配列数を入力して保存してください。 5. 該当のコンポーネントを再起動してください。 (エラー要因がvalueの場合) 1. メッセージに含まれているカラム名を確認してください。 2. Publishしている機器側の設定で、手順1で確認したデータの値を、データ型の範囲内に収まるように変更してください。 (エラー要因がkey_nameの場合) 1. メッセージに含まれているキー名称を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「データ名」に入力できる名前を確認してください。 4. Publishしている機器側の設定で、手順1で確認したカラムの名前を、手順3で確認した名前に変更してください。

15.2.7.8 CA_FILE_ERROR

項目	説明
メッセージ	証明書ファイルエラー
内容	設定項目「CA証明書」にアップロードされた証明書が不正です。

項目	説明
処置方法	1. MQTTブローカー側の設定で、正しいCA証明書を取得してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「CA証明書」に、手順1で取得した証明書をアップロードして保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.8 MQTTエミッタ

15.2.8.1 AUTHENTICATION_FAILURE

項目	説明
メッセージ	認証エラー
内容	MQTTブローカーへの認証が失敗しました。
処置方法	(設定項目「匿名アクセス」をOFFにする場合) 1. MQTTブローカー側の設定で、正しいユーザー情報(ユーザー名・パスワード)を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「匿名アクセス」をOFFにしてください。 4. 設定項目「ユーザー名」「パスワード」に、手順1で確認したユーザー名・パスワードを入力して保存してください。 5. 該当のコンポーネントを再起動してください。 (設定項目「匿名アクセス」をONにする場合) 1. MQTTブローカー側の設定で、匿名アクセスが許可されていることを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「匿名アクセス」をONにして保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.8.2 UNSUPPORTED_DATA_TYPE

項目	説明
メッセージ	未対応のデータ型
内容	送信するデータの中に、未対応のデータ型のものが含まれています。

項目	説明
処置方法	1. 該当のコンポーネントの入力ポートを右クリックしてください。 2. 設定項目「設定方法」に「データ定義を受信して設定する」を指定してください。 3. データ一覧の中から、MQTTエミッタが対応していない型のデータを確認してください。 4. 手順3で確認したデータを送信しているコンポーネントに対して、次のいずれかの処置を行ってください。 ・フローリンクを削除する ・フローリンクのフィルター機能で、手順3で確認したデータが通過しないように設定する ・コンポーネントを停止もしくは削除する ・手順3で確認したデータを送信しないようにコンポーネントの設定を行う
備考	MQTTエミッタが対応しているデータ型は次の通りです。 ・BOOLEAN ・INT8 ・INT16 ・INT32 ・INT64 ・UINT8 ・UINT16 ・UINT32 ・UINT64 ・FLOAT ・DOUBLE ・STRING ・BINARY ・FILE

15.2.8.3 PUBLISH_ERROR

項目	説明
メッセージ	送信エラー
内容	データの送信に失敗しました。
処置方法	(対処方法1) SpeeDBee Synapseが稼働しているマシンが、ネットワークに接続されていることを確認してください。 (対処方法2) 1. MQTTブローカー側の設定で、通信可能なホスト・ポートを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「ホスト」「ポート」に、手順1で確認したホスト・ポートを入力して保存してくださ

項目	説明
	い。 4. 該当のコンポーネントを再起動してください。 (対処方法3) システム管理者に相談し、MQTTブローカー側の設定に不備がないか確認してください。

15.2.9 Fileエミッタ

15.2.9.1 UNSUPPORTED_DATA_TYPE

項目	説明
メッセージ	未対応のデータ型
内容	送信するデータの中に、未対応のデータ型のものが含まれています。
処置方法	1. 該当のコンポーネントの入力ポートを確認してください。 2. 次のものではないコンポーネントとフローリンクでつながっている場合、そのフローリンクを削除してください。 ・CSVシリアライザ ・JSONシリアライザ

15.2.10 Cloudエミッタ(AWS用)

15.2.10.1 AUTHENTICATION_FAILURE

項目	説明
メッセージ	認証エラー
内容	AWS IoT Coreへの認証が失敗しました。
処置方法	1. AWS IoT Coreの設定で、正しいホスト・ポートを確認し、正しいCA証明書・SSL証明書・秘密鍵を取得してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 次の設定項目に、手順1で確認した情報を入力し、取得したファイルをアップロードして保存してください。 ・ホスト ・ポート

項目	説明
	・CA証明書 ・SSL証明書 ・秘密鍵 4. 該当のコンポーネントを再起動してください。

15.2.10.2 PUBLISH_ERROR

項目	説明
メッセージ	送信エラー
内容	AWS IoT Coreへのデータの送信に失敗しました。
処置方法	(対処方法1) SpeeDBee Synapseが稼働しているマシンが、ネットワークに接続されていることを確認してください。 (対処方法2) 1. AWS IoT Coreの設定で、正しいホスト・ポートを確認し、正しいCA証明書・SSL証明書・秘密鍵を取得してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 次の設定項目に、手順1で確認した情報を入力し、取得したファイルをアップロードして保存してください。 ・ホスト ・ポート ・CA証明書 ・SSL証明書 ・秘密鍵 4. 該当のコンポーネントを再起動してください。 (対処方法3) システム管理者に相談し、AWS Iot Core側の設定に不備がないか確認してください。

15.2.10.3 UNSUPPORTED_DATA_TYPE

項目	説明
メッセージ	未対応のデータ型
内容	送信するデータの中に、未対応のデータ型のものが含まれています。

項目	説明
処置方法	1. 該当のコンポーネントの入力ポートを確認してください。 2. 次のものではないコンポーネントとフローリンクでつながっている場合、そのフローリンクを削除してください。 ・CSVシリアライザ ・JSONシリアライザ

15.2.11 Cloudエミッタ(Azure用)

15.2.11.1 AUTHENTICATION_FAILURE

項目	説明
メッセージ	認証エラー
内容	Azure IoT Hubへの認証が失敗しました。
処置方法	1. Azure IoT Hub側の設定で、正しい接続文字列を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「接続文字列」に、手順1で確認した接続文字列を入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.11.2 PUBLISH_ERROR

項目	説明
メッセージ	送信エラー
内容	データの送信に失敗しました。
処置方法	(対処方法1) SpeeDBee Synapseが稼働しているマシンが、ネットワークに接続されていることを確認してください。 (対処方法2) 1. Azure IoT Hub側の設定で、正しい接続文字列を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「接続文字列」に、手順1で確認した接続文字列を入力して保存してください。 4. 該当のコンポーネントを再起動してください。

項目	説明
	(対処方法3) システム管理者に相談し、Azure IoT Hub側の設定に不備がないか確認してください。

15.2.11.3 UNSUPPORTED_DATA_TYPE

項目	説明
メッセージ	未対応のデータ型
内容	送信するデータの中に、未対応のデータ型のものが含まれています。
処置方法	1. 該当のコンポーネントの入力ポートを確認してください。 2. 次のものではないコンポーネントとフローリンクでつながっている場合、そのフローリンクを削除してください。 ・CSVシリアライザ ・JSONシリアライザ

15.2.12 FTP/FTPS/SFTPエミッタ

15.2.12.1 AUTHENTICATION_FAILURE

項目	説明
メッセージ	認証エラー
内容	FTP/FTPS/SFTPサーバーへの認証が失敗しました。
処置方法	(設定項目「匿名アクセス」をOFFにする場合) 1. FTP/FTPS/SFTPサーバー側の設定で、正しいユーザー情報(ユーザー名・パスワード)を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「匿名アクセス」をOFFにしてください。 4. 設定項目「ユーザー名」「パスワード」に、手順1で確認したユーザー名・パスワードを入力して保存してください。 5. 該当のコンポーネントを再起動してください。 (設定項目「匿名アクセス」をONにする場合) 1. FTP/FTPS/SFTPサーバー側の設定で、匿名アクセスが許可されていることを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。

項目	説明
	3. 設定項目「匿名アクセス」をONにして保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.12.2 SEND_ERROR

項目	説明
メッセージ	送信エラー
内容	データの送信に失敗しました。
処置方法	(対処方法1) SpeeDBee Synapseが稼働しているマシンが、ネットワークに接続されていることを確認してください。 (対処方法2) 1. FTP/FTPS/SFTPサーバー側の設定で、通信可能なホスト・ポートを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「ホスト」「ポート」に、手順1で確認したホスト・ポートを入力して保存してください。 4. 該当のコンポーネントを再起動してください。 (対処方法3) システム管理者に相談し、FTP/FTPS/SFTPサーバー側の設定に不備がないか確認してください。

15.2.12.3 PERMISSION_ERROR

項目	説明
メッセージ	ディレクトリ操作エラー
内容	FTP/FTPS/SFTPサーバー上のディレクトリの操作に失敗しました。(権限エラーなど)
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「出力先ディレクトリ」に入力されているディレクトリを確認してください。 3. システム管理者に相談し、FTP/FTPS/SFTPサーバー上で、手順3で確認したディレクトリに対して書込み権限の付与などを行ってください。

15.2.13 JSONシリアライザ

15.2.13.1 MAX_SIZE_ERROR

項目	説明
メッセージ	上限サイズエラー
内容	ファイルサイズが設定項目「上限サイズ」を超えるため、ファイルを作成できません。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「上限サイズ」に、より大きい値を入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.14 CSVシリアライザ

15.2.14.1 MAX_SIZE_ERROR

項目	説明
メッセージ	上限サイズエラー
内容	ファイルサイズが設定項目「上限サイズ」を超えるため、ファイルを作成できません。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「上限サイズ」に、より大きい値を入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.15 イベントトリガ

15.2.15.1 IGNORE_INVALID_TIMESTAMP_DETECT

項目	説明
メッセージ	無効なタイムスタンプのデータからのイベント検知を無視した
内容	開始イベント、または終了イベントを検知後に、無効なタイムスタンプのデータから別のトリガが発生し、該当のトリガによる検知を無視した事を示すエラーです。無効なタイムスタンプとは、最初の検知時と同じタイムスタンプ、またはそれより過去のタイムスタンプのデータを示します。イベントトリガは、このようなタイムスタンプのデータからの検知を無視します。

項目	説明
処置方法	開始イベントと終了イベントが同時成立する条件式を見直してください。開始イベントと終了イベントが別のタイミングで発生する場合に、このエラーは発生しません。

15.2.16 シェルコマンド

15.2.16.1 COMMAND_EXEC_ERROR

項目	説明
メッセージ	コマンド実行エラー
内容	設定項目「コマンド」に入力されたコマンドの実行に失敗しました。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「コマンド」に、実行可能なコマンドを入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.17 PLC書き込み

15.2.17.1 CONNECTION_REFUSED

項目	説明
メッセージ	接続が切断
内容	PLCとの接続が切断されました。
処置方法	(処置方法1) コンポーネントを再起動してください。 (処置方法2) 1. PLC側の設定で、通信可能なホスト・ポートを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「ホスト」「ポート」に、手順1で確認したホスト・ポートを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.17.2 NETWORK_TIMEOUT

項目	説明
メッセージ	タイムアウト ※次の情報を含みます。 ・アドレス ・ポート番号 ・設定されているタイムアウト値 ・操作種別
内容	(操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」の時間内にPLCへ接続できませんでした。 (操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」の時間内にPLCからデータを受信できませんでした。
処置方法	1. メッセージに含まれている操作種別を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. (手順1で確認した操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」を大きくしてください。 (手順1で確認した操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」を大きくしてください。 4. 該当のコンポーネントを再起動してください。

15.2.17.3 REGISTER_ADDRESS_ERROR

項目	説明
メッセージ	指定レジスタが不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	ビット指定できないレジスタアドレスが、ビット指定されて設定項目「アドレス」に入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、ビット指定しないレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.17.4 REGISTER_ADDRESS_RANGE_ERROR

項目	説明
メッセージ	不正なレジスタアドレス ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「アドレス」に、PLCのプロトコルでは扱えないレジスタアドレスが入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、PLCプロトコルで扱えるレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。 ※利用できるアドレス、その範囲については、該当機器のマニュアル等を参照して下さい。

15.2.17.5 REGISTER_ADDRESS_DATA_TYPE_ERROR

項目	説明
メッセージ	データ型が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「データ型」に、PLC側の設定とは異なるデータ型が指定されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. PLCの設定で、手順1で確認したレジスタのデータ型を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「データ型」に、手順2で確認したデータ型を指定して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.17.6 REGISTER_ADDRESS_SIZE_ERROR

項目	説明
メッセージ	連続領域数が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「連続領域数」に、PLC側の設定とは異なる連続領域数が入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. PLCの設定で、手順1で確認したレジスタの連続領域数を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「連続領域数」に、手順2で確認した連続領域数を入力して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.17.7 REGISTER_WRITE_FIX_VALUE_ERROR

項目	説明
メッセージ	固定値が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス
内容	設定項目「書き込む値」が不正です。 ※設定項目「値種別」が「固定値」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「書き込む値」が、次の条件を満たしていることを確認して保存してください。 ・書き込む値の型が設定項目「データ型」と一致していること ・書き込む値が文字の場合、文字数が設定項目「連続領域数」以下であること 5. 該当のコンポーネントを再起動してください。

15.2.17.8 REGISTER_WRITE_NO_VALUE_ERROR

項目	説明
メッセージ	レジスタ書込み値の取得エラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス
内容	設定項目「書き込む値」に指定されたデータの取得に失敗しました。 ※設定項目「値種別」が「コンポーネント」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの入力ポートをクリックし、データモニタを開いてください。 3. データが1件以上登録されたカラムを確認してください。 4. 該当のコンポーネントの設定画面を開いてください。 5. 手順1で確認したレジスタの設定項目「書き込む値」に、手順3で確認したカラムを指定して保存してください。 6. 該当のコンポーネントを再起動してください。

15.2.17.9 REGISTER_WRITE_ERROR

項目	説明
メッセージ	レジスタ書込みエラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・プロトコルエラー番号
内容	レジスタへのWriteアクセスに失敗しました。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレス、プロトコルエラー番号を確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したプロトコルエラー番号の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合) システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.18 Modbus書き込み

15.2.18.1 CONNECTION_REFUSED

項目	説明
メッセージ	接続が切断
内容	機器との接続が切断されました。
処置方法	(処置方法1) コンポーネントを再起動してください。 (処置方法2) 1. 機器側の設定で、通信可能なホスト・ポートを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「ホスト」「ポート」に、手順1で確認したホスト・ポートを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.18.2 NETWORK_TIMEOUT

項目	説明
メッセージ	タイムアウト ※次の情報を含みます。 ・アドレス ・ポート番号 ・設定されているタイムアウト値 ・操作種別
内容	(操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」の時間内に機器へ接続できませんでした。 (操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」の時間内に機器からデータを受信できませんでした。
処置方法	1. メッセージに含まれている操作種別を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. (手順1で確認した操作種別がconnect、closeの場合) 設定項目「接続タイムアウト」を大きくしてください。 (手順1で確認した操作種別がreceive、sendの場合) 設定項目「読込タイムアウト」を大きくしてください。 4. 該当のコンポーネントを再起動してください。

15.2.18.3 REGISTER_ADDRESS_ERROR

項目	説明
メッセージ	指定レジスタが不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	ビット指定できないレジスタアドレスが、ビット指定されて設定項目「アドレス」に入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、ビット指定しないレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.18.4 REGISTER_ADDRESS_RANGE_ERROR

項目	説明
メッセージ	不正なレジスタアドレス ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「アドレス」に、機器のプロトコルでは扱えないレジスタアドレスが入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「アドレス」に、機器プロトコルで扱えるレジスタアドレスを入力して保存してください。 4. 該当のコンポーネントを再起動してください。 ※利用できるアドレス、その範囲については、該当機器のマニュアル等を参照して下さい。

15.2.18.5 REGISTER_ADDRESS_DATA_TYPE_ERROR

項目	説明
メッセージ	データ型が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「データ型」に、機器側の設定とは異なるデータ型が指定されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 機器の設定で、手順1で確認したレジスタのデータ型を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「データ型」に、手順2で確認したデータ型を指定して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.18.6 REGISTER_ADDRESS_SIZE_ERROR

項目	説明
メッセージ	連続領域数が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・操作種別
内容	設定項目「連続領域数」に、機器側の設定とは異なる連続領域数が入力されています。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 機器の設定で、手順1で確認したレジスタの連続領域数を確認してください。 3. 該当のコンポーネントの設定画面を開いてください。 4. 手順1で確認したレジスタの設定項目「連続領域数」に、手順2で確認した連続領域数を入力して保存してください。 5. 該当のコンポーネントを再起動してください。

15.2.18.7 REGISTER_WRITE_FIX_VALUE_ERROR

項目	説明
メッセージ	固定値が不正 ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス
内容	設定項目「書き込む値」が不正です。 ※設定項目「値種別」が「固定値」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認したレジスタの設定項目「書き込む値」が、次の条件を満たしていることを確認して保存してください。 ・書き込む値の型が設定項目「データ型」と一致していること ・書き込む値が文字の場合、文字数が設定項目「連続領域数」以下であること 5. 該当のコンポーネントを再起動してください。

15.2.18.8 REGISTER_WRITE_NO_VALUE_ERROR

項目	説明
メッセージ	レジスタ書込み値の取得エラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス
内容	設定項目「書き込む値」に指定されたデータの取得に失敗しました。 ※設定項目「値種別」が「コンポーネント」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレスを確認してください。 2. 該当のコンポーネントの入力ポートをクリックし、データモニタを開いてください。 3. データが1件以上登録されたカラムを確認してください。 4. 該当のコンポーネントの設定画面を開いてください。 5. 手順1で確認したレジスタの設定項目「書き込む値」に、手順3で確認したカラムを指定して保存してください。 6. 該当のコンポーネントを再起動してください。

15.2.18.9 REGISTER_WRITE_ERROR

項目	説明
メッセージ	レジスタ書込みエラー ※次の情報を含みます。 ・レジスタ名 ・レジスタアドレス ・プロトコルエラー番号
内容	レジスタへのWriteアクセスに失敗しました。
処置方法	1. メッセージに含まれているレジスタ名、レジスタアドレス、プロトコルエラー番号を確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したプロトコルエラー番号の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合) システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.19 EtherNet/IP書き込み

15.2.19.1 WRITE_ERROR

項目	説明
メッセージ	書込みエラー ※次の情報を含みます。 ・エラー要因 ・変数名 or クラスID ・インスタンスID ・アトリビュートID
内容	書込みに失敗しました。
処置方法	1. メッセージに含まれているエラー要因、変数名、クラスID、インスタンスID、アトリビュートIDを確認してください。 2. 機器の通信仕様のマニュアルに、手順1で確認したエラー要因の対処方法が存在するか確認してください。 3. (対処方法が存在する場合) 対処方法を実施してください。 (対処方法が存在しない場合、または対処方法に関する説明がよく分からない場合)

項目	説明
	システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.19.2 UNSUPPORTED_FUNCTION_ERROR

項目	説明
メッセージ	サポートしていない機能
内容	現状サポートしていない機能です。CIPObjectでの書き込みはサポートしていません。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 変数の設定項目「設定方法」が、次の条件を満たしていることを確認して保存してください。 ・「変数名」となっていること（「CIPObject」での書き込みはサポートしていません。） 3. 該当のコンポーネントを再起動してください。

15.2.19.3 FIX_DATA_TYPE_INCOLLECT

項目	説明
メッセージ	固定値書込みのデータ型不一致 ※次の情報を含みます。 ・データ型 ・書込む値
内容	固定値の書込む値が、書込み先のデータ型と一致していません。 ※設定項目「値種別」が「固定値」の場合に起こるエラーです。
処置方法	1. メッセージに含まれているデータ型、書込む値を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認した変数の設定項目「書込む値」が、次の条件を満たしていることを確認して保存してください。 ・書き込む値の型が設定項目「データ型」と一致していること 4. 該当のコンポーネントを再起動してください。

15.2.19.4 STRUCT_ERROR

項目	説明
メッセージ	指定した構造体が存在しない ※次の情報を含みます。 ・変数名
内容	指定した構造体が存在しません。
処置方法	1. メッセージに含まれている変数名を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 手順1で確認した変数の設定項目「変数名」が、次の条件を満たしていることを確認して保存してください。 ・指定した変数名が正しいこと ・指定した変数が構造体であること 4. 該当のコンポーネントを再起動してください。

15.2.20 メール送信

15.2.20.1 SMTP_AUTHENTICATION_ERROR

項目	説明
メッセージ	SMTP認証エラー ※次の情報を含みます。 ・ユーザー名 ・パスワード
内容	SMTPサーバーへの認証が失敗しました。
処置方法	1. SMTPサーバー側の設定で、正しいSMTPユーザー情報(ユーザー名・パスワード)を確認してください。 2. 該当のコンポーネントの設定画面を開いてください。 3. 設定項目「SMTPユーザー名」「SMTPパスワード」に、手順1で確認したユーザー名・パスワードを入力して保存してください。 4. 該当のコンポーネントを再起動してください。

15.2.21 コアインターナル

15.2.21.1 DELETE_STORAGE_GROUPS_ERROR

項目	説明
メッセージ	過去データ削除エラー
内容	永続化されたカラムの古いデータの削除に失敗しました。ストレージの異常などの可能性があります。
処置方法	システム管理者に相談し、ストレージの状態などを確認してください。(故障の可能性など)

15.2.21.2 DELETE_FILE_TYPE_OLD_DATA_ERROR

項目	説明
メッセージ	ファイル型カラムデータ削除エラー
内容	ファイル型カラムの古いデータの削除に失敗しました。ストレージの異常などの可能性があります。 （ファイル型カラムとは、シリアライザが保存するファイルを指します。）
処置方法	システム管理者に相談し、ストレージの状態などを確認してください。(故障の可能性など)

15.2.21.3 ACCESS_FILE_TYPE_DATA_DIR_ERROR

項目	説明
メッセージ	ファイル型カラムデータ削除エラー ※次の情報を含みます。 ・ファイル数
内容	ディレクトリにアクセスできないため、ファイル型カラムの古いデータの削除に失敗しました。
処置方法	システム管理者に相談し、ストレージの状態などを確認してください。(故障の可能性など)

15.2.21.4 SYNC_COLUMNS_TO_STORAGE_ERROR

項目	説明
メッセージ	永続化データ書き込みエラー

項目	説明
内容	永続化されたカラムのストレージ書き込みに失敗しました。ストレージの異常などの可能性があります。
処置方法	システム管理者に相談し、ストレージの状態などを確認してください。(故障の可能性など)

15.2.21.5 CORE_INTERRUPT

項目	説明
メッセージ	前回実行時異常終了検出 ※次の情報を含みます。 ・コア開始時刻 ・最終状態 ・最終状態遷移時刻
内容	SpeeDBee Synapseが稼働しているマシンの電源が落ちるなどして、正常な終了手続きを行わずにコアが停止しました。
処置方法	SpeeDBee Synapseに登録されているコンポーネントが、正常に稼働することを確認してください。

15.2.21.6 SPEEDBEE_ERROR_ACCESS

項目	説明
メッセージ	DBアクセスエラー検出
内容	SpeeDBee Synapseの時系列データベースのファイルアクセスに失敗しました。
処置方法	(対処方法1) 頻発する場合は、「サーバー設定」を参考にデータベースを初期化してください。 (対処方法2) 対処方法1を実施しても解消しない場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.21.7 SPEEDBEE_ERROR_CORRUPT

項目	説明
メッセージ	DBファイル破損検出
内容	SpeeDBee Synapseの時系列データベースのファイルが破損しています。
処置方法	(対処方法1) 頻発する場合は、「サーバー設定」を参考にデータベースを初期化してください。 (対処方法2) 対処方法1を実施しても解消しない場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.21.8 SPEEDBEE_ERROR_INTR

項目	説明
メッセージ	DB電源断検出区分
内容	SpeeDBee Synapseが稼働しているマシンの電源が落ちるなどして、時系列データベースの操作が中断されました。
処置方法	SpeeDBee Synapseに登録されているコンポーネントが、正常に稼働することを確認してください。

15.2.21.9 SPEEDBEE_ERROR_NO_REPORTS

項目	説明
メッセージ	DB欠損ログ取得エラー
内容	SpeeDBee Synapseの時系列データベースの欠損検知ログの取得に失敗しました。
処置方法	(対処方法1) 頻発する場合は、「サーバー設定」を参考にデータベースを初期化してください。 (対処方法2) 対処方法1を実施しても解消しない場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.22 DBクエリサービス

15.2.22.1 PARAMETER_ERROR_IPADDRESS

項目	説明
メッセージ	IPアドレス 設定エラー
内容	設定項目「アドレス」が未入力です。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「アドレス」に、IPアドレスを入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.22.2 PARAMETER_ERROR_PORT_NUMBER

項目	説明
メッセージ	ポート番号 設定エラー
内容	設定項目「ポート」が未入力です。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「ポート」に、ポート番号を入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.22.3 PARAMETER_ERROR_HTTPS_CERTIFICATION

項目	説明
メッセージ	HTTPS証明書エラー
内容	設定メニュー「HTTP/HTTPS通信」の設定項目「SSL/TLS証明書」にアップロードされたHTTPS証明書ファイルが存在しません。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.22.4 PARAMETER_ERROR_HTTPS_SECRETKEY

項目	説明
メッセージ	HTTPS秘密鍵エラー
内容	設定メニュー「HTTP/HTTPS通信」の設定項目「秘密鍵」にアップロードされたHTTPS秘密鍵ファイルが存在しません。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.22.5 PARAMETER_ERROR_JWS_SETTING

項目	説明
メッセージ	JWS検証の有効無効 設定エラー
内容	設定項目「トークン認証」が未指定です。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.22.6 PARAMETER_ERROR_JWS_VERIFICATION

項目	説明
メッセージ	JWS検証用ファイルエラー
内容	JWS検証用のファイルに不備があります。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.22.7 DB_QUERY_HTTP_SERVER_START_ERROR

項目	説明
メッセージ	DBクエリサービス 起動エラー

項目	説明
内容	HTTPサーバーの起動に失敗しました。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「ホスト」「ポート」を確認してください。 3. システム管理者に相談し、手順2で確認したホスト・ポートが、他のサービスと競合していないか確認してください。

15.2.23 エラーマネージャー

15.2.23.1 PARAMETER_ERROR_INTERVAL

項目	説明
メッセージ	処理間隔 設定エラー
内容	設定項目「処理間隔」に不正な値が設定されています。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「処理間隔」に、5から180までの値を入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.23.2 PARAMETER_ERROR_CORE_PORT

項目	説明
メッセージ	コアポート番号 設定エラー
内容	コアのポート番号に不正な値が設定されています。
処置方法	通常は発生しないエラーです。システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.23.3 NETWORK_TIMEOUT_ERROR

項目	説明
メッセージ	通信タイムアウトエラー
内容	通信タイムアウトが発生しました。
処置方法	1. 該当のコンポーネントの設定画面を開いてください。 2. 設定項目「タイムアウト」に、より大きい値を入力して保存してください。 3. 該当のコンポーネントを再起動してください。

15.2.24 イベントデータマネージャー

15.2.24.1 RESTORE_DB

項目	説明
メッセージ	データベースの破損により、バックアップからデータを復元
内容	i-Triggerの内部データベースアクセス時に、破損を検知し、バックアップされているデータベースからデータを復元したことを知らせる警告です。
処置方法	特別な処置は必要ありません。ただし、一部のデータは消失したことになります。

15.2.25 VIDEOコレクタ（i-Trigger 拡張パッケージ）

15.2.25.1 READ_FRAME_ERROR

項目	説明
メッセージ	フレームの読み込みに失敗
内容	ネットワークカメラから受信したストリーミングデータの読み込みに失敗しました。
処置方法	通常は、処置は必要ありません。ただし、このエラーが解消しない場合は、次の点を確認してください。 1. 接続しているカメラが、i-Triggerに対応しているカメラか確認する。（参照） 2. カメラ側の設定を見直してください。解像度を低くしたり、フレームレートの数字を下げるなどをお試しください。

15.2.25.2 MAKE_CONTAINER_ERROR

項目	説明
メッセージ	mp4の作成中にエラーが発生
内容	mp4ファイルの作成中に、内部エラーが発生しました。
処置方法	通常は発生しないエラーです。症状が続く場合は、システム管理者に相談し、「製造元への問い合わせ」の手順に従って製造元に問い合わせください。

15.2.26 Event Recorder（i-Trigger 拡張パッケージ）

このコンポーネントで発生するエラーは、すべて「コンポーネント共通」のものとなります。

15.3 カスタムコンポーネントPython-APIリファレンス

カスタムコンポーネントをPython言語で開発する際のAPI仕様です。 カスタムコンポーネントの開発手順については下記を参照してください。

Pythonカスタムコンポーネント作成

15.3.1 speedbeesynapse.component.base

Hiveコンポーネントベースモジュール

このモジュールはSpeeDBee Synapseのコンポーネントを Pythonで実装する場合にインポートするモジュールです。

15.3.2 HiveComponentBase Objects

```
class HiveComponentBase()
```

コンポーネント用基底クラス

すべてのコンポーネントはこのクラスを継承して定義する必要があります。

Attributes:

- `in_portX` *HiveInPort* - 入力ポートX番を表すインスタンス変数。実際にはin_port1, in_port2のように、`X`には数字が入ります。
- `out_portX` *HiveOutPort* - 出力ポートX番を表すインスタンス変数。実際にはout_port1, out_port2のように、`X`には数字が入ります。

15.3.2.1 premain

```
def premain(param: str | dict) -> typing.Optional[ErrorInfoBase]
```

コンポーネントメイン処理直前に呼ばれる関数

コンポーネントの開始時、メイン処理の直前に呼ばれます。 ここでパラメータを読み取ってカラムの作成や独自の初期化処理等を行うことが可能です。

ただし、この関数における処理は全コンポーネント内で排他制御が掛けられますので長時間処理に時間がかかると、その間、他のコンポーネントのpremainが動作できないことに留意してください。その代わり、ここでカラムを作成する場合はメイン処理で作成する場合に比べて高速に動作します。

正常に初期化処理等が終了した場合は、何も返さず本メソッドを終了してください。その後`main`メソッドがコールされます。

何らかの理由により、コンポーネントの実行を継続できない場合は、ErrorInfoBaseを継承したクラスのインスタンスを返却してください。それにより画面上にもエラーの内容が表示され、コンポーネントは停止します。
（`main`,`postmain`メソッドは呼ばれません）

このメソッドの定義は必須ではありません。

Arguments:

- `param` *str | dict* - コンポーネントへのパラメータ

Returns:

- `None` - 正常に終了した
- `ErrorInfoBase` - なんらかの理由でコンポーネントを開始できない場合、ErrorInfoBaseを継承したエラーオブジェクトを返す

15.3.2.2 main

```
def main(param: str | dict)
```

コンポーネントメイン処理関数

コンポーネントのメイン処理を実装するためのメソッドです。 HiveComponentBaseを継承したクラスではこのメソッドを必ずオーバーライドして、実装してください。

この関数の終了は、コンポーネントの処理が完了したことを意味します。正常に終了する際は何も返す必要はありません。 エラーにより終了する場合は、ErrorInfoBaseを継承したクラスのインスタンスを返してください。返されたインスタンスの情報は画面上に停止理由として表示されます。

このメソッドの外に例外が送出された場合、不明なエラーとしてコンポーネントは停止します。その挙動が望ましくない場合は、例外をキャッチして処理を継続するか、ErrorInfoBaseを継承したクラスのインスタンスを返して本メソッドを終了してください。

Arguments:

- `param` *str | dict* - コンポーネントへのパラメータ

Returns:

- `None` - 正常に終了した
- `ErrorInfoBase` - なんらかの理由でコンポーネントを開始できない場合、ErrorInfoBaseを継承したエラーオブジェクトを返す

15.3.2.3 postmain

```
def postmain(param: str | dict)
```

コンポーネントメイン処理直後に呼ばれる関数

コンポーネントの停止時など、メイン処理関数が終了した直後に呼ばれます。
メイン処理がエラー等で終了した場合も必ず呼ばれるため、使用したリソースの解放等を行うことを想定しています。
このメソッドの定義は必須ではありません。

Arguments:

- `param` *str | dict* - コンポーネントへのパラメータ

15.3.2.4 is_runnable

```
def is_runnable()
```

コンポーネントの実行継続可否判定

このコンポーネントのメイン処理を継続してよいかを判定します。 コンポーネントは定期的（少なくとも5秒以内）にこの関数をコールし、メインの処理を継続してよいか確認する必要があります。 この関数がTrueを返すときはそのままメイン処理を継続することができますが、 Falseを返すときは、速やかにメイン処理を終了する必要があります。

Returns:

- Trueなら継続可能、Flaseならコンポーネントの停止要求が出ている

15.3.2.5 get_timestamp

```
def get_timestamp() -> int
```

現在時刻の取得

ナノ秒単位のUNIXタイムで表現された現在時刻を取得します。

Returns:

- 現在時刻のタイムスタンプ。

15.3.2.6 notify_stop

```
def notify_stop()
```

コンポーネントの停止要求通知

このコンポーネントがシステムから停止要求を受けた場合にコールされます。
HiveComponentBaseを継承したクラスでは必要に応じてこのメソッドをオーバーライドしてください。
メソッドがコールされた場合には、速やかにコンポーネントのメイン処理を停止させる必要があります。

メイン処理内で`is_runnable()`を定期的にコールして継続可否をチェックするのであれば、この関数を実装する必要はありません。

15.3.2.7 register_status

```
def register_status(status: RunningStatus)
```

ステータス情報の登録

エラー情報が追加されたステータスを登録します

Arguments:

- `status` - エラー情報が登録されたステータスオブジェクト

15.3.2.8 Status

```
def Status()
```

ステータス情報の生成

エラー情報を登録するためのステータスオブジェクトを生成します。

15.3.2.9 interval_iteration

```
def interval_iteration(interval, base_time=0)
```

定期処理用イテレータ

コンポーネントにて一定時間間隔で何らかの処理をするためのイテレータです。
指定された基準時刻より、指定した定期処理間隔でイテレータから値がyieldされます。ただし実際に処理が開始されるタイミングは厳密に予定時刻と一致するわけではなく、OSやデバイスの負荷やクロック精度に影響されます。

また、ユーザーが実装する定期的な処理の実行に、定期処理間隔より長い時間を要した場合、次の定期処理がスキップされることがあります。この場合、スキップされた定期処理の回数がyieldの第二要素として受け取れます。

本関数はコンポーネントの停止要求を受けた場合にはイテレーションを終了します。

Arguments:

- `interval` - 定期処理間隔（単位：ナノ秒）
- `base_time` - イテレーション開始基準時刻を表すUNIXタイム（単位：ナノ秒）

Yields:

- `[0]` - その時のイテレーションが実行される予定だった時刻を表すUNIXタイム（単位：ナノ秒）
- `[1]` - 直前のイテレーションのスキップ数

Examples:

以下のようにfor文でイテレータとして使用できます。

```
for [ts, skip] in self.interval_iteration(1000000000):
    #
    # 定期的に実行したい処理
    #
```

この場合は1秒間隔で「定期的に実行したい処理」の部分が実行されます。

15.3.2.10 save_running_info

```
def save_running_info(key: str, data: bytes)
```

任意の情報を保存する

任意のバイト列をシステムに記録します。
保存されたデータは、同一のコンポーネントであればシステムの再起動後もload_running_info()で参照する事ができます。また、指定されたkeyにより異なるデータとして保存されるため、複数保存したい場合はそれぞれkeyを分けてください。

Arguments:

- `key` *str* - 保存する情報を識別するためのキー
- `data` *bytes* - bytes型の保存したい情報

Raises:

- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.2.11 load_running_info

```
def load_running_info(key: str) -> typing.Optional[bytes]
```

任意の情報を読み込む

Arguments:

- `key` *str* - 読み込みたい情報を識別するためのキー

Returns:

- `data` - byte型の読み込んだデータ
- `None` - データがない

Raises:

- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.2.12 get_common_variable

```
def get_common_variable(varname: str) -> typing.Optional[str]
```

