

Enterprise Analyzer チュートリアル

クエリー作成 編

1 目的

複雑で多くのアプリケーションを分析する際、問題と推測される箇所をあらかじめ特定し、改修の際にかかる工数を算出するために数量を把握したり、デグレードを起こさないために影響度を把握したりすることはアプリケーション管理者や開発者にとって大切な業務です。

本チュートリアルでは Enterprise Analyzer を利用して、アプリケーション分析を行うためのクエリーを作成し、その機能と手順の習得を目的としています。

2 前提

- 2.1 本チュートリアルで使用したマシン OS : Windows Server 2022 Standard Edition
- 2.2 使用マシンに Enterprise Analyzer 11.0 がインストールされていること
- 2.3 Enterprise Analyzer 11.0 基本機能の紹介と利用方法チュートリアルが完了していること

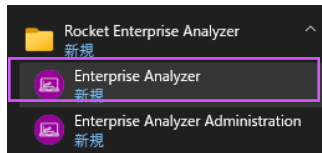
3 チュートリアル手順の概要

- 3.1 Enterprise Analyzer の起動
- 3.2 既存ルールによる分析
- 3.3 オリジナルルールによる分析
- 3.4 ルールのエクスポート
- 3.5 Eclipse 版によるルールのインポート
- 3.6 Visual Studio 版によるルールのインポート
- 3.7 最後に

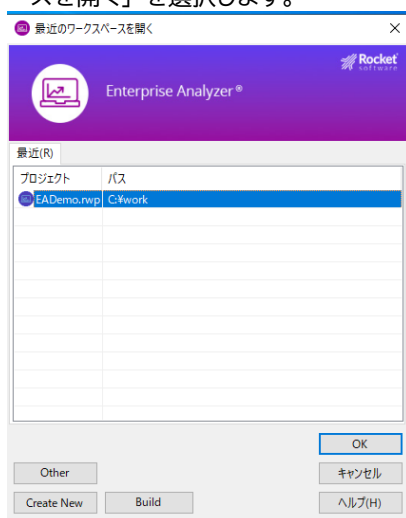
3.1 Enterprise Analyzer の起動

作成済みのワークスペースに存在するプロジェクトを開きます。ワークスペースの作成については「Enterprise Analyzer 利用ガイド」チュートリアルをご参照ください。

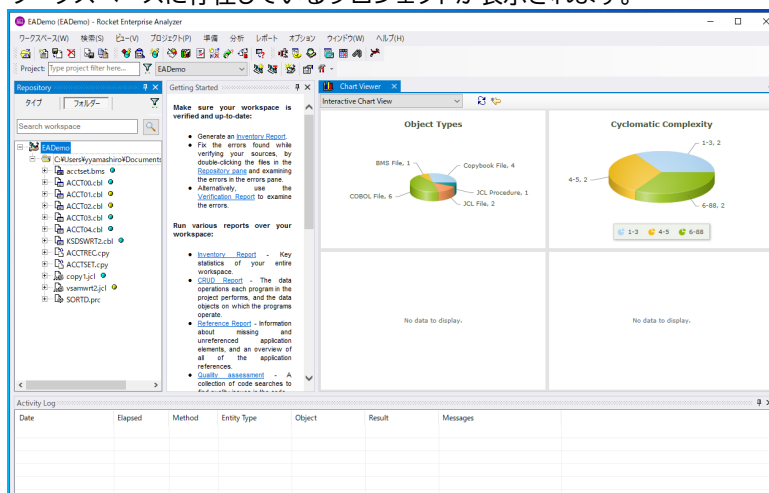
3.1.1 Windows メニューから [Enterprise Analyzer] を起動します。



3.1.2 起動と同時に「ワークスペースを開く」ウィンドウが表示されますので、使用するワークスペースを選択して [OK] ボタンをクリックします。もし、閉じている場合は、メニューから「ワークスペース」メニューの「ワークスペースを開く」を選択します。



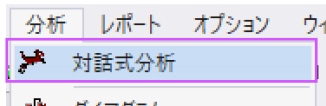
3.1.3 ワークスペースに存在しているプロジェクトが表示されます。



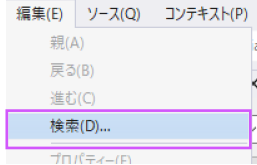
3.2 既存ルールによる分析

ある一定のルールによるプログラムコードの検出を行います。ルールは、製品に備わっている既存ルールもありますが、オリジナルのルールを作成して適用することも可能です。まずは既存ルールで検索を実行してみます。

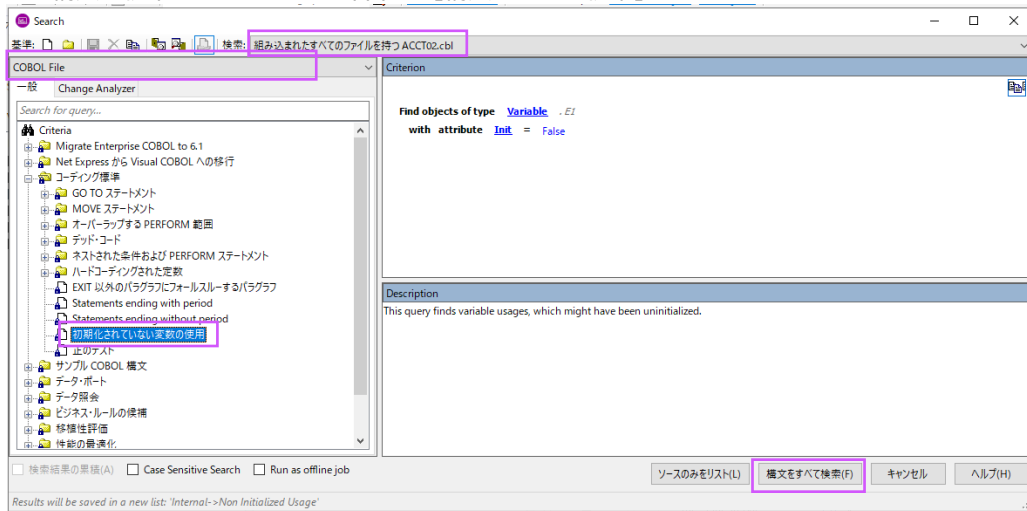
- 3.2.1 [Repository]にて“ACCT00.cbl”をクリックして選択してから[分析]メニュー > [対話式分析] を選択します。



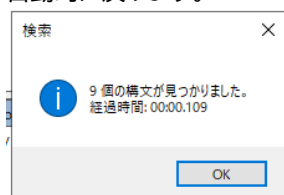
- 3.2.2 [対話式分析] ウィンドウが表示されますので、既存ルールを適用するために[編集]プルダウンメニューの[検索]を選択します。



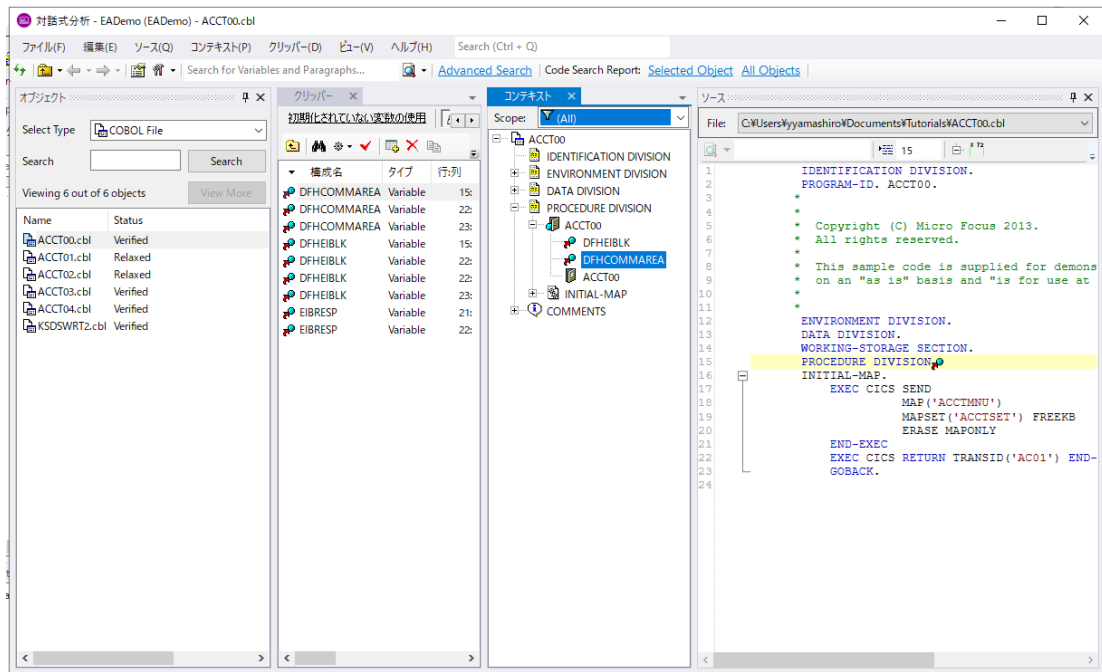
- 3.2.3 [検索] ウィンドウが表示されましたら、一番上のアイコンメニューにある[検索]に[すべてのオブジェクト]を選択、アイコンメニューの下[検索]には[COBOL File]を選択した状態で、左側ツリービューの[コーディング標準]を展開して[初期化されていない変数の使用]を選択すると[基準]ペインに検索構文が表示されます。この構文で検索をかけてみます。右下の[構文をすべて検索]ボタンをクリックします。



- 3.2.4 検索が終了すると下記メッセージが表示されますので[OK]ボタンをクリックすると[対話式分析]ウィンドウへ自動的に戻ります。



- 3.2.5 [クリッパー]ペインに検索結果箇所が表示されます。“ACCT00.cbl”をダブルクリックすると[コンテキスト]ペインには該当の構文が、[ソース]ペインには該当行がポイントされて表示されます。他の結果行をダブルクリックすることで同様に詳細が表示できます。



3.2.6 [クリッパー] ペインの一番上にある [Internal] をクリックすると、過去に検索した結果が表示されます。再度結果を表示したい場合は該当行をダブルクリックしてください。



3.3 オリジナルルールによる分析

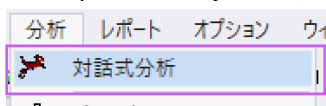
オリジナルルールはスクラッチで作成することも可能ですが、まずはどのような既存ルールがあるのか、どのように記述するのかを含め、作成したい条件と類似している既存ルールをコピーして作成することをお勧めします。

本チュートリアルでは [ON SIZE ERROR 記述のない COMPUTE 命令を含む算術式] を検索することを目的に、類似している既存ルールをコピーして新規オリジナルルールを作成した後、検索を実行します。