コンポーネント共通変数取得

Arguments:

- `varname` *str* - 取得したい共通変数の変数名

Returns:

- 指定した共通変数の値
- `None` - 指定した変数が存在しない

Raises:

- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.2.13 create_lock

```
def create_lock(key: typing.Union[str, bytes],
                interbase: bool = False,
                timeout: typing.Optional[float] = None) -> SynapseLock
```

コンポーネント間での排他制御のためのロック獲得関数

本メソッドは、引数`key`で指定された文字列orバイト列で識別されるリソースの排他制御のためのロックを獲得します。 同一の`key`におけるロックは１つのスレッドでしか獲得できません。 獲得したロックは、返却されるロックオブジェクトの`unlock()`メソッドで解放できます。 もしくは、本メソッドを`with`句で使用すればスコープを抜けた際に解放されます。 また、ロックを獲得したままコンポーネントが停止した場合にも自動でロックが解放されます。

引数`interbase`がtrueの場合、同じ`key`を使うことで異なるコンポーネントベース間でも排他制御が可能になります。 この場合はC-APIの`hive_lock()`を使ったカスタムコンポーネントとの排他制御も利用可能です。 `interbase`がfalseの場合、同一のコンポーネントベースでのみ排他制御されます。

本メソッドを呼び出したときにすでに外のコンポーネントインスタンスがロックを獲得している場合、メソッド内で実行がブロックされます。 ブロックする最大時間は、引数`timeout`にて秒単位で指定可能です。タイムアウト時間を過ぎてもロックを獲得できない場合は`HiveTimeoutError`例外が送出されます。 `timeout`に`0`以下の値を指定した場合は、実行がブロックされることはなく、ロックを獲得できなければ即時タイムアウトとなります。
`timeout`にNoneを指定した場合はタイムアウトせずにロック獲得できるまで待ち続けます。

ロック獲得を待っている間に実行しているコンポーネントインスタンスが停止要求を受けた場合、 タイムアウト時間にかかわらず、HiveLockCanceled例外を送出します。

同一スレッドで同じ`key`に対して二重にロックすることはできません。その場合はロック解放待ち状態になりますのでタイムアウトするか停止要求を受けるまでブロックし続けます。

Arguments:

- `key` *str|bytes* - 排他制御対象を識別するための任意の文字列orバイト列
- `interbase` *bool* - 異なるコンポーネントベース間で排他制御を行う場合は`True`
- `timeout` *float|None* - ロックが即時獲得できない場合に解放を待つ場合のタイムアウト時間

Returns:

SynapseLock オブジェクト

Raises:

- `HiveTimeoutError` - 指定されたタイムアウト時間内にロックを獲得できなかった
- `HiveLockCanceled` - ロックを獲得する前にコンポーネントの停止要求を受けた
- `HiveApiError` - Hiveフレームワークの各種エラー

Examples:

以下のようにwith句を使う事ができます。

```
while self.is_runnable();
    try:
        with self.create_lock('lock-serial-device', False, 10.0):
            #
            # 排他制御が必要な処理
            #
    except HiveTimeoutError as exc:
        self.log.debug('wait lock')
        continue
    except HiveLockCanceled:
        return
```

15.3.3 DataType Objects

```
class DataType(IntEnum)
```

データ型定義Enumクラス

データ型を示すEnumクラスです。カラム作成時に指定するDataTypeとして、このアトリビュートを使用してください。

Attributes:

- `NONE(0)` - どの型にも一致しない未定義の型
- `BOOLEAN(1)` - 真偽値型

- `INT8(2)` - 8bit符号あり整数型
- `INT16(3)` - 16bit符号あり整数型
- `INT32(4)` - 32bit符号あり整数型
- `INT64(5)` - 64bit符号あり整数型
- `UINT8(6)` - 8bit符号なし整数型
- `UINT16(7)` - 16bit符号なし整数型
- `UINT32(8)` - 32bit符号なし整数型
- `UINT64(9)` - 64bit符号なし整数型
- `FLOAT(10)` - 32bit浮動小数点型
- `DOUBLE(11)` - 64bit浮動小数点型
- `STRING(13)` - 文字列
- `BINARY(14)` - バイナリ
- `FILE(16)` - ファイル

15.3.4 ColumnOption Objects

```
class ColumnOption()
```

カラムオプション

出力ポートのカラムに対するオプション設定

Attributes:

- `column_type` *ColumnType* - カラムタイプ(LOW、MIDDLE、HIGH)
- `samplingrate` *float* - サンプリングレート
- `compression_unit` *int* - 圧縮単位
- `fixedsize_binary` *int* - カラムデータサイズ
- `max_binarysize` *int* - LO可変長タイプのデータ最大長

15.3.5 FileTypeMetaInfo Objects

```
class FileTypeMetaInfo(typing.TypedDict)
```

ファイル型データメタ情報

ファイル型カラムに登録するファイルのメタ情報をもつクラスです。 いずれのメンバも省略可能です。

Attributes:

- `media_type` *ColumnType* - メディアタイプ(application/jsonやtext/csvなどを設定できる)
- `filename` *float* - ファイル名のヒント（ファイルエミッタに繋いだ場合にこのファイル名が使われる）
- `begin_timestamp` *int* - このファイルに収められたデータの時間範囲の先頭時刻（UNIXタイム、単位：ナノ秒）
- `end_timestamp` *int* - このファイルに収められたデータの時間範囲の末尾時刻（UNIXタイム、単位：ナノ秒）

15.3.6 OutColumn Objects

```
class OutColumn()
```

出力ポートのカラム

出力ポートのカラムに対するデータの登録等はこのクラスのインスタンスを介して行います

Attributes:

- `name` *str* - 作成時に指定したカラム名
- `data_type` *DataType* - 作成時に指定したカラムのデータ型
- `data_array` *int* - 作成時に指定したカラムの要素数 スカラ型の場合は0となります
- `options` *ColumnOption* - 作成時に指定したカラムのオプション

15.3.6.1 insert

```
def insert(*args) -> None
```

カラムへの値登録

このカラムに指定したタイムスタンプ`ts`の値を登録します。
タイムスタンプを省略した場合は現在時刻として扱われます。
dataには、このカラム作成時のデータ型と一致するデータを渡してください。異なるデータを渡した場合、暗黙に変換可能なものは変換されますが、そうでない場合はエラーとなります。

Arguments:

- `data` - 登録する値
- `ts` - この値を登録するタイムスタンプ

Raises:

- `TypeError` - このカラムのデータ型に一致しない、不正なデータが渡された
- `InvalidTimestamp` - タイムスタンプが現在時刻と著しく離れている、もしくは前に登録したタイムスタンプより過去になっている

15.3.6.2 open_file

```
@contextmanager
def open_file()
```

新規ファイルの作成

ファイル型として作成されたカラムに登録する新規のファイルをオープンし、そのファイルオブジェクトを返します。

Returns:

- `FileObject` - オープンされたファイルオブジェクト

Raises:

- `TypeError` - FILE型でないカラムに対してこのメソッドが呼ばれた
- `HiveApiError` - Hiveフレームワークの各種エラー
- `OSError` - ファイルオープンに関連する各種エラー

Examples:

以下のようにwith句で使用できます。

```
file_column = self.out_port1.Column("clmfile", DataType.FILE)
with file_column.open_file() as fo:
    fo.write("filedata - line 1\n".encode('utf-8))
    fo.write("filedata - line 2\n".encode('utf-8))
    fo.write("filedata - line 3\n".encode('utf-8))

    file_column.insert_file(fo, ts)
```

15.3.6.3 insert_file

```
def insert_file(fo: typing.IO[typing.AnyStr],
                ts: int = 0,
                meta: typing.Optional[FileTypeMetaInfo] = None) -> None
```

ファイルのDB登録

ファイルをファイル型のカラムへ登録します。

Arguments:

- `file` - オープンされたファイルオブジェクト
- `ts` - この値を登録するタイムスタンプ
- `meta` - 登録するファイルのメタ情報

Raises:

- `ValueError` - 登録しようとしたファイルがすでにクローズされている

- `HiveApiError` - Hiveフレームワークの各種エラー
- `OSError` - ファイル操作に関連する各種エラー

15.3.6.4 insert_file_move

```
def insert_file_move(pathlike,
                     ts: int = 0,
                     meta: typing.Optional[FileTypeMetaInfo] = None) -> None
```

既存ファイルのDB登録（移動）

すでに作成済みのファイルをファイル型のカラムへ登録します。 指定されたファイルは、DBの管理ディレクトリに移動されます。引数で指定したパスにはファイルが残らないことに注意してください。 ディレクトリは登録できません。

Arguments:

- `pathlike` - 登録対象のファイルのパスを表すpath-likeオブジェクト
- `ts` - この値を登録するタイムスタンプ
- `meta` - 登録するファイルのメタ情報

Raises:

- `ValueError` - 指定したパス
- `HiveApiError` - Hiveフレームワークの各種エラー
- `OSError` - ファイル操作に関連する各種エラー

15.3.6.5 insert_file_copy

```
def insert_file_copy(pathlike,
                     ts: int = 0,
                     meta: typing.Optional[FileTypeMetaInfo] = None) -> None
```

既存ファイルのDB登録（コピー）

すでに作成済みのファイルをファイル型のカラムへ登録します。 指定されたファイルはDBの管理ディレクトリにコピーされ、そのコピー先のファイルがカラムに登録されます。 以降、引数で指定した元ファイルのパスは参照されません。 ディレクトリは登録できません。

Arguments:

- `pathlike` - 登録対象のファイルのパスを表すpath-likeオブジェクト
- `ts` - この値を登録するタイムスタンプ
- `meta` - 登録するファイルのメタ情報

Raises:

- `ValueError` - 指定したパス
- `HiveApiError` - Hiveフレームワークの各種エラー
- `OSError` - ファイル操作に関連する各種エラー

15.3.6.6 insert_file_ref

```
def insert_file_ref(pathlike,
                    ts: int = 0,
                    meta: typing.Optional[FileTypeMetaInfo] = None) -> None
```

既存ファイルのDB登録（参照）

すでに作成済みのファイルをファイル型のカラムへ登録します。 指定されたファイルは、そのパスのままカラムに登録されます。 そのためファイルを削除してしまうと、このカラムの情報を他のコンポーネントが参照する際にもファイルを参照できなくなります。 ディレクトリは登録できません。

Arguments:

- `pathlike` - 登録対象のファイルのパスを表すpath-likeオブジェクト
- `ts` - この値を登録するタイムスタンプ
- `meta` - 登録するファイルのメタ情報

Raises:

- `ValueError` - 指定したパス
- `HiveApiError` - Hiveフレームワークの各種エラー
- `OSError` - ファイル操作に関連する各種エラー

15.3.7 AggregationType Objects

```
class AggregationType(IntEnum)
```

集約種別定義Enumクラス

入力ポートから取得するカラムデータの集約種別を表します

Attributes:

- `NONE(0)` - なし
- `OLDER(1)` - 最古値
- `NEWER(2)` - 最新値
- `COUNT(3)` - 件数
- `SUM(4)` - 総和

- `SUMSQ(5)` - 2乗和
- `SUMSQD(6)` - 偏差平方和
- `MIN(7)` - 最小
- `MAX(8)` - 最大
- `RANGE(9)` - 範囲
- `MEAN(10)` - 算術平均値
- `VAR(11)` - 分散
- `STDEV(12)` - 標準偏差
- `UVAR(13)` - 不偏分散
- `USTDEV(14)` - 標本不偏標準偏差
- `STDER(15)` - 標準誤差
- `CV(16)` - 変動係数
- `MEDIAN(17)` - 中央値
- `QUARTMIN(18)` - 四分位数
- `QUART1(19)` - 四分位数
- `QUART2(20)` - 四分位数
- `QUART3(21)` - 四分位数
- `QUARTMAX(22)` - 四分位数
- `MODE(23)` - 最頻値
- `MODECOUNT(24)` - 最頻値個数
- `MDEV(25)` - 平均偏差
- `HMEAN(26)` - 調和平均
- `GMEAN(27)` - 幾何平均
- `KURT(28)` - 尖度
- `SKEW(29)` - 歪度

15.3.8 AggregationTypeSet Objects

```
class AggregationTypeSet()
```

入力ポートのカラムからデータ取得する際に有効になっている統計情報を示すフラグのセット。

`AggregationType`の属性を`test()`メソッドに渡すことでこの統計情報が有効になっているかを確認できます。 また、イテレーターを使って有効になっている統計情報のリストを取得することも可能です。

Examples:

`AggregationType`の属性を`test()`メソッドに渡すことでこの統計情報が有効になっているかを確認できます。

```
if a_type_set.test(AggregationType.MAX):
  pring(max: ON")
else:
  pring(max: OFF")
```

また、イテレーターを使って有効になっている統計情報のリストを取得することも可能です。

```
for a_type in aggregation_type_set:
    print(a_type.name)
```

15.3.9 InColumn Objects

```
class InColumn()
```

入力ポートのカラム

このクラスからは情報を取得するのみです。カラムに対してデータを登録するなどの操作はできません。

Attributes:

- `source_name` *str* - カラムを生成したコンポーネントの名前
- `data_name` *str* - カラムの名前
- `data_type` *DataType* - カラムの型情報
- `array_size` *int* - カラムの配列要素数（スカラの場合は0）
- `raw_data` *bool* - カラムからのデータ取得時に未加工のデータが含まれるか否かを示す真偽値
- `stat_type_set` *AggregationTypeSet* - カラムからのデータ取得時に有効となっている統計情報を示すフラグセット
- `output_name` *str* - カラムの出力名。指定がない場合はNone

15.3.9.1 get_latest_value

```
def get_latest_value(use_storage: bool = False) -> (int, typing.Any)
```

カラムの最新値取得

カラムに登録されている最新のデータを読み込み、そのタイムスタンプと値のタプルを返します。

Arguments:

- `use_storage` *bool* - ※現時点では未サポート
  ストレージに永続化されたデータからの読み込みを行う場合にTrueを指定してください。
  Falseの場合、メモリ上からのみ読み込みを行うため、永続化されたデータが有ってもデータなしを返す可能性が

あります。
Trueの場合でも、メモリ上に最新データがあればストレージのアクセスは行いません。

Returns:

- `(0, None)`: カラムから読み込める値がない場合
- `(>0, !=None)`: 読み込まれたタイムスタンプと値

Raises:

- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.10 InPortReader Objects

```
class InPortReader()
```

入力ポートリーダー

入力ポートのカラムから、データを読み込むクラスです。
このクラスは直接インスタンス化せず、`self.in_portX.ContinuousReader()`、もしくは、`self.in_portX.TimeRangeReader()`関数を使用してください。

15.3.10.1 read

```
def read() -> typing.Optional[HiveWindowData]
```

カラムデータ読み込み

入力ポートからカラムデータを読み込みます。

Returns:

- `HiveWindowData` - 読み込んだウィンドウデータを返します。
- `None` - 現時点で読み込みできない場合はNoneが返されます。

Raises:

- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.11 HiveOutPort Objects

```
class HiveOutPort()
```

出力ポートを表すクラス

出力ポートに対するカラムの作成等はこのクラスのインスタンスを介して行います。
このクラスのインスタンスを直接生成することはできません。コンポーネントの開始時に、アトリビュート `self.out_port1` のように最初からアクセスできますのでこれを使ってください。

15.3.11.1 Column

```
def Column(name: str,
           data_type: DataType,
           data_array: int = 0,
           opt: typing.Optional[ColumnOption] = None) -> OutColumn
```

カラム生成

この出力ポート内に、指定された名称のカラムを生成します。

Arguments:

- `name` - 作成するカラム名
- `data_type` - 作成するカラムのデータ型
- `data_array` - 作成するカラムの要素数(スカラ型の場合は0を指定)
- `opt` - カラムオプション

Returns:

- OutColumn

Raises:

- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.11.2 register_confirmed_time

```
def register_confirmed_time(ts: int = 0) -> None
```

出力ポートへの確定時刻登録

この出力ポートのカラムへのデータ登録が、指定した時刻 `ts` までは確定したことを通知します。この登録により、指定した時刻 `ts` より前の時刻のデータは読み込むことができることを、フローリンクを通して繋がっている別のコンポーネントが知ることができます。

※このAPIの呼び出しは必須ではありません。

Arguments:

- `ts` - 出力ポートのカラムへのデータ登録が確定している時刻 ※省略した場合は現在時刻になります

15.3.12 HiveInPort Objects

```
class HiveInPort()
```

入力ポートを表すクラス

入力ポートからデータを取得するような処理はこのクラスのインスタンスを介して行います

15.3.12.1 get_columns

```
def get_columns() -> list[InColumn]
```

カラムリストの取得

この入力ポートに接続されている全カラムの情報を取得します

Returns:

- list[InColumn]

15.3.12.2 ContinuousReader

```
def ContinuousReader(start: int) -> InPortReader
```

継続カラムリーダーの生成

この入力ポートに接続されているカラムから、継続的にデータを読み込むContinuousReaderを生成します

Arguments:

- `start` - 入力ポートからの読み込みを開始する時刻（UNIXタイム、単位：ナノ秒）

Returns:

- ContinuousReader

Examples:

以下のようにwith句でreaderを取得して使用できます。

```
with self.in_port1.ContinuousReader(start=self.get_timestamp()) as reader:
    while self.is_runnable():
        window_data = reader.read()
        if not window_data:
            continue
            :
```

15.3.12.3 TimeRangeReader

```
def TimeRangeReader(start: int, end: int) -> InPortReader
```

時間範囲データリーダーの生成

指定したカラム、もしくは入力ポートに接続されている全カラムから、指定した時間範囲のデータを読み込むTimeRangeReaderを生成します

Arguments:

- `start` - 入力ポートからの読み込みを開始する時刻（UNIXタイム、単位：ナノ秒）
- `end` - 入力ポートからの読み込みを終了する時刻（UNIXタイム、単位：ナノ秒）
- `column` - データを取得するカラム(指定がない場合には入力ポートの全カラム)

Returns:

- data

Examples:

以下のようにwith句でreaderを取得して使用できます。

```
with self.in_port1.ContinuousReader(start=self.get_timestamp()) as reader:
    for window_data in reader:
        :
```

15.3.13 RunningStatus Objects

```
class RunningStatus()
```

15.3.13.1 add_error

```
def add_error(err: ErrorInfoBase)
```

エラー情報登録

ステータスオブジェクトにエラー情報を登録します。 このオブジェクトに登録しただけでは保存されません。最終的にはHiveComponentBase.register_statusを呼び出してシステムに登録してください。

Arguments:

- `err` - エラー情報

15.3.13.2 ErrorType

```
def ErrorType(error_id: str, *param_names: str)
```

エラー情報クラス生成

エラー情報を管理するクラスを生成します。 実際のエラー情報は、この関数で生成したクラスからインスタンスを生成してください。

Arguments:

- `error_id` - エラー情報を識別する文字列
- `param_names` - エラー情報のパラメータの名称

15.3.14 ErrorInfoBase Objects

```
class ErrorInfoBase(Exception)
```

ステータスに登録するためのエラー情報

コンポーネントのエラーの種別や詳細を保持するためのクラスです。
このクラスは、直接インスタンス化したり継承したりせずに、`ErrorType`関数を使って生成したクラスをインスタンス化してください。

Attributes:

- `errorTypeIndex` *int* - エラーパラメータのタイプ
- `params` *list[ErrorParam]* - エラーパラメータ

15.3.15 HiveLogLevel Objects

```
class HiveLogLevel(IntEnum)
```

ログレベル型定義Enumクラス

ログレベルを示すEnumクラスです。

Attributes:

- `ERROR(0)` - エラーログ
- `WARNING(1)` - 警告ログ
- `INFO(2)` - 通常情報ログ
- `DEBUG(3)` - デバッグ用ログ
- `TRACE(4)` - トレースログ

15.3.16 HiveLog Objects

```
class HiveLog()
```

ログ出力クラス

15.3.16.1 get_current_level

```
def get_current_level()
```

ログレベル取得

Returns:

- `HiveLogLevel` - 現在のログレベルEnum

15.3.16.2 error

```
def error(*msg)
```

エラーログ出力

Arguments:

- `msg` - ログ出力する文字列

15.3.16.3 warning

```
def warning(*msg)
```

警告ログ出力

Arguments:

- `msg` - ログ出力する文字列

15.3.16.4 info

```
def info(*msg)
```

通常情報ログ出力

Arguments:

- `msg` - ログ出力する文字列

15.3.16.5 debug

```
def debug(*msg)
```

デバッグログ出力

Arguments:

- `msg` - ログ出力する文字列

15.3.16.6 trace

```
def trace(*msg)
```

トレースログ出力

Arguments:

- `msg` - ログ出力する文字列

15.3.16.7 exception

```
def exception(exc: Exception)
```

例外専用エラーログ出力

Arguments:

- `exc` - ログ出力する例外

15.3.17 HiveWindowData Objects

```
class HiveWindowData()
```

ウィンドウデータクラス

入力ポートからカラムのデータを取得する際に、特定の時間単位でまとめられたデータのかたまりです。 時間単位がどの程度になるかは入力ポートの設定により異なります。

Attributes:

- `time_range` *tuple[int:2]* - このウィンドウの時間範囲を表す２要素のタプル。最初の要素が開始時刻、２番目の要素が終了時刻を表す。どちらもナノ秒単位。
- `window_id` *int* - ウィンドウを識別する数値。ウィンドウ毎に異なる値が取得されるが、常に１つずつ増加するとは限らないことに注意。
- `event_id` *int* - イベント範囲を識別する数値。イベント定義されていない場合やイベント範囲外では0に固定。イベント中は1以上の値が入る。

- `records` *list[HiveRecord]* - このウィンドウ内のレコードリスト
- `columns` *list[InColumn]* - 取得した入力ポートに接続されたカラムの一覧。HiveInPort.get_columns()で取得したものと同じ。
- `event` *HiveInPortEventSetting* - 入力ポートに設定されたイベント情報

15.3.18 HiveRecordType Objects

```
class HiveRecordType(IntEnum)
```

レコード種別Enumクラス

レコード種別を示すEnumクラスです。

Attributes:

- `RAW_DATA(0)` - 未加工データ
- `STATISTICS(1)` - 集約データ

15.3.19 HiveRecord Objects

```
class HiveRecord()
```

レコードクラス

入力ポートからカラムのデータを取得する際に、同一時刻のデータをまとめたもの。
別カラムのデータでも同一時刻のものがあれば全て１つのインスタンスにまとめられます。レコードは２種類、未加工データと集約データがあり、`record_type`で識別できます。

Attributes:

- `record_type` *HiveRecordType* - このウィンドウ内のレコードリスト
- `timestamp` *int* - このレコードに含まれるデータのタイムスタンプ。UNIXタイム（単位ナノ秒）
- `data` *list[HiveRecordData]* - このレコードのデータのリスト

15.3.20 HiveRecordData Objects

```
class HiveRecordData()
```

レコードデータクラス

１レコード内の各カラムのデータ。

Attributes:

- `value` *any* - 実データ。数値、文字列、配列など、カラムの型に応じて異なる形式。
- `column` *InColumn* - 対象カラムを示すオブジェクト
- `stat_type` *AggregationType* - このデータの集約種別

15.3.21 HiveCompError Objects

```
class HiveCompError()
```

コンポーネントエラー情報クラス

コンポーネント実行中、もしくは、エラーで停止した場合のエラー情報

Attributes:

- `error_type` *str* - コンポーネントのエラーの種別を示す文字列
- `parameters` *list[any]* - エラーの付加情報

15.3.22 HiveCompStatus Objects

```
class HiveCompStatus()
```

コンポーネントステータスクラス

コンポーネントのステータスとエラー情報

Attributes:

- `status` *str* - コンポーネントの実行状態を表す文字列
- `errors` *list[HiveCompError]* - コンポーネント実行中、もしくは、エラーで停止した場合のエラー情報

15.3.23 HiveInPortEventSetting Objects

```
class HiveInPortEventSetting()
```

入力ポートのイベント設定情報クラス

入力ポートに設定されたイベント情報を保持するクラス このクラスの属性値を変更しても実際の入力ポート設定には反映されません

Attributes:

- `source_name` *str* - イベント判定の対象となっているコンポーネント名
- `data_name` *str* - イベント判定の対象となっているカラム名

- `pre_time` *int* - イベント区間前方の拡張時間範囲
- `post_time` *int* - イベント区間後方の拡張時間範囲

15.3.24 SynapseLock Objects

```
class SynapseLock()
```

コンポーネント間排他制御のためのロックオブジェクト

`HiveComponentBase.create_lock()` でロックを獲得したときに取得できるオブジェクトです。 通常はこのオブジェクトを操作する必要はありません。 `create_lock()` の使用例のように、 `with` 句を使用していればそのスコープを抜けた際に自動でロックが解放されます。

`with` 句を使用しない場合は、このクラスのメソッド `acquire()`, `release()` でロックの獲得解放が可能です。

15.3.24.1 acquire

```
def acquire() -> None
```

ロックを獲得する

このオブジェクトが生成されたときの引数 `key, timeout, interbase` を使ってロックを要求します。 ロック獲得成功した場合は何も返さずにメソッドが終了します。

Raises:

- `HiveTimeoutError` - 指定されたタイムアウト時間内にロックを獲得できなかった
- `HiveLockCanceled` - ロックを獲得する前にコンポーネントの停止要求を受けた
- `HiveApiError` - Hiveフレームワークの各種エラー

15.3.24.2 release

```
def release() -> None
```

ロックを解放する

このオブジェクトが保持しているロックを解放し、他のコンポーネント/スレッドがロックを獲得できるようにします。

15.3.25 HiveTimeoutError Objects

```
class HiveTimeoutError(Exception)
```

関数、メソッド内でブロッキングする場合のタイムアウト時間に達した場合に送出される例外です

15.3.26 HiveLockCanceled Objects

```
class HiveLockCanceled(Exception)
```

SynapseLockオブジェクトにおいて、ロック獲得要求中にコンポーネントが停止された場合に送出される例外です

15.4 カスタムコンポーネントC-APIリファレンス

カスタムコンポーネントをC言語で開発する際のAPI仕様です。 カスタムコンポーネントの開発手順については下記を参照してください。

C言語カスタムコンポーネント作成

15.4.1 出力ポート関数

15.4.1.1 hive_outport_create_column() … 新規カラム作成

コンポーネントが収集したデータをSpeeDBee Synapseに登録するためのカラムのハンドルを取得する関数

定義

```
HIVE_OUTCOLUMN hive_outport_create_column(HIVE_COMPONENT component, int out_port_no, const char
*name, HIVE_DATA_TYPE type, HIVE_COLUMN_OPTIONS opt)
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`int out_port_no`	カラムを作成するポート番号 OUTPORT1, OUTPORT2などのマクロ定数を指定します。通常はOUTPORT1だけを使用すればOKです。
3	`const char *name`	カラム名 作成するカラムの名前を指定します
4	`HIVE_DATA_TYPE type`	データ型 作成するカラムの型を指定します
5	`HIVE_COLUMN_OPTIONS opt`	カラムオプション 作成するカラムのオプションを指定します。詳細はHIVE_COLUMN_OPTIONSを参照してください。

戻り値

- カラムハンドル`HIVE_OUTCOLUMN`: カラムを扱うハンドル。のちにカラムにデータを登録したり、カラム設定を変更する際に使用します
- NULL: カラムの作成に失敗

説明

コンポーネントが収集したデータをSpeeDBee Synapseに登録するためのカラムのハンドルを取得する関数です。カラムにデータを登録する際にはこの関数が返したカラムハンドルを使用します。

カラム名には"/"は使用できません。その他の文字は使用可能ですが、ASCII以外の文字を使用する場合はUTF-8エンコードされている必要があります。また、カラム名は127バイトまでとしてください。

１つのコンポーネント内で同一のカラム名のカラムを取得した場合、最後に取得したカラムハンドルのみが有効となります。 そのため、本関数で同じカラム名のカラムハンドルを取得すれば、カラムの設定を変更できます。ただし事前に取得していたカラムハンドルは無効になりますので使用しないでください。

カラムのオプション指定についての詳細はHIVE_COLUMN_OPTIONSを参照してください。

15.4.1.2 hive_outport_register_confirmed_time() … ステータス情報へのカラムデータ確定時刻の登録

出力ポートにカラムデータの確定時刻を登録する関数

定義

```
void hive_outport_register_confirmed_time(HIVE_COMPONENT component, int out_port_no, hive_time_t
ts);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`int out_port_no`	登録対象の出力ポート番号
3	`hive_time_t ts`	登録する時刻

戻り値

なし

説明

出力ポートにカラムデータの確定時刻を登録します。

この情報が登録されることにより、このコンポーネントは、登録した時刻より前のデータはすべて登録が完了したことをフレームワークに伝えることができます。
これによりデータを受け取る側のコンポーネントの処理が最適化され、より少ない遅延でデータを扱うことができるようになります。

データを登録するコンポーネントでは、何らかのタイムスタンプのデータを登録した後で、今後それより前のタイムスタンプのデータを登録することがないことが確定したタイミングでその時刻を登録してください。
このAPIを使わずとも各種コンポーネントは動作可能ですが、データを受けとる側のコンポーネントでデータを受け取る際に多少の遅延が発生します。

逆に、このAPIであるタイムスタンプを登録した場合、そのタイムスタンプより前のタイムスタンプのデータを登録しないように注意が必要です。 登録してしまうと、そのデータを受け取るコンポーネントでデータを取りこぼす可能性があります。

15.4.2 出力カラム関数

15.4.2.1 hive_outcolumn_insert() … データ登録

カラムにデータを登録する関数

定義

```
bool hive_outcolumn_insert(HIVE_OUTCOLUMN column, const void *data);
```

引数

No	項目	説明
1	`HIVE_OUTCOLUMN column`	カラムハンドル hive_outport_create_column()で取得したカラムハンドルを指定します
2	`void *data`	登録するデータへのポインタ

戻り値

- true: 登録成功
- false: 登録失敗

説明

第一引数に指定したカラムに単一のデータを登録します。 第二引数`data`には、登録するカラムのデータ型に合わせたメモリアドレスを指定する必要があります。

例

- uint16型のスカラの場合、2バイトのメモリ領域の先頭アドレス
- float型の5要素配列の場合、20バイトのメモリ領域の先頭アドレス

本関数で登録したデータのタイムスタンプは、本関数呼び出し時の現在時刻となります。時刻を指定する必要がある場合はhive_outcolumn_insert_ts()を使用してください。

15.4.2.2 hive_outcolumn_insert_ts() … タイムスタンプ指定データ登録

カラムにタイムスタンプ指定のデータを登録する関数

定義

```
bool hive_outcolumn_insert_ts(HIVE_OUTCOLUMN column, const void *data, hive_time_t ts);
```

引数

No	項目	説明
1	`HIVE_OUTCOLUMN column`	カラムハンドル hive_outport_create_column()で取得したカラムハンドルを指定します
2	`const void *data`	登録するデータへのポインタ
3	`hive_time_t ts`	登録するデータのタイムスタンプ

戻り値

- true: 登録成功
- false: 登録失敗

説明

第一引数、第二引数はhive_outcolumn_insert()と同様です。
第三引数`ts`にはタイムスタンプを指定してください。現在時刻のタイムスタンプはhive_timestamp()で取得できます。

本関数は、複数のカラムに対して同一のタイムスタンプでデータを登録するような場面での利用を想定しています。

使用例：同じタイムスタンプで複数カラムにデータを登録する

```
 :
hive_time_t ts0 = hive_timestamp();
hive_outcolumn_insert_ts(clmA, dataA, ts0);
hive_outcolumn_insert_ts(clmB, dataB, ts0);
hive_outcolumn_insert_ts(clmC, dataC, ts0);
 :
```

hive_outcolumn_insert()を使用した場合、clmA, clmB, clmCそれぞれで異なるタイムスタンプになってしまいます。

本システムでは、同一のカラムに登録するデータのタイムスタンプは、基本的に時系列順に登録する必要があります。 そのため、あるカラムに時刻`T0`のデータを登録したあとで`T0`より前のタイムスタンプのデータを同じカラムに登録することはできません。

15.4.2.3 hive_outcolumn_insert_binary() … バイナリ型データ登録

BINARY型カラムに可変長バイナリデータを登録する関数

定義

```
bool hive_outcolumn_insert_binary(HIVE_OUTCOLUMN column, const void *data, size_t n);
```

引数

No	項目	説明
1	`HIVE_OUTCOLUMN column`	カラムハンドル hive_outport_create_column()で取得したカラムハンドルを指定します
2	`const void *data`	登録するデータへのポインタ
3	`size_t n`	登録するデータのバイト数

戻り値

- true: 登録成功
- false: 登録失敗

説明

BINARY型として作成されたカラムに対して任意長のデータを登録する関数です。
第一引数、第二引数はhive_outcolumn_insert()と同様です。第三引数`n`には、登録するデータのバイト数を指定してください。

本関数で登録したデータのタイムスタンプは、本関数呼び出し時の現在時刻となります。 時刻を指定する必要がある場合はhive_outcolumn_insert_binary_ts()を使用してください。

15.4.2.4 hive_outcolumn_insert_binary_ts() … タイムスタンプ指定バイナリ型データ登録

BINARY型カラムに可変長バイナリデータをタイムスタンプ指定で登録する関数

定義

```
bool hive_outcolumn_insert_binary_ts(HIVE_OUTCOLUMN column, const void *data, size_t n,
hive_time_t ts);
```

引数

No	項目	説明
1	`HIVE_OUTCOLUMN column`	カラムハンドル hive_outport_create_column()で取得したカラムハンドルを指定します
2	`const void *data`	登録するデータへのポインタ
3	`size_t n`	登録するデータのバイト数
4	`hive_time_t ts`	登録するデータのタイムスタンプ

戻り値

- true: 登録成功
- false: 登録失敗

説明

第一引数、第二引数、第三引数はhive_outcolumn_insert_binary()と同様です。
第三引数`ts`にはタイムスタンプを指定してください。タイムスタンプについての注意事事項は
hive_outcolumn_insert_ts()を参照してください。

15.4.2.5 hive_outcolumn_insert_multisamples() … 複数サンプルデータ登録

１つのカラムに複数のタイムスタンプのデータを一括で登録する関数

定義

```
bool hive_outcolumn_insert_multisamples(HIVE_OUTCOLUMN column, const void *data, hive_time_t ts,
int sample_count);
```

引数

No	項目	説明
1	`HIVE_OUTCOLUMN column`	カラムハンドル hive_outport_create_column()が返したカラムハンドルを指定します
2	`const void *data`	登録するデータへのポインタ
3	`hive_time_t ts`	登録するデータのタイムスタンプ
4	`int sample_count`	登録するデータのサンプル数

戻り値

- true: 登録成功
- false: 登録失敗

説明

本関数はカラムオプションにてカラムタイプMIDDLEでサンプリングレートを指定した場合のみ使用できます。
第一、第二、第三引数はhive_outcolumn_insert_ts()と同様です。
第四引数`sample_count`には、登録するデータのサンプル数を指定してください。登録する複数のデータは、第二引数の`data`が指すメモリ上に連続して配置されている必要があります。

使用例：連続する複数サンプルのデータを登録する

```
  HIVE_COLUMN_OPTIONS opt = {
    .type = column_type_middle,
    .sampling_rate = 10;
  };
  HIVE_OUTCOLUMN clmA = hive_outport_create_column(compo, "clmA",
HIVE_DATA_SCALAR(HIVE_TYPE_UINT16), opt)
  uint16_t data[10] = {.}
   :
  get_data(data, 10);
  hive_time_t ts0 = hive_timestamp();
  hive_outcolumn_insert_multisamples(clmA, data, ts0, 10);
   :
```

上記の例の場合、data[0]は指定したタイムスタンプ`ts0`で登録され、続く`data[1]`は`ts0 + 0.1秒`, data[2]は`ts0 + 0.2秒`として登録されます。
各タイムスタンプの間隔は`opt.sampling_rate`から算出されます。