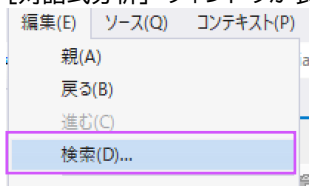


フォルダやルールに左のような鍵マークが付いているものは編集できません。

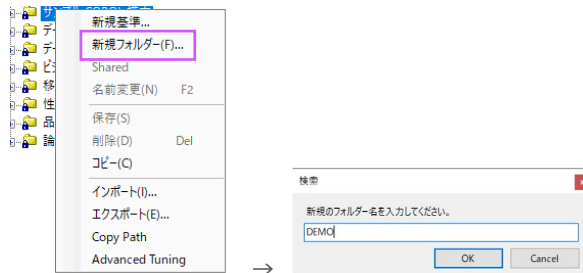
3.3.1 前項で起動した Enterprise Analyzer Administration ウィンドウの[管理]メニュー > [Micro Focus Enterprise Analyzer の起動]を選択します。



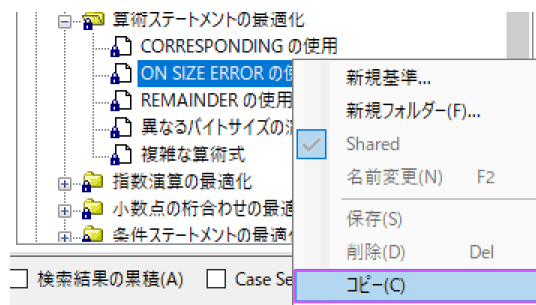
3.3.2 [対話式分析] ウィンドウが表示されますので [編集] プルダウンメニューの [検索...] を選択します。



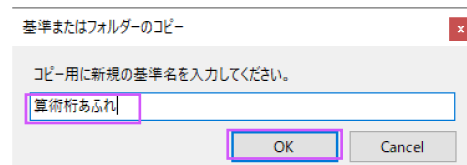
- 3.3.3 編集可能なオリジナルルール用のフォルダを作成します。[検索] ウィンドウが表示されましたら、左側ツリービュー上で右クリックをして [新規フォルダ] を選択して任意の名称を入力します。本チュートリアルでは “DEMO” と入力します。



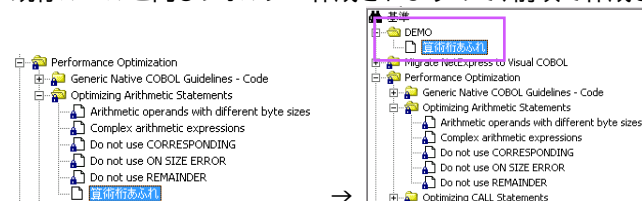
- 3.3.4 作成したい条件に類似している、既存ルールの [性能の最適化] > [算術ステートメントの最適化] > [ON SIZE ERROR の使用] を選択して右クリックから [コピー] を選択します。



- 3.3.5 オリジナルルールの名称を入力します。任意ですが、本チュートリアルでは “算術桁あふれ” と入力して [OK] ボタンをクリックします。



- 3.3.6 既存ルールと同じフォルダへ作成されますので、前項で作成した [DEMO] フォルダへドラッグ & ドロップします。



- 3.3.7 [ON SIZE ERROR 記述のない COMPUTE 命令を含む算術式] の検索を行うため、COMPUTE 命令による桁あふれの可能性のあるコードを任意の例題プログラムへ追加します。本チュートリアルでは ACCT01.CBL へ次のコードを追加します。

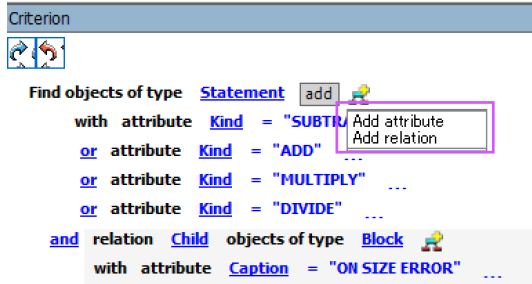
```
01 DEMO-WORK.
```

```
03 DEMO-VAL1 PIC 9(10) VALUE 1111111111.
```

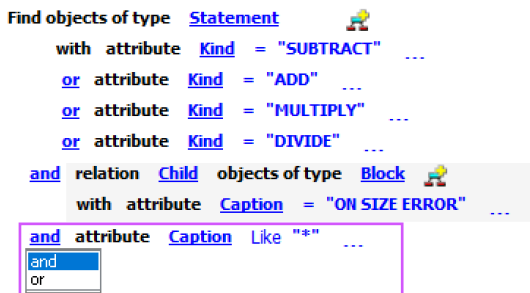
```
03 DEMO-VAL2 PIC 9(10) VALUE 3333333333.
```

```
COMPUTE DEMO-RLT = DEMO-VAL1 + DEMO-VAL2.
```

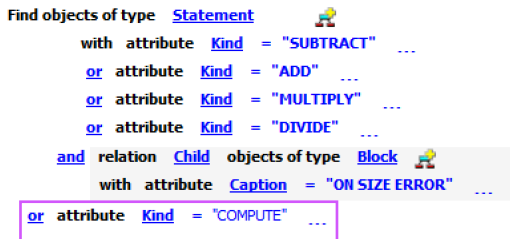
- 3.3.8 作成した「算術桁あふれ」ルールを選択した状態で右側のペインで編集を行います。現在記述している算術式に COMPUTE 命令を追加します。[Statement] 横の [add] から [Add attribute] をクリックします。