15.4.2.6 hive_outcolumn_open_file() … ファイル作成

ファイル型カラム用のファイルを作成する関数

定義

```
HIVE_FILE *hive_outcolumn_open_file(HIVE_OUTCOLUMN column);
```

引数

No	項目	説明
1	`HIVE_OUTCOLUMN column`	カラムハンドル hive_outport_create_column()が返したカラムハンドルを指定します

戻り値

- HIVE_FILEポインタ `HIVE_FILE` : 作成されたファイルポインタを持つHIVE_FILEのポインタ。
- NULL: ファイルの作成に失敗

説明

本関数はファイル型カラムへのデータ登録のためのファイルを作成し、そのファイルポインタを持つ `HIVE_FILE` ポインタを返します。
返された `HIVE_FILE` 内のファイルポインタを使用してファイルへ書き込みを行ってください。その後、hive_outcolumn_insert_file()を使用してカラムにファイルを登録できます。

使用例：ファイル作成とデータの書き込み

```
  HIVE_OUTCOLUMN clmfile  = hive_outport_create_column(comp, OUTPORT1, "clmfile",
HIVE_DATA_SCALAR(HIVE_TYPE_FILE),   HIVE_COLUMN_OPTION_NONE);
   :
  HIVE_FILE *hf = hive_outcolumn_open_file(clmfile);
  if (hf == NULL) {
    HIVE_API_ERROR error = hive_get_api_error();
    HIVE_LOG_ERROR("hive_outcolumn_open_file error(0x%x,%d,%d)", error.code, error.system_errno,
error.db_errno);
  }
   :
  for (int i = 0; i < 10; i++) {
    fprintf(hf->fp, "data: %d\n");
  }
   :
  HIVE_FILE_METAINFO meta = {
   .filename = "a",
   .media_type = "text/plain",
   .begin_timestamp = 1,
   .end_timestamp = 2,
  };

  if (!hive_outcolumn_insert_file(hf, 0, &meta)) {
    HIVE_API_ERROR error = hive_get_api_error();
    HIVE_LOG_ERROR("hive_outcolumn_insert_file error(0x%x,%d,%d)", error.code,
error.system_errno, error.db_errno);
  }
```

本関数が返した`HIVE_FILE`ポインタは、必ず hive_outcolumn_insert_file()かhive_outcolumn_dispose_file()を使用してリソースを解放してください。そうしない 場合、リソースが解放されずにメモリリークの原因となります。

15.4.2.7 hive_outcolumn_insert_file() … ファイル型データの登録

ファイル型カラムにデータを登録する関数

定義

```
bool hive_outcolumn_insert_file(HIVE_FILE *hf, hive_time_t ts, const HIVE_FILE_METAINFO *meta);
```

引数

No	項目	説明
1	`HIVE_FILE *hf`	HIVE_FILEのポインタ hive_outcolumn_open_file()が返したHIVE_FILEポインタを指定します
2	`hive_time_t ts`	登録するデータのタイムスタンプ
3	`HIVE_FILE_METAINFO *meta`	登録するファイルのメタ情報

戻り値

- true: 登録成功
- false: 登録失敗

説明

本関数はファイル型カラムにデータを登録します。
第一引数`hf`にはhive_outcolumn_open_file()で作成した`HIVE_FILE`ポインタを指定します。 第二引数`ts`にはタイムスタンプを指定してください。`0`が指定されると現在時刻で登録されます。 第三引数`meta`には`HIVE_FILE_METAINFO`に登録するファイルのメタ情報を入れ、指定してください。 メタ情報は、このファイル型カラムのデータを受け取る側で参照されますが、不要ならNULL指定も可能です。

使用例はhive_outcolumn_open_file()を参照してください。

15.4.2.8 hive_outcolumn_dispose_file() … ファイル破棄

ファイルを破棄する関数

定義

```
void hive_outcolumn_dispose_file(HIVE_FILE *hf);
```

引数

No	項目	説明
1	`HIVE_FILE *hf`	HIVE_FILEのポインタ hive_outcolumn_open_file()が返したHIVE_FILEポインタを指定します

戻り値

なし

説明

本関数はhive_outcolumn_open_file()で作成したファイル型カラム用のファイルとHIVE_FILEを破棄します。
hive_outcolumn_open_file()でファイルをオープンしたものの、途中で不要となった場合には本関数を呼び出してファイルを破棄してください。

15.4.3 入力カラム関数

15.4.3.1 hive_inport_get_column_list() … 入力カラム情報取得

入力ポートのカラム情報を取得する

定義

```
const HIVE_INCOLUMN_LIST *hive_inport_get_column_list(HIVE_COMPONENT component, int inPortNo);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`int in_port_no`	入力ポート番号

戻り値

- HIVE_INCOLUMN_LIST: カラム情報のリスト
- NULL: メモリ不足等によりカラム情報が取得できない場合

説明

本関数は、指定した入力ポートに接続された、他のコンポーネントのカラム情報（コンポーネント名、カラム名、型など）を取得します。
戻り値として受け取った構造体のメモリはシステムが内部で管理しているため、変更、解放等行わないでください。また、他の`hive_inport_xxxx`APIをコールすると内容が変化することがありますので、保存しておく必要がある場合は別のメモリ領域にコピーするなどの操作を行ってください。

15.4.3.2 hive_inport_continuous_reader() … カラムデータ継続検索用のReader取得

入力ポートのカラムから継続的にデータを取得するためのReaderを取得する

定義

```
HIVE_CONTINUOUS_READER *hive_inport_continuous_reader(HIVE_COMPONENT component, int in_port_no,
hive_time_t search_begin);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`int in_port_no`	入力ポート番号
3	`hive_time_t search_begin`	検索範囲開始時刻

戻り値

- HIVE_CONTINUOUS_READER: カラムデータ継続検索用のReader

説明

本関数は、指定した入力ポートに接続された、他のコンポーネントのカラムデータを継続して検索するためのReaderを取得します。
取得したReaderをhive_continuous_reader_read()で使用することでカラムデータを検索、取得することができます、戻り値として受け取った構造体のメモリはシステムが内部で管理しているため、変更、解放等行わないでください。

使用例：

```
HIVE_CONTINUOUS_READER *creader;
creader = hive_inport_continuous_reader(comp, INPORT1, hive_timestamp());

while (hive_component_runnable(comp)) {
  const HIVE_COLUMN_READ_RESULT *read_result = hive_continuous_reader_read(creader);
  if (!read_result) { continue; }

  HIVE_RECORD_ITERATOR iter = HIVE_GET_RECORD_ITERATOR(read_result);
  const HIVE_RECORD *record;
  while ((record = HIVE_RECORD_ITERATOR_GET_NEXT(&iter)) != NULL)
  {
    for (int i=0; i < record->data_count; i++) {
      const HIVE_RECORD_DATA *rd = HIVE_RECORD_GET_DATA(record, i);
      if (!rd) { continue; }

      const char *sourcename   = read_result->incolumn_list->incolumns[i].source_component;
      const char *dataname     = read_result->incolumn_list->incolumns[i].data_name;
      HIVE_DATA_TYPE type      = read_result->incolumn_list->incolumns[i].data_type;
      hive_aggregation_stat_type_t stat = read_result->incolumn_list->incolumns[i].stat_type;
      const char *dptr         = rd->data;
      size_t size              = rd->data_size;

      process_record_data(sourcename, dataanme, type, stat_type, dptr, size);
    }
  }
}
hive_continuous_reader_release(creader);
```

15.4.3.3 hive_continuous_reader_read() … カラムデータ継続検索

Readerで指定された入力ポートに接続されているカラムのデータを継続して検索、取得する

定義

```
const HIVE_COLUMN_READ_RESULT *hive_continuous_reader_read(HIVE_CONTINUOUS_READER *contReader);
```

引数

No	項目	説明
1	`HIVE_CONTINUOUS_READER *contReader`	hive_inport_continuous_reader()で取得したReader

戻り値

- HIVE_COLUMN_READ_RESULT: カラムデータ検索結果

説明

本関数は、hive_inport_continuous_reader()で取得したReaderからカラムデータを取得します。
呼び出しを行うごとに検索時間が更新され、継続的にカラムデータを取得することができます。
戻り値として受け取った構造体のメモリはシステムが内部で管理しているため、変更、解放等行わないでください。

15.4.3.4 hive_continuous_reader_release() … カラムデータ継続検索用のReaderの解放

hive_inport_continuous_reader()で取得したReaderを解放する

定義

```
const void hive_continuous_reader_release(HIVE_CONTINUOUS_READER *contReader);
```

引数

No	項目	説明
1	`HIVE_CONTINUOUS_READER *contReader`	hive_inport_continuous_reader()で取得したReader

戻り値

なし

説明

本関数は、hive_inport_continuous_reader()で取得したReaderを解放します。
コンポーネントの処理中にReaderが不要となるなどの場合に呼び出してください。

15.4.3.5 hive_inport_timerange_reader() … カラムデータ時間範囲検索用のReader取得

入力ポートのカラムから時間範囲を指定してデータを取得するためのReaderを取得する

定義

```
HIVE_TIMERANGE_READER *hive_inport_timerange_reader(HIVE_COMPONENT component, const HIVE_INCOLUMN
*incolumn, hive_time_t searchBegin, hive_time_t searchEnd);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。

No	項目	説明
2	`HIVE_INCOLUMN *incolumn`	データ取得するカラム
3	`hive_time_t search_begin`	検索範囲開始時刻
3	`hive_time_t search_end`	検索範囲終了時刻

戻り値

- HIVE_TIMERANGE_READER: カラムデータ時間範囲検索用のReader

説明

本関数は、指定したカラムのデータを時間範囲を指定して検索するためのReaderを取得します。
カラムを指定しない場合には入力ポートに接続された、他のコンポーネントのカラムすべてを検索対象とします。
取得したReaderをhive_timerange_reader_read()で使用することでカラムデータを検索、取得することができます。
戻り値として受け取った構造体のメモリはシステムが内部で管理しているため、変更、解放等行わないでください。

使用例：

```
hive_time_t search_bigin_ts = hive_timestamp() - 2*HIVE_TIME_1S;

while (hive_component_runnable(comp)) {
  HIVE_TIMERANGE_READER *reader = hive_inport_timerange_reader(comp, incolumn, search_bigin_ts,
search_bigin_ts + 1*HIVE_TIME_1S);

  const HIVE_COLUMN_READ_RESULT *read_result;
  while ((read_result = hive_timerange_reader_read(reader)) != NULL) {
    if (!hive_component_runnable(comp)) { break; }

    HIVE_LOG_INFO("time range: %lu ~ %lu", begin, end);

    HIVE_RECORD_ITERATOR iter = HIVE_GET_RECORD_ITERATOR(read_result);
    const HIVE_RECORD *record;
    while ((record = HIVE_RECORD_ITERATOR_GET_NEXT(&iter)) != NULL)
    {
      Logbuff logb;
      logbuff_reset(&logb);
      logbuff_write(&logb, " %c %19lu", (record->record_type==0)?'D':'S', record->timestamp);

      for (int i=0; i < record->data_count; i++) {
        const HIVE_RECORD_DATA *rd = HIVE_RECORD_GET_DATA(record, i);
        if (!rd) {
          logbuff_write(&logb, "%10s", "(no data)");
          continue;
        }
        const char *sourcename   = read_result->incolumn_list->incolumns[i].source_component;
        const char *dataname     = read_result->incolumn_list->incolumns[i].data_name;
        HIVE_DATA_TYPE type      = read_result->incolumn_list->incolumns[i].data_type;
        hive_aggregation_stat_type_t stat = read_result->incolumn_list->incolumns[i].stat_type;
        const char *dptr         = rd->data;
        size_t size              = rd->data_size;

        process_record_data(sourcename, dataanme, type, stat_type, dptr, size);
      }
    }
  }
  hive_timerange_reader_release(reader);
  usleep(1*1000*1000);
  search_next_ts += 1*HIVE_TIME_1S;
}
```

15.4.3.6 hive_timerange_reader_read() … カラムデータ時間範検索

Readerで指定されたカラム、または入力ポートに接続されている全カラムのデータを指定された時間範囲で検索、取得する

定義

```
const HIVE_COLUMN_READ_RESULT *hive_timerange_reader_read(HIVE_TIMERANGE_READER *rangeReader);
```

引数

No	項目	説明
1	`HIVE_TIMERANGE_READER *rangeReader`	hive_inport_timerange_reader()で取得したReader

戻り値

- HIVE_COLUMN_READ_RESULT: カラムデータ検索結果

説明

本関数は、hive_inport_timerange_reader()で取得したReaderからカラムデータを取得します。
Readerの取得の際に指定した時間範囲のカラムデータを取得することができます。
戻り値として受け取った構造体のメモリはシステムが内部で管理しているため、変更、解放等行わないでください。

15.4.3.7 hive_timerange_reader_release() … カラムデータ時間範囲検索用のReaderの解放

hive_inport_timerange_reader()で取得したReaderを解放する

定義

```
const void hive_timerange_reader_release(HIVE_TIMERANGE_READER *reader);
```

引数

No	項目	説明
1	`HIVE_TIMERANGE_READER *reader`	hive_inport_timerange_reader()で取得したReader

戻り値

なし

説明

本関数は、hive_inport_timerange_reader()で取得したReaderを解放します。
コンポーネントの処理中にReaderが不要となるなどの場合に呼び出してください。

15.4.3.8 HIVE_GET_RECORD_ITERATOR() … レコードイテレータ取得

ウィンドウデータからレコードのイテレータを取得

定義

```
HIVE_RECORD_ITERATOR HIVE_GET_RECORD_ITERATOR(const HIVE_COLUMN_READ_RESULT *data);
```

引数

No	項目	説明
1	`const HIVE_COLUMN_READ_RESULT *data`	hive_continuous_reader_read()で取得したウィンドウデータ

戻り値

- HIVE_RECORD_ITERATOR: レコードイテレータ

説明

本関数は、hive_continuous_reader_read()で取得したウィンドウデータが持つ、レコードのイテレータを取得します。
取得したイテレータに対してHIVE_RECORD_ITERATOR_GET_NEXT()を呼び出すことで、ウィンドウに含まれるレコードを順次取得することができます。

使用例：

```
while (hive_component_runnable(component)) {
  const HIVE_RECORD *record;
  const HIVE_COLUMN_READ_RESULT *read_result;

  read_result = hive_continuous_reader_read(creader);
  if (!read_result) { continue; }

  HIVE_RECORD_ITERATOR iter = HIVE_GET_RECORD_ITERATOR(read_result);
  while ((record = HIVE_RECORD_ITERATOR_GET_NEXT(&iter)) != NULL)
  {
    for (int i=0; i < record->data_count; i++) {
      const HIVE_RECORD_DATA *rd = HIVE_RECORD_GET_DATA(record, i);
    }
  }
}
```

15.4.3.9 HIVE_RECORD_ITERATOR_GET_NEXT() … レコードイテレータ取得

レコード取得

定義

```
const HIVE_RECORD *HIVE_RECORD_ITERATOR_GET_NEXT(HIVE_RECORD_ITERATOR *iter);
```

引数

No	項目	説明
1	`HIVE_RECORD_ITERATOR *iter`	HIVE_GET_RECORD_ITERATOR()で取得したイテレータへのポインタ

戻り値

- `HIVE_RECORD*`：レコードポインタ
- `NULL`：すべてのレコードを取得済み

説明

本関数は、ウィンドウデータに含まれるレコードを順次取得するために、HIVE_GET_RECORD_ITERATOR()と組み合わせて使用します。
使用例はHIVE_GET_RECORD_ITERATOR()を参考にしてください。

15.4.3.10 HIVE_RECORD_GET_DATA() … レコードイテレータ取得

レコードデータ取得

定義

```
const HIVE_RECORD_DATA *HIVE_RECORD_GET_DATA(const HIVE_RECORD *record, size_t index);
```

引数

No	項目	説明
1	`HIVE_RECORD *record`	HIVE_RECORD_ITERATOR_GET_NEXT()で取得したレコードへのポインタ
2	`size_t index`	取得するレコードデータのインデックス。`record->data_count`未満の値にすること

戻り値

- `HIVE_RECORD_DATA*`：レコードデータポインタ

説明

本関数は、レコードに含まれる個々のデータを取得します。
使用例はHIVE_GET_RECORD_ITERATOR()を参考にしてください。

15.4.3.11 hive_incolumn_get_latest_value() … カラムデータ最新値の取得

入力ポートのカラムから最新のデータを取得する

定義

```
bool hive_incolumn_get_latest_value(const HIVE_INCOLUMN *incolumn, hive_time_t *rts, char
**dataPointer, size_t *dataSize);
```

引数

No	項目	説明
1	`HIVE_INCOLUMN *incolumn`	データ取得するカラム
2	`hive_time_t *rst`	取得したデータのタイムスタンプを入れるポインタ
3	`char **dataPointer`	取得したデータの先頭アドレスを格納するポインタ
4	`size_t *dataSize`	取得したデータのサイズを格納するポインタ

戻り値

- true: 最新値取得成功
- false: 最新値取得失敗

説明

本関数は、指定したカラムの最新データを取得します。
取得に失敗した場合には、戻り値はfalseに、最新データが無いなど取得ができない場合には、取得データのタイムスタンプは0、データサイズも0となります。読み込んだデータ領域は`HIVE_INCOLUMN`内に保持しています。そのため複数スレッドから同じ`HIVE_INCOLUMN`に対してこの関数を呼ぶ場合、排他制御が必要です。

15.4.4 コンポーネントステータス関数

15.4.4.1 hive_status_clear() … ステータス情報クリア

ステータス情報をクリアし、エラーがない状態にする

定義

```
void hive_status_clear(HIVE_STATUS status);
```

引数

No	項目	説明
1	`HIVE_STATUS status`	ステータス構造体へのポインタ

戻り値

なし

説明

ステータス構造体に登録されたエラー情報等を消去して、何もエラーが無い、正常状態に戻します。
この状態のステータス構造体がhive_component_register_status()で登録されると、本コンポーネントは正常に動作していると判断されます。

15.4.4.2 hive_status_add_error() … ステータス情報へのエラー登録

ステータス情報にエラーを登録する

定義

```
void hive_status_add_error(HIVE_STATUS status, HIVE_ERROR_INFO error);
```

引数

No	項目	説明
1	`HIVE_STATUS status`	ステータス構造体へのポインタ
2	`HIVE_ERROR_INFO error`	登録するエラーの番号 事前に定義済みのエラー番号を指定できます

戻り値

なし

説明

ステータス構造体にエラーを登録します。 `error` には、HIVE_ERROR()マクロ関数で生成した値を指定してください。

使用例：

```
const static HIVE_ERROR_TYPE error_types[] = {
  { .name="CONNECTION REFUSED",       .parameter_names={"b","c"}},
  { .name="REGISTER READ ERROR",      .parameter_names={"f","g"}},
  { .name=NULL },
};
const int ERROR_CONNECTION_REFUSED       = 0;
const int ERROR_REGISTER_READ_ERROR      = 1;

int proc(COMPONENT *comp, HIVE_STATUS status) {
         :
  hvie_status_clear(status);
  if (detect_error1())
    hive_status_add_error(status, HIVE_ERROR(ERROR_CONNECTION_REFUSED, STR("hostname:port")));

  if (detect_error2())
    hive_status_add_error(status, HIVE_ERROR(ERROR_REGISTER_READ_ERROR, STR("REGDATA_A"),
STR("DT0001")));

         :
  hive_component_register_status(comp, status);
}
```

15.4.4.3 hive_status_set_total_xxx_xxx() … ステータス情報への実行時補助情報の登録

ステータス情報に実行時補助情報を登録する

定義

```
void hive_status_set_total_emission_count(HIVE_STATUS status, uint64_t value);
void hive_status_set_total_emission_bytes(HIVE_STATUS status, uint64_t value);
void hive_status_set_total_received_count(HIVE_STATUS status, uint64_t value);
void hive_status_set_total_received_bytes(HIVE_STATUS status, uint64_t value);
```

引数

No	項目	説明
1	`HIVE_STATUS status`	ステータス構造体へのポインタ
2	`uint64_t value`	登録値

戻り値

なし

説明

ステータス構造体にコンポーネント実行時補助情報を登録します。 現時点で定義済みの補助情報は下記のとおりです。

- 累計送信数(`hive_status_set_total_emission_count`)
  コンポーネントが外部に送信したデータの累計数
- 累計送信バイト数(`hive_status_set_total_emission_bytes`)
  コンポーネントが外部に送信したデータの累計バイト数
- 累計受信数(`hive_status_set_total_received_count`)
  コンポーネントが外部から受信したデータの累計数
- 累計受信バイト数(`hive_status_set_total_received_bytes`)
  コンポーネントが外部から受信したデータの累計バイト数

この情報はUIにて補助的に表示する目的で使用されます。 そのためAPIを呼び出さずともコンポーネントの動作に影響はありません。

15.4.4.4 hive_component_register_status() … コンポーネントへのステータス登録

コンポーネントにステータス情報を登録する

定義

```
void hive_component_register_status(HIVE_COMPONENT component, const HIVE_STATUS status);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`HIVE_STATUS status`	ステータス構造体へのポインタ

戻り値

なし

説明

コンポーネントにステータス情報を登録します。
このときのステータス情報にエラーが何も登録されていない場合、コンポーネントは正常に稼働しているとシステムが判断します。ステータス情報にエラーが登録されていると、コンポーネントにエラーが有ることがシステムに報告されます。

システムは最後に登録したステータス情報から、コンポーネントの状態を判定します。コンポーネントは、エラーが解消した場合にはエラーがクリアされたステータス情報をこの関数で登録し直さなければなりません。

15.4.4.5 HIVE_ERROR() … エラー情報の生成

エラー情報構造体を生成するマクロ関数

定義

```
HIVE_ERROR_INFO HIVE_ERROR(int error_no, ...);
```

引数

No	項目	説明
1	`int error_no`	コンポーネントが定義しているエラー種別情報のインデックスを指定してください。。
2	…	発生したエラーの付属情報を指定してください。整数、浮動小数点数、文字列のいずれかを指定できますが、下記の説明の通り利用方法が特殊です。

戻り値

- エラー情報構造体 `HIVE_ERROR_INFO`

説明

発生したエラーの情報を持つ構造体を生成します。 （ヒープ領域ではなくスタック領域確保するため明示的に解放する必要はありません）

第一引数にはコンポーネントで定義しているエラー種別情報のインデックスを指定します。この情報からエラー種別が判別されます。 発生したエラーの詳細をシステムに知らせるために付属情報を引数で指定できます。その場合付属情報の型に応じて以下のようにマクロを利用して引数指定してください。

```
HIVE_ERROR_INFO error = HIVE_ERROR(ERROR_CONNECTION_REFUSED, STR(hostname), I64(portno));
```

付属情報には型に応じて以下のマクロを指定できます。

- `I64(x)` : 64bit符号付き整数
- `U64(x)` : 64bit符号無し整数
- `DBL(x)` : 倍精度浮動小数点数
- `STR(x)` : 文字列ポインタ

付属情報に文字列を使う場合は、生成したHIVE_ERROR_INFOをhive_component_register_status()で登録するまではメモリを解放しないようにしてください。

15.4.4.6 hive_status_alloc() … ステータス情報構造体の確保

ステータス情報構造体を確保する

定義

```
HIVE_STATUS hive_status_alloc(HIVE_COMPONENT component);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。

戻り値

- ステータス情報構造体`HIVE_STATUS`のポインタ
- NULL: メモリ不足

説明

ステータス情報を記録する構造体を確保します。
通常のコンポーネント開発ではこの関数を利用する必要はありません。コンポーネントの開始前に自動でステータス情報`HIVE_STATUS`が確保され、`mainloop`の引数として渡されます。

15.4.4.7 hive_status_free() … ステータス情報構造体の解放

ステータス情報構造体を解放する

定義

```
void hive_status_free(HIVE_STATUS status);
```

引数

No	項目	説明
1	`HIVE_STATUS status`	解放するステータス情報構造体のポインタ

戻り値

なし

説明

hive_status_alloc()で確保したステータス情報構造体を解放します。
通常のコンポーネント開発ではこの関数を利用する必要はありません。コンポーネントの開始前に自動で確保されたステータス情報`HIVE_STATUS`は、コンポーネント停止時に自動で解放されます。

15.4.5 ログ関連関数

15.4.5.1 HIVE_LOG_XXXX() … ログ出力マクロ

コンポーネント用のログ出力マクロ関数

定義

```
void HIVE_LOG_ERROR(const char *format, ...);
void HIVE_LOG_WARNING(const char *format, ...);
void HIVE_LOG_INFO(const char *format, ...);
void HIVE_LOG_DEBUG(const char *format, ...);
void HIVE_LOG_TRACE(const char *format, ...);
```

引数

No	引数	説明
1	`const char *format`	ログ出力書式指定文字列 ログに出力する書式指定文字列を指定します。標準Cライブラリのprintfと同じ形式です。
2	-	書式指定文字列の変換指示子により変換されるパラメータ

戻り値

なし

説明

SpeeDBee Synapse本体のログ出力に文字列を出力するためのマクロ関数です。
引数`format`、および、それに続く可変長引数は、標準Cライブラリのprintf()と同様に使用できます。
マクロは５種類用意されており、それぞれログレベルに対応しています。

15.4.5.2 hive_current_loglevel() … ログレベル取得

システムの現在のログレベルを取得する

定義

```
HIVE_FW_API HIVE_LOG_LEVEL hive_get_loglevel();
```

引数 なし

戻り値

- HIVE_LOG_LEVEL_ERROR(0): エラーログのみをログ出力する状態
- HIVE_LOG_LEVEL_WARNING(1): エラーおよび警告ログをログ出力する状態
- HIVE_LOG_LEVEL_INFO(2): 通常レベルのログを出力する状態
- HIVE_LOG_LEVEL_DEBUG(3): デバッグ用のログを出力する状態
- HIVE_LOG_LEVEL_TRACE(4): 詳細なトレースログを出力する状態

説明

システムの現在のログレベルを取得します。 ERRORが最低値、TRACEが最高値となっており、戻り値の値より小さい値のログはログファイルに出力されます。

15.4.6 その他の関数

15.4.6.1 hive_component_interval_call() … 定期処理関数

一定の周期で指定した関数を継続的に実行する関数

定義

```
bool hive_component_interval_call(HIVE_COMPONENT component, hive_interval_proc_t proc, void
*data, hive_time_t base_time, uint64_t interval_usec);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`hive_interval_proc_t proc`	定期処理関数 定期的に実行する処理を定義した関数のポインタを指定します。
3	`void *data`	ユーザーデータ 任意のデータポインタを指定できます。procがコールされる際に引数に渡されます。
4	`hive_time_t base_time`	定期処理関数の実行開始時刻 proc関数を実行開始する基準時刻 0を指定すると現在時刻となる

No	項目	説明
5	`uint64_t interval_usec`	定期処理関数実行の周期 proc関数をコールする周期をusec単位で指定します。

戻り値

- true: proc関数が一度もエラーとならずに、正常にコンポーネントが停止した場合
- false: proc関数内でエラーが発生した場合

説明

一定の周期で何らかの処理を実行する場合に、この関数を使用してください。

15.4.6.2 hive_component_save_running_info() … 実行時の情報を保存

コンポーネントの実行時の情報を保存する関数

定義

```
bool hive_component_save_running_info(HIVE_COMPONENT component, const char *key, const void
*data, size_t size);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`const char *key`	保存するデータを特定するための情報 異なる`key`を指定することでそれぞれ別の情報を保持できます。
3	`const void *data`	保存するデータ
4	`size_t size`	保存するデータのサイズ

戻り値

- true: 保存成功
- false: 保存失敗

説明

コンポーネント実行時の任意の情報を保存します。
エミッタなどで一度停止した後で、続きから処理したい場合のためにデータを保持しておくことが可能です。

15.4.6.3 hive_component_load_running_info() … 保存した実行時の情報を読み込む

hive_component_save_running_info()で保存したデータを読み込む関数

定義

```
const void *hive_component_load_running_info(HIVE_COMPONENT component, const char *key, size_t
*size);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`const char *key`	保存したデータを特定するための情報
3	`size_t *size`	読み込んだデータのサイズが返る 出力引数

戻り値

読み込んだデータの先頭アドレス。データが存在しない場合にはnullが返されます。
このメモリはフレームワーク側で管理しているため、ユーザーが解放する必要はありません。ただし、同じAPIを再度コールすると以前のメモリ領域が上書きされますので、必要な場合はユーザー側でコピーして持つようにしてください。

説明

hive_component_save_running_info()で保存したデータを読み出します。 実行中のコンポーネントを停止した後、再度続きから処理したい場合に保存したデータを読み込むことで、データの途中から処理を再開することが出来ます。

15.4.6.4 hive_component_runnable() … コンポーネント継続可否

コンポーネントの継続可否を判定する

定義

```
bool hive_component_runnable(HIVE_COMPONENT component);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。

戻り値

- true: コンポーネントのメイン処理を継続可能
- false: コンポーネントの停止要求が出ている

説明

本関数は、コンポーネントのメイン処理を継続可能かを判定するために使用します。
コンポーネントのメイン処理は、この関数がtrueを返す間、処理を継続し、falseを返した場合には速やかに終了する必要があります。そのためメイン処理内では定期的に（最長でも５秒間隔程度）この関数を呼び出して戻り値を確認してください。

この関数がfalseを返す状態にも関わらず、コンポーネントが所定の時間停止しない場合、コンポーネントのスレッドはシステムにより強制停止されることがあります。

15.4.6.5 hive_timestamp() … 現在時刻取得

システムの現在時刻を取得する

定義

```
hive_time_t hive_timestamp();
```

引数

なし

戻り値

- システムの現在時刻

説明

システムの現在時刻を取得します。
タイムスタンプの単位についてはhive_time_tを参照してください。

15.4.6.6 hive_get_api_error() … 発生したエラーの情報を取得

コンポーネントの実行時に発生したエラーの情報を取得する関数

定義

```
HIVE_API_ERROR hive_get_api_error();
```

引数 なし

戻り値

- HIVE_API_ERROR: エラー情報

説明

本関数はコンポーネントの実行時に発生したエラーの情報を取得します。 エラー情報の内容は`HIVE_API_ERROR`構造体を参照してください。

使用例：エラー情報の取得

```
  :
  HIVE_OUTCOLUMN clm  = hive_outport_create_column(comp, OUTPORT1, "clm",
HIVE_DATA_SCALAR(HIVE_TYPE_INT8),   HIVE_COLUMN_OPTION_NONE);
  if (clm == nullptr) {
    HIVE_API_ERROR error = hive_get_api_error();
    HIVE_LOG_ERROR("hive_outport_create_column error(0x%x,%d,%d)", error.code,
error.system_errno, error.db_errno);
  }
   :
```

15.4.6.7 HIVE_DATA_TYPE_NAME() … データ型の名前(文字列)を取得

SpeeDBee Synapseのデータ型の名前(文字列)を取得する関数

定義

```
const char *HIVE_DATA_TYPE_NAME(hive_primitive_type_t primitive_type);
```

引数

No	項目	説明
1	`hive_primitive_type_t primitive_type`	データ型

戻り値

- 渡されたデータ型の名前が文字列として返される

説明

本関数はSpeeDBee Synapseのデータ型(hive_primitive_type_t)の名前を文字列として取得します。

使用例：データ型の名前(文字列)を取得

```
  HIVE_LOG_INFO("HIVE_DATA_TYPE_NAME: %s", HIVE_DATA_TYPE_NAME(HIVE_TYPE_INT16));
  :
```

15.4.6.8 HIVE_AGGREGATION_STAT_TYPE_NAME() … 統計種別の名前(文字列)を取得

SpeeDBee Synapseの統計種別の名前(文字列)を取得する関数

定義

```
const char *HIVE_AGGREGATION_STAT_TYPE_NAME(hive_aggregation_stat_type_t stat_type);
```

引数

No	項目	説明
1	`hive_aggregation_stat_type_t stat_type`	統計種別

戻り値

- 渡された統計種別の名前が文字列として返される

説明

本関数はSpeeDBee Synapseの統計種別の名前を文字列として取得します。

使用例：統計種別の名前(文字列)を取得

```
  HIVE_LOG_INFO("HIVE_AGGREGATION_STAT_TYPE_NAME: %s",
HIVE_AGGREGATION_STAT_TYPE_NAME(HIVE_AGGREGATION_STAT_TYPE_BASICSUM));
  :
```

15.4.6.9 HIVE_DATA_TYPE_FROM_NAME() … データ名(文字列)からデータ型と配列数を取得

データ名(文字列)からSpeeDBee Synapseのデータ型と配列数を取得する関数

定義

```
HIVE_DATA_TYPE HIVE_DATA_TYPE_FROM_NAME(const char *typeName);
```

引数

No	項目	説明
1	`char *typeName`	データ名

戻り値

- HIVE_DATA_TYPE: SpeeDBee Synapseのデータ型と配列数の情報

説明

本関数はデータ名(文字列)からSpeeDBee Synapseのデータ型と配列数を取得します。

使用例：データ型と配列数を取得

```
HIVE_DATA_TYPE data_type = HIVE_DATA_TYPE_FROM_NAME("int16[3]");
HIVE_LOG_INFO("HIVE_DATA_TYPE: type: %d array: %d", data_type.type, data_type.array);
:
```

15.4.6.10 hive_get_common_variable() … コンポーネント共通変数取得

コンポーネント共通変数を取得する関数

定義

```
const char *hive_get_common_variable(HIVE_COMPONENT component, const char *varname);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
1	`char *varname`	取得したい共通変数の変数名

戻り値

指定した変数名に設定されている値（文字列）の先頭アドレス。データが存在しない場合にはnullが返されます。このメモリはフレームワーク側で管理しているため、ユーザーが解放する必要はありません。ただし、同じAPIを再度コールすると以前のメモリ領域が上書きされますので、必要な場合はユーザー側でコピーして持つようにしてください。

説明

本関数は変数名を指定して、設定されている共通変数の値を取得します。

使用例：データ型と配列数を取得

```
  const char *project_dir = hive_get_common_variable(component, "PROJECT_DIR");
  HIVE_LOG_INFO("project_dir: %s", project_dir);
  :
```

15.4.6.11 hive_lock() … コンポーネント間排他制御

コンポーネント間での排他制御のためのロック獲得関数

定義

```
HIVE_LOCK_RESULT hive_lock(HIVE_COMPONENT component, const char *key, bool interbase, uint64_t
timeout);
HIVE_LOCK_RESULT synapse_lock(HIVE_COMPONENT component, const char *key, bool interbase, uint64_t
timeout);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`const char *key`	排他制御対象を識別するための任意の文字列
3	`bool interbase`	異なるコンポーネントベース間で排他制御を行う場合は `true`
4	`uint64_t timeout`	ロックの解放を待つ場合のタイムアウト時間

戻り値

- HIVE_LOCK_OK: ロック獲得成功
- HIVE_LOCK_TIMEOUT: 指定されたタイムアウト時間内にロックを獲得できなかった
- HIVE_LOCK_CANCELED: ロックを獲得する前にコンポーネントの停止要求を受けた

説明

本関数は、引数 `key` で指定された文字列で識別されるリソースの排他制御のためのロックを獲得します。 同一の `key` におけるロックは１つのスレッドでしか獲得できません。 獲得したロックは、対応するAPI関数hive_unlock()で解放できます。 もしくは、ロックを獲得したままコンポーネントの実行が終了した場合に、自動でロックが解放されます。

使用例：デバイスファイルを使用するロジックの排他制御

```
HIVE_LOCK_RESULT result = hive_lock(component, "/dev/serial", false, 10000000000);
if (result == HIVE_LOCK_TIMEOUT) {
  HIVE_LOG_ERROR("could not take lock.");
  return false;
} else if (result == HIVE_LOCK_CANCELED) {
  return true;
}