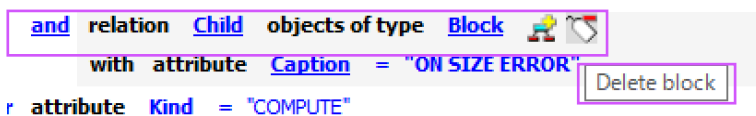
- 3.3.9 最下行に追加されるので and を or 変更し、[Caption] を [Kind] に [like] を [=] に [*] を [COMPUTE] に設定して [OK] ボタンをクリックします。



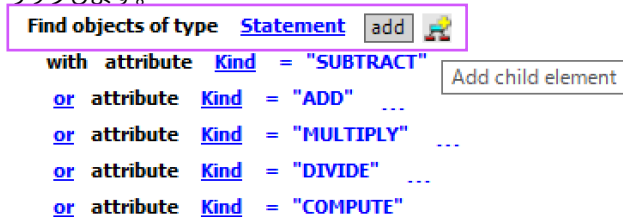
- 3.3.10 “Find objects of type Statement” 段落に COMPUTE 命令が加わりました。



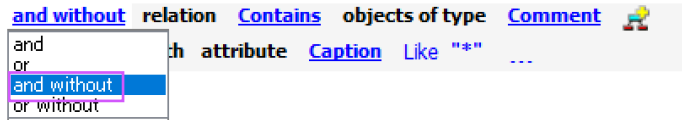
- 3.3.11 次に “and relation Child objects of type Block” 段落を一旦削除します。[Delete block] のタグをクリックします。



- 3.3.12 一旦削除したので追加した or 句が並びます。次に、削除した and 句を追加するので [Add relation] をクリックします。

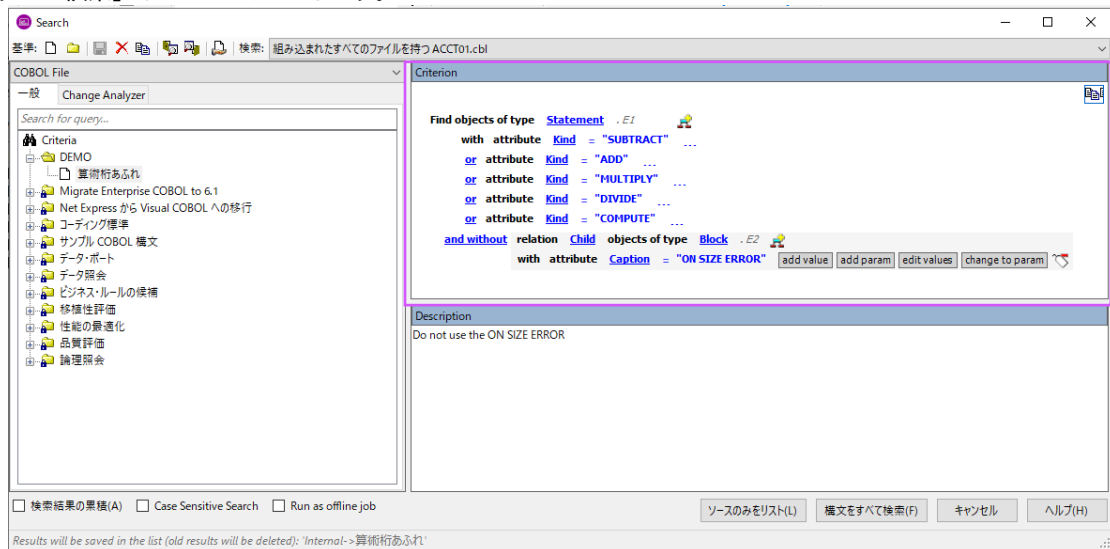


3.3.13 デフォルトが [or] で作成されるので [and without] に変更します。

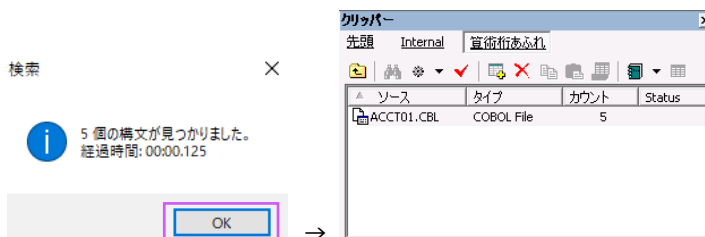


3.3.14 [Contains] を [Child] に、[Comment] を [Block] に変更します。[Like] は [=] に、[値] を [*] から [ON SIZE ERROR] に設定します。

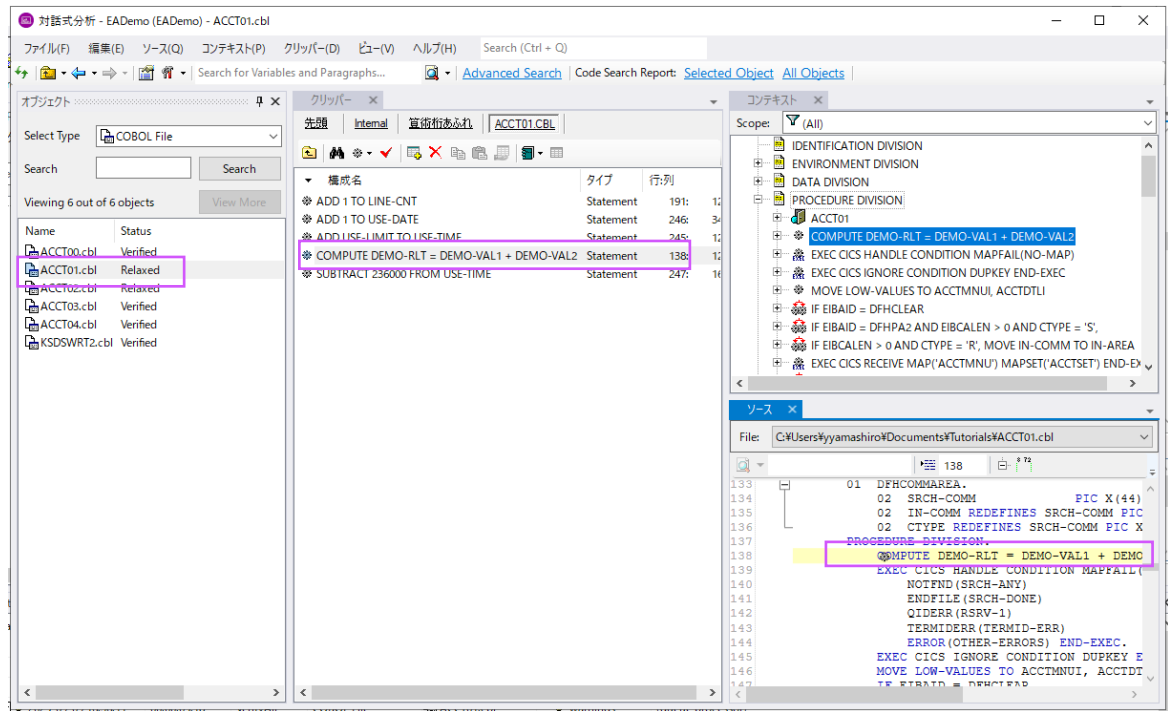
3.3.15 右側のペインには COMPUTE 命令を含む条件式が表示されます。アイコンメニューの [検索] には [すべてのオブジェクト] が、アイコンメニュー下の [検索] には [COBOL File] が選択されていることを確認後 [構文をすべて検索] ボタンをクリックします。



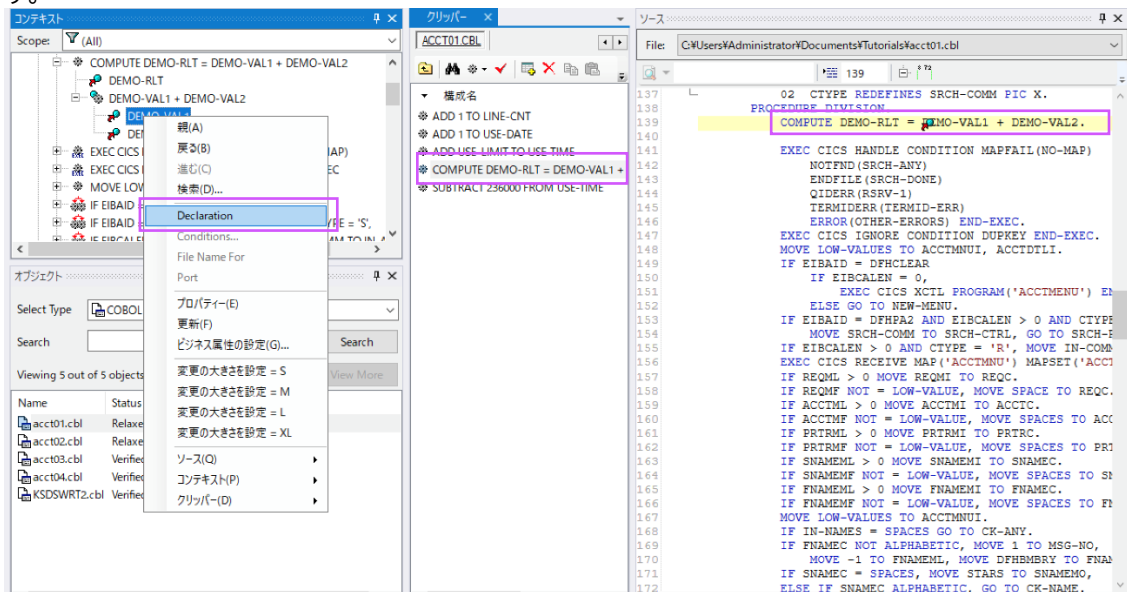
3.3.16 検索が終了すると結果が表示されます。[OK] ボタンをクリックすると [対話式分析] ウィンドウへ戻り、[クリッパー] ペインに検索内容が表示されます。



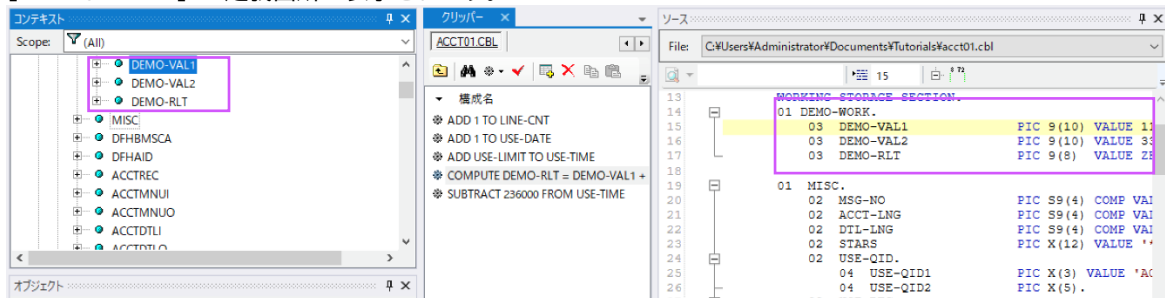
3.3.17 [クリッパー] ペインの [ACCT01.CBL] 行をダブルクリックすると検索が終了すると結果が表示されます。[クリッパー] ペインには使用命令が、[コンテキスト] ペインには該当箇所の構成が、[ソース] ペインには該当行が表示されます。