  // この区間は単一のコンポーネントインスタンスのみ実行可能

hive_unlock(component, "/dev/serial", false);
```

引数`key`は任意の文字列を指定可能です。

引数`interbase`がtrueの場合、異なるコンポーネントベースでも、同じ`key`を使えば排他制御が可能になります。この場合はPython-APIの`HiveComponentBase::lock()`を使ったカスタムコンポーネントとの排他制御も利用可能です。 `interbase`がfalseの場合、同一のコンポーネントベースでのみ排他制御されます。

本関数を呼び出したときにすでに外のコンポーネントインスタンスがロックを獲得している場合、関数内で実行がブロックされます。 ブロックする最大時間は、引数`timeout`にてナノ秒単位で指定可能です。(実質的な単位は実行環境のクロック精度に依存します)
`0`を指定した場合、ロックが獲得できないとすぐに関数がリターンします。タイムアウトせずにロック獲得できるまで待つ場合は、引数`timeout`に`HIVE_LOCK_NO_TIMEOUT`を指定してください。

同一スレッドでも同じ`key`に対して二重にロックすることはできません。その場合はロック解放待ち状態になりますのでタイムアウトするか停止要求を受けるまでブロックし続けます。

15.4.6.12 hive_unlock() … コンポーネント間排他制御

コンポーネント間での排他制御のために獲得したロックを解放する

定義

```
HIVE_LOCK_RESULT hive_unlock(HIVE_COMPONENT component, const char *key, bool interbase);
HIVE_LOCK_RESULT synapse_unlock(HIVE_COMPONENT component, const char *key, bool interbase);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス premain/main/postmainの関数引数に渡されたコンポーネントインスタンスをそのまま渡してください。
2	`const char *key`	排他制御対象を識別するための任意の文字列

No	項目	説明
3	`bool interbase`	異なるコンポーネントベース間で排他制御を行う場合は `true`

戻り値

- HIVE_LOCK_OK: アンロック成功
- HIVE_LOCK_ERROR: エラー

説明

本関数は、`hive_lock` で獲得したロックを解放します。 引数 `key`, `interbase` には、ロック獲得時に指定した値と同じ値を使用してください。

使用例：デバイスファイルを使用するロジックの排他制御

```
  HIVE_LOCK_RESULT result = hive_lock(component, "lock-serialdevice", false, 10000000000);
  if (result == HIVE_LOCK_TIMEOUT) {
    HIVE_LOG_ERROR("could not take lock.");
    return false;
  } else if (result == HIVE_LOCK_CANCELED) {
    return true;
  }

    // この区間は単一のコンポーネントインスタンスのみ実行可能

  hive_unlock(component, "lock-serialdevice", false);
```

15.4.7 型・構造体・関数

15.4.7.1 hive_time_t ... HIVE時刻型

システムの時刻を表す整数型

定義

```
typedef uint64_t hive_time_t;
```

説明

時刻を表す整数値です。

符号なしの64bit整数値となっており、 UNIXエポック(1970年1月1日0時0分0秒(UTC))を0としたときの経過ナノ秒数で表現されています。

時刻	値
1970年1月1日0時0分0秒.0(UTC)	0
1970年1月1日9時0分0秒.0(JST)	0
1970年1月1日9時0分1 秒.0(JST)	1000000000
1970年1月1日9時1分0 秒.0(JST)	60000000000
1970年1月1日10時0分0秒.0(JST)	3600000000000
2022年1月31日13時0分0秒.0(JST)	1643688000000000000
2022年1月31日13時0分1秒.0(JST)	1643688001000000000
2022年1月31日13時1分0秒.0(JST)	1643688060000000000
2022年1月31日14時0分0秒.0(JST)	1643691600000000000

15.4.7.2 hive_primitive_type_t … HIVEカタムプリミティブ型

カラムのプリミティブ型の列挙型定義

定義

```
typedef enum {
  HIVE_TYPE_NONE = 0,
  HIVE_TYPE_BOOLEAN,
  HIVE_TYPE_INT8,
  HIVE_TYPE_INT16,
  HIVE_TYPE_INT32,
  HIVE_TYPE_INT64,
  HIVE_TYPE_UINT8,
  HIVE_TYPE_UINT16,
  HIVE_TYPE_UINT32,
  HIVE_TYPE_UINT64,
  HIVE_TYPE_FLOAT,
  HIVE_TYPE_DOUBLE,
  HIVE_TYPE_TIMESTAMP,         // unsupported
  HIVE_TYPE_STRING,
  HIVE_TYPE_BINARY,
  HIVE_TYPE_COMPLEX,           // unsupported
  HIVE_TYPE_FILE,
  HIVE_TYPE_JSON,              // unsupported
  HIVE_TYPE_BSON,              // unsupported
  HIVE_TYPE_MESSAGEPACK,       // unsupported
  HIVE_TYPE_COMPONENT_STATUS,  // unsupported

  HIVE_TYPE_USERDEFINED = 100,// unsupported
} hive_primitive_type_t;
```

メンバ

No	項目	値	説明
1	`HIVE_TYPE_NONE`	0	不正な型
2	`HIVE_TYPE_BOOLEAN`	1	真偽値型
3	`HIVE_TYPE_INT8`	2	8bit符号あり整数
4	`HIVE_TYPE_INT16`	3	16bit符号あり整数
5	`HIVE_TYPE_INT32`	4	32bit符号あり整数
6	`HIVE_TYPE_INT64`	5	64bit符号あり整数
7	`HIVE_TYPE_UINT8`	6	8bit符号なし整数
8	`HIVE_TYPE_UINT16`	7	16bit符号なし整数
9	`HIVE_TYPE_UINT32`	8	32bit符号なし整数

No	項目	値	説明
10	`HIVE_TYPE_UINT64`	9	64bit符号なし整数
11	`HIVE_TYPE_FLOAT`	10	32bit浮動小数点数
12	`HIVE_TYPE_DOUBLE`	11	64bit浮動小数点数
13	`HIVE_TYPE_STRING`	13	文字列
14	`HIVE_TYPE_BINARY`	14	バイナリデータ
15	`HIVE_TYPE_FILE`	16	ファイルデータ

説明

出力ポートにカラムを作成するときや、入力ポートからカラム情報を取得するときに、そのカラムの基本型を表します。

15.4.7.3 HIVE_DATA_TYPE ... HIVEカラムデータ型

カラムのデータ型を識別する構造体

定義

```
typedef struct _tag_HIVE_DATA_TYPE {
  uint8_t type;
  uint8_t array;
} HIVE_DATA_TYPE;
```

メンバ

No	項目	説明
1	`uint8_t type`	プリミティブ型
2	`uint8_t array`	配列要素数

説明

出力ポートにカラムを作成するときや、入力ポートからカラム情報を取得するときに、そのカラムのデータ型を示す構造体です。
カラムのデータ型には、スカラの他に配列も指定可能なので、配列を指定する場合はその要素数が`array`となります。ただし、`array=0`はスカラ型であることを示します。

以下のプリミティブ型はスカラのみに対応しています。配列にはできないため`array`は常に0としてください。

- `HIVE_TYPE_TIMESTAMP`
- `HIVE_TYPE_STRING`
- `HIVE_TYPE_BINARY`
- `HIVE_TYPE_FILE`

15.4.7.4 HIVE_COMPONENT … コンポーネントインスタンス

コンポーネントのインスタンス情報

定義

```
typedef struct {
  void *user_data;
} *HIVE_COMPONENT;
```

メンバ

No	項目	説明
1	`void *user_data`	コンポーネント側で自由に参照・編集できる専用メンバ

説明

コンポーネントの実行情報を保持する構造体へのポインタです。
上記定義の他、コンポーネントを制御するシステム側が個々のコンポーネントを参照/識別するためのものも別メンバとして定義されています。

各種API呼び出し時に必要となりますのでその引数として使用してください。

`user_data`はユーザーが任意に参照・編集できます。 コンポーネントのコールバック関数を超えて保持したいデータがある場合はこちらを使用してください。

15.4.7.5 hive_interval_proc_t … 定期処理関数シグネチャ

API hive_component_interval_callを使用する際に指定する関数インタフェースのtypedefです。

定義

```
typedef bool (*hive_interval_proc_t)(HIVE_COMPONENT component, void *data, hive_time_t now, int skipped, HIVE_STATUS status);
```

引数

No	項目	説明
1	`HIVE_COMPONENT component`	コンポーネントのインスタンス `hive_component_interval_call`の呼び出し時に渡した`component`です。
2	`void *data`	ユーザーデータ `hive_component_interval_call`で指定した任意のポインタです。
3	`hive_time_t now`	各定期処理実行時に、この関数が呼ばれた時刻のタイムスタンプ
4	`int skipped`	定期処理が間に合わずにスキップされた場合のスキップ数
5	`HIVE_STATUS status`	エラー情報等の登録に使用する`status`

戻り値

- true: 定期処理をこのあとも継続する場合
- false: 定期処理を停止する場合

説明

API hive_component_interval_callを使用する際に指定する関数インタフェースのtypedefです。
定期処理を行う関数はこの宣言と一致するように定義してください。

この関数の戻り値により、このあとの定期処理が継続されるか、停止するかが決まります。エラーでこのあとの処理が継続できないような場合以外はtrueを返すようにしてください。

15.4.7.6 HIVE_COLUMN_READ_RESULT … カラム情報読み込み結果（ウィンドウデータ）

入力ポートからカラムの値を読み込んだ結果を格納する構造体

定義

```
typedef struct hive_HIVE_COLUMN_READ_RESULT {
  hive_time_t timerange_begin;
  hive_time_t timerange_end;
  const HIVE_INCOLUMN_LIST *incolumn_list;
  uint64_t window_id;
  uint64_t event_id;
  const struct hive_HIVE_RECORD *statistics_record;
  bool is_continued;
  const char *buffer;
  size_t buffer_size;
} HIVE_COLUMN_READ_RESULT;
```

メンバ

No	項目	説明
1	`hive_time_t timerange_begin`	この構造体に含まれるデータの時刻範囲の先頭時刻
2	`hive_time_t timerange_end`	この構造体に含まれるデータの時刻範囲の終端時刻
3	`const HIVE_INCOLUMN_LIST *incolumn_list`	入力ポートに接続されたカラムの一覧
6	`uint64_t window_id`	このデータのウィンドウID
6	`uint64_t event_id`	このデータのイベントID
6	`HIVE_RECORD *statistics_record`	集約レコード
6	`bool is_continued`	ウィンドウが次のREAD_RESULTに継続するか否か
4	`const char *buffer`	システムが内部で使用する領域。アクセスしないこと
5	`size_t buffer_size`	システムが内部で使用する領域。アクセスしないこと

説明

API hive_continuous_reader_read()を使用した際に戻り値として受け取るデータ型です。
この１つの構造体で１ウィンドウ分のデータを保持しています。ウィンドウ範囲は入力ポートの設定により変動します。

この構造体からデータを取得する場合、HIVE_GET_RECORD_ITERATOR()やHIVE_RECORD_ITERATOR_GET_NEXT()を使用してください。

15.4.7.7 HIVE_RECORD … カラムデータレコード

ウィンドウデータに含まれる同一タイムスタンプの各カラムのデータをまとめた構造体

定義

```
typedef struct hive_HIVE_RECORD {
  uint8_t record_type;
  uint8_t unused;
  uint16_t data_count;
  uint32_t data_size;
  uint64_t timestamp;
  uint32_t offset[0];
} HIVE_RECORD;
```

メンバ

No	項目	説明
1	`uint8_t record_type`	レコード種別
2	`uint8_t unused`	未使用領域
3	`uint16_t data_count`	レコード内のデータ数
4	`uint32_t data_size`	レコード内の全データのバイト数
5	`uint64_t timestamp`	レコードのタイムスタンプ
6	`uint32_t offset[0]`	システムが内部で使用する領域。アクセスしないこと

説明

入力ポートから取得したウィンドウデータ内の同一タイムスタンプのデータセットです。
HIVE_RECORD_ITERATOR_GET_NEXT()を使用して取得できます。
このレコードに含まれるデータ数は`data_count`にて取得できますが、ウィンドウデータのincolumn_listのカラム数と同じになります。
レコード内の各データには`HIVE_RECORD_GET_DATA(record, i)`でアクセスできます。

15.4.7.8 HIVE_RECORD_DATA … レコードデータ

レコードに含まれる個々のデータを持つ構造体

定義

```
typedef struct hive_HIVE_RECORD_DATA {
  uint16_t index;
  uint16_t stat_type;
  uint32_t data_size;
  uint64_t timestamp;
  char data[0];
} HIVE_RECORD_DATA;
```

メンバ

No	項目	説明
1	`uint16_t index`	未使用領域
2	`uint16_t stat_type`	本データの集約種別
3	`uint32_t data_size`	データサイズ
4	`uint64_t timestamp`	タイムスタンプ
5	`char data[0]`	データ

説明

レコード内の個々のデータを保持する構造体です。
メンバ `data` は、0要素のchar配列で定義されていますが、実際には `data_size` 分のメモリ領域が確保されています。
ただし、 `data_size` が0の場合、このタイムスタンプのデータがないことを示します。その場合も `data` のポインタは `NULL` にはならないことに注意してください。

15.4.7.9 HIVE_COLUMN_OPTIONS … カラムオプション

カラム作成時のオプション情報を格納する構造体

定義

```
typedef struct _tag_HIVE_COLUMN_OPTIONS {
  hive_column_type_t type;
  double sampling_rate;
  int compression_unit;
  int fixed_size_binary;
  int max_binary_size;
} HIVE_COLUMN_OPTIONS;

#define HIVE_COLUMN_OPTION_NONE ((HIVE_COLUMN_OPTIONS){column_type_low,0,0,0,0})
#define HIVE_COLUMN_OPTION_MIDDLE(r) ((HIVE_COLUMN_OPTIONS){column_type_middle,r,0,0,0})
```

メンバ

No	項目	説明
1	`hive_column_type_t type`	カラム種別（`column_type_low` or `column_type_middle`）
2	`double sampling_rate`	サンプリングレート
3	`int compression_unit`	圧縮単位
4	`int fixed_size_binary`	バイナリ型固定サイズ
5	`int max_binary_size`	バイナリ型最大サイズ

説明

hive_outport_create_column()でカラムを作成する際に、第５引数に指定するオプション情報です。必須ではありません。

マクロ

マクロ`HIVE_COLUMN_OPTION_NONE`はデフォルトのオプション定義です。通常はこれを使用してください。
マクロ`HIVE_COLUMN_OPTION_MIDDLE(r)`はミドル種別のカラムを定義するためのオプション情報です。ミドルカラムの場合、サンプリングレートの指定が必須となります。

15.4.7.10 HIVE_API_ERROR … APIエラー

APIエラー情報構造体

定義

```
typedef struct hive_HIVE_API_ERROR {
  uint32_t code;
  int system_errno;
  int db_errno;
} HIVE_API_ERROR;
```

メンバ

No	項目	説明
1	`uint32_t code`	エラーの種別を表す数値
2	`int system_errno`	システムのエラー番号
3	`int db_errno`	データベースのエラー番号

説明

APIエラー発生時に、そのエラーの種別や詳細情報を保持する構造体です。
エラーが発生した場合、`hive_get_api_error()` 関数によりこの情報を取得できます。

`code` には、発生したエラーを識別する数値が記録されています。

APIエラーがシステムエラーの場合、そのエラー番号が `system_errno` に格納されます。システムエラーではない場合は `system_errno` は0です。

データベースシステムのエラーが起因となっていた場合、そのエラー番号が `db_errno` に格納されます。データベースシステム起因でない場合は `db_errno` は0です。

15.5 カスタムUI JS-APIリファレンス

カスタムUIをHTMLで作成する際のJavaScriptのAPI仕様です。カスタムUIをHTMLで作成する手順については下記を参照してください。

カスタムUI作成 HTMLを利用した定義ファイルの作成

15.5.1 コールバック関数

パラメータ設定ダイアログが表示されるとき、および、設定を保存して閉じるときに、 SpeeDBee SynapseのUI画面本体とパラメータ情報をやり取りするためのコールバック関数が必要になります。 下記に説明する２つの関数 `onLoad`, `onSave` は、カスタムUI(HTML)内のJavaScriptで定義してください。 SpeeDBee SynapseのUI画面本体からコールされます。

15.5.1.1 onLoad … カスタムUI読み込みフック

カスタムコンポーネントの設定ダイアログがユーザーに表示された際にコールされるフック関数

定義

```
onLoad?: (editable: boolean, parameter: {}, newComponent: boolean, componentId?: string) =>
Promise<void>;
```

引数

- editable: 現在表示している設定ダイアログを編集可能とする場合はtrue, 編集不可の場合はfalse
- parameter: 保存済みのパラメータオブジェクト
- newComponent: コンポーネントの新規作成時にtrue, 更新時はfalse
- componentId: 対象コンポーネントインスタンスのID

戻り値

なし

説明

カスタムコンポーネントの設定ダイアログがユーザーに表示された際にコールされる関数を登録するプロパティです。 保存済みのパラメータを取得するために必要ですので、以下のようにして必ず登録してください。

```
window.customUiLib.onLoad = function (editable, parameter, newComponent, componentId) {
  :
};
```

この関数の引数として、以前保存したパラメータオブジェクトを受け取ることができます。 このオブジェクトを元に、HTML内の入力ボックス等に初期値を設定してください。

Runモードの場合は`editable`が`false`となっています。この場合はユーザーが設定を変更できないように、 入力項目等は非活性にするなどの対応をしてください。 非活性にしない場合ユーザーが画面上で値を変更できてしまいます。ただし、その場合でもパラメータの保存はできません。

第四引数の`componentId`は、`window.customUiLib.api`オブジェクト内のAPIを使用する際の引数として使用できます。 ただしコンポーネントの新規作成時には、まだコンポーネントが作成されていないため`componentId`は引数として渡されません。

15.5.1.2 onSave … パラメーター保存フック

カスタムコンポーネントの設定ダイアログで、ユーザーが保存ボタンを押してパラメーターを保存する際に自動で呼び出されるフック関数

定義

```
onSave?: () => Promise<{} | null>;
```

引数

なし

戻り値

- null: パラメータの保存をキャンセルする場合にnullを返します。
- {}: 保存したいパラメータを返します。

説明

ユーザーが保存ボタンを押した際に自動で実行される関数です。 保存時に行いたい処理を一つの関数として実装し、下記のように`onSave`に登録してください。

```
window.customUiLib.onSave = function () {
  :
  return ..;
};
```

この関数がコールされたら、保存するパラメータ情報をオブジェクトとして返却する必要があります。 オブジェクトはJSON化可能なデータとしてください。 返却されたオブジェクトが当該コンポーネントのパラメータとして保存され、設定ダイアログは閉じられます。

nullを返却した場合、パラメータの保存はキャンセルされます。この場合はダイアログも閉じられません。 カスタムUI画面でユーザーが入力したデータ等のフォーマットをチェックし、入力内容に問題がある場合などに、 この方法で保存をキャンセルできます。 ただしキャンセルされたことは自動ではユーザーに通知されませんので、どのような要因で保存をキャンセルしたのかについては カスタムUIのHTML内でエラー表示等を行うようにしてください。

15.5.2 SpeeDBee Synapse設定取得関数

`window.customUiLib.api` オブジェクトには、SpeeDBee Synapseの画面上で設定した様々な情報を取得することができるユーティリティ関数が定義されています。 これらの関数の使用は必須ではありませんが、適切に利用することで入力補助などの用途でユーザーの利便性、操作性を向上させることができます。

15.5.2.1 getComponentbases … コンポーネントベース一覧取得

コンポーネントベース情報の一覧を取得する関数

定義

```
window.customUiLib.api.getComponentbases: () => Promise<ComponentbaseResponse[]>;
```

引数

なし

戻り値

- ComponentbaseResponse[]: コンポーネントベース情報の一覧

説明

本関数は、コンポーネントベース情報を一覧で取得します。

15.5.2.2 getPanels … パネル一覧取得

パネル一覧を取得する関数

定義

```
window.customUiLib.api.getPanels: () => Promise<PanelResponse[]>;
```

引数

なし

戻り値

- PanelResponse[]: パネル情報の一覧

説明

本関数は、全パネルの情報を一覧で取得します。

15.5.2.3 getPanel … パネル情報取得

パネル情報を取得する関数

定義

```
window.customUiLib.api.getPanel: (id: number) => Promise<PanelResponse>;
```

引数

No	項目	説明
1	`id` *number*	パネルID

戻り値

- PanelResponse: パネル情報

説明

本関数は、指定したパネルIDのパネル情報を取得します。

15.5.2.4 getComponents … コンポーネント一覧取得

コンポーネント一覧を取得する関数

定義

```
window.customUiLib.api.getComponents: (panelId?: number) => Promise<ComponentResponse[]>;
```

引数

No	項目	説明
1	`panelId` *number | undefined*	取得するパネルのID（未指定の場合は全パネル）

戻り値

- ComponentResponse[]: コンポーネント一覧

説明

本関数は、コンポーネント情報を一覧で取得します。
パネルIDを指定すると、そのパネルに配置しているコンポーネントだけを取得できます。

15.5.2.5 getComponent … コンポーネント情報取得

コンポーネント情報を取得する関数

定義

```
window.customUiLib.api.getComponent: (id: string) => Promise<ComponentResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- ComponentResponse: コンポーネント情報

説明

本関数は、指定したコンポーネントIDのコンポーネント情報を取得します。

15.5.2.6 getInPort … 入力ポート設定取得

入力ポート設定を取得する関数

定義

```
window.customUiLib.api.getInPort: (id: string) => Promise<InportResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- InportResponse: コンポーネントの入力ポート設定

説明

本関数は、コンポーネントの入力ポート設定を取得します。
入力ポート設定には、送られてくるデータ対しての設定や集約、遅延時間、イベントについての設定が含まれています。

15.5.2.7 getInPortData … 入力ポート情報取得

入力ポートで受信しているデータについての情報を取得する関数

定義

```
window.customUiLib.api.getInPortData: (id: string) => Promise<PortDataResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- PortDataResponse: コンポーネントの入力ポートで受信しているデータの情報

説明

本関数は、コンポーネントの入力ポートで受信しているデータについての情報を取得します。
入力ポート情報には、送られてくるデータがどのような型か、配列かどうか等の情報が含まれています。

15.5.2.8 getInPortDataValue … 入力ポートデータ取得

入力ポートで受信しているデータを取得する関数

定義

```
window.customUiLib.api.getInPortDataValue: (id: string) => Promise<PortDataValueResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- PortDataValueResponse: コンポーネントの入力ポートで受信しているデータ

説明

本関数は、コンポーネントの入力ポートで受信しているデータやタイムスタンプを取得します。
取得できる値は各データごとに最新の一件となります。

15.5.2.9 getSystemPort … システムポート設定取得

システムポート設定を取得する関数

定義

```
window.customUiLib.api.getSystemPort: (id: string) => Promise<OutportResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- OutportResponse: コンポーネントのシステムポート設定

説明

本関数は、コンポーネントのシステムポート設定を取得します。
システムポート設定には、送信するデータの永続化、数値変換、登録条件や付与したコメントの設定が含まれています。

15.5.2.10 getSystemPortData … システムポート情報取得

システムポートから送信しているデータについての情報を取得する関数

定義

```
window.customUiLib.api.getSystemPortData: (id: string) => Promise<PortDataResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- PortDataResponse: コンポーネントのシステムポートから送信しているデータの情報

説明

本関数は、コンポーネントのシステムポートから送信しているデータについて取得します。
システムポート情報には、送るデータはどのような型か、配列かどうか等の情報が含まれています。

15.5.2.11 getSystemPortDataValue … システムポートデータ取得

システムポートで送信している最新のデータを取得する関数

定義

```
window.customUiLib.api.getSystemPortDataValue: (id: string) => Promise<PortDataValueResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- PortDataValueResponse: コンポーネントのシステムポートから送信しているデータ

説明

本関数は、コンポーネントのシステムポートから送信しているデータ、タイムスタンプを取得します。 取得する値は各データごとに最新の一件となります。

15.5.2.12 getOutPort … 出力ポート設定取得

出力ポート設定を取得する関数

定義

```
window.customUiLib.api.getOutPort: (id: string) => Promise<OutportResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- OutportResponse: コンポーネントの出力ポート設定

説明

本関数は、コンポーネントの出力ポート設定を取得します。
出力ポート設定には、送信するデータの永続化、数値変換、登録条件や付与したコメントの設定が含まれています。

15.5.2.13 getOutPortData … 出力ポート情報取得

出力ポートから送信しているデータについての情報を取得する関数

定義

```
window.customUiLib.api.getOutPortData: (id: string) => Promise<PortDataResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- PortDataResponse: コンポーネントの出力ポートから送信しているデータの情報

説明

本関数は、コンポーネントの出力ポートから送信しているデータについて取得します。
出力ポート情報には、送るデータはどのような型か、配列かどうか等の情報が含まれています。

15.5.2.14 getOutPortDataValue … 出力ポートデータ取得

出力ポートで送信している最新のデータを取得する関数

定義

```
window.customUiLib.api.getOutPortDataValue: (id: string) => Promise<PortDataValueResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するコンポーネントのID

戻り値

- PortDataValueResponse: コンポーネントの出力ポートから送信しているデータ

説明

本関数は、コンポーネントの出力ポートから送信しているデータ、タイムスタンプを取得します。 取得する値は各データごとに最新の一件となります。

15.5.2.15 runMethod … コンポーネントインスタンスメソッド実行

コンポーネントインスタンスメソッドを実行する関数です。

> **注意**
>
> メソッドを実行できるのは管理者以上です。

定義

```
runMethod: (id: string, methodName: string, parameter: string | object | null) =>
Promise<ComponentMethodResponse>;
```

引数

No	項目	説明
1	`id` *string*	実行するコンポーネントのID
2	`methodName` *string*	取得するメソッドの名称
3	`parameter` *string | object | null*	実行するメソッドに渡すパラメーター

戻り値

- ComponentMethodResponse: 実行したインスタンスメソッドの結果と戻り値

説明

本関数は、コンポーネントに実装されているインスタンスメソッドを実行します。
インスタンスメソッドについては を参照してください。

15.5.2.16 getFlowlinks … フローリンク一覧取得

フローリンク一覧を取得する関数

定義

```
window.customUiLib.api.getFlowlinks: (componentId?: string) => Promise<FlowlinkResponse[]>;
```

引数

No	項目	説明
1	`componentId` *string*	取得するコンポーネントのID（未指定の場合は全フローリンク）

戻り値

- FlowlinkResponse[]: フローリンク一覧

説明

本関数は、フローリンク情報を取得します。
コンポーネントIDを指定した場合はそのコンポーネントに接続されているフローリンクを、未指定の場合は全フローリンクを取得します。

15.5.2.17 getFlowlink … フローリンク情報取得

フローリンク情報を取得する関数

定義

```
window.customUiLib.api.getFlowlink: (id: string) => Promise<FlowlinkResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するフローリンクのID

戻り値

- FlowlinkResponse: フローリンク情報

説明

本関数は、フローリンク情報をIDで検索して取得します。

15.5.2.18 getFlowlinkFilter … フローリンクフィルター情報取得

フローリンクに設定されているフィルター情報を取得する関数

定義

```
window.customUiLib.api.getFlowlinkFilter: (id: string) => Promise<FlowlinkFiltersResponse>;
```

引数

No	項目	説明
1	`id` *string*	取得するフローリンクのID

戻り値

- FlowlinkFiltersResponse: フローリンク情報

説明

本関数は、フローリンクに設定されているフィルター情報をIDで検索して取得します。

15.5.3 型

15.5.3.1 ComponentbaseResponse

SpeeDBee Synapseに登録されているコンポーネントベース情報

定義

```
type ComponentbaseResponse = {
    id: string,
    name: string,
    tag: string,
    inports: number,
    outports: number,
    required_point: number,
    error_types: {
        id: string,
        parameter_names: string[],
    }[],
}
```

構成

No	項目	説明
1	`id`	コンポーネントベースID
2	`name`	コンポーネントベース名
3	`tag`	タグ
4	`inports`	入力ポート数
5	`outports`	出力ポート数

No	項目	説明
6	`required_point`	起動時に必要なポイント数
7	`error_types`	エラー種別オブジェクトの一覧

> *error_types*
>
> | No | 項目 | 説明 |
> | --- | --- | --- |
> | 1 | `id` | エラーID |
> | 2 | `parameter_names` | パラメータ名 |

説明

各コンポーネントのベース情報のデータ型です。
getComponentbasesを使用した際に戻り値として一覧を取得できます。

15.5.3.2 PanelResponse

パネル情報

定義

```
type PanelResponse = {
    id: number,
    order: number,
    name: string,
    type: string,
    options: any,
    components: any[],
};
```

構成

No	項目	説明
1	`id`	パネルID
2	`order`	表示順
3	`name`	名称

No	項目	説明
4	`type`	種別
5	`options`	オプション
6	`components`	コンポーネント情報

説明

登録されている各パネルのデータ型です。
getPanels、getPanelを使用した際に戻り値として取得できます。

15.5.3.3 ComponentResponse

コンポーネント情報

定義

```
type ComponentResponse = {
    id: string,
    baseid: string,
    name: string,
    enable: boolean,
    parameter: any | undefined,
    options: any | undefined,
};
```

構成

No	項目	説明
1	`id`	コンポーネントID
2	`baseid`	コンポーネントベースID
3	`name`	名称
4	`enable`	有効/無効
5	`parameter`	パラメーター
6	`options`	オプション

説明

ユーザーが設定した各コンポーネントを取得する際のデータ型です。
getComponents、getComponentを使用した際に戻り値として取得できます。

15.5.3.4 InportResponse

入力ポート設定

定義

```
type InportResponse = {
    setting_option: 'reception' | 'regexp',
    column_data: {
        target: {
            source_name: string,
            data_name: string,
            data_types?: string[],
        }[],
        raw_data: boolean,
        typical_value?: string,
        basic_statistics: string[],
    }[],
    window: {
        window_type: 'tumbling' | 'value_change',
        time_span?: number,
        columns?: {
            source_name: string,
            data_name: string,
            data_types: string[]
        },
        max_duration?: number
    },
    delaytime: number,
    event?: {
        source_name: string,
        data_name: string,
        pre_event_time: number,
        post_event_time: number
    },
    column_order_output?: {
        specify_columns: {
            source_name: string,
            data_name: string,
            outputname?: string,
        }[],
        order_type: string
    }
};
```

構成

No	項目	説明
1	`setting_option`	フィルターを正規表現で行うか、データごとに行うかどうか
2	`column_data`	入力データ情報
3	`window`	ウィンドウ設定
4	`delaytime`	遅延時間(ナノ秒)
5	`event`	入力ポートへのデータを外部のイベントによりON/OFFする設定
6	`column_order_output`	入力ポートに入ってきたデータを並べ替える設定と別名の設定

column_data

No	項目	説明
1	`target`	集約対象となるソースとデータのパターン
2	`raw_data`	未加工データの利用有無
3	`typical_value`	代表値の設定。最新値(newer)または最古値(older)
4	`basic_statistics`	基本統計データ

target

No	項目	説明
1	`source_name`	対象とするコンポーネント名のパターンを示す正規表現
2	`data_name`	対象とするデータ名のパターンを示す正規表現
3	`data_types`	対象とするデータ型のリスト(省略/nullは任意の型にマッチ)

window

No	項目	説明
1	`window_type`	ウィンドウ種別
2	`time_span`	検索時間幅(ナノ秒)
3	`columns`	対象とするデータ情報の一覧
4	`max_duration`	ウィンドウの最大時間幅(ナノ秒)

columns

No	項目	説明
1	`source_name`	対象とするコンポーネント名のパターンを示す正規表現
2	`data_name`	対象とするデータ名のパターンを示す正規表現
3	`data_types`	対象とするデータ型のリスト(省略/nullは任意の型にマッチ)

event

No	項目	説明
1	`source_name`	外部イベントのカラムを生成するコンポーネント名
2	`data_name`	外部イベントのカラム名(bool型のみ有効)
3	`pre_event_time`	プレイベント時間(ナノ秒)
4	`post_event_time`	ポストイベント時間(ナノ秒)

column_order_output

No	項目	説明
1	`specify_columns`	並び順と出力名
2	`order_type`	指定外のデータの並び順

specify_columns

No	項目	説明
1	`source_name`	コンポーネント名
2	`data_name`	データ名
3	`outputname`	カラムに付与する出力名

説明

コンポーネントの入力ポート設定のデータ型です。
getInPortを使用した際に戻り値として取得できます。

15.5.3.5 OutportResponse

システム、出力ポート設定

定義

```
type OutportResponse = {
    setting_option: 'reception' | 'regexp';
    pattern_settings: {
        target: {
            source_name?: string,
            data_name: string,
            data_types?: string[],
        }[],
        storage_type: string,
        file_index: boolean,
        conversion?: {
            data_type?: string,
            expression?: string,
        },
        value_filter?: {
            expression: string,
        },
        comment?: string,
    }[];
};
```

構成

No	項目	説明
1	`target`	対象データを指定するためのパターン定義
2	`storage_type`	永続化
3	`file_index`	検索用インデックスの作成フラグ
4	`conversion`	数値変換
5	`value_filter`	登録条件
6	`comment`	コメント

target

No	項目	説明
1	`source_name`	対象とするコンポーネント名のパターンを示す正規表現
2	`data_name`	対象とするデータ名のパターンを示す正規表現
3	`data_types`	対象とするデータ型のリスト(省略/nullは任意の型にマッチ)

conversion

No	項目	説明
1	`data_type`	変換後のデータ型
2	`expression`	変換式

value_filter

No	項目	説明
1	`expression`	条件式

コンポーネントの出力ポート設定のデータ型です。
getSystemPort、getOutPortを使用した際に戻り値として取得できます。

15.5.3.6 PortDataResponse

入出力ポートが受信、送信しているデータについての情報

定義

```
type PortDataResponse = {
    id: string;
    name: string;
    data?: {
        name: string;
        source?: string;
        column_type: string,
        sampling_rate?: number,
        data_type: string;
        data_array?: number;
    }[]
};
```

構成

No	項目	説明
1	`id`	コンポーネントID
2	`name`	名称
3	`data`	入力ポートに受信しているデータ情報

> *data*
>
> | No | 項目 | 説明 |
> |---|---|---|
> | 1 | `name` | データ名 |
> | 2 | `source` | コンポーネント名 |
> | 3 | `column_type` | カラム種別 |
> | 4 | `sampling_rate` | サンプリングレート |
> | 5 | `data_type` | データ型 |
> | 6 | `data_array` | 配列数 |

説明

コンポーネントの入出力ポートが受信、送信しているデータについてのデータ型です。
getInPortData、getSystemPortData、getOutPortDataを使用した際に戻り値として取得できます。

15.5.3.7 PortDataValueResponse

入出力ポートが受信、送信しているデータ

定義

```
type PortDataValueResponse = {
    id: string;
    name: string;
    data?: {
        value: any | any[];
        timestamp?: number;
        name: string;
        source?: string;
        data_type: string;
        data_array?: number;
    }[]
};
```

構成

No	項目	説明
1	`id`	コンポーネントID
2	`name`	名称
3	`data`	入力ポートに受信しているデータ

> *data*
>
> | No | 項目 | 説明 |
> | --- | --- | --- |
> | 3 | `value` | 値 |
> | 4 | `timestamp` | タイムスタンプ（ナノ秒） |
> | 1 | `name` | データ名 |
> | 2 | `source` | コンポーネント名 |
> | 5 | `data_type` | データ型 |
> | 6 | `data_array` | 配列数 |

説明

コンポーネントの入出力ポートが受信、送信しているデータのデータ型です。
getInPortDataValue、getSystemPortDataValue、getOutPortDataValueを使用した際に戻り値として取得できます。

15.5.3.8 ComponentMethodResponse

コンポーネントのインスタンスメソッドを実行した後の結果

定義

```
type ComponentMethodResponse = {
    method_name: string,
    result: string | object | null
};
```

構成

No	項目	説明
1	`method_name`	実行した関数名
2	`result`	実行した関数の戻り値

説明

コンポーネントのインスタンスメソッドを実行した後に返されるデータのデータ型です。
`result`の内容はカスタムコンポーネント側の実装によって変わるため、これ以降に行う処理に合わせてカスタムコンポーネント側の戻り値を実装してください。

15.5.3.9 FlowlinkResponse

フローリンク情報

定義

```
type FlowlinkResponse = {
    id: string,
    source?: {
        id: string,
        port: number,
    },
    destination?: {
        id: string,
        port: number,
    }
};
```

構成

No	項目	説明
1	`id`	フローリンクID
2	`source`	接続元の情報
3	`destination`	接続先の情報

> *source, destination*
>
> | No | 項目 | 説明 |
> | --- | --- | --- |
> | 1 | `id` | 接続先のポートID |
> | 2 | `port` | ポート番号 |

説明

コンポーネントが接続しているフローリンクのデータ型です。
getFlowlinks、getFlowlinkを使用した際に戻り値として取得できます。

15.5.3.10 FlowlinkFiltersResponse

フローリンクのフィルター情報

定義

```
type FlowlinkFiltersResponse = {
    setting_option: 'reception' | 'regexp',
    pattern_settings: {
        target: {
            data_name: string,
            data_types?: string[]
        },
        pass: boolean,
    }[],
};
```

構成

No	項目	説明
1	`setting_option`	フィルターを正規表現で行うか、データごとに行うかどうか
2	`pattern_settings`	フィルター設定オブジェクトの一覧

> *pattern_settins*
>
> | No | 項目 | 説明 |
> | --- | --- | --- |
> | 1 | `target` | 対象データを指定するためのパターン定義 |
> | 2 | `pass` | 値を通すかどうかの判定用フラグ |
>
> > *target*
> >
> > | No | 項目 | 説明 |
> > | --- | --- | --- |
> > | 1 | `data_name` | 対象とするデータ名のパターンを示す正規表現 |
> > | 2 | `data_types` | 対象とするデータ型のリスト(省略/nullは任意の型にマッチ) |

説明

コンポーネントが接続しているフローリンクに設定されたフィルターのデータ型です。 getFlowlinkFilterを使用した際に戻り値として取得できます。

SpeeDBee Synapseは、オープンソースを利用しています。 利用しているオープンソースについては、下記を参照してください。

- cpp-httplib

```
The MIT License (MIT)

Copyright (c) 2017 yhirose

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- cppcodec

```
Copyright (c) 2015 Topology Inc.
Copyright (c) 2018 Jakob Petsovits
Copyright (c) various other contributors, see individual files

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- ghc

```
Copyright (c) 2018, Steffen Schümann <s.schuemann@pobox.com>

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- martinmoene-optional-lite

```
Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization
obtaining a copy of the software and accompanying documentation covered by
this license (the "Software") to use, reproduce, display, distribute,
execute, and transmit the Software, and to prepare derivative works of the
Software, and to permit third-parties to whom the Software is furnished to
do so, all subject to the following:

The copyright notices in the Software and this entire statement, including
the above license grant, this restriction and the following disclaimer,
must be included in all copies of the Software, in whole or in part, and
all derivative works of the Software, unless such copies or derivative
works are solely in the form of machine-executable object code generated by
a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT
SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE
FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
DEALINGS IN THE SOFTWARE.
```

- martinmoene-variant-lite

```
Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization
obtaining a copy of the software and accompanying documentation covered by
this license (the "Software") to use, reproduce, display, distribute,
execute, and transmit the Software, and to prepare derivative works of the
Software, and to permit third-parties to whom the Software is furnished to
do so, all subject to the following:

The copyright notices in the Software and this entire statement, including
the above license grant, this restriction and the following disclaimer,
must be included in all copies of the Software, in whole or in part, and
all derivative works of the Software, unless such copies or derivative
works are solely in the form of machine-executable object code generated by
a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT
SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE
FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
DEALINGS IN THE SOFTWARE.
```

- nlohmann-json

```
MIT License

Copyright (c) 2013-2022 Niels Lohmann

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- tiny-AES-c

```
This is free and unencumbered software released into the public domain.

Anyone is free to copy, modify, publish, use, compile, sell, or
distribute this software, either in source code form or as a compiled
binary, for any purpose, commercial or non-commercial, and by any
means.

In jurisdictions that recognize copyright laws, the author or authors
of this software dedicate any and all copyright interest in the
software to the public domain. We make this dedication for the benefit
of the public at large and to the detriment of our heirs and
successors. We intend this dedication to be an overt act of
relinquishment in perpetuity of all present and future rights to this
software under copyright law.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR
OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR
OTHER DEALINGS IN THE SOFTWARE.

For more information, please refer to <http://unlicense.org/>
```

- PicoSHA2

```
MIT License

Copyright (c) 2017 okdshin

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- takamin/win-c

```
The MIT License (MIT)

Copyright (c) 2015 Koji Takami

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- tronkko/dirent

```
The MIT License (MIT)

Copyright (c) 1998-2019 Toni Ronkko

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- libuuid

```
Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, and the entire permission notice in its entirety,
   including the disclaimer of warranties.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote
   products derived from this software without specific prior
   written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESS OR IMPLIED
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ALL OF
WHICH ARE HEREBY DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
USE OF THIS SOFTWARE, EVEN IF NOT ADVISED OF THE POSSIBILITY OF SUCH
DAMAGE.
```

- libpython

```
1. This LICENSE AGREEMENT is between the Python Software Foundation
("PSF"), and the Individual or Organization ("Licensee") accessing and
otherwise using this software ("Python") in source or binary form and
its associated documentation.

2. Subject to the terms and conditions of this License Agreement, PSF
hereby grants Licensee a nonexclusive, royalty-free, world-wide
license to reproduce, analyze, test, perform and/or display publicly,
prepare derivative works, distribute, and otherwise use Python
alone or in any derivative version, provided, however, that PSF's
License Agreement and PSF's notice of copyright, i.e., "Copyright (c)
2001, 2002, 2003, 2004, 2005, 2006 Python Software Foundation; All Rights
Reserved" are retained in Python alone or in any derivative version
prepared by Licensee.

3. In the event Licensee prepares a derivative work that is based on
or incorporates Python or any part thereof, and wants to make
the derivative work available to others as provided herein, then
Licensee hereby agrees to include in any such work a brief summary of
the changes made to Python.

4. PSF is making Python available to Licensee on an "AS IS"
basis. PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR
IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND
DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS
FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON WILL NOT
INFRINGE ANY THIRD PARTY RIGHTS.

5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON
FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS
A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON,
OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material
breach of its terms and conditions.

7. Nothing in this License Agreement shall be deemed to create any
relationship of agency, partnership, or joint venture between PSF and
Licensee. This License Agreement does not grant permission to use PSF
trademarks or trade name in a trademark sense to endorse or promote
products or services of Licensee, or any third party.

8. By copying, installing or otherwise using Python, Licensee
agrees to be bound by the terms and conditions of this License
Agreement.

BEOPEN.COM LICENSE AGREEMENT FOR PYTHON 2.0
-----------------

BEOPEN PYTHON OPEN SOURCE LICENSE AGREEMENT VERSION 1

1. This LICENSE AGREEMENT is between BeOpen.com ("BeOpen"), having an
office at 160 Saratoga Avenue, Santa Clara, CA 95051, and the
Individual or Organization ("Licensee") accessing and otherwise using
this software in source or binary form and its associated
documentation ("the Software").

2. Subject to the terms and conditions of this BeOpen Python License
```

```
Agreement, BeOpen hereby grants Licensee a non-exclusive,
royalty-free, world-wide license to reproduce, analyze, test, perform
and/or display publicly, prepare derivative works, distribute, and
otherwise use the Software alone or in any derivative version,
provided, however, that the BeOpen Python License is retained in the
Software, alone or in any derivative version prepared by Licensee.

3. BeOpen is making the Software available to Licensee on an “AS IS”
basis. BEOPEN MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR
IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, BEOPEN MAKES NO AND
DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS
FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE WILL NOT
INFRINGE ANY THIRD PARTY RIGHTS.

4. BEOPEN SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF THE
SOFTWARE FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS
AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE, OR ANY
DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

5. This License Agreement will automatically terminate upon a material
breach of its terms and conditions.

6. This License Agreement shall be governed by and interpreted in all
respects by the law of the State of California, excluding conflict of
law provisions. Nothing in this License Agreement shall be deemed to
create any relationship of agency, partnership, or joint venture
between BeOpen and Licensee. This License Agreement does not grant
permission to use BeOpen trademarks or trade names in a trademark
sense to endorse or promote products or services of Licensee, or any
third party. As an exception, the “BeOpen Python” logos available at
http://www.pythonlabs.com/logos.html may be used according to the
permissions granted on that web page.

7. By copying, installing or otherwise using the software, Licensee
agrees to be bound by the terms and conditions of this License
Agreement.

CNRI OPEN SOURCE LICENSE AGREEMENT (for Python 1.6b1)
--------------------

IMPORTANT: PLEASE READ THE FOLLOWING AGREEMENT CAREFULLY.

BY CLICKING ON “ACCEPT” WHERE INDICATED BELOW, OR BY COPYING,
INSTALLING OR OTHERWISE USING PYTHON 1.6, beta 1 SOFTWARE, YOU ARE
DEEMED TO HAVE AGREED TO THE TERMS AND CONDITIONS OF THIS LICENSE
AGREEMENT.

1. This LICENSE AGREEMENT is between the Corporation for National
Research Initiatives, having an office at 1895 Preston White Drive,
Reston, VA 20191 (“CNRI”), and the Individual or Organization
(“Licensee”) accessing and otherwise using Python 1.6, beta 1
software in source or binary form and its associated documentation,
as released at the www.python.org Internet site on August 4, 2000
(“Python 1.6b1”).

2. Subject to the terms and conditions of this License Agreement, CNRI
hereby grants Licensee a non-exclusive, royalty-free, world-wide
license to reproduce, analyze, test, perform and/or display
publicly, prepare derivative works, distribute, and otherwise use
```

```
Python 1.6b1 alone or in any derivative version, provided, however,
that CNRIs License Agreement is retained in Python 1.6b1, alone or
in any derivative version prepared by Licensee.

Alternately, in lieu of CNRIs License Agreement, Licensee may
substitute the following text (omitting the quotes): “Python 1.6,
beta 1, is made available subject to the terms and conditions in
CNRIs License Agreement. This Agreement may be located on the
Internet using the following unique, persistent identifier (known
as a handle): 1895.22/1011. This Agreement may also be obtained
from a proxy server on the Internet using the
URL:http://hdl.handle.net/1895.22/1011”.

3. In the event Licensee prepares a derivative work that is based on
or incorporates Python 1.6b1 or any part thereof, and wants to make
the derivative work available to the public as provided herein,
then Licensee hereby agrees to indicate in any such work the nature
of the modifications made to Python 1.6b1.

4. CNRI is making Python 1.6b1 available to Licensee on an “AS IS”
basis. CNRI MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR
IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, CNRI MAKES NO AND
DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR
FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON 1.6b1
WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. CNRI SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF THE
SOFTWARE FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR
LOSS AS A RESULT OF USING, MODIFYING OR DISTRIBUTING PYTHON 1.6b1,
OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY
THEREOF.

6. This License Agreement will automatically terminate upon a material
breach of its terms and conditions.

7. This License Agreement shall be governed by and interpreted in all
respects by the law of the State of Virginia, excluding conflict of
law provisions. Nothing in this License Agreement shall be deemed
to create any relationship of agency, partnership, or joint venture
between CNRI and Licensee. This License Agreement does not grant
permission to use CNRI trademarks or trade name in a trademark
sense to endorse or promote products or services of Licensee, or
any third party.

8. By clicking on the “ACCEPT” button where indicated, or by copying,
installing or otherwise using Python 1.6b1, Licensee agrees to be
bound by the terms and conditions of this License Agreement.

ACCEPT

CWI LICENSE AGREEMENT FOR PYTHON 0.9.0 THROUGH 1.2
--------------------

Copyright (c) 1991 – 1995, Stichting Mathematisch Centrum Amsterdam,
The Netherlands. All rights reserved.

Permission to use, copy, modify, and distribute this software and its
documentation for any purpose and without fee is hereby granted,
provided that the above copyright notice appear in all copies and that
```

```
both that copyright notice and this permission notice appear in
supporting documentation, and that the name of Stichting Mathematisch
Centrum or CWI not be used in advertising or publicity pertaining to
distribution of the software without specific, written prior
permission.

STICHTING MATHEMATISCH CENTRUM DISCLAIMS ALL WARRANTIES WITH REGARD TO
THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND
FITNESS, IN NO EVENT SHALL STICHTING MATHEMATISCH CENTRUM BE LIABLE
FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
```

- pycomm3

```
The MIT License (MIT)

Copyright (c) 2020 Ian Ottoway
Copyright (c) 2014 Agostino Ruscito

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- paramiko

```
          GNU LESSER GENERAL PUBLIC LICENSE
               Version 2.1, February 1999

 Copyright (C) 1991, 1999 Free Software Foundation, Inc.
     51 Franklin Street, Suite 500, Boston, MA  02110-1335  USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL.  It also counts
 as the successor of the GNU Library Public License, version 2, hence
 the version number 2.1.]

                Preamble

  The licenses for most software are designed to take away your
freedom to share and change it.  By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

  This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it.  You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

  When we speak of free software, we are referring to freedom of use,
not price.  Our General Public Licenses are designed to make sure that
you have the freedom to distribute copies of free software (and charge
for this service if you wish); that you receive source code or can get
it if you want it; that you can change the software and use pieces of
it in new free programs; and that you are informed that you can do
these things.

  To protect your rights, we need to make restrictions that forbid
distributors to deny you these rights or to ask you to surrender these
rights.  These restrictions translate to certain responsibilities for
you if you distribute copies of the library or if you modify it.

  For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you.  You must make sure that they, too, receive or can get the source
code.  If you link other code with the library, you must provide
complete object files to the recipients, so that they can relink them
with the library after making changes to the library and recompiling
it.  And you must show them these terms so they know their rights.

  We protect your rights with a two-step method: (1) we copyright the
library, and (2) we offer you this license, which gives you legal
permission to copy, distribute and/or modify the library.

  To protect each distributor, we want to make it very clear that
there is no warranty for the free library.  Also, if the library is
modified by someone else and passed on, the recipients should know
that what they have is not the original version, so that the original
author's reputation will not be affected by problems that might be
introduced by others.
```

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE

```
   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

  0. This License Agreement applies to any software library or other
program which contains a notice placed by the copyright holder or
other authorized party saying it may be distributed under the terms of
this Lesser General Public License (also called "this License").
Each licensee is addressed as "you".

  A "library" means a collection of software functions and/or data
prepared so as to be conveniently linked with application programs
(which use some of those functions and data) to form executables.

  The "Library", below, refers to any such software library or work
which has been distributed under these terms.  A "work based on the
Library" means either the Library or any derivative work under
copyright law: that is to say, a work containing the Library or a
portion of it, either verbatim or with modifications and/or translated
straightforwardly into another language.  (Hereinafter, translation is
included without limitation in the term "modification".)

  "Source code" for a work means the preferred form of the work for
making modifications to it.  For a library, complete source code means
all the source code for all modules it contains, plus any associated
interface definition files, plus the scripts used to control compilation
and installation of the library.

  Activities other than copying, distribution and modification are not
covered by this License; they are outside its scope.  The act of
running a program using the Library is not restricted, and output from
such a program is covered only if its contents constitute a work based
on the Library (independent of the use of the Library in a tool for
writing it).  Whether that is true depends on what the Library does
and what the program that uses the Library does.

  1. You may copy and distribute verbatim copies of the Library's
complete source code as you receive it, in any medium, provided that
you conspicuously and appropriately publish on each copy an
appropriate copyright notice and disclaimer of warranty; keep intact
all the notices that refer to this License and to the absence of any
warranty; and distribute a copy of this License along with the
Library.

  You may charge a fee for the physical act of transferring a copy,
and you may at your option offer warranty protection in exchange for a
fee.


  2. You may modify your copy or copies of the Library or any portion
of it, thus forming a work based on the Library, and copy and
distribute such modifications or work under the terms of Section 1
above, provided that you also meet all of these conditions:

    a) The modified work must itself be a software library.

    b) You must cause the files modified to carry prominent notices
    stating that you changed the files and the date of any change.

    c) You must cause the whole of the work to be licensed at no
    charge to all third parties under the terms of this License.
```

    d) If a facility in the modified Library refers to a function or a
    table of data to be supplied by an application program that uses
    the facility, other than as an argument passed when the facility
    is invoked, then you must make a good faith effort to ensure that,
    in the event an application does not supply such function or
    table, the facility still operates, and performs whatever part of
    its purpose remains meaningful.

    (For example, a function in a library to compute square roots has
    a purpose that is entirely well-defined independent of the
    application. Therefore, Subsection 2d requires that any
    application-supplied function or table used by this function must
    be optional: if the application does not supply it, the square
    root function must still compute square roots.)

These requirements apply to the modified work as a whole. If
identifiable sections of that work are not derived from the Library,
and can be reasonably considered independent and separate works in
themselves, then this License, and its terms, do not apply to those
sections when you distribute them as separate works. But when you
distribute the same sections as part of a whole which is a work based
on the Library, the distribution of the whole must be on the terms of
this License, whose permissions for other licensees extend to the
entire whole, and thus to each and every part regardless of who wrote
it.

Thus, it is not the intent of this section to claim rights or contest
your rights to work written entirely by you; rather, the intent is to
exercise the right to control the distribution of derivative or
collective works based on the Library.

In addition, mere aggregation of another work not based on the Library
with the Library (or with a work based on the Library) on a volume of
a storage or distribution medium does not bring the other work under
the scope of this License.

  3. You may opt to apply the terms of the ordinary GNU General Public
License instead of this License to a given copy of the Library. To do
this, you must alter all the notices that refer to this License, so
that they refer to the ordinary GNU General Public License, version 2,
instead of to this License. (If a newer version than version 2 of the
ordinary GNU General Public License has appeared, then you can specify
that version instead if you wish.) Do not make any other change in
these notices.

  Once this change is made in a given copy, it is irreversible for
that copy, so the ordinary GNU General Public License applies to all
subsequent copies and derivative works made from that copy.

  This option is useful when you wish to copy part of the code of
the Library into a program that is not a library.

  4. You may copy and distribute the Library (or a portion or
derivative of it, under Section 2) in object code or executable form
under the terms of Sections 1 and 2 above provided that you accompany
it with the complete corresponding machine-readable source code, which
must be distributed under the terms of Sections 1 and 2 above on a

```
medium customarily used for software interchange.

  If distribution of object code is made by offering access to copy
from a designated place, then offering equivalent access to copy the
source code from the same place satisfies the requirement to
distribute the source code, even though third parties are not
compelled to copy the source along with the object code.

  5. A program that contains no derivative of any portion of the
Library, but is designed to work with the Library by being compiled or
linked with it, is called a "work that uses the Library".  Such a
work, in isolation, is not a derivative work of the Library, and
therefore falls outside the scope of this License.

  However, linking a "work that uses the Library" with the Library
creates an executable that is a derivative of the Library (because it
contains portions of the Library), rather than a "work that uses the
library".  The executable is therefore covered by this License.
Section 6 states terms for distribution of such executables.

  When a "work that uses the Library" uses material from a header file
that is part of the Library, the object code for the work may be a
derivative work of the Library even though the source code is not.
Whether this is true is especially significant if the work can be
linked without the Library, or if the work is itself a library.  The
threshold for this to be true is not precisely defined by law.

  If such an object file uses only numerical parameters, data
structure layouts and accessors, and small macros and small inline
functions (ten lines or less in length), then the use of the object
file is unrestricted, regardless of whether it is legally a derivative
work.  (Executables containing this object code plus portions of the
Library will still fall under Section 6.)

  Otherwise, if the work is a derivative of the Library, you may
distribute the object code for the work under the terms of Section 6.
Any executables containing that work also fall under Section 6,
whether or not they are linked directly with the Library itself.


  6. As an exception to the Sections above, you may also combine or
link a "work that uses the Library" with the Library to produce a
work containing portions of the Library, and distribute that work
under terms of your choice, provided that the terms permit
modification of the work for the customer's own use and reverse
engineering for debugging such modifications.

  You must give prominent notice with each copy of the work that the
Library is used in it and that the Library and its use are covered by
this License.  You must supply a copy of this License.  If the work
during execution displays copyright notices, you must include the
copyright notice for the Library among them, as well as a reference
directing the user to the copy of this License.  Also, you must do one
of these things:

    a) Accompany the work with the complete corresponding
    machine-readable source code for the Library including whatever
    changes were used in the work (which must be distributed under
    Sections 1 and 2 above); and, if the work is an executable linked
```

with the Library, with the complete machine-readable "work that
uses the Library", as object code and/or source code, so that the
user can modify the Library and then relink to produce a modified
executable containing the modified Library. (It is understood
that the user who changes the contents of definitions files in the
Library will not necessarily be able to recompile the application
to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the
Library. A suitable mechanism is one that (1) uses at run time a
copy of the library already present on the user's computer system,
rather than copying library functions into the executable, and (2)
will operate properly with a modified version of the library, if
the user installs one, as long as the modified version is
interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at
least three years, to give the same user the materials
specified in Subsection 6a, above, for a charge no more
than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy
from a designated place, offer equivalent access to copy the above
specified materials from the same place.

e) Verify that the user has already received a copy of these
materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the
Library" must include any data and utility programs needed for
reproducing the executable from it. However, as a special exception,
the materials to be distributed need not include anything that is
normally distributed (in either source or binary form) with the major
components (compiler, kernel, and so on) of the operating system on
which the executable runs, unless that component itself accompanies
the executable.

It may happen that this requirement contradicts the license
restrictions of other proprietary libraries that do not normally
accompany the operating system. Such a contradiction means you cannot
use both them and the Library together in an executable that you
distribute.

7. You may place library facilities that are a work based on the
Library side-by-side in a single library together with other library
facilities not covered by this License, and distribute such a combined
library, provided that the separate distribution of the work based on
the Library and of the other library facilities is otherwise
permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work
based on the Library, uncombined with any other library
facilities. This must be distributed under the terms of the
Sections above.

b) Give prominent notice with the combined library of the fact
that part of it is a work based on the Library, and explaining
where to find the accompanying uncombined form of the same work.

  8. You may not copy, modify, sublicense, link with, or distribute
the Library except as expressly provided under this License. Any
attempt otherwise to copy, modify, sublicense, link with, or
distribute the Library is void, and will automatically terminate your
rights under this License. However, parties who have received copies,
or rights, from you under this License will not have their licenses
terminated so long as such parties remain in full compliance.

  9. You are not required to accept this License, since you have not
signed it. However, nothing else grants you permission to modify or
distribute the Library or its derivative works. These actions are
prohibited by law if you do not accept this License. Therefore, by
modifying or distributing the Library (or any work based on the
Library), you indicate your acceptance of this License to do so, and
all its terms and conditions for copying, distributing or modifying
the Library or works based on it.

  10. Each time you redistribute the Library (or any work based on the
Library), the recipient automatically receives a license from the
original licensor to copy, distribute, link with or modify the Library
subject to these terms and conditions. You may not impose any further
restrictions on the recipients' exercise of the rights granted herein.
You are not responsible for enforcing compliance by third parties with
this License.

  11. If, as a consequence of a court judgment or allegation of patent
infringement or for any other reason (not limited to patent issues),
conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License. If you cannot
distribute so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you
may not distribute the Library at all. For example, if a patent
license would not permit royalty-free redistribution of the Library by
all those who receive copies directly or indirectly through you, then
the only way you could satisfy both it and this License would be to
refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any
particular circumstance, the balance of the section is intended to apply,
and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any
patents or other property right claims or to contest validity of any
such claims; this section has the sole purpose of protecting the
integrity of the free software distribution system which is
implemented by public license practices. Many people have made
generous contributions to the wide range of software distributed
through that system in reliance on consistent application of that
system; it is up to the author/donor to decide if he or she is willing
to distribute software through any other system and a licensee cannot
impose that choice.

This section is intended to make thoroughly clear what is believed to
be a consequence of the rest of this License.

  12. If the distribution and/or use of the Library is restricted in

certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

  13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

  14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

                NO WARRANTY

  15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

  16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

             END OF TERMS AND CONDITIONS

           How to Apply These Terms to Your New Libraries

  If you develop a new library, and you want it to be of the greatest

```
possible use to the public, we recommend making it free software that
everyone can redistribute and change.  You can do so by permitting
redistribution under these terms (or, alternatively, under the terms of the
ordinary General Public License).

  To apply these terms, attach the following notices to the library.  It is
safest to attach them to the start of each source file to most effectively
convey the exclusion of warranty; and each file should have at least the
"copyright" line and a pointer to where the full notice is found.

    <one line to give the library's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This library is free software; you can redistribute it and/or
    modify it under the terms of the GNU Lesser General Public
    License as published by the Free Software Foundation; either
    version 2.1 of the License, or (at your option) any later version.

    This library is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
    Lesser General Public License for more details.

    You should have received a copy of the GNU Lesser General Public
    License along with this library; if not, write to the Free Software
    Foundation, Inc., 51 Franklin Street, Suite 500, Boston, MA  02110-1335  USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary.  Here is a sample; alter the names:

  Yoyodyne, Inc., hereby disclaims all copyright interest in the
  library `Frob' (a library for tweaking knobs) written by James Random Hacker.

  <signature of Ty Coon>, 1 April 1990
  Ty Coon, President of Vice

That's all there is to it!
```

- wheel

```
MIT License

Copyright (c) 2012 Daniel Holth <dholth@fastmail.fm> and contributors

Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR
OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR
OTHER DEALINGS IN THE SOFTWARE.
```

- paho.MQTT

```
This project is dual licensed under the Eclipse Public License 2.0 and the
Eclipse Distribution License 1.0 as described in the epl-v20 and edl-v10 files.
```

- pahoMQTT

```
Eclipse Public License - v 2.0

This program and the accompanying materials
are made available under the terms of the Eclipse Public License v2.0
and Eclipse Distribution License v1.0 which accompany this distribution.

The Eclipse Public License is available at
  https://www.eclipse.org/legal/epl-2.0/
and the Eclipse Distribution License is available at
  http://www.eclipse.org/org/documents/edl-v10.php.

For an explanation of what dual-licensing means to you, see:
https://www.eclipse.org/legal/eplfaq.php#DUALLIC
```

- OpenSSH

```
                                 Apache License
                           Version 2.0, January 2004
                        https://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction,
   and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by
   the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all
   other entities that control, are controlled by, or are under common
   control with that entity. For the purposes of this definition,
   "control" means (i) the power, direct or indirect, to cause the
   direction or management of such entity, whether by contract or
   otherwise, or (ii) ownership of fifty percent (50%) or more of the
   outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity
   exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation
   source, and configuration files.

   "Object" form shall mean any form resulting from mechanical
   transformation or translation of a Source form, including but
   not limited to compiled object code, generated documentation,
   and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or
   Object form, made available under the License, as indicated by a
   copyright notice that is included in or attached to the work
   (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object
   form, that is based on (or derived from) the Work and for which the
   editorial revisions, annotations, elaborations, or other modifications
   represent, as a whole, an original work of authorship. For the purposes
   of this License, Derivative Works shall not include works that remain
   separable from, or merely link (or bind by name) to the interfaces of,
   the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including
   the original version of the Work and any modifications or additions
   to that Work or Derivative Works thereof, that is intentionally
   submitted to Licensor for inclusion in the Work by the copyright owner
   or by an individual or Legal Entity authorized to submit on behalf of
   the copyright owner. For the purposes of this definition, "submitted"
   means any form of electronic, verbal, or written communication sent
   to the Licensor or its representatives, including but not limited to
   communication on electronic mailing lists, source code control systems,
   and issue tracking systems that are managed by, or on behalf of, the
   Licensor for the purpose of discussing and improving the Work, but
```

```
      excluding communication that is conspicuously marked or otherwise
      designated in writing by the copyright owner as "Not a Contribution."

      "Contributor" shall mean Licensor and any individual or Legal Entity
      on behalf of whom a Contribution has been received by Licensor and
      subsequently incorporated within the Work.

   2. Grant of Copyright License. Subject to the terms and conditions of
      this License, each Contributor hereby grants to You a perpetual,
      worldwide, non-exclusive, no-charge, royalty-free, irrevocable
      copyright license to reproduce, prepare Derivative Works of,
      publicly display, publicly perform, sublicense, and distribute the
      Work and such Derivative Works in Source or Object form.

   3. Grant of Patent License. Subject to the terms and conditions of
      this License, each Contributor hereby grants to You a perpetual,
      worldwide, non-exclusive, no-charge, royalty-free, irrevocable
      (except as stated in this section) patent license to make, have made,
      use, offer to sell, sell, import, and otherwise transfer the Work,
      where such license applies only to those patent claims licensable
      by such Contributor that are necessarily infringed by their
      Contribution(s) alone or by combination of their Contribution(s)
      with the Work to which such Contribution(s) was submitted. If You
      institute patent litigation against any entity (including a
      cross-claim or counterclaim in a lawsuit) alleging that the Work
      or a Contribution incorporated within the Work constitutes direct
      or contributory patent infringement, then any patent licenses
      granted to You under this License for that Work shall terminate
      as of the date such litigation is filed.

   4. Redistribution. You may reproduce and distribute copies of the
      Work or Derivative Works thereof in any medium, with or without
      modifications, and in Source or Object form, provided that You
      meet the following conditions:

      (a) You must give any other recipients of the Work or
          Derivative Works a copy of this License; and

      (b) You must cause any modified files to carry prominent notices
          stating that You changed the files; and

      (c) You must retain, in the Source form of any Derivative Works
          that You distribute, all copyright, patent, trademark, and
          attribution notices from the Source form of the Work,
          excluding those notices that do not pertain to any part of
          the Derivative Works; and

      (d) If the Work includes a "NOTICE" text file as part of its
          distribution, then any Derivative Works that You distribute must
          include a readable copy of the attribution notices contained
          within such NOTICE file, excluding those notices that do not
          pertain to any part of the Derivative Works, in at least one
          of the following places: within a NOTICE text file distributed
          as part of the Derivative Works; within the Source form or
          documentation, if provided along with the Derivative Works; or,
          within a display generated by the Derivative Works, if and
          wherever such third-party notices normally appear. The contents
          of the NOTICE file are for informational purposes only and
          do not modify the License. You may add Your own attribution
```

notices within Derivative Works that You distribute, alongside
or as an addendum to the NOTICE text from the Work, provided
that such additional attribution notices cannot be construed
as modifying the License.

You may add Your own copyright statement to Your modifications and
may provide additional or different license terms and conditions
for use, reproduction, or distribution of Your modifications, or
for any such Derivative Works as a whole, provided Your use,
reproduction, and distribution of the Work otherwise complies with
the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

```
   END OF TERMS AND CONDITIONS
```

- json-c

```
Copyright (c) 2009-2012 Eric Haszlakiewicz

Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

----------------------------------------------------------------

Copyright (c) 2004, 2005 Metaparadigm Pte Ltd

Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- Protocol Buffers

```
Copyright 2008 Google Inc.  All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

    * Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above
copyright notice, this list of conditions and the following disclaimer
in the documentation and/or other materials provided with the
distribution.
    * Neither the name of Google Inc. nor the names of its
contributors may be used to endorse or promote products derived from
this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Code generated by the Protocol Buffer compiler is owned by the owner
of the input file used when generating it.  This code is not
standalone and requires a support library to be linked with it.  This
support library is itself covered by the above license.
```

- libcares(c-ares)

```
MIT License

Copyright (c) 1998 Massachusetts Institute of Technology
Copyright (c) 2007 - 2023 Daniel Stenberg with many contributors, see AUTHORS
file.

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so,
subject to the following conditions:

The above copyright notice and this permission notice (including the next
paragraph) shall be included in all copies or substantial portions of the
Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

- grpc

```
                                 Apache License
                           Version 2.0, January 2004
                        http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction,
   and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by
   the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all
   other entities that control, are controlled by, or are under common
   control with that entity. For the purposes of this definition,
   "control" means (i) the power, direct or indirect, to cause the
   direction or management of such entity, whether by contract or
   otherwise, or (ii) ownership of fifty percent (50%) or more of the
   outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity
   exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation
   source, and configuration files.

   "Object" form shall mean any form resulting from mechanical
   transformation or translation of a Source form, including but
   not limited to compiled object code, generated documentation,
   and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or
   Object form, made available under the License, as indicated by a
   copyright notice that is included in or attached to the work
   (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object
   form, that is based on (or derived from) the Work and for which the
   editorial revisions, annotations, elaborations, or other modifications
   represent, as a whole, an original work of authorship. For the purposes
   of this License, Derivative Works shall not include works that remain
   separable from, or merely link (or bind by name) to the interfaces of,
   the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including
   the original version of the Work and any modifications or additions
   to that Work or Derivative Works thereof, that is intentionally
   submitted to Licensor for inclusion in the Work by the copyright owner
   or by an individual or Legal Entity authorized to submit on behalf of
   the copyright owner. For the purposes of this definition, "submitted"
   means any form of electronic, verbal, or written communication sent
   to the Licensor or its representatives, including but not limited to
   communication on electronic mailing lists, source code control systems,
   and issue tracking systems that are managed by, or on behalf of, the
   Licensor for the purpose of discussing and improving the Work, but
```

```
   excluding communication that is conspicuously marked or otherwise
   designated in writing by the copyright owner as "Not a Contribution."

   "Contributor" shall mean Licensor and any individual or Legal Entity
   on behalf of whom a Contribution has been received by Licensor and
   subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   copyright license to reproduce, prepare Derivative Works of,
   publicly display, publicly perform, sublicense, and distribute the
   Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   (except as stated in this section) patent license to make, have made,
   use, offer to sell, sell, import, and otherwise transfer the Work,
   where such license applies only to those patent claims licensable
   by such Contributor that are necessarily infringed by their
   Contribution(s) alone or by combination of their Contribution(s)
   with the Work to which such Contribution(s) was submitted. If You
   institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the
   Work or Derivative Works thereof in any medium, with or without
   modifications, and in Source or Object form, provided that You
   meet the following conditions:

   (a) You must give any other recipients of the Work or
       Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices
       stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works
       that You distribute, all copyright, patent, trademark, and
       attribution notices from the Source form of the Work,
       excluding those notices that do not pertain to any part of
       the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its
       distribution, then any Derivative Works that You distribute must
       include a readable copy of the attribution notices contained
       within such NOTICE file, excluding those notices that do not
       pertain to any part of the Derivative Works, in at least one
       of the following places: within a NOTICE text file distributed
       as part of the Derivative Works; within the Source form or
       documentation, if provided along with the Derivative Works; or,
       within a display generated by the Derivative Works, if and
       wherever such third-party notices normally appear. The contents
       of the NOTICE file are for informational purposes only and
       do not modify the License. You may add Your own attribution
```

```
        notices within Derivative Works that You distribute, alongside
        or as an addendum to the NOTICE text from the Work, provided
        that such additional attribution notices cannot be construed
        as modifying the License.