3.3.18 意図的に追加した“COMPUTE”命令行がありますので、この行をダブルクリックします。[コンテキスト] ペインで COMPUTE 命令のツリーを展開し、[DEMO-VAL1] を選択して右クリック後 [Declaration] を選択します。



3.3.19 [DEMO-VAL1] の定義箇所が表示されます。

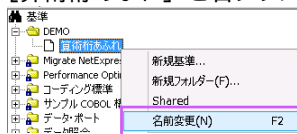


3.4 ルールのエクスポート

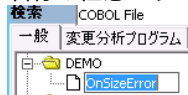
Enterprise Analyzer で使用しているルールをエクスポート後、開発環境製品である Enterprise Developer や Visual COBOL へインポートすることで、コーディングやビルド時に分析することが可能になります。本チュートリアルでは前項で作成したオリジナルルールをエクスポートして Enterprise Developer for Eclipse へインポートします。

- 3.4.1 Enterprise Analyzer の【検索】ウィンドウを開き、作成したオリジナルルールをエクスポートします。まずは名称を英語名へ変更します。
※開発環境製品の 3.0 以降のバージョンで日本語名称はサポートされていますが、これより前のバージョンをご使用の場合はルール名に英語名称を指定してください。

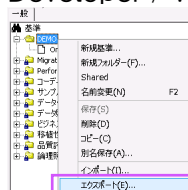
【算術桁あふれ】を右クリックして【名前の変更】を選択します



- 3.4.2 名称は任意ですが、本チュートリアルでは“OnSizeError”とします。



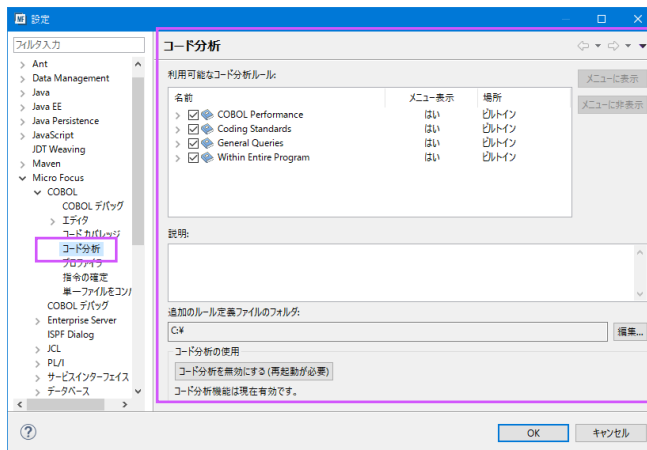
- 3.4.3 ルールが含まれるフォルダを再度右クリックして【エクスポート】を選択します。
注:必ずフォルダレベルからエクスポートしてください。フォルダレベルからエクスポートすることで Enterprise Developer / Visual COBOL 上にインポートする時、フォルダ名がそのままルールセットとして採用されます。



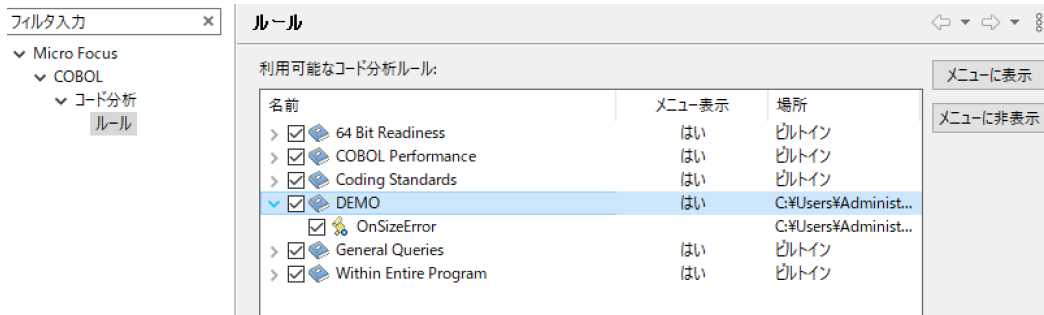
- 3.4.4 任意のパスを指定すると xml ファイルが生成されます。ファイル名称は“DEMO.xml”とします。
※ファイル名には、英語名称を指定してください。

3.5 Eclipse 版によるルールのインポート

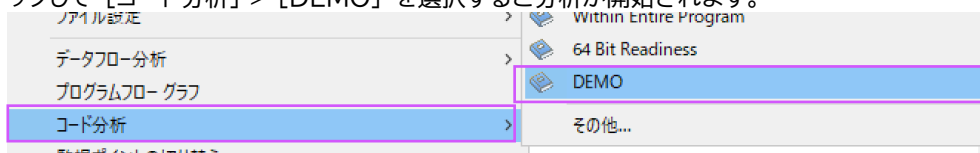
- 3.5.1 生成された xml ファイルを開発環境製品である Enterprise Developer for Eclipse へ取り込みます。Visual COBOL も同様の手順で取り込むことが可能です。
- 3.5.2 Enterprise Developer for Eclipse を起動後、任意のプロジェクトを作成し、P5 の定義内容と命令を含んだソースコードを追加します。
- 3.5.3 Enterprise Developer for Eclipse を起動後【ウィンドウ】>【設定】ウィンドウを表示して、左側ツリービューの【Micro Focus】>【COBOL】>【コード分析】を選択するとビルトインされている分析ルールが表示されます。