    You may add Your own copyright statement to Your modifications and
    may provide additional or different license terms and conditions
    for use, reproduction, or distribution of Your modifications, or
    for any such Derivative Works as a whole, provided Your use,
    reproduction, and distribution of the Work otherwise complies with
    the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
    any Contribution intentionally submitted for inclusion in the Work
    by You to the Licensor shall be under the terms and conditions of
    this License, without any additional terms or conditions.
    Notwithstanding the above, nothing herein shall supersede or modify
    the terms of any separate license agreement you may have executed
    with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
    names, trademarks, service marks, or product names of the Licensor,
    except as required for reasonable and customary use in describing the
    origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
    agreed to in writing, Licensor provides the Work (and each
    Contributor provides its Contributions) on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
    implied, including, without limitation, any warranties or conditions
    of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
    PARTICULAR PURPOSE. You are solely responsible for determining the
    appropriateness of using or redistributing the Work and assume any
    risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
    whether in tort (including negligence), contract, or otherwise,
    unless required by applicable law (such as deliberate and grossly
    negligent acts) or agreed to in writing, shall any Contributor be
    liable to You for damages, including any direct, indirect, special,
    incidental, or consequential damages of any character arising as a
    result of this License or out of the use or inability to use the
    Work (including but not limited to damages for loss of goodwill,
    work stoppage, computer failure or malfunction, or any and all
    other commercial damages or losses), even if such Contributor
    has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
    the Work or Derivative Works thereof, You may choose to offer,
    and charge a fee for, acceptance of support, warranty, indemnity,
    or other liability obligations and/or rights consistent with this
    License. However, in accepting such obligations, You may act only
    on Your own behalf and on Your sole responsibility, not on behalf
    of any other Contributor, and only if You agree to indemnify,
    defend, and hold each Contributor harmless for any liability
    incurred by, or claims asserted against, such Contributor by reason
    of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS
```

```
   APPENDIX: How to apply the Apache License to your work.

      To apply the Apache License to your work, attach the following
      boilerplate notice, with the fields enclosed by brackets "[]"
      replaced with your own identifying information. (Don't include
      the brackets!)  The text should be enclosed in the appropriate
      comment syntax for the file format. We also recommend that a
      file or class name and description of purpose be included on the
      same "printed page" as the copyright notice for easier
      identification within third-party archives.

   Copyright [yyyy] [name of copyright owner]

   Licensed under the Apache License, Version 2.0 (the "License");
   you may not use this file except in compliance with the License.
   You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

   Unless required by applicable law or agreed to in writing, software
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
   See the License for the specific language governing permissions and
   limitations under the License.

-------------------------------------------------------------

BSD 3-Clause License

Copyright 2016, Google Inc.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its
contributors may be used to endorse or promote products derived from this
software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
THE POSSIBILITY OF SUCH DAMAGE.

-------------------------------------------------------------
```

```
Mozilla Public License Version 2.0
==================================

1. Definitions
--------------

1.1. "Contributor"
    means each individual or legal entity that creates, contributes to
    the creation of, or owns Covered Software.

1.2. "Contributor Version"
    means the combination of the Contributions of others (if any) used
    by a Contributor and that particular Contributor's Contribution.

1.3. "Contribution"
    means Covered Software of a particular Contributor.

1.4. "Covered Software"
    means Source Code Form to which the initial Contributor has attached
    the notice in Exhibit A, the Executable Form of such Source Code
    Form, and Modifications of such Source Code Form, in each case
    including portions thereof.

1.5. "Incompatible With Secondary Licenses"
    means

    (a) that the initial Contributor has attached the notice described
        in Exhibit B to the Covered Software; or

    (b) that the Covered Software was made available under the terms of
        version 1.1 or earlier of the License, but not also under the
        terms of a Secondary License.

1.6. "Executable Form"
    means any form of the work other than Source Code Form.

1.7. "Larger Work"
    means a work that combines Covered Software with other material, in
    a separate file or files, that is not Covered Software.

1.8. "License"
    means this document.

1.9. "Licensable"
    means having the right to grant, to the maximum extent possible,
    whether at the time of the initial grant or subsequently, any and
    all of the rights conveyed by this License.

1.10. "Modifications"
    means any of the following:

    (a) any file in Source Code Form that results from an addition to,
        deletion from, or modification of the contents of Covered
        Software; or

    (b) any new file in Source Code Form that contains any Covered
        Software.
```

1.11. "Patent Claims" of a Contributor
    means any patent claim(s), including without limitation, method,
    process, and apparatus claims, in any patent Licensable by such
    Contributor that would be infringed, but for the grant of the
    License, by the making, using, selling, offering for sale, having
    made, import, or transfer of either its Contributions or its
    Contributor Version.

1.12. "Secondary License"
    means either the GNU General Public License, Version 2.0, the GNU
    Lesser General Public License, Version 2.1, the GNU Affero General
    Public License, Version 3.0, or any later versions of those
    licenses.

1.13. "Source Code Form"
    means the form of the work preferred for making modifications.

1.14. "You" (or "Your")
    means an individual or a legal entity exercising rights under this
    License. For legal entities, "You" includes any entity that
    controls, is controlled by, or is under common control with You. For
    purposes of this definition, "control" means (a) the power, direct
    or indirect, to cause the direction or management of such entity,
    whether by contract or otherwise, or (b) ownership of more than
    fifty percent (50%) of the outstanding shares or beneficial
    ownership of such entity.

2. License Grants and Conditions

2.1. Grants

Each Contributor hereby grants You a world-wide, royalty-free,
non-exclusive license:

(a) under intellectual property rights (other than patent or trademark)
    Licensable by such Contributor to use, reproduce, make available,
    modify, display, perform, distribute, and otherwise exploit its
    Contributions, either on an unmodified basis, with Modifications, or
    as part of a Larger Work; and

(b) under Patent Claims of such Contributor to make, use, sell, offer
    for sale, have made, import, and otherwise transfer either its
    Contributions or its Contributor Version.

2.2. Effective Date

The licenses granted in Section 2.1 with respect to any Contribution
become effective for each Contribution on the date the Contributor first
distributes such Contribution.

2.3. Limitations on Grant Scope

The licenses granted in this Section 2 are the only rights granted under
this License. No additional rights or licenses will be implied from the
distribution or licensing of Covered Software under this License.
Notwithstanding Section 2.1(b) above, no patent license is granted by a
Contributor:

(a) for any code that a Contributor has removed from Covered Software;
    or

(b) for infringements caused by: (i) Your and any other third party's
    modifications of Covered Software, or (ii) the combination of its
    Contributions with other software (except as part of its Contributor
    Version); or

(c) under Patent Claims infringed by Covered Software in the absence of
    its Contributions.

This License does not grant any rights in the trademarks, service marks,
or logos of any Contributor (except as may be necessary to comply with
the notice requirements in Section 3.4).

2.4. Subsequent Licenses

No Contributor makes additional grants as a result of Your choice to
distribute the Covered Software under a subsequent version of this
License (see Section 10.2) or under the terms of a Secondary License (if
permitted under the terms of Section 3.3).

2.5. Representation

Each Contributor represents that the Contributor believes its
Contributions are its original creation(s) or it has sufficient rights
to grant the rights to its Contributions conveyed by this License.

2.6. Fair Use

This License is not intended to limit any rights You have under
applicable copyright doctrines of fair use, fair dealing, or other
equivalents.

2.7. Conditions

Sections 3.1, 3.2, 3.3, and 3.4 are conditions of the licenses granted
in Section 2.1.

3. Responsibilities

3.1. Distribution of Source Form

All distribution of Covered Software in Source Code Form, including any
Modifications that You create or to which You contribute, must be under
the terms of this License. You must inform recipients that the Source
Code Form of the Covered Software is governed by the terms of this
License, and how they can obtain a copy of this License. You may not
attempt to alter or restrict the recipients' rights in the Source Code
Form.

3.2. Distribution of Executable Form

If You distribute Covered Software in Executable Form then:

(a) such Covered Software must also be made available in Source Code
    Form, as described in Section 3.1, and You must inform recipients of
    the Executable Form how they can obtain a copy of such Source Code

Form by reasonable means in a timely manner, at a charge no more
than the cost of distribution to the recipient; and

(b) You may distribute such Executable Form under the terms of this
License, or sublicense it under different terms, provided that the
license for the Executable Form does not attempt to limit or alter
the recipients' rights in the Source Code Form under this License.

3.3. Distribution of a Larger Work

You may create and distribute a Larger Work under terms of Your choice,
provided that You also comply with the requirements of this License for
the Covered Software. If the Larger Work is a combination of Covered
Software with a work governed by one or more Secondary Licenses, and the
Covered Software is not Incompatible With Secondary Licenses, this
License permits You to additionally distribute such Covered Software
under the terms of such Secondary License(s), so that the recipient of
the Larger Work may, at their option, further distribute the Covered
Software under the terms of either this License or such Secondary
License(s).

3.4. Notices

You may not remove or alter the substance of any license notices
(including copyright notices, patent notices, disclaimers of warranty,
or limitations of liability) contained within the Source Code Form of
the Covered Software, except that You may alter any license notices to
the extent required to remedy known factual inaccuracies.

3.5. Application of Additional Terms

You may choose to offer, and to charge a fee for, warranty, support,
indemnity or liability obligations to one or more recipients of Covered
Software. However, You may do so only on Your own behalf, and not on
behalf of any Contributor. You must make it absolutely clear that any
such warranty, support, indemnity, or liability obligation is offered by
You alone, and You hereby agree to indemnify every Contributor for any
liability incurred by such Contributor as a result of warranty, support,
indemnity or liability terms You offer. You may include additional
disclaimers of warranty and limitations of liability specific to any
jurisdiction.

4. Inability to Comply Due to Statute or Regulation

If it is impossible for You to comply with any of the terms of this
License with respect to some or all of the Covered Software due to
statute, judicial order, or regulation then You must: (a) comply with
the terms of this License to the maximum extent possible; and (b)
describe the limitations and the code they affect. Such description must
be placed in a text file included with all distributions of the Covered
Software under this License. Except to the extent prohibited by statute
or regulation, such description must be sufficiently detailed for a
recipient of ordinary skill to be able to understand it.

5. Termination

5.1. The rights granted under this License will terminate automatically

```
if You fail to comply with any of its terms. However, if You become
compliant, then the rights granted under this License from a particular
Contributor are reinstated (a) provisionally, unless and until such
Contributor explicitly and finally terminates Your grants, and (b) on an
ongoing basis, if such Contributor fails to notify You of the
non-compliance by some reasonable means prior to 60 days after You have
come back into compliance. Moreover, Your grants from a particular
Contributor are reinstated on an ongoing basis if such Contributor
notifies You of the non-compliance by some reasonable means, this is the
first time You have received notice of non-compliance with this License
from such Contributor, and You become compliant prior to 30 days after
Your receipt of the notice.

5.2. If You initiate litigation against any entity by asserting a patent
infringement claim (excluding declaratory judgment actions,
counter-claims, and cross-claims) alleging that a Contributor Version
directly or indirectly infringes any patent, then the rights granted to
You by any and all Contributors for the Covered Software under Section
2.1 of this License shall terminate.

5.3. In the event of termination under Sections 5.1 or 5.2 above, all
end user license agreements (excluding distributors and resellers) which
have been validly granted by You or Your distributors under this License
prior to termination shall survive termination.

************************************************************************
*                                                                      *
*  6. Disclaimer of Warranty                                           *
*  -------------------------                                           *
*                                                                      *
*  Covered Software is provided under this License on an "as is"       *
*  basis, without warranty of any kind, either expressed, implied, or  *
*  statutory, including, without limitation, warranties that the       *
*  Covered Software is free of defects, merchantable, fit for a        *
*  particular purpose or non-infringing. The entire risk as to the     *
*  quality and performance of the Covered Software is with You.        *
*  Should any Covered Software prove defective in any respect, You     *
*  (not any Contributor) assume the cost of any necessary servicing,   *
*  repair, or correction. This disclaimer of warranty constitutes an   *
*  essential part of this License. No use of any Covered Software is   *
*  authorized under this License except under this disclaimer.         *
*                                                                      *
************************************************************************

************************************************************************
*                                                                      *
*  7. Limitation of Liability                                          *
*  --------------------------                                          *
*                                                                      *
*  Under no circumstances and under no legal theory, whether tort      *
*  (including negligence), contract, or otherwise, shall any           *
*  Contributor, or anyone who distributes Covered Software as          *
*  permitted above, be liable to You for any direct, indirect,         *
*  special, incidental, or consequential damages of any character      *
*  including, without limitation, damages for lost profits, loss of    *
*  goodwill, work stoppage, computer failure or malfunction, or any    *
*  and all other commercial damages or losses, even if such party      *
*  shall have been informed of the possibility of such damages. This   *
*  limitation of liability shall not apply to liability for death or   *
```

```
*  personal injury resulting from such party's negligence to the       *
*  extent applicable law prohibits such limitation. Some               *
*  jurisdictions do not allow the exclusion or limitation of           *
*  incidental or consequential damages, so this exclusion and          *
*  limitation may not apply to You.                                    *
*                                                                      *
************************************************************************

8. Litigation
-------------

Any litigation relating to this License may be brought only in the
courts of a jurisdiction where the defendant maintains its principal
place of business and such litigation shall be governed by laws of that
jurisdiction, without reference to its conflict-of-law provisions.
Nothing in this Section shall prevent a party's ability to bring
cross-claims or counter-claims.

9. Miscellaneous
----------------

This License represents the complete agreement concerning the subject
matter hereof. If any provision of this License is held to be
unenforceable, such provision shall be reformed only to the extent
necessary to make it enforceable. Any law or regulation which provides
that the language of a contract shall be construed against the drafter
shall not be used to construe this License against a Contributor.

10. Versions of the License
---------------------------

10.1. New Versions

Mozilla Foundation is the license steward. Except as provided in Section
10.3, no one other than the license steward has the right to modify or
publish new versions of this License. Each version will be given a
distinguishing version number.

10.2. Effect of New Versions

You may distribute the Covered Software under the terms of the version
of the License under which You originally received the Covered Software,
or under the terms of any subsequent version published by the license
steward.

10.3. Modified Versions

If you create software not governed by this License, and you want to
create a new license for such software, you may create and use a
modified version of this License if you rename the license and remove
any references to the name of the license steward (except to note that
such modified license differs from this License).

10.4. Distributing Source Code Form that is Incompatible With Secondary
Licenses

If You choose to distribute Source Code Form that is Incompatible With
Secondary Licenses under the terms of this version of the License, the
notice described in Exhibit B of this License must be attached.
```

```
Exhibit A - Source Code Form License Notice
-------------------------------------------

  This Source Code Form is subject to the terms of the Mozilla Public
  License, v. 2.0. If a copy of the MPL was not distributed with this
  file, You can obtain one at http://mozilla.org/MPL/2.0/.

If it is not possible or desirable to put the notice in a particular
file, then You may include the notice in a location (such as a LICENSE
file in a relevant directory) where a recipient would be likely to look
for such a notice.

You may add additional accurate notices of copyright ownership.

Exhibit B - "Incompatible With Secondary Licenses" Notice
---------------------------------------------------------

  This Source Code Form is "Incompatible With Secondary Licenses", as
  defined by the Mozilla Public License, v. 2.0.
```

- re2

```
// Copyright (c) 2009 The RE2 Authors. All rights reserved.
//
// Redistribution and use in source and binary forms, with or without
// modification, are permitted provided that the following conditions are
// met:
//
//    * Redistributions of source code must retain the above copyright
// notice, this list of conditions and the following disclaimer.
//    * Redistributions in binary form must reproduce the above
// copyright notice, this list of conditions and the following disclaimer
// in the documentation and/or other materials provided with the
// distribution.
//    * Neither the name of Google Inc. nor the names of its
// contributors may be used to endorse or promote products derived from
// this software without specific prior written permission.
//
// THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
// "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
// LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
// A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
// OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
// SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
// LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
// DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
// THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
// (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
// OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

- abseil

```
                                 Apache License
                           Version 2.0, January 2004
                        https://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction,
   and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by
   the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all
   other entities that control, are controlled by, or are under common
   control with that entity. For the purposes of this definition,
   "control" means (i) the power, direct or indirect, to cause the
   direction or management of such entity, whether by contract or
   otherwise, or (ii) ownership of fifty percent (50%) or more of the
   outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity
   exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation
   source, and configuration files.

   "Object" form shall mean any form resulting from mechanical
   transformation or translation of a Source form, including but
   not limited to compiled object code, generated documentation,
   and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or
   Object form, made available under the License, as indicated by a
   copyright notice that is included in or attached to the work
   (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object
   form, that is based on (or derived from) the Work and for which the
   editorial revisions, annotations, elaborations, or other modifications
   represent, as a whole, an original work of authorship. For the purposes
   of this License, Derivative Works shall not include works that remain
   separable from, or merely link (or bind by name) to the interfaces of,
   the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including
   the original version of the Work and any modifications or additions
   to that Work or Derivative Works thereof, that is intentionally
   submitted to Licensor for inclusion in the Work by the copyright owner
   or by an individual or Legal Entity authorized to submit on behalf of
   the copyright owner. For the purposes of this definition, "submitted"
   means any form of electronic, verbal, or written communication sent
   to the Licensor or its representatives, including but not limited to
   communication on electronic mailing lists, source code control systems,
   and issue tracking systems that are managed by, or on behalf of, the
   Licensor for the purpose of discussing and improving the Work, but
```

```
      excluding communication that is conspicuously marked or otherwise
      designated in writing by the copyright owner as "Not a Contribution."

      "Contributor" shall mean Licensor and any individual or Legal Entity
      on behalf of whom a Contribution has been received by Licensor and
      subsequently incorporated within the Work.

   2. Grant of Copyright License. Subject to the terms and conditions of
      this License, each Contributor hereby grants to You a perpetual,
      worldwide, non-exclusive, no-charge, royalty-free, irrevocable
      copyright license to reproduce, prepare Derivative Works of,
      publicly display, publicly perform, sublicense, and distribute the
      Work and such Derivative Works in Source or Object form.

   3. Grant of Patent License. Subject to the terms and conditions of
      this License, each Contributor hereby grants to You a perpetual,
      worldwide, non-exclusive, no-charge, royalty-free, irrevocable
      (except as stated in this section) patent license to make, have made,
      use, offer to sell, sell, import, and otherwise transfer the Work,
      where such license applies only to those patent claims licensable
      by such Contributor that are necessarily infringed by their
      Contribution(s) alone or by combination of their Contribution(s)
      with the Work to which such Contribution(s) was submitted. If You
      institute patent litigation against any entity (including a
      cross-claim or counterclaim in a lawsuit) alleging that the Work
      or a Contribution incorporated within the Work constitutes direct
      or contributory patent infringement, then any patent licenses
      granted to You under this License for that Work shall terminate
      as of the date such litigation is filed.

   4. Redistribution. You may reproduce and distribute copies of the
      Work or Derivative Works thereof in any medium, with or without
      modifications, and in Source or Object form, provided that You
      meet the following conditions:

      (a) You must give any other recipients of the Work or
          Derivative Works a copy of this License; and

      (b) You must cause any modified files to carry prominent notices
          stating that You changed the files; and

      (c) You must retain, in the Source form of any Derivative Works
          that You distribute, all copyright, patent, trademark, and
          attribution notices from the Source form of the Work,
          excluding those notices that do not pertain to any part of
          the Derivative Works; and

      (d) If the Work includes a "NOTICE" text file as part of its
          distribution, then any Derivative Works that You distribute must
          include a readable copy of the attribution notices contained
          within such NOTICE file, excluding those notices that do not
          pertain to any part of the Derivative Works, in at least one
          of the following places: within a NOTICE text file distributed
          as part of the Derivative Works; within the Source form or
          documentation, if provided along with the Derivative Works; or,
          within a display generated by the Derivative Works, if and
          wherever such third-party notices normally appear. The contents
          of the NOTICE file are for informational purposes only and
          do not modify the License. You may add Your own attribution
```

```
        notices within Derivative Works that You distribute, alongside
        or as an addendum to the NOTICE text from the Work, provided
        that such additional attribution notices cannot be construed
        as modifying the License.

    You may add Your own copyright statement to Your modifications and
    may provide additional or different license terms and conditions
    for use, reproduction, or distribution of Your modifications, or
    for any such Derivative Works as a whole, provided Your use,
    reproduction, and distribution of the Work otherwise complies with
    the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
    any Contribution intentionally submitted for inclusion in the Work
    by You to the Licensor shall be under the terms and conditions of
    this License, without any additional terms or conditions.
    Notwithstanding the above, nothing herein shall supersede or modify
    the terms of any separate license agreement you may have executed
    with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
    names, trademarks, service marks, or product names of the Licensor,
    except as required for reasonable and customary use in describing the
    origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
    agreed to in writing, Licensor provides the Work (and each
    Contributor provides its Contributions) on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
    implied, including, without limitation, any warranties or conditions
    of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
    PARTICULAR PURPOSE. You are solely responsible for determining the
    appropriateness of using or redistributing the Work and assume any
    risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
    whether in tort (including negligence), contract, or otherwise,
    unless required by applicable law (such as deliberate and grossly
    negligent acts) or agreed to in writing, shall any Contributor be
    liable to You for damages, including any direct, indirect, special,
    incidental, or consequential damages of any character arising as a
    result of this License or out of the use or inability to use the
    Work (including but not limited to damages for loss of goodwill,
    work stoppage, computer failure or malfunction, or any and all
    other commercial damages or losses), even if such Contributor
    has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
    the Work or Derivative Works thereof, You may choose to offer,
    and charge a fee for, acceptance of support, warranty, indemnity,
    or other liability obligations and/or rights consistent with this
    License. However, in accepting such obligations, You may act only
    on Your own behalf and on Your sole responsibility, not on behalf
    of any other Contributor, and only if You agree to indemnify,
    defend, and hold each Contributor harmless for any liability
    incurred by, or claims asserted against, such Contributor by reason
    of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS
```

```
   APPENDIX: How to apply the Apache License to your work.

      To apply the Apache License to your work, attach the following
      boilerplate notice, with the fields enclosed by brackets "[]"
      replaced with your own identifying information. (Don't include
      the brackets!)  The text should be enclosed in the appropriate
      comment syntax for the file format. We also recommend that a
      file or class name and description of purpose be included on the
      same "printed page" as the copyright notice for easier
      identification within third-party archives.

   Copyright [yyyy] [name of copyright owner]

   Licensed under the Apache License, Version 2.0 (the "License");
   you may not use this file except in compliance with the License.
   You may obtain a copy of the License at

       https://www.apache.org/licenses/LICENSE-2.0

   Unless required by applicable law or agreed to in writing, software
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
   See the License for the specific language governing permissions and
   limitations under the License.
```

- その他WEBUIで利用しているオープンソース

```
Name                          Version         License
 PyYAML                        6.0.2           MIT License
 aiofiles                      23.2.1          Apache Software License
 aiohappyeyeballs              2.4.4           Other/Proprietary License; Python Software Foundation
License
 aiohttp                       3.10.11         Apache Software License
 aiohttp-sse-client            0.2.1           Apache Software License
 aiorwlock                     1.3.0           Apache Software License
 aiosignal                     1.3.1           Apache Software License
 annotated-types               0.7.0           MIT License
 anyio                         4.5.2           MIT License
 async-timeout                 5.0.1           Apache Software License
 attrs                         25.3.0          MIT License
 bcrypt                        4.3.0           Apache Software License
 certifi                       2025.4.26       Mozilla Public License 2.0 (MPL 2.0)
 cffi                          1.17.1          MIT License
 click                         8.1.8           BSD License
 cryptography                  45.0.3          Apache-2.0 OR BSD-3-Clause
 ecdsa                         0.19.1          MIT
 exceptiongroup                1.3.0           MIT License
 execnet                       2.1.1           MIT License
 fastapi                       0.115.8         MIT License
 frozenlist                    1.5.0           Apache Software License
 grafana-api-sdk               0.3.1           Apache Software License
 h11                           0.14.0          MIT License
 httpcore                      0.17.3          BSD License
 httptools                     0.6.4           MIT License
 httpx                         0.24.1          BSD License
 idna                          3.10            BSD License
 importlib_resources           6.4.5           Apache Software License
 iniconfig                     2.1.0           MIT License
 itsdangerous                  2.1.2           BSD License
 jsonschema-specifications     2023.12.1       MIT License
 multidict                     6.1.0           Apache Software License
 mypy_extensions               1.1.0           MIT License
 packaging                     25.0            Apache Software License; BSD License
 passlib                       1.7.4           BSD
 pkgutil_resolve_name          1.3.10          MIT License
 pluggy                        1.5.0           MIT License
 propcache                     0.2.0           Apache Software License
 pyasn1                        0.6.1           BSD License
 pycodestyle                   2.12.1          MIT License
 pycparser                     2.22            BSD License
 pycryptodomex                 3.23.0          BSD License; Public Domain
 pydantic                      2.10.6          MIT License
 pydantic_core                 2.27.2          MIT License
 python-dotenv                 1.0.1           BSD License
 python-jose                   3.3.0           MIT License
 python-multipart              0.0.6           Apache Software License
 pyzipper                      0.3.6           MIT License
 referencing                   0.35.1          MIT License
 rpds-py                       0.20.1          MIT License
 rsa                           4.9.1           Apache Software License
 six                           1.17.0          MIT License
 sniffio                       1.3.1           Apache Software License; MIT License
 sse-starlette                 1.6.5           BSD License
 starlette                     0.44.0          BSD License
 types-pyasn1                  0.6.0.20240913  Apache Software License
```

```
 typing_extensions          4.13.2          Python Software Foundation License
 uvicorn                    0.23.2          BSD License
 uvloop                     0.21.0          Apache Software License; MIT License
 watchfiles                 0.24.0          MIT License
 websockets                 13.1            BSD License
 yarl                       1.15.2          Apache Software License
 zipp                       3.20.2          MIT License

--------------------------------------------------------------------

{
  "@babel/helper-string-parser@7.25.9": {
    "licenses": "MIT",
    "repository": "https://github.com/babel/babel",
    "publisher": "The Babel Team",
    "url": "https://babel.dev/team"
  },
  "@babel/helper-validator-identifier@7.25.9": {
    "licenses": "MIT",
    "repository": "https://github.com/babel/babel",
    "publisher": "The Babel Team",
    "url": "https://babel.dev/team"
  },
  "@babel/parser@7.26.8": {
    "licenses": "MIT",
    "repository": "https://github.com/babel/babel",
    "publisher": "The Babel Team",
    "url": "https://babel.dev/team"
  },
  "@babel/types@7.26.8": {
    "licenses": "MIT",
    "repository": "https://github.com/babel/babel",
    "publisher": "The Babel Team",
    "url": "https://babel.dev/team"
  },
  "@jridgewell/sourcemap-codec@1.5.0": {
    "licenses": "MIT",
    "repository": "https://github.com/jridgewell/sourcemap-codec",
    "publisher": "Rich Harris"
  },
  "@kurkle/color@0.3.4": {
    "licenses": "MIT",
    "repository": "https://github.com/kurkle/color",
    "publisher": "Jukka Kurkela"
  },
  "@quasar/extras@1.16.17": {
    "licenses": "MIT",
    "repository": "https://github.com/quasarframework/quasar",
    "publisher": "Razvan Stoenescu"
  },
  "@vue/compiler-core@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/compiler-dom@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
```

```
  },
  "@vue/compiler-sfc@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/compiler-ssr@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/reactivity@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/runtime-core@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/runtime-dom@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/server-renderer@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/shared@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "chart.js@4.4.7": {
    "licenses": "MIT",
    "repository": "https://github.com/chartjs/Chart.js"
  },
  "cross-env@7.0.3": {
    "licenses": "MIT",
    "repository": "https://github.com/kentcdodds/cross-env",
    "publisher": "Kent C. Dodds",
    "email": "me@kentcdodds.com",
    "url": "https://kentcdodds.com"
  },
  "cross-spawn@7.0.6": {
    "licenses": "MIT",
    "repository": "https://github.com/moxystudio/node-cross-spawn",
    "publisher": "André Cruz",
    "email": "andre@moxy.studio"
  },
  "csstype@3.1.3": {
    "licenses": "MIT",
    "repository": "https://github.com/frenic/csstype",
    "publisher": "Fredrik Nicol",
    "email": "fredrik.nicol@gmail.com"
  },
```

```
  "drawflow@0.0.59": {
    "licenses": "MIT",
    "repository": "https://github.com/jerosoler/Drawflow",
    "publisher": "Jero soler"
  },
  "emittery@1.1.0": {
    "licenses": "MIT",
    "repository": "https://github.com/sindresorhus/emittery",
    "publisher": "Sindre Sorhus",
    "email": "sindresorhus@gmail.com",
    "url": "https://sindresorhus.com"
  },
  "entities@4.5.0": {
    "licenses": "BSD-2-Clause",
    "repository": "https://github.com/fb55/entities",
    "publisher": "Felix Boehm",
    "email": "me@feedic.com"
  },
  "estree-walker@2.0.2": {
    "licenses": "MIT",
    "repository": "https://github.com/Rich-Harris/estree-walker",
    "publisher": "Rich Harris"
  },
  "hls.js@1.5.20": {
    "licenses": "Apache-2.0",
    "repository": "https://github.com/video-dev/hls.js"
  },
  "isexe@2.0.0": {
    "licenses": "ISC",
    "repository": "https://github.com/isaacs/isexe",
    "publisher": "Isaac Z. Schlueter",
    "email": "i@izs.me",
    "url": "http://blog.izs.me/"
  },
  "magic-string@0.30.17": {
    "licenses": "MIT",
    "repository": "https://github.com/rich-harris/magic-string",
    "publisher": "Rich Harris"
  },
  "maska@2.1.11": {
    "licenses": "MIT",
    "repository": "https://github.com/beholdr/maska",
    "publisher": "Alexander Shabunevich"
  },
  "nanoid@3.3.8": {
    "licenses": "MIT",
    "repository": "https://github.com/ai/nanoid",
    "publisher": "Andrey Sitnik",
    "email": "andrey@sitnik.ru"
  },
  "path-key@3.1.1": {
    "licenses": "MIT",
    "repository": "https://github.com/sindresorhus/path-key",
    "publisher": "Sindre Sorhus",
    "email": "sindresorhus@gmail.com",
    "url": "sindresorhus.com"
  },
  "picocolors@1.1.1": {
    "licenses": "ISC",
```

```
    "repository": "https://github.com/alexeyraspopov/picocolors",
    "publisher": "Alexey Raspopov"
  },
  "postcss@8.5.2": {
    "licenses": "MIT",
    "repository": "https://github.com/postcss/postcss",
    "publisher": "Andrey Sitnik",
    "email": "andrey@sitnik.ru"
  },
  "quasar@2.17.7": {
    "licenses": "MIT",
    "repository": "https://github.com/quasarframework/quasar",
    "publisher": "Razvan Stoenescu",
    "email": "razvan.stoenescu@gmail.com",
    "url": "https://quasar.dev"
  },
  "shebang-command@2.0.0": {
    "licenses": "MIT",
    "repository": "https://github.com/kevva/shebang-command",
    "publisher": "Kevin Mårtensson",
    "email": "kevinmartensson@gmail.com",
    "url": "github.com/kevva"
  },
  "shebang-regex@3.0.0": {
    "licenses": "MIT",
    "repository": "https://github.com/sindresorhus/shebang-regex",
    "publisher": "Sindre Sorhus",
    "email": "sindresorhus@gmail.com",
    "url": "sindresorhus.com"
  },
  "sortablejs@1.15.6": {
    "licenses": "MIT",
    "repository": "https://github.com/SortableJS/Sortable"
  },
  "source-map-js@1.2.1": {
    "licenses": "BSD-3-Clause",
    "repository": "https://github.com/7rulnik/source-map-js",
    "publisher": "Valentin 7rulnik Semirulnik",
    "email": "v7rulnik@gmail.com"
  },
  "typescript@5.7.3": {
    "licenses": "Apache-2.0",
    "repository": "https://github.com/microsoft/TypeScript",
    "publisher": "Microsoft Corp."
  },
  "vue-chartjs@5.3.2": {
    "licenses": "MIT",
    "repository": "https://github.com/apertureless/vue-chartjs",
    "publisher": "Jakub Juszczak",
    "email": "jakub@posteo.de"
  },
  "vue@3.5.13": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "when-json-met-bigint@0.27.0": {
    "licenses": "MIT",
    "repository": "https://github.com/haoadoresorange/when-json-met-bigint",
```

```
    "publisher": "haoadoresorange"
  },
  "which@2.0.2": {
    "licenses": "ISC",
    "repository": "https://github.com/isaacs/node-which",
    "publisher": "Isaac Z. Schlueter",
    "email": "i@izs.me",
    "url": "http://blog.izs.me"
  }
}
```

i-Triggerで利用しているオープンソースは、下記を参照してください。

- FFmpeg

  This software uses code of FFmpeg licensed under the LGPLv2.1

  its source can be downloaded here

```
License
Most files in FFmpeg are under the GNU Lesser General Public License version 2.1 or later
(LGPL v2.1+). Read the file COPYING.LGPLv2.1 for details. Some other files have MIT/X11/BSD-
style licenses. In combination the LGPL v2.1+ applies to FFmpeg.

Some optional parts of FFmpeg are licensed under the GNU General Public License version 2 or
later (GPL v2+). See the file COPYING.GPLv2 for details. None of these parts are used by
default, you have to explicitly pass --enable-gpl to configure to activate them. In this case,
FFmpeg's license changes to GPL v2+.

Specifically, the GPL parts of FFmpeg are:

libpostproc
optional x86 optimization in the files
libavcodec/x86/flac_dsp_gpl.asm
libavcodec/x86/idct_mmx.c
libavfilter/x86/vf_removegrain.asm
the following building and testing tools
compat/solaris/make_sunver.pl
doc/t2h.pm
doc/texi2pod.pl
libswresample/tests/swresample.c
tests/checkasm/*
tests/tiny_ssim.c
the following filters in libavfilter:
signature_lookup.c
vf_blackframe.c
vf_boxblur.c
vf_colormatrix.c
vf_cover_rect.c
vf_cropdetect.c
vf_delogo.c
vf_eq.c
vf_find_rect.c
vf_fspp.c
vf_histeq.c
vf_hqdn3d.c
vf_kerndeint.c
vf_lensfun.c (GPL version 3 or later)
vf_mcdeint.c
vf_mpdecimate.c
vf_nnedi.c
vf_owdenoise.c
vf_perspective.c
vf_phase.c
vf_pp.c
vf_pp7.c
vf_pullup.c
vf_repeatfields.c
vf_sab.c
vf_signature.c
vf_smartblur.c
vf_spp.c
vf_stereo3d.c
vf_super2xsai.c
vf_tinterlace.c
vf_uspp.c
vf_vaguedenoiser.c
```

vsrc_mptestsrc.c
Should you, for whatever reason, prefer to use version 3 of the (L)GPL, then the configure parameter --enable-version3 will activate this licensing option for you. Read the file COPYING.LGPLv3 or, if you have enabled GPL parts, COPYING.GPLv3 to learn the exact legal terms that apply in this case.

There are a handful of files under other licensing terms, namely:

The files libavcodec/jfdctfst.c, libavcodec/jfdctint_template.c and libavcodec/jrevdct.c are taken from libjpeg, see the top of the files for licensing details. Specifically note that you must credit the IJG in the documentation accompanying your program if you only distribute executables. You must also indicate any changes including additions and deletions to those three files in the documentation.
tests/reference.pnm is under the expat license.
External libraries
FFmpeg can be combined with a number of external libraries, which sometimes affect the licensing of binaries resulting from the combination.

Compatible libraries
The following libraries are under GPL version 2:

avisynth
frei0r
libcdio
libdavs2
librubberband
libvidstab
libx264
libx265
libxavs
libxavs2
libxvid
When combining them with FFmpeg, FFmpeg needs to be licensed as GPL as well by passing --enable-gpl to configure.

The following libraries are under LGPL version 3:

gmp
libaribb24
liblensfun
When combining them with FFmpeg, use the configure option --enable-version3 to upgrade FFmpeg to the LGPL v3.

The VMAF, mbedTLS, RK MPI, OpenCORE and VisualOn libraries are under the Apache License 2.0. That license is incompatible with the LGPL v2.1 and the GPL v2, but not with version 3 of those licenses. So to combine these libraries with FFmpeg, the license version needs to be upgraded by passing --enable-version3 to configure.

The smbclient library is under the GPL v3, to combine it with FFmpeg, the options --enable-gpl and --enable-version3 have to be passed to configure to upgrade FFmpeg to the GPL v3.

Incompatible libraries
There are certain libraries you can combine with FFmpeg whose licenses are not compatible with the GPL and/or the LGPL. If you wish to enable these libraries, even in circumstances that their license may be incompatible, pass --enable-nonfree to configure. This will cause the resulting binary to be unredistributable.

```
The Fraunhofer FDK AAC and OpenSSL libraries are under licenses which are incompatible with
the GPLv2 and v3. To the best of our knowledge, they are compatible with the LGPL.
```

- その他WEBUIで利用しているオープンソース(i-Trigger)

```
{
  "@babel/parser@7.24.8": {
    "licenses": "MIT",
    "repository": "https://github.com/babel/babel",
    "publisher": "The Babel Team",
    "url": "https://babel.dev/team"
  },
  "@jridgewell/sourcemap-codec@1.4.15": {
    "licenses": "MIT",
    "repository": "https://github.com/jridgewell/sourcemap-codec",
    "publisher": "Rich Harris"
  },
  "@kurkle/color@0.3.2": {
    "licenses": "MIT",
    "repository": "https://github.com/kurkle/color",
    "publisher": "Jukka Kurkela"
  },
  "@quasar/extras@1.16.12": {
    "licenses": "MIT",
    "repository": "https://github.com/quasarframework/quasar",
    "publisher": "Razvan Stoenescu"
  },
  "@vue/compiler-core@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/compiler-dom@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/compiler-sfc@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/compiler-ssr@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/reactivity@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/runtime-core@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/runtime-dom@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/server-renderer@3.4.33": {
```

```
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "@vue/shared@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "chart.js@4.4.2": {
    "licenses": "MIT",
    "repository": "https://github.com/chartjs/Chart.js"
  },
  "chartjs-adapter-moment@1.0.1": {
    "licenses": "MIT",
    "repository": "https://github.com/chartjs/chartjs-adapter-moment"
  },
  "chartjs-plugin-annotation@3.0.1": {
    "licenses": "MIT",
    "repository": "https://github.com/chartjs/chartjs-plugin-annotation",
    "publisher": "Evert Timberg",
    "email": "evert.timberg@gmail.com"
  },
  "cross-env@7.0.3": {
    "licenses": "MIT",
    "repository": "https://github.com/kentcdodds/cross-env",
    "publisher": "Kent C. Dodds",
    "email": "me@kentcdodds.com",
    "url": "https://kentcdodds.com"
  },
  "cross-spawn@7.0.3": {
    "licenses": "MIT",
    "repository": "https://github.com/moxystudio/node-cross-spawn",
    "publisher": "André Cruz",
    "email": "andre@moxy.studio"
  },
  "csstype@3.1.3": {
    "licenses": "MIT",
    "repository": "https://github.com/frenic/csstype",
    "publisher": "Fredrik Nicol",
    "email": "fredrik.nicol@gmail.com"
  },
  "drawflow@0.0.59": {
    "licenses": "MIT",
    "repository": "https://github.com/jerosoler/Drawflow",
    "publisher": "Jero soler"
  },
  "emittery@1.0.1": {
    "licenses": "MIT",
    "repository": "https://github.com/sindresorhus/emittery",
    "publisher": "Sindre Sorhus",
    "email": "sindresorhus@gmail.com",
    "url": "https://sindresorhus.com"
  },
  "entities@4.5.0": {
    "licenses": "BSD-2-Clause",
    "repository": "https://github.com/fb55/entities",
    "publisher": "Felix Boehm",
    "email": "me@feedic.com"
```

```
  },
  "estree-walker@2.0.2": {
    "licenses": "MIT",
    "repository": "https://github.com/Rich-Harris/estree-walker",
    "publisher": "Rich Harris"
  },
  "hls.js@1.5.8": {
    "licenses": "Apache-2.0",
    "repository": "https://github.com/video-dev/hls.js"
  },
  "isexe@2.0.0": {
    "licenses": "ISC",
    "repository": "https://github.com/isaacs/isexe",
    "publisher": "Isaac Z. Schlueter",
    "email": "i@izs.me",
    "url": "http://blog.izs.me/"
  },
  "magic-string@0.30.10": {
    "licenses": "MIT",
    "repository": "https://github.com/rich-harris/magic-string",
    "publisher": "Rich Harris"
  },
  "maska@2.1.9": {
    "licenses": "MIT",
    "repository": "https://github.com/beholdr/maska",
    "publisher": "Alexander Shabunevich"
  },
  "moment@2.29.4": {
    "licenses": "MIT",
    "repository": "https://github.com/moment/moment",
    "publisher": "Iskren Ivov Chernev",
    "email": "iskren.chernev@gmail.com",
    "url": "https://github.com/ichernev"
  },
  "nanoid@3.3.7": {
    "licenses": "MIT",
    "repository": "https://github.com/ai/nanoid",
    "publisher": "Andrey Sitnik",
    "email": "andrey@sitnik.ru"
  },
  "path-key@3.1.1": {
    "licenses": "MIT",
    "repository": "https://github.com/sindresorhus/path-key",
    "publisher": "Sindre Sorhus",
    "email": "sindresorhus@gmail.com",
    "url": "sindresorhus.com"
  },
  "picocolors@1.0.1": {
    "licenses": "ISC",
    "repository": "https://github.com/alexeyraspopov/picocolors",
    "publisher": "Alexey Raspopov"
  },
  "postcss@8.4.41": {
    "licenses": "MIT",
    "repository": "https://github.com/postcss/postcss",
    "publisher": "Andrey Sitnik",
    "email": "andrey@sitnik.ru"
  },
  "quasar@2.16.6": {
```

```
    "licenses": "MIT",
    "repository": "https://github.com/quasarframework/quasar",
    "publisher": "Razvan Stoenescu",
    "email": "razvan.stoenescu@gmail.com",
    "url": "https://quasar.dev"
  },
  "shebang-command@2.0.0": {
    "licenses": "MIT",
    "repository": "https://github.com/kevva/shebang-command",
    "publisher": "Kevin Mårtensson",
    "email": "kevinmartensson@gmail.com",
    "url": "github.com/kevva"
  },
  "shebang-regex@3.0.0": {
    "licenses": "MIT",
    "repository": "https://github.com/sindresorhus/shebang-regex",
    "publisher": "Sindre Sorhus",
    "email": "sindresorhus@gmail.com",
    "url": "sindresorhus.com"
  },
  "sortablejs@1.15.0": {
    "licenses": "MIT",
    "repository": "https://github.com/SortableJS/Sortable"
  },
  "source-map-js@1.2.0": {
    "licenses": "BSD-3-Clause",
    "repository": "https://github.com/7rulnik/source-map-js",
    "publisher": "Valentin 7rulnik Semirulnik",
    "email": "v7rulnik@gmail.com"
  },
  "typescript@5.0.4": {
    "licenses": "Apache-2.0",
    "repository": "https://github.com/Microsoft/TypeScript",
    "publisher": "Microsoft Corp."
  },
  "vue-chartjs@5.3.0": {
    "licenses": "MIT",
    "repository": "https://github.com/apertureless/vue-chartjs",
    "publisher": "Jakub Juszczak",
    "email": "jakub@posteo.de"
  },
  "vue@3.4.33": {
    "licenses": "MIT",
    "repository": "https://github.com/vuejs/core",
    "publisher": "Evan You"
  },
  "when-json-met-bigint@0.27.0": {
    "licenses": "MIT",
    "repository": "https://github.com/haoadoresorange/when-json-met-bigint",
    "publisher": "haoadoresorange"
  },
  "which@2.0.2": {
    "licenses": "ISC",
    "repository": "https://github.com/isaacs/node-which",
    "publisher": "Isaac Z. Schlueter",
    "email": "i@izs.me",
    "url": "http://blog.izs.me"
```

```
    }
}
```

15.7 索引

15.7.1 C

Cloudエミッタ(AWS用)

各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
セキュア利用ガイド/暗号化通信の利用
付録/エラーコード一覧/Cloudエミッタ(AWS用)

Cloudエミッタ(Azure用)

各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
セキュア利用ガイド/暗号化通信の利用
付録/エラーコード一覧/Cloudエミッタ(Azure用)

CSVシリアライザ

導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/ロジック（イベント）/イベントトリガ
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/CSVシリアライザ

15.7.2 D

DBクエリサービス

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面操作/設定メニュー/アクセストークン発行
各種機能説明/システムコンポーネント/DBクエリサービス
各種機能説明/DBクエリ（SQL）/DBクエリの概要
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/Web-API仕様
各種機能説明/DBクエリ（SQL）/DBクエリサービス利用時の注意点
応用的な利用方法/Grafanaとの連携/グラフの作成/データの登録
応用的な利用方法/Grafanaとの連携/グラフの作成/DBクエリサービスの起動
応用的な利用方法/Grafanaとの連携/Grafanaデータソースの手動設定
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンス稼働環境に関する注意事項
付録/エラーコード一覧/DBクエリサービス

15.7.3 E

Edit Mode

導入フロー/初期設定/管理者パスワードの変更
各種機能説明/フローエディタ/モードについて
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー
各種機能説明/フローエディタ/画面操作/パネル/追加
各種機能説明/フローエディタ/画面操作/パネル/名前の変更
各種機能説明/フローエディタ/画面操作/パネル/並び替え
各種機能説明/フローエディタ/画面操作/パネル/削除
各種機能説明/フローエディタ/画面操作/パネル/複製
各種機能説明/フローエディタ/画面操作/コンポーネント/追加
各種機能説明/フローエディタ/画面操作/コンポーネント/編集
各種機能説明/フローエディタ/画面操作/コンポーネント/削除
各種機能説明/フローエディタ/画面操作/コンポーネント/コピー&ペースト
各種機能説明/フローエディタ/画面操作/コンポーネント/複数選択
各種機能説明/フローエディタ/画面操作/フローリンク
各種機能説明/各種ログ/コンポーネント個別ログ/個別ログレベル設定
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/編集制限の制御
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントのテスト実行
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/登録/更新
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/メール送信の設定
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス稼働環境のセットアップ

EtherNet/IPコレクタ

各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの概要
各種機能説明/コレクタ/EtherNet/IPコレクタ
付録/エラーコード一覧/EtherNet/IPコレクタ

EtherNet/IP書き込み

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/アクション/EtherNet/IP書き込み
付録/エラーコード一覧/EtherNet/IP書き込み

Event Recorder

各種機能説明/フローエディタ/画面操作/設定メニュー/データディレクトリ設定
各種機能説明/システムコンポーネント/イベントデータマネージャー
i-Trigger（拡張パッケージ）/動画再生用ビューワ/イベント録画再生
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder
付録/エラーコード一覧/Event Recorder（i-Trigger 拡張パッケージ）

15.7.4 F

FFT

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/FFT

Fileエミッタ

SpeeDBee Synapseの概要/運用までの流れ
導入フロー/コンポーネントの設定/Fileエミッタの設定
導入フロー/収集サービスの実行/動作確認
各種機能説明/フローエディタ/画面操作/設定メニュー/コンポーネント共通
各種機能説明/フローエディタ/画面操作/設定メニュー/データディレクトリ設定
各種機能説明/エミッタ/Fileエミッタ
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/Fileエミッタ

FTP/FTPS/SFTPエミッタ

各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
セキュア利用ガイド/暗号化通信の利用
付録/エラーコード一覧/FTP/FTPS/SFTPエミッタ

15.7.5 G

Grafana

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/Grafanaセットアップ
各種機能説明/フローエディタ/画面操作/設定メニュー/アクセストークン発行
各種機能説明/コンポーネント共通の設定項目/出力ポート/永続化の設定
各種機能説明/コンポーネント共通の設定項目/システムポート/永続化の設定
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
応用的な利用方法/Grafanaとの連携/Grafana連携の概要
応用的な利用方法/Grafanaとの連携/Grafanaのインストール
応用的な利用方法/Grafanaとの連携/Grafanaのセットアップ
応用的な利用方法/Grafanaとの連携/グラフの作成
応用的な利用方法/Grafanaとの連携/グラフの閲覧
応用的な利用方法/Grafanaとの連携/Grafanaデータソースの手動設定
実行ライセンス/実行ライセンス利用環境の動作仕様

15.7.6 I

i-Trigger

導入フロー/インストール（Windows）/インストールおよびアップデート
各種機能説明/システムコンポーネント/イベントデータマネージャー
i-Trigger（拡張パッケージ）/インストール/インストールおよびアップデート
i-Trigger（拡張パッケージ）/インストール/アンインストール
付録/エラーコード一覧/イベントデータマネージャー
付録/エラーコード一覧/VIDEOコレクタ（i-Trigger 拡張パッケージ）
付録/エラーコード一覧/Event Recorder（i-Trigger 拡張パッケージ）

15.7.7 J

JSONシリアライザ

各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面操作/設定メニュー/コンポーネント共通
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/シリアライザ/JSONシリアライザ
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/JSONシリアライザ

15.7.8 M

Modbus TCPコレクタ

各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの概要
各種機能説明/コレクタ/Modbus TCPコレクタ
付録/エラーコード一覧/Modbus TCPコレクタ

Modbus書き込み

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/アクション/Modbus書き込み
付録/エラーコード一覧/Modbus書き込み

MQTTコレクタ

各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの概要
各種機能説明/コレクタ/MQTTコレクタ
セキュア利用ガイド/暗号化通信の利用
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス稼働環境のセットアップ
付録/エラーコード一覧/MQTTコレクタ

MQTTエミッタ

各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/フローエディタ/画面操作/設定メニュー/ポイントについて
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/エミッタ/MQTTエミッタ
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/MQTTエミッタ

MQTTプロトコル

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/エミッタ/MQTTエミッタ

15.7.9 P

PLCコレクタ

各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの概要
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの設定・切替
各種機能説明/コレクタ/PLCコレクタ
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
トラブルシュート/よくあるトラブルとその対処方法/PLCコレクタに接続できない場合の対処方法
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル
付録/エラーコード一覧/PLCコレクタ

PLC書き込み

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/アクション/PLC書き込み
付録/エラーコード一覧/PLC書き込み

15.7.10 R

Run Mode

導入フロー/初期設定/ユーザーの追加
導入フロー/収集サービスの実行/動作確認
各種機能説明/フローエディタ/モードについて
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/フローエディタ/画面操作/ユーザー管理/追加
各種機能説明/フローエディタ/画面操作/ユーザー管理/編集
各種機能説明/フローエディタ/画面操作/パネル/全体パネル
各種機能説明/フローエディタ/画面操作/パネル/パネルステータス
各種機能説明/フローエディタ/画面操作/コンポーネント/設定内容の確認
各種機能説明/フローエディタ/画面操作/システムリソース
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/パラメータダイアログが表示されたときに実行される関数
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/編集制限の制御
実行ライセンス/実行ライセンス利用環境の動作仕様

15.7.11 S

SQL

各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/機能一覧/データベース
各種機能説明/システムコンポーネント/DBクエリサービス
各種機能説明/コンポーネント共通の設定項目/出力ポート/永続化の設定
各種機能説明/コンポーネント共通の設定項目/入力ポート/集約の設定
各種機能説明/コンポーネント共通の設定項目/システムポート/永続化の設定
各種機能説明/DBクエリ（SQL）/DBクエリの概要
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SET
応用的な利用方法/グローバル変数/グローバル変数の概要

15.7.12 V

VIDEOコレクタ

各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面操作/設定メニュー/データディレクトリ設定
i-Trigger（拡張パッケージ）/インストール/アンインストール
i-Trigger（拡張パッケージ）/動画再生用ビューワ/常時録画再生
i-Trigger（拡張パッケージ）/動画再生用ビューワ/イベント録画再生/再生
i-Trigger（拡張パッケージ）/動画再生用ビューワ/ライブ再生
i-Trigger（拡張パッケージ）/関連コンポーネント/VIDEOコレクタ
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder
付録/エラーコード一覧/VIDEOコレクタ（i-Trigger 拡張パッケージ）

15.7.13 W

WEBUI

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/各種ログ/ログの種別
各種機能説明/各種ログ/ダウンロード
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント/実行

WebAPI

各種機能説明/システムコンポーネント/DBクエリサービス
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPIの概要
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPIの実行方法
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様

15.7.14 あ

アクション

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode

アクセストークン

各種機能説明/フローエディタ/画面操作/設定メニュー/アクセストークン発行
各種機能説明/システムコンポーネント/DBクエリサービス
各種機能説明/DBクエリ（SQL）/DBクエリサービス利用時の注意点
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/共通事項
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/全コンポーネントの実行状態取得
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/個別コンポーネントの実行状態取得
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行開始
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行停止
応用的な利用方法/Grafanaとの連携/グラフの作成/データの登録
応用的な利用方法/Grafanaとの連携/Grafanaデータソースの手動設定
セキュア利用ガイド/トークンの厳重管理
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンス稼働環境に関する注意事項

HTTP/HTTPS通信

導入フロー/インストール（Linux）/インストールおよびアップデート
導入フロー/インストール（Windows）/インストールおよびアップデート
各種機能説明/フローエディタ/画面操作/設定メニュー/HTTP/HTTPS通信
セキュア利用ガイド/HTTPSの利用
付録/エラーコード一覧/DBクエリサービス

移動平均

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/移動平均
各種機能説明/ロジック（リアルタイム分析/データ操作）/データ変換

イベントデータ

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/システムコンポーネント/イベントデータマネージャー
応用的な利用方法/グローバル変数/設定例/イベントデータの条件判定に使用する閾値を変化させる
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/メール送信の設定
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder/設定項目
i-Trigger（拡張パッケージ）/関連コンポーネント/イベントデータマネージャー
付録/エラーコード一覧/イベントデータマネージャー

イベントトリガ

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/コンポーネント共通の設定項目/入力ポート/イベントの設定
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder
i-Trigger（拡張パッケージ）/関連コンポーネント/イベントトリガ
付録/エラーコード一覧/イベントトリガ

永続化

各種機能説明/エンジン機能の全体像/機能一覧/データベース
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/コンポーネント共通の設定項目/出力ポート
各種機能説明/コンポーネント共通の設定項目/システムポート
各種機能説明/演算式・条件式/演算式と条件式（共通）
応用的な利用方法/グローバル変数/グローバル変数の構文
付録/エラーコード一覧/コアインターナル
付録/カスタムコンポーネントPython-APIリファレンス/InColumn Objects/get_latest_value
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPort … システムポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPort … 出力ポート設定取得
付録/カスタムUI JS-APIリファレンス/型/OutportResponse

エミッタ

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
SpeeDBee Synapseの概要/運用までの流れ
導入フロー/コンポーネントの設定/Fileエミッタの設定
導入フロー/収集サービスの実行/動作確認
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/コンポーネント共通
各種機能説明/フローエディタ/画面操作/設定メニュー/データディレクトリ設定
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/フローエディタ/画面操作/設定メニュー/ポイントについて
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/エミッタ/MQTTエミッタ
各種機能説明/エミッタ/Fileエミッタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
各種機能説明/開発用コンポーネント/ダミーエミッタ
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの概要
セキュア利用ガイド/暗号化通信の利用
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/MQTTエミッタ
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/FTP/FTPS/SFTPエミッタ
付録/カスタムコンポーネントPython-APIリファレンス/FileTypeMetaInfo Objects
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_save_running_info() … 実行時の情報を保存

エラーマネージャー

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/システムコンポーネント/エラーマネージャー
付録/エラーコード一覧/エラーマネージャー

演算コレクタ

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/コレクタ/演算コレクタ
各種機能説明/演算式・条件式/演算式・条件式の概要
付録/エラーコード一覧/コンポーネント共通
付録/エラーコード一覧/演算コレクタ

演算式

各種機能説明/コレクタ/演算コレクタ
各種機能説明/演算式・条件式/演算式・条件式の概要
各種機能説明/演算式・条件式/演算式と条件式（共通）
各種機能説明/演算式・条件式/演算式
応用的な利用方法/グローバル変数/グローバル変数の概要
付録/エラーコード一覧/演算コレクタ

15.7.15 か

カスタムコンポーネント

はじめに/対象読者
SpeeDBee Synapseの概要/運用までの流れ
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/カスタム(Python)
各種機能説明/フローエディタ/画面操作/設定メニュー/カスタム(C)
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/ロジック（リアルタイム分析/データ操作）/FFT
各種機能説明/ステータスタイムライン/ステータス一覧
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
各種機能説明/設定情報の管理(プロジェクト)/インポート
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/カスタムコンポーネントとは
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントベースとコンポーネントインスタンス
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントインスタンスのライフサイクル
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントの入力ポートと出力ポート
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/前提知識
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/サンプルカスタムコンポーネント
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント用API
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント用API
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用JSONファイルの作成/作成方法
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/作成方法
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/HTML形式の詳細
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/パラメータダイアログが表示されたときに実行される関数
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/パラメータダイアログの保存時に実行される関数

応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/PythonカスタムコンポーネントへのカスタムUI(JSON)の適用 {:python}
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/C言語カスタムコンポーネントへのカスタムUI(JSON)の適用 {:C}
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの概要
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネント管理ディレクトリの初期化
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントの追加
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントのテスト実行
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/パッケージング
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除
後方互換に関して/後方互換の対象
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス設定の出力
実行ライセンス/実行ライセンス稼働環境に関する注意事項
付録/カスタムコンポーネントPython-APIリファレンス/HiveComponentBase Objects/create_lock
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_lock() … コンポーネント間排他制御
付録/カスタムUI JS-APIリファレンス/コールバック関数/onLoad … カスタムUI読み込みフック
付録/カスタムUI JS-APIリファレンス/コールバック関数/onSave … パラメーター保存フック
付録/カスタムUI JS-APIリファレンス/型/ComponentMethodResponse

基本統計

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/基本統計
付録/カスタムUI JS-APIリファレンス/型/InportResponse

グラフ

SpeeDBee Synapseの概要/運用までの流れ
各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/データモニタ/分布
応用的な利用方法/Grafanaとの連携/Grafana連携の概要
応用的な利用方法/Grafanaとの連携/グラフの作成
応用的な利用方法/Grafanaとの連携/グラフの閲覧
応用的な利用方法/Grafanaとの連携/グラフのデータ件数上限の変更
i-Trigger（拡張パッケージ）/動画再生用ビューワ/イベント録画再生/再生

グローバル変数

各種機能説明/コレクタ/EtherNet/IPコレクタ
各種機能説明/アクション/EtherNet/IP書き込み
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SET
各種機能説明/演算式・条件式/演算式と条件式（共通）
各種機能説明/演算式・条件式/演算式
各種機能説明/演算式・条件式/条件式
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネント情報定義
応用的な利用方法/グローバル変数/グローバル変数の概要
応用的な利用方法/グローバル変数/グローバル変数の構文
応用的な利用方法/グローバル変数/設定例

コア

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/エンジン機能の全体像/機能一覧/データベース
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/エンジン機能の全体像/動作仕様/正常停止時
各種機能説明/エンジン機能の全体像/動作仕様/異常停止時(電断など)
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/コレクタ/リソースコレクタ
各種機能説明/システムコンポーネント/コアインターナル
各種機能説明/各種ログ/ログの種別
各種機能説明/各種ログ/ダウンロード
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント/実行
応用的な利用方法/Grafanaとの連携/グラフの作成/DBクエリサービスの起動
応用的な利用方法/グローバル変数/グローバル変数の構文
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法
付録/エラーコード一覧/コアインターナル
付録/エラーコード一覧/エラーマネージャー

コアインターナル

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/エンジン機能の全体像/動作仕様/異常停止時(電断など)
各種機能説明/システムコンポーネント/コアインターナル
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法
付録/エラーコード一覧/コアインターナル

コレクタ

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
SpeeDBee Synapseの概要/運用までの流れ
導入フロー/コンポーネントの設定/リソースコレクタの設定
導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/収集サービスの実行/動作確認
導入フロー/収集サービスの実行/設定チューニング
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/データディレクトリ設定
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの概要
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの設定・切替
各種機能説明/コレクタ/リソースコレクタ
各種機能説明/コレクタ/PLCコレクタ
各種機能説明/コレクタ/Modbus TCPコレクタ
各種機能説明/コレクタ/EtherNet/IPコレクタ
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/コレクタ/演算コレクタ
各種機能説明/シリアライザ/JSONシリアライザ
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/開発用コンポーネント/ダミーコレクタ
各種機能説明/コンポーネント共通の設定項目/出力ポート/数値変換の設定
各種機能説明/コンポーネント共通の設定項目/出力ポート/登録条件の設定
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/INSERT
各種機能説明/演算式・条件式/演算式・条件式の概要
各種機能説明/演算式・条件式/演算式と条件式（共通）
各種機能説明/演算式・条件式/演算式
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの概要
トラブルシュート/よくあるトラブルとその対処方法/PLCコレクタに接続できない場合の対処方法
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
セキュア利用ガイド/暗号化通信の利用
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル
i-Trigger（拡張パッケージ）/インストール/アンインストール
i-Trigger（拡張パッケージ）/動画再生用ビューワ/常時録画再生
i-Trigger（拡張パッケージ）/動画再生用ビューワ/イベント録画再生/再生
i-Trigger（拡張パッケージ）/動画再生用ビューワ/ライブ再生
i-Trigger（拡張パッケージ）/関連コンポーネント/VIDEOコレクタ
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder
付録/エラーコード一覧/コンポーネント共通
付録/エラーコード一覧/リソースコレクタ
付録/エラーコード一覧/PLCコレクタ

付録/エラーコード一覧/Modbus TCPコレクタ
付録/エラーコード一覧/EtherNet/IPコレクタ
付録/エラーコード一覧/演算コレクタ
付録/エラーコード一覧/MQTTコレクタ
付録/エラーコード一覧/VIDEOコレクタ（i-Trigger 拡張パッケージ）

コンポーネント

はじめに/対象読者
SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
SpeeDBee Synapseの概要/運用までの流れ
導入フロー/初期設定/ユーザーの追加
導入フロー/コンポーネントの設定/リソースコレクタの設定
導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
導入フロー/収集サービスの実行/収集サービスの開始/停止
導入フロー/収集サービスの実行/設定チューニング
導入フロー/収集サービスの実行/収集サービスの終了
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/フローエディタ/画面操作/設定メニュー/全般的な設定
各種機能説明/フローエディタ/画面操作/設定メニュー/コンポーネント共通
各種機能説明/フローエディタ/画面操作/設定メニュー/データディレクトリ設定
各種機能説明/フローエディタ/画面操作/設定メニュー/カスタム(Python)
各種機能説明/フローエディタ/画面操作/設定メニュー/カスタム(C)
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/フローエディタ/画面操作/設定メニュー/ポイントについて
各種機能説明/フローエディタ/画面操作/ユーザー管理/追加
各種機能説明/フローエディタ/画面操作/ユーザー管理/編集
各種機能説明/フローエディタ/画面操作/パネル/コンポーネント検索
各種機能説明/フローエディタ/画面操作/パネル/表示倍率の調整
各種機能説明/フローエディタ/画面操作/パネル/全体パネル
各種機能説明/フローエディタ/画面操作/パネル/パネルステータス
各種機能説明/フローエディタ/画面操作/コンポーネント
各種機能説明/フローエディタ/画面操作/フローリンク/追加
各種機能説明/フローエディタ/画面操作/データモニタ
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの概要
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの設定・切替
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの動作
各種機能説明/コレクタ/リソースコレクタ
各種機能説明/コレクタ/PLCコレクタ
各種機能説明/コレクタ/Modbus TCPコレクタ
各種機能説明/コレクタ/EtherNet/IPコレクタ
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/コレクタ/演算コレクタ
各種機能説明/シリアライザ/JSONシリアライザ
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/エミッタ/MQTTエミッタ

各種機能説明/エミッタ/Fileエミッタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/基本統計
各種機能説明/ロジック（リアルタイム分析/データ操作）/FFT
各種機能説明/ロジック（リアルタイム分析/データ操作）/移動平均
各種機能説明/ロジック（リアルタイム分析/データ操作）/データ変換
各種機能説明/アクション/シェルコマンド
各種機能説明/アクション/メール送信
各種機能説明/アクション/PLC書き込み
各種機能説明/アクション/Modbus書き込み
各種機能説明/アクション/EtherNet/IP書き込み
各種機能説明/開発用コンポーネント/ダミーコレクタ
各種機能説明/開発用コンポーネント/ダミーエミッタ
各種機能説明/システムコンポーネント/DBクエリサービス
各種機能説明/システムコンポーネント/コアインターナル
各種機能説明/システムコンポーネント/エラーマネージャー
各種機能説明/システムコンポーネント/イベントデータマネージャー
各種機能説明/コンポーネント共通の設定項目/出力ポート
各種機能説明/コンポーネント共通の設定項目/入力ポート
各種機能説明/コンポーネント共通の設定項目/システムポート
各種機能説明/ステータスタイムライン/ステータス一覧
各種機能説明/ステータスタイムライン/ステータスタイムライン
各種機能説明/各種ログ/ログの種別
各種機能説明/各種ログ/コンポーネント個別ログ
各種機能説明/各種ログ/ダウンロード
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/INSERT
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
各種機能説明/設定情報の管理(プロジェクト)/インポート
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/カスタムコンポーネントとは
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントベースとコンポーネントインスタンス
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントインスタンスのライフサイクル
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントの入力ポートと出力ポート
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/前提知識
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/サンプルカスタムコンポーネント
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント用API

応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント用API
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/カスタムUIの作成方法
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用JSONファイルの作成/作成方法
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/作成方法
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/HTML形式の詳細
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/パラメータダイアログが表示されたときに実行される関数
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/パラメータダイアログの保存時に実行される関数
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/編集制限の制御
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/PythonカスタムコンポーネントへのカスタムUI(JSON)の適用 {:python}
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/C言語カスタムコンポーネントへのカスタムUI(JSON)の適用 {:C}
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの概要
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネント管理ディレクトリの初期化
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントの追加
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントのテスト実行
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/パッケージング
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/共通事項
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/全コンポーネントの実行状態取得
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/個別コンポーネントの実行状態取得
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行開始
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行停止
応用的な利用方法/Grafanaとの連携/グラフの作成/データの登録
応用的な利用方法/Grafanaとの連携/グラフの閲覧
応用的な利用方法/Grafanaとの連携/Grafanaデータソースの手動設定
応用的な利用方法/グローバル変数/グローバル変数の概要

トラブルシュート/異常発生時共通の手順
トラブルシュート/よくあるトラブルとその対処方法/PLCコレクタに接続できない場合の対処方法
トラブルシュート/異常が発生した場合に検知する方法
セキュア利用ガイド/HTTPSの利用
セキュア利用ガイド/暗号化通信の利用
後方互換に関して/後方互換の対象
実行ライセンス/ライセンスの種類
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンスの運用方法/運用までの流れ
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス設定の出力
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス稼働環境のセットアップ
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/動作確認
実行ライセンス/実行ライセンス稼働環境に関する注意事項
i-Trigger（拡張パッケージ）/動画再生用ビューワ/イベント録画再生/再生
i-Trigger（拡張パッケージ）/関連コンポーネント/VIDEOコレクタ
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder
i-Trigger（拡張パッケージ）/関連コンポーネント/イベントトリガ
i-Trigger（拡張パッケージ）/関連コンポーネント/イベントデータマネージャー
付録/エラーコード一覧/コンポーネント共通
付録/エラーコード一覧/PLCコレクタ
付録/エラーコード一覧/Modbus TCPコレクタ
付録/エラーコード一覧/EtherNet/IPコレクタ
付録/エラーコード一覧/演算コレクタ
付録/エラーコード一覧/MQTTコレクタ
付録/エラーコード一覧/MQTTエミッタ
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/FTP/FTPS/SFTPエミッタ
付録/エラーコード一覧/JSONシリアライザ
付録/エラーコード一覧/CSVシリアライザ
付録/エラーコード一覧/シェルコマンド
付録/エラーコード一覧/PLC書き込み
付録/エラーコード一覧/Modbus書き込み
付録/エラーコード一覧/EtherNet/IP書き込み
付録/エラーコード一覧/メール送信
付録/エラーコード一覧/コアインターナル
付録/エラーコード一覧/DBクエリサービス
付録/エラーコード一覧/エラーマネージャー
付録/エラーコード一覧/Event Recorder（i-Trigger 拡張パッケージ）
付録/カスタムコンポーネントPython-APIリファレンス/HiveComponentBase Objects
付録/カスタムコンポーネントPython-APIリファレンス/DataType Objects
付録/カスタムコンポーネントPython-APIリファレンス/OutColumn Objects/insert_file_ref
付録/カスタムコンポーネントPython-APIリファレンス/InColumn Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveOutPort Objects
付録/カスタムコンポーネントPython-APIリファレンス/ErrorInfoBase Objects

付録/カスタムコンポーネントPython-APIリファレンス/HiveCompError Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveCompStatus Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveInPortEventSetting Objects
付録/カスタムコンポーネントPython-APIリファレンス/SynapseLock Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveLockCanceled Objects
付録/カスタムコンポーネントC-APIリファレンス/出力ポート関数/hive_outport_create_column() … 新規カラム作成
付録/カスタムコンポーネントC-APIリファレンス/出力ポート関数/hive_outport_register_confirmed_time() … ステータス情報へのカラムデータ確定時刻の登録
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_inport_get_column_list() … 入力カラム情報取得
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_inport_continuous_reader() … カラムデータ継続検索用のReader取得
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_continuous_reader_release() … カラムデータ継続検索用のReaderの解放
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_inport_timerange_reader() … カラムデータ時間範囲検索用のReader取得
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_timerange_reader_release() … カラムデータ時間範囲検索用のReaderの解放
付録/カスタムコンポーネントC-APIリファレンス/コンポーネントステータス関数
付録/カスタムコンポーネントC-APIリファレンス/ログ関連関数/HIVE_LOG_XXXX() … ログ出力マクロ
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_interval_call() … 定期処理関数
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_save_running_info() … 実行時の情報を保存
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_load_running_info() … 保存した実行時の情報を読み込む
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_runnable() … コンポーネント継続可否
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_get_api_error() … 発生したエラーの情報を取得
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_get_common_variable() … コンポーネント共通変数取得
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_lock() … コンポーネント間排他制御
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_unlock() … コンポーネント間排他制御
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_COMPONENT … コンポーネントインスタンス
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/hive_interval_proc_t … 定期処理関数シグネチャ
付録/カスタムUI JS-APIリファレンス/コールバック関数/onLoad … カスタムUI読み込みフック
付録/カスタムUI JS-APIリファレンス/コールバック関数/onSave … パラメーター保存フック
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getComponentbases … コンポーネントベース一覧取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getComponents … コンポーネント一覧取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getComponent … コンポーネント情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPort … 入力ポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPortData … 入力ポート情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPortDataValue … 入力ポートデータ取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPort … システムポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPortData … システムポート情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPortDataValue … システムポートデータ取得

付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPort … 出力ポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPortData … 出力ポート情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPortDataValue … 出力ポートデータ取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/runMethod … コンポーネントインスタンスメソッド実行
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getFlowlinks … フローリンク一覧取得
付録/カスタムUI JS-APIリファレンス/型/ComponentbaseResponse
付録/カスタムUI JS-APIリファレンス/型/PanelResponse
付録/カスタムUI JS-APIリファレンス/型/ComponentResponse
付録/カスタムUI JS-APIリファレンス/型/InportResponse
付録/カスタムUI JS-APIリファレンス/型/OutportResponse
付録/カスタムUI JS-APIリファレンス/型/PortDataResponse
付録/カスタムUI JS-APIリファレンス/型/PortDataValueResponse
付録/カスタムUI JS-APIリファレンス/型/ComponentMethodResponse
付録/カスタムUI JS-APIリファレンス/型/FlowlinkResponse
付録/カスタムUI JS-APIリファレンス/型/FlowlinkFiltersResponse

コンポーネントベース

各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントベースとコンポーネントインスタンス
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/サンプルカスタムコンポーネント/実行
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネントクラス定義
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/`HiveComponent`クラスのライフサイクル
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/コンポーネントパラメータ
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネント情報定義
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントのテスト実行
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/カスタムコンポーネントPython-APIリファレンス/HiveComponentBase Objects/create_lock
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_lock() … コンポーネント間排他制御
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_unlock() … コンポーネント間排他制御
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getComponentbases … コンポーネントベース一覧取得
付録/カスタムUI JS-APIリファレンス/型/ComponentbaseResponse
付録/カスタムUI JS-APIリファレンス/型/ComponentResponse