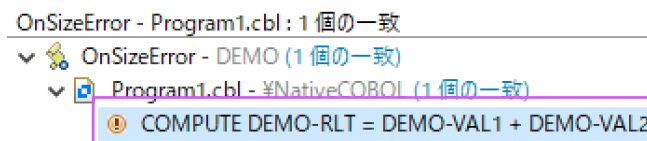
3.5.4 [追加のルール定義ファイルフォルダ] の [編集] ボタンをクリックして、エクスポートした xml ファイルが置かれているパスを指定すると “DEMO” がルールセットとして表示されていることが確認できます。チェックをオンにして [メニューに表示] をクリックします。[メニュー表示] に [はい] が表示されていることを確認後 [OK] ボタンをクリックします。



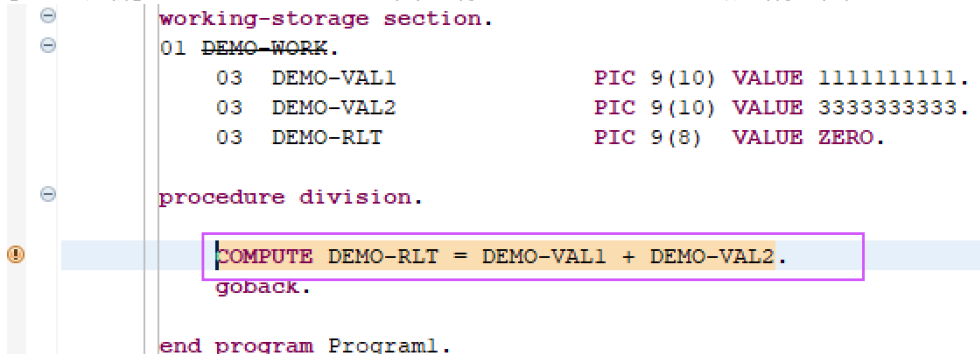
3.5.5 Eclipse のプロジェクトへ P5 の定義内容と命令を含んだソースコードをインポートします。このファイルを右クリックして [コード分析] > [DEMO] を選択すると分析が開始されます。



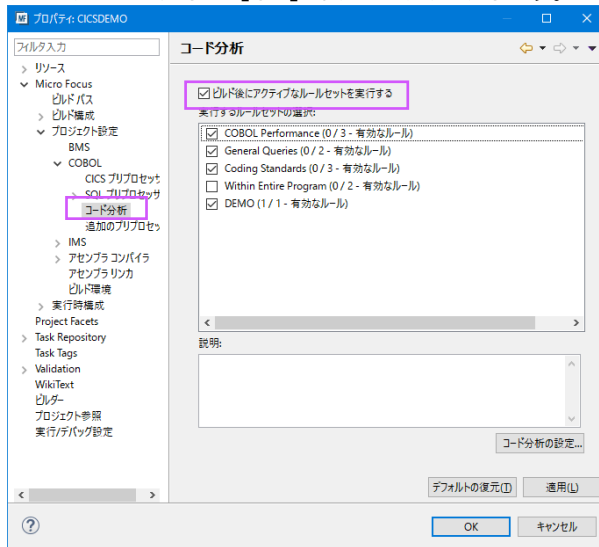
3.5.6 分析が終了すると Eclipse の [コード分析] タブに結果が表示されます。



3.5.7 [コード分析] タブの COMPUTE 命令の行をダブルクリックすると該当行に位置づけられます。

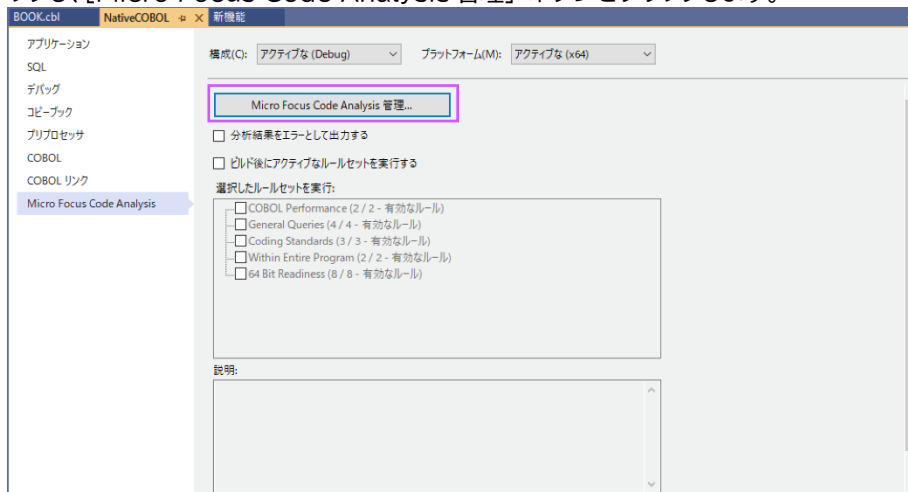


- 3.5.8 また、Eclipse のプロジェクト設定では、ビルド後に自動的にコード分析を行うことができます。プロジェクトプロパティウィンドウの左側ツリービューで [Micro Focus] > [プロジェクト設定] > [COBOL] > [コード分析] を選択し、[ビルド後にアクティブなルールセットを実行する] のチェックをオンにします。分析実行したいルールのチェックをオンにして [OK] ボタンをクリックします。