15.7.16 さ

シェルコマンド

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/アクション/シェルコマンド
付録/エラーコード一覧/シェルコマンド

時系列データベース

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/機能一覧/データベース
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/コンポーネント共通の設定項目/出力ポート/永続化の設定
各種機能説明/コンポーネント共通の設定項目/システムポート/永続化の設定
各種機能説明/DBクエリ（SQL）/DBクエリの概要
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SHOW
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/INSERT
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法
付録/エラーコード一覧/コアインターナル

システムコンポーネント

各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode

システムパネル

応用的な利用方法/Grafanaとの連携/グラフの作成/DBクエリサービスの起動
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/メール送信の設定

システムポート

各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/フローリンク/追加
各種機能説明/コンポーネント共通の設定項目/システムポート
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPort … システムポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPortData … システムポート情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPortDataValue … システムポートデータ取得

シミュレーションモード

各種機能説明/フローエディタ/画面操作/シミュレーションモード

収集サービス

導入フロー/収集サービスの実行/収集サービスの開始/停止
導入フロー/収集サービスの実行/設定チューニング
導入フロー/収集サービスの実行/収集サービスの終了

集約

各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/コレクタ/演算コレクタ
各種機能説明/コンポーネント共通の設定項目/入力ポート
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの概要
付録/カスタムコンポーネントPython-APIリファレンス/AggregationType Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveRecordType Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveRecord Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveRecordData Objects
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_COLUMN_READ_RESULT … カラム情報読み込み結果（ウィンドウデータ）
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_RECORD_DATA … レコードデータ
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPort … 入力ポート設定取得
付録/カスタムUI JS-APIリファレンス/型/InportResponse

出力ポート

導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
導入フロー/収集サービスの実行/動作確認
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/フローリンク/追加
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/基本統計
各種機能説明/ロジック（リアルタイム分析/データ操作）/FFT
各種機能説明/ロジック（リアルタイム分析/データ操作）/移動平均
各種機能説明/ロジック（リアルタイム分析/データ操作）/データ変換
各種機能説明/アクション/PLC書き込み
各種機能説明/アクション/Modbus書き込み
各種機能説明/アクション/EtherNet/IP書き込み
各種機能説明/コンポーネント共通の設定項目/出力ポート
各種機能説明/演算式・条件式/演算式・条件式の概要
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントの入力ポートと出力ポート
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/サンプルカスタムコンポーネント/サンプルの内容
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/サンプルカスタムコンポーネント/実行
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネントクラス定義
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/出力ポートのカラム作成/データ登録(`sin_and_random`)
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント/サンプルの内容
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント/実行
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネント情報定義
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/出力ポートのカラム作成/データ登録（`count_and_random`）
付録/カスタムコンポーネントPython-APIリファレンス/HiveComponentBase Objects
付録/カスタムコンポーネントPython-APIリファレンス/ColumnOption Objects
付録/カスタムコンポーネントPython-APIリファレンス/OutColumn Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveOutPort Objects
付録/カスタムコンポーネントC-APIリファレンス/出力ポート関数
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/hive_primitive_type_t … HIVEカタムプリミティブ型
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_DATA_TYPE … HIVEカラムデータ型
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPort … 出力ポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPortData … 出力ポート情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPortDataValue … 出力ポートデータ取得

付録/カスタムUI JS-APIリファレンス/型/ComponentbaseResponse
付録/カスタムUI JS-APIリファレンス/型/OutportResponse
付録/カスタムUI JS-APIリファレンス/型/PortDataResponse
付録/カスタムUI JS-APIリファレンス/型/PortDataValueResponse

条件式

各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/コンポーネント共通の設定項目/出力ポート/登録条件の設定
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
各種機能説明/演算式・条件式/演算式・条件式の概要
各種機能説明/演算式・条件式/演算式と条件式（共通）
各種機能説明/演算式・条件式/演算式
各種機能説明/演算式・条件式/条件式
応用的な利用方法/グローバル変数/グローバル変数の概要
応用的な利用方法/グローバル変数/グローバル変数の構文
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
付録/エラーコード一覧/コンポーネント共通
付録/エラーコード一覧/イベントトリガ
付録/カスタムUI JS-APIリファレンス/型/OutportResponse

シリアライザ

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
SpeeDBee Synapseの概要/運用までの流れ
導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/フローエディタ/画面操作/設定メニュー/コンポーネント共通
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/シリアライザ/JSONシリアライザ
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/エミッタ/MQTTエミッタ
各種機能説明/エミッタ/Fileエミッタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
各種機能説明/ロジック（イベント）/イベントトリガ
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの概要
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/JSONシリアライザ
付録/エラーコード一覧/CSVシリアライザ
付録/エラーコード一覧/コアインターナル

数値変換

各種機能説明/コンポーネント共通の設定項目/出力ポート
各種機能説明/演算式・条件式/演算式・条件式の概要
応用的な利用方法/グローバル変数/グローバル変数の概要
応用的な利用方法/グローバル変数/グローバル変数の構文
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPort … システムポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPort … 出力ポート設定取得
付録/カスタムUI JS-APIリファレンス/型/OutportResponse

ステータスタイムライン

各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/エンジン機能の全体像/動作仕様/異常停止時(電断など)
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/ステータスタイムライン/ステータスタイムライン
トラブルシュート/異常発生時共通の手順
トラブルシュート/よくあるトラブルとその対処方法/PLCコレクタに接続できない場合の対処方法
実行ライセンス/実行ライセンス利用環境の動作仕様


設定メニュー

導入フロー/初期設定/管理者パスワードの変更
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー
各種機能説明/設定情報の管理(プロジェクト)/新規作成
各種機能説明/設定情報の管理(プロジェクト)/複製
各種機能説明/設定情報の管理(プロジェクト)/インポート
各種機能説明/設定情報の管理(プロジェクト)/エクスポート
各種機能説明/設定情報の管理(プロジェクト)/切替
各種機能説明/設定情報の管理(プロジェクト)/情報更新
各種機能説明/設定情報の管理(プロジェクト)/削除
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/登録
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/削除
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/表示
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/ダウンロード
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/保護の付与
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/保護の解除
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/外部ライブラリの追加
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/外部ライブラリの削除
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する/登録
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する/削除
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する/ダウンロード
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する/保護の付与
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する/保護の解除
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/PythonカスタムコンポーネントへのカスタムUI(JSON)の適用 {:python}/登録
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/PythonカスタムコンポーネントへのカスタムUI(JSON)の適用 {:python}/削除
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/C言語カスタムコンポーネントへのカスタムUI(JSON)の適用 {:C}/登録
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/C言語カスタムコンポーネントへのカスタムUI(JSON)の適用 {:C}/削除
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/登録/更新

応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/ダウンロード
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/削除
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/保護の付与
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/保護の解除
応用的な利用方法/Grafanaとの連携/Grafanaのセットアップ
応用的な利用方法/Grafanaとの連携/グラフの作成/データの登録
トラブルシュート/よくあるトラブルとその対処方法/時刻設定に関する問題と対処方法/SpeeDBee SynapseがWindowsで動作している場合
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス設定の出力
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス稼働環境のセットアップ
付録/エラーコード一覧/DBクエリサービス

全体パネル

各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/フローエディタ/画面操作/パネル/全体パネル

操作ログ

各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/各種ログ/ログの種別
各種機能説明/各種ログ/操作ログ
各種機能説明/各種ログ/ダウンロード
セキュア利用ガイド/操作ログの確認

15.7.17 た

データ収集フロー

各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/フローエディタ/モードについて
各種機能説明/フローエディタ/画面操作/コンポーネント/起動/停止
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/編集制限の制御

データ変換

各種機能説明/ロジック（リアルタイム分析/データ操作）/データ変換

登録条件

各種機能説明/コンポーネント共通の設定項目/出力ポート
各種機能説明/演算式・条件式/演算式・条件式の概要
応用的な利用方法/グローバル変数/グローバル変数の概要
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getSystemPort … システムポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getOutPort … 出力ポート設定取得
付録/カスタムUI JS-APIリファレンス/型/OutportResponse

15.7.18 な

入力ポート

導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/コンポーネント/追加
各種機能説明/フローエディタ/画面操作/フローリンク/追加
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/コレクタ/演算コレクタ
各種機能説明/シリアライザ/JSONシリアライザ
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/エミッタ/MQTTエミッタ
各種機能説明/エミッタ/Fileエミッタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/基本統計
各種機能説明/ロジック（リアルタイム分析/データ操作）/FFT
各種機能説明/ロジック（リアルタイム分析/データ操作）/移動平均
各種機能説明/ロジック（リアルタイム分析/データ操作）/データ変換
各種機能説明/アクション/シェルコマンド
各種機能説明/アクション/メール送信
各種機能説明/アクション/PLC書き込み
各種機能説明/アクション/Modbus書き込み
各種機能説明/アクション/EtherNet/IP書き込み
各種機能説明/開発用コンポーネント/ダミーエミッタ
各種機能説明/コンポーネント共通の設定項目/入力ポート
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/コンポーネントの入力ポートと出力ポート
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/サンプルカスタムコンポーネント/サンプルの内容
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネントクラス定義
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/入力ポートからのデータ読み込み（`read_and_log`）
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/サンプルカスタムコンポーネント/サンプルの内容
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/カスタムコンポーネント情報定義
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/入力ポートからのデータ読み込み（`read_and_log`）
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定

トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder
付録/エラーコード一覧/コンポーネント共通
付録/エラーコード一覧/MQTTエミッタ
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/PLC書き込み
付録/エラーコード一覧/Modbus書き込み
付録/カスタムコンポーネントPython-APIリファレンス/HiveComponentBase Objects
付録/カスタムコンポーネントPython-APIリファレンス/AggregationType Objects
付録/カスタムコンポーネントPython-APIリファレンス/AggregationTypeSet Objects
付録/カスタムコンポーネントPython-APIリファレンス/InColumn Objects
付録/カスタムコンポーネントPython-APIリファレンス/InPortReader Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveInPort Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveWindowData Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveRecord Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveInPortEventSetting Objects
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_inport_get_column_list() … 入力カラム情報取得
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_inport_continuous_reader() … カラムデータ継続検索用のReader取得
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_continuous_reader_read() … カラムデータ継続検索
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_inport_timerange_reader() … カラムデータ時間範囲検索用のReader取得
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_timerange_reader_read() … カラムデータ時間範検索
付録/カスタムコンポーネントC-APIリファレンス/入力カラム関数/hive_incolumn_get_latest_value() … カラムデータ最新値の取得
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/hive_primitive_type_t … HIVEカタムプリミティブ型
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_DATA_TYPE … HIVEカラムデータ型
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_COLUMN_READ_RESULT … カラム情報読み込み結果（ウィンドウデータ）
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_RECORD … カラムデータレコード
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPort … 入力ポート設定取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPortData … 入力ポート情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getInPortDataValue … 入力ポートデータ取得
付録/カスタムUI JS-APIリファレンス/型/ComponentbaseResponse
付録/カスタムUI JS-APIリファレンス/型/InportResponse
付録/カスタムUI JS-APIリファレンス/型/PortDataResponse
付録/カスタムUI JS-APIリファレンス/型/PortDataValueResponse

ネットワーク

導入フロー/インストール（Linux）/インストールおよびアップデート
導入フロー/インストール（Windows）/インストールおよびアップデート
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/コレクタ/リソースコレクタ
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/システムコンポーネント/DBクエリサービス
トラブルシュート/よくあるトラブルとその対処方法/PLCコレクタに接続できない場合の対処方法/IPアドレスの確認
トラブルシュート/よくあるトラブルとその対処方法/PLCコレクタに接続できない場合の対処方法/ポートの確認
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル
i-Trigger（拡張パッケージ）/関連コンポーネント/VIDEOコレクタ
付録/エラーコード一覧/コンポーネント共通
付録/エラーコード一覧/リソースコレクタ
付録/エラーコード一覧/MQTTエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/エラーコード一覧/FTP/FTPS/SFTPエミッタ
付録/エラーコード一覧/VIDEOコレクタ（i-Trigger 拡張パッケージ）

15.7.19 は

パネル

導入フロー/コンポーネントの設定/リソースコレクタの設定
導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
導入フロー/収集サービスの実行/収集サービスの開始/停止
導入フロー/収集サービスの実行/動作確認
導入フロー/収集サービスの実行/設定チューニング
各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/サーバー設定
各種機能説明/フローエディタ/画面操作/ユーザー管理/追加
各種機能説明/フローエディタ/画面操作/ユーザー管理/編集
各種機能説明/フローエディタ/画面操作/パネル
各種機能説明/フローエディタ/画面操作/コンポーネント/追加
各種機能説明/フローエディタ/画面操作/コンポーネント/コピー&ペースト
各種機能説明/フローエディタ/画面操作/コンポーネント/複数選択
各種機能説明/フローエディタ/画面操作/コンポーネント/起動/停止
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/共通事項
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/全コンポーネントの実行状態取得
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/個別コンポーネントの実行状態取得
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行開始
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行停止
応用的な利用方法/Grafanaとの連携/グラフの作成/DBクエリサービスの起動
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/メール送信の設定
後方互換に関して/後方互換の対象
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/実行ライセンス設定の出力
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getPanels … パネル一覧取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getPanel … パネル情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getComponents … コンポーネント一覧取得
付録/カスタムUI JS-APIリファレンス/型/PanelResponse

ビューワ

各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/システムコンポーネント/イベントデータマネージャー
i-Trigger（拡張パッケージ）/動画再生用ビューワ/ビューワへのアクセス
i-Trigger（拡張パッケージ）/関連コンポーネント/VIDEOコレクタ
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder

フィルタ

各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面操作/フローリンク/フィルタ
各種機能説明/コンポーネント共通の設定項目/フィルタ
各種機能説明/コンポーネント共通の設定項目/フローリンクアイコン
実行ライセンス/実行ライセンス利用環境の動作仕様
付録/エラーコード一覧/MQTTエミッタ
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getFlowlinkFilter … フローリンクフィルター情報取得
付録/カスタムUI JS-APIリファレンス/型/InportResponse
付録/カスタムUI JS-APIリファレンス/型/FlowlinkFiltersResponse

フローエディタ

各種機能説明/フローエディタ/モードについて
各種機能説明/フローエディタ/画面項目
各種機能説明/フローエディタ/画面操作
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネントのテスト実行
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築

フローリンク

導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/エンジン機能の全体像/動作仕様/データ収集に関連する動作仕様について
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/フローリンク
各種機能説明/フローエディタ/画面操作/シミュレーションモード/シミュレーションモードの動作
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/コンポーネント共通の設定項目/出力ポート
各種機能説明/コンポーネント共通の設定項目/フィルタ
各種機能説明/コンポーネント共通の設定項目/フローリンクアイコン
各種機能説明/コンポーネント共通の設定項目/入力ポート
各種機能説明/コンポーネント共通の設定項目/システムポート
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/MQTTエミッタ
付録/エラーコード一覧/Fileエミッタ
付録/エラーコード一覧/Cloudエミッタ(AWS用)
付録/エラーコード一覧/Cloudエミッタ(Azure用)
付録/カスタムコンポーネントPython-APIリファレンス/HiveOutPort Objects/register_confirmed_time
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getFlowlinks … フローリンク一覧取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getFlowlink … フローリンク情報取得
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数/getFlowlinkFilter … フローリンクフィルター情報取得
付録/カスタムUI JS-APIリファレンス/型/FlowlinkResponse
付録/カスタムUI JS-APIリファレンス/型/FlowlinkFiltersResponse

プロフィール

導入フロー/初期設定/ユーザーの追加
各種機能説明/フローエディタ/画面操作/ユーザー管理/プロフィール

ヘッダ

導入フロー/初期設定/管理者パスワードの変更
導入フロー/初期設定/プロジェクト名の設定
導入フロー/コンポーネントの設定/リソースコレクタの設定
導入フロー/収集サービスの実行/収集サービスの開始/停止/SpeeDBee Synapse上の全コンポーネントの起動/停止
導入フロー/収集サービスの実行/動作確認
導入フロー/収集サービスの実行/設定チューニング
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/コンポーネント/起動/停止
各種機能説明/フローエディタ/画面操作/シミュレーションモード/運転ファイルのフォーマット
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/各種ログ/操作ログ/閲覧
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/インクルードヘッダ
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/登録/更新
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPIの実行方法
応用的な利用方法/Grafanaとの連携/Grafanaのセットアップ
応用的な利用方法/Grafanaとの連携/グラフの閲覧
応用的な利用方法/Grafanaとの連携/Grafanaデータソースの手動設定
i-Trigger（拡張パッケージ）/動画再生用ビューワ/ビューワへのアクセス
i-Trigger（拡張パッケージ）/動画再生用ビューワ/イベント録画再生/削除

ポイント

各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/ソフトウェア情報
各種機能説明/フローエディタ/画面操作/設定メニュー/ポイントについて
各種機能説明/コレクタ/リソースコレクタ
各種機能説明/コレクタ/PLCコレクタ
各種機能説明/コレクタ/Modbus TCPコレクタ
各種機能説明/コレクタ/EtherNet/IPコレクタ
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/コレクタ/演算コレクタ
各種機能説明/シリアライザ/JSONシリアライザ
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/エミッタ/MQTTエミッタ
各種機能説明/エミッタ/Fileエミッタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/ロジック（リアルタイム分析/データ操作）/基本統計
各種機能説明/ロジック（リアルタイム分析/データ操作）/FFT
各種機能説明/ロジック（リアルタイム分析/データ操作）/移動平均
各種機能説明/ロジック（リアルタイム分析/データ操作）/データ変換
各種機能説明/アクション/シェルコマンド
各種機能説明/アクション/メール送信
各種機能説明/アクション/PLC書き込み
各種機能説明/アクション/Modbus書き込み
各種機能説明/アクション/EtherNet/IP書き込み
各種機能説明/開発用コンポーネント/ダミーコレクタ/基本情報
各種機能説明/開発用コンポーネント/ダミーエミッタ/基本情報
各種機能説明/システムコンポーネント/DBクエリサービス
各種機能説明/システムコンポーネント/コアインターナル
各種機能説明/システムコンポーネント/エラーマネージャー
各種機能説明/システムコンポーネント/イベントデータマネージャー
応用的な利用方法/制御プログラムとの連携（WebAPI）/WebAPI仕様/コンポーネント一括実行開始
i-Trigger（拡張パッケージ）/関連コンポーネント/VIDEOコレクタ/基本情報
i-Trigger（拡張パッケージ）/関連コンポーネント/Event Recorder/基本情報
付録/エラーコード一覧/リソースコレクタ
付録/カスタムUI JS-APIリファレンス/型/ComponentbaseResponse

15.7.20 ま

メール送信

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/アクション/メール送信
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/特定のコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/イベントデータの設定
トラブルシュート/異常が発生した場合に検知する方法/Synapse内のいずれかのコンポーネントで異常が発生した場合に検知する方法/メール送信の設定
トラブルシュート/異常が発生した場合に検知する方法/電断などにより時系列データベースのデータ欠損を検知する方法/メール送信の設定
セキュア利用ガイド/暗号化通信の利用
付録/エラーコード一覧/メール送信

モード

導入フロー/初期設定/管理者パスワードの変更
導入フロー/コンポーネントの設定/リソースコレクタの設定
導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/コンポーネントの設定/Fileエミッタの設定
導入フロー/収集サービスの実行/動作確認
導入フロー/収集サービスの実行/設定チューニング
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/エンジン機能の全体像/動作仕様/ネットワーク異常、復旧時の動作仕様について
各種機能説明/フローエディタ/モードについて
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/シミュレーションモード
各種機能説明/エミッタ/MQTTエミッタ
各種機能説明/エミッタ/Cloudエミッタ(AWS用)
各種機能説明/エミッタ/Cloudエミッタ(Azure用)
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SynapseへのSCCPKGファイルの登録、および、削除/登録/更新
応用的な利用方法/Grafanaとの連携/Grafanaのセットアップ
応用的な利用方法/Grafanaとの連携/グラフの作成/データの登録
実行ライセンス/実行ライセンス稼働環境に関する注意事項
付録/カスタムUI JS-APIリファレンス/コールバック関数/onLoad … カスタムUI読み込みフック

15.7.21 や


ユーザー

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
SpeeDBee Synapseの概要/運用までの流れ
導入フロー/インストール（Linux）/インストールおよびアップデート
導入フロー/初期設定/管理者パスワードの変更
導入フロー/初期設定/ユーザーの追加
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/エンジン機能の全体像/機能一覧/WEBUI
各種機能説明/エンジン機能の全体像/機能一覧/データベース
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/フローエディタ/モードについて
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/HTTP/HTTPS通信
各種機能説明/フローエディタ/画面操作/設定メニュー/プロキシ設定
各種機能説明/フローエディタ/画面操作/ユーザー管理
各種機能説明/フローエディタ/画面操作/パネル
各種機能説明/フローエディタ/画面操作/コンポーネント/起動/停止
各種機能説明/コレクタ/MQTTコレクタ
各種機能説明/エミッタ/MQTTエミッタ
各種機能説明/エミッタ/FTP/FTPS/SFTPエミッタ
各種機能説明/アクション/メール送信
各種機能説明/システムコンポーネント/コアインターナル
各種機能説明/各種ログ/ログの種別
各種機能説明/各種ログ/操作ログ
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネントの概要/カスタムコンポーネントとは
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネントを使用する/設定画面
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネントを使用する/設定画面
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/カスタムUIの作成方法
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムUI作成/SynapseカスタムUI用HTML作成、適用方法/編集制限の制御
応用的な利用方法/Grafanaとの連携/Grafana連携の概要
応用的な利用方法/Grafanaとの連携/Grafanaのインストール
応用的な利用方法/Grafanaとの連携/Grafanaのセットアップ
セキュア利用ガイド/ユーザーの管理
後方互換に関して/後方互換の対象
後方互換に関して/後方互換
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンス稼働環境に関する注意事項
付録/エラーコード一覧/MQTTコレクタ
付録/エラーコード一覧/MQTTエミッタ
付録/エラーコード一覧/FTP/FTPS/SFTPエミッタ
付録/エラーコード一覧/メール送信

付録/カスタムコンポーネントPython-APIリファレンス/HiveComponentBase Objects/interval_iteration
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_interval_call() … 定期処理関数
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_component_load_running_info() … 保存した実行時の情報を読み込む
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_get_common_variable() … コンポーネント共通変数取得
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/HIVE_COMPONENT … コンポーネントインスタンス
付録/カスタムコンポーネントC-APIリファレンス/型・構造体・関数/hive_interval_proc_t … 定期処理関数シグネチャ
付録/カスタムUI JS-APIリファレンス/コールバック関数/onLoad … カスタムUI読み込みフック
付録/カスタムUI JS-APIリファレンス/コールバック関数/onSave … パラメーター保存フック
付録/カスタムUI JS-APIリファレンス/SpeeDBee Synapse設定取得関数
付録/カスタムUI JS-APIリファレンス/型/ComponentResponse

15.7.22 ら

ライセンス

はじめに/ご承諾事項
導入フロー/初期設定/管理者パスワードの変更
各種機能説明/エンジン機能の全体像/機能一覧/コア
各種機能説明/エンジン機能の全体像/動作仕様/基本動作
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/設定メニュー/実行ライセンス設定ダウンロード
各種機能説明/フローエディタ/画面操作/設定メニュー/ソフトウェア情報
各種機能説明/フローエディタ/画面操作/設定メニュー/ポイントについて
各種機能説明/設定情報の管理(プロジェクト)/プロジェクトの概要
応用的な利用方法/カスタムコンポーネントの作成・利用/カスタムコンポーネント開発環境(SCCDE)/SCCDEの使い方とパッケージング/カスタムコンポーネント管理ディレクトリの初期化
実行ライセンス/ライセンスの種類
実行ライセンス/実行ライセンス利用環境の動作仕様
実行ライセンス/実行ライセンスの運用方法
実行ライセンス/実行ライセンス稼働環境に関する注意事項

リアルタイム分析

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
SpeeDBee Synapseの概要/運用までの流れ
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/フローエディタ/画面項目/Edit Mode

リソースコレクタ

導入フロー/コンポーネントの設定/リソースコレクタの設定
導入フロー/コンポーネントの設定/CSVシリアライザの設定
導入フロー/収集サービスの実行/動作確認
導入フロー/収集サービスの実行/設定チューニング
各種機能説明/コレクタ/リソースコレクタ
各種機能説明/シリアライザ/JSONシリアライザ
各種機能説明/シリアライザ/CSVシリアライザ
各種機能説明/ロジック（イベント）/イベントデータ
各種機能説明/ロジック（イベント）/イベントトリガ
各種機能説明/コンポーネント共通の設定項目/出力ポート/数値変換の設定
各種機能説明/コンポーネント共通の設定項目/出力ポート/登録条件の設定
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/SELECT
各種機能説明/DBクエリ（SQL）/DBクエリの詳細/構文仕様/INSERT
実行ライセンス/実行ライセンスの運用方法/簡易チュートリアル/開発ライセンスを使ったシステム構築
付録/エラーコード一覧/リソースコレクタ

ログアウト

導入フロー/初期設定/ユーザーの追加
各種機能説明/フローエディタ/画面操作/ユーザー管理/ログアウト

ログイン

導入フロー/インストール（Linux）/インストールおよびアップデート
導入フロー/インストール（Linux）/SpeeDBee Synapseの制御
導入フロー/インストール（Linux）/アンインストール
導入フロー/初期設定/管理者パスワードの変更
導入フロー/初期設定/ユーザーの追加
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/フローエディタ/画面項目/Run Mode
各種機能説明/フローエディタ/画面操作/ユーザー管理/追加
各種機能説明/フローエディタ/画面操作/ユーザー管理/編集
各種機能説明/フローエディタ/画面操作/ユーザー管理/プロフィール
各種機能説明/フローエディタ/画面操作/ユーザー管理/ログイン
応用的な利用方法/Grafanaとの連携/Grafanaのセットアップ
i-Trigger（拡張パッケージ）/インストール/インストールおよびアップデート
i-Trigger（拡張パッケージ）/インストール/アンインストール

ログ出力

各種機能説明/フローエディタ/画面操作/設定メニュー/ログ設定
各種機能説明/各種ログ/コンポーネント個別ログ
応用的な利用方法/カスタムコンポーネントの作成・利用/Pythonカスタムコンポーネント作成/カスタムコンポーネント実装詳細/ログ出力
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/ライフサイクルコールバック定義
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/入力ポートからのデータ読み込み（`read_and_log`）
応用的な利用方法/カスタムコンポーネントの作成・利用/C言語カスタムコンポーネント作成/カスタムコンポーネント実装詳細/ログ出力
付録/カスタムコンポーネントPython-APIリファレンス/HiveLog Objects
付録/カスタムコンポーネントC-APIリファレンス/ログ関連関数/HIVE_LOG_XXXX() … ログ出力マクロ
付録/カスタムコンポーネントC-APIリファレンス/ログ関連関数/hive_current_loglevel() … ログレベル取得

ログレベル

各種機能説明/フローエディタ/画面操作/設定メニュー/ログ設定
各種機能説明/各種ログ/コンポーネント個別ログ
付録/カスタムコンポーネントPython-APIリファレンス/HiveLogLevel Objects
付録/カスタムコンポーネントPython-APIリファレンス/HiveLog Objects/get_current_level
付録/カスタムコンポーネントC-APIリファレンス/ログ関連関数/HIVE_LOG_XXXX() … ログ出力マクロ
付録/カスタムコンポーネントC-APIリファレンス/ログ関連関数/hive_current_loglevel() … ログレベル取得

ロジック

SpeeDBee Synapseの概要/SpeeDBee Synapseの構成
各種機能説明/エンジン機能の全体像/機能一覧/コンポーネント
各種機能説明/フローエディタ/画面項目/Edit Mode
各種機能説明/ロジック（イベント）/イベントトリガ
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_lock() … コンポーネント間排他制御
付録/カスタムコンポーネントC-APIリファレンス/その他の関数/hive_unlock() … コンポーネント間排他制御

15.8 DBクエリサービスWeb-API仕様

この章では、DBクエリサービスのWeb-API仕様について説明します。

Core Interface (0.1.0)

Download OpenAPI specification: Download

コアDBクエリ専用API

DBクエリ実行

DBクエリを実行しその実行結果を返す。 クエリを実行した上でその内容がエラーとなる場合は、ステータスコード `200` で `result-type:error` のレスポンスを返す。
クエリの実行自体ができない場合は、 `200` 以外のステータスコードを返す。

HEADER PARAMETERS

- `X-hive-api-key` — string
  トークン認証を有効にした場合に指定するトークン文字列。

REQUEST BODY SCHEMA: application/json
required

DBクエリ

- `query` — required — string
  クエリ文字列
- `timeout` — int [1 .. 180]
  Default: `30`
  クエリ処理のタイムアウト(秒)、指定時間をすぎるとタイムアウトとなる
- `timestamp_unit` — string
  Default: `"nanosecond"`
  Enum: `"nanosecond"` `"microsecond"` `"millisecond"` `"second"`
  返却結果のタイムスタンプの単位
- `omit_option` — string
  Default: `0`
  Enum: `0` `1` `2`
  データ無し、NULL値のデータを省略するか指定する(0=NULLとデータ無し省略する,1 or 2=省略しない)

Responses

200 DBクエリ結果

RESPONSE SCHEMA: application/json

- query (required)
  string
  クエリ文字列
- timestamp_unit (required)
  string
  タイムスタンプの単位指定
- success (required)
  boolean
  クエリの実行が成功したことを示す
- cutoff
  boolean
  クエリ結果が切り捨てられたかどうかを示す
- error
  object (DbQueryError)
  クエリの実行エラーの情報を保持したオブジェクト
  - code
    string
    エラー識別子
  - message
    string
    エラーメッセージ
- result
  object (DbQueryResult)
  クエリの実行結果がある場合に設定されるオブジェクト
  - columns
    Array of strings (HiveColumnListResult) [items]
    カラム一覧取得要求時のみ設定されるオブジェクト
  - record_columns
    Array of objects (DbQueryResultColumnMeta)
    カラムのメタ情報一覧

    Array [
    - name (required)
      string
    - data_type (required)
      string (DbQueryDataType)
      Enum: "bool" "int8" "int16" "int32" "int64" "uint8" "uint16" "uint32" "uint64" "float" "double" "timestamp" "string" "binary"

data_array
required

integer (DbQueryDataArray) >= 1

]

records

Array of objects (DbQueryResultRecord)

Array [

timestamp
required

int
レコードのタイムスタンプ

window_id
required

int
参照時のウィンドウを示すID

record_type
required

string (DbQueryResultRecordType)
Enum: "D" "S"

data

Array of objects

Array [

name
required

string

value
required

any

data_type
required

string (DbQueryDataType)
Enum: "bool" "int8" "int16" "int32" "int64" "uint8" "uint16" "uint32" "uint64" "float" "double" "timestamp" "string" "binary"

data_array

integer (DbQueryDataArray) >= 1

]

]

500 本機能は現在無効化されている

RESPONSE SCHEMA: application/json

id
required

string

code required	integer	
codeHex required	string `0x[0-9a-f]{8}`	
message required	string	
parameters	Array of any	
extendedInfo	any	

POST /dbquery

Request samples

Payload

Content type
application/json

Copy

```
{
    "query": "SELECT ¥"comp1:data1¥", ¥"comp2:dataA¥" FROM SPDB;",
    "timeout": 15,
    "timestamp_unit": "nanosecond"
}
```

Response samples

200 500

Content type
application/json

Example
DBクエリ結果

Copy Expand all Collapse all

```
{
    "query": "SELECT ¥"comp1:data1¥", ¥"comp2:dataA¥" FROM SPDB;",
    "timestamp_unit": "nanosecond",
    "success": true,
```

```
  - "result": {
      + "record_columns": [ … ],
      + "records": [ … ]
    }
}
```


DBクエリ実行（ストリーム形式）

DBクエリを実行しその実行結果をストリーム形式で返す。 本APIではSELECTなどのサーバーの状態に変化を与えないクエリのみに対応する。 現時点では非対応。（将来的に追加する可能性あり）

REQUEST BODY SCHEMA: application/json
required

DBクエリ

- `query` (required)
  string
  クエリ文字列
- `timeout`
  int [1 .. 180]
  Default: 30
  クエリ処理のタイムアウト(秒)、指定時間をすぎるとタイムアウトとなる
- `timestamp_unit`
  string
  Default: "nanosecond"
  Enum: "nanosecond" "microsecond" "millisecond" "second"
  返却結果のタイムスタンプの単位
- `omit_option`
  string
  Default: 0
  Enum: 0 1 2
  データ無し、NULL値のデータを省略するか指定する(0=NULLとデータ無し省略する,1 or 2=省略しない)

Responses

200 DBクエリ結果

RESPONSE SCHEMA: application/jsonlines

Array [

- timestamp
  required
  int
  レコードのタイムスタンプ
- window_id
  required
  int
  参照時のウィンドウを示すID
- record_type
  required
  string (DbQueryResultRecordType)
  Enum: `"D"` `"S"`
- data
  Array of objects

Array [

- name
  required
  string
- value
  required
  any
- data_type
  required
  string (DbQueryDataType)
  Enum: `"bool"` `"int8"` `"int16"` `"int32"` `"int64"` `"uint8"` `"uint16"` `"uint32"` `"uint64"` `"float"` `"double"` `"timestamp"` `"string"` `"binary"`
- data_array
  integer (DbQueryDataArray) `>= 1`

]

]

500 本機能は現在無効化されている

RESPONSE SCHEMA: application/json

- id
  required
  string
- code
  required
  integer
- codeHex
  required
  string `0x[0-9a-f]{8}`
- message
  required
  string
- parameters
  Array of any
- extendedInfo
  any

GET /dbquery

Request samples

Payload

Content type
application/json

Copy

```
{
    "query": "SELECT ¥"comp1:data1¥", ¥"comp2:dataA¥" FROM SPDB;",
    "timeout": 15,
    "timestamp_unit": "nanosecond"
}
```

Response samples

200 500

Content type
application/jsonlines

Copy Expand all Collapse all

```
[
 - {
        "timestamp": null,
        "window_id": null,
        "record_type": "D",
     + "data": [ … ]
    }
]
```

オムロン株式会社 インダストリアルオートメーションビジネスカンパニー

製品に関するお問い合わせ先（DX シリーズ専用）

お客様相談室

メールでのお問い合わせ（お問い合わせフォーム）

https://www.fa.omron.co.jp/dx1/contact/

対応時間：9:00～17:00（土・日・12/31～1/3を除く）

マニュアル動画

https://www.fa.omron.co.jp/dx1/video-manual/ja/

その他のお問い合わせ：納期・価格・サンプル・仕様書は貴社のお取引先、または貴社担当オムロン販売員にご相談ください。オムロン制御機器販売店やオムロン販売拠点は、Webページでご案内しています。

オムロン制御機器の最新情報をご覧いただけます。緊急時のご購入にもご利用ください。 **www.fa.omron.co.jp**

●本誌に記載のない条件や環境での使用、および原子力制御・鉄道・航空・車両・燃焼装置・医療機器・娯楽機械・安全機器、その他人命や財産に大きな影響が予測されるなど、特に安全性が要求される用途に使用される際には、当社の意図した特別な商品用途の場合や特別の合意がある場合を除き、当社は当社商品に対して一切保証をいたしません。

●本製品の内、外国為替及び外国貿易法に定める輸出許可、承認対象貨物（又は技術）に該当するものを輸出（又は非居住者に提供）する場合は同法に基づく輸出許可、承認（又は役務取引許可）が必要です。

オムロン商品のご用命は

Man. No. SBSB-540A 2025年9月現在

©OMRON Corporation 2025 All Rights Reserved.
お断りなく仕様などを変更することがありますのでご了承ください