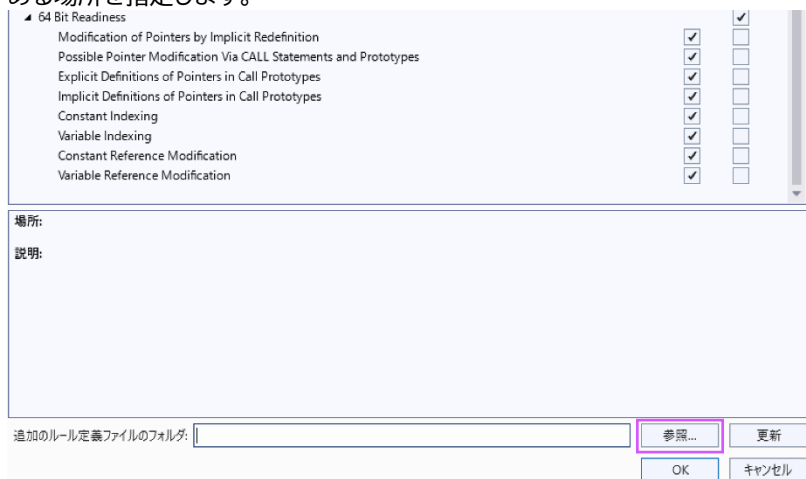


3.6 Visual Studio 版によるルールのインポート

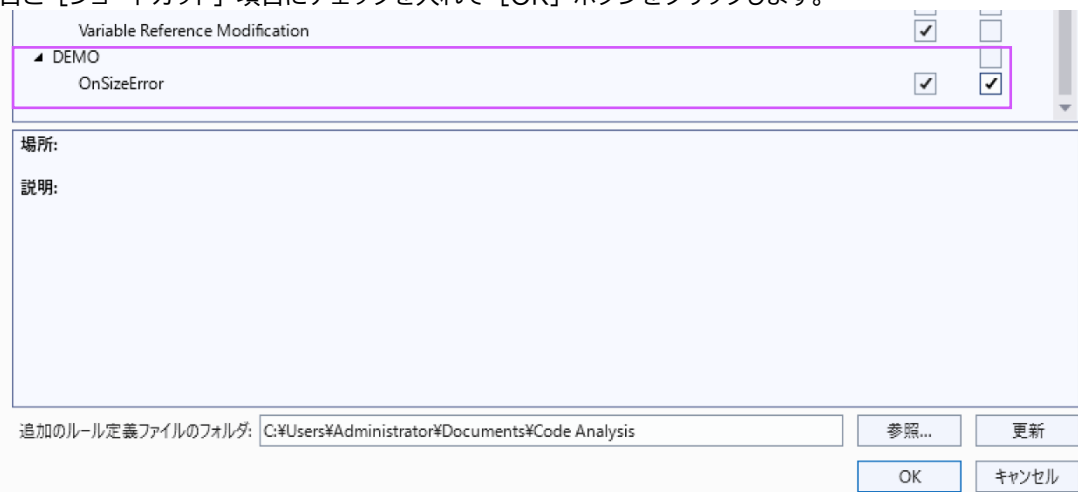
- 3.6.1 生成された xml ファイルを開発環境製品である Enterprise Developer for Visual Studio へ取り込みます。Visual COBOL も同様の手順で取り込むことが可能です。
- 3.6.2 Enterprise Developer for Visual Studio を起動後、任意のプロジェクトを作成し、P5 の定義内容と命令を含んだソースコードを追加します。
- 3.6.3 ソリューションエクスプローラーより [プロパティ] を表示して、[Micro Focus Code Analysis] ペインをクリックし、[Micro Focus Code Analysis 管理] ボタンをクリックします。



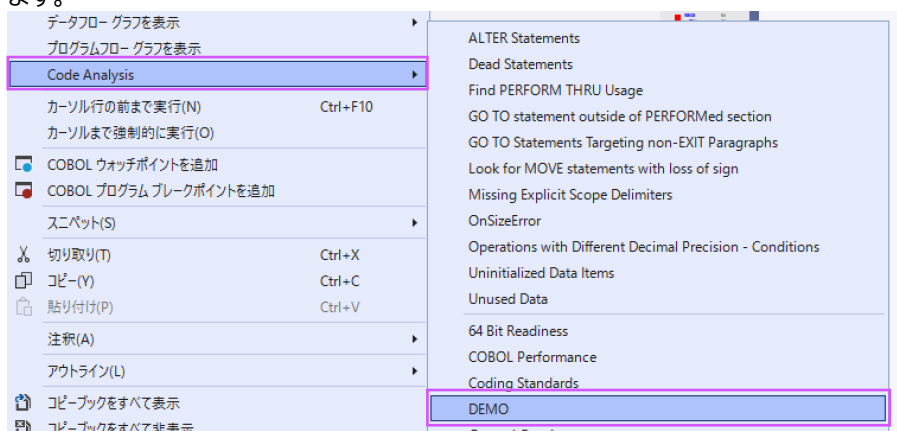
3.6.4 最下部にある「追加のルール定義ファイルのフォルダ」右横にある「参照」ボタンをクリックして XML ファイルがある場所を指定します。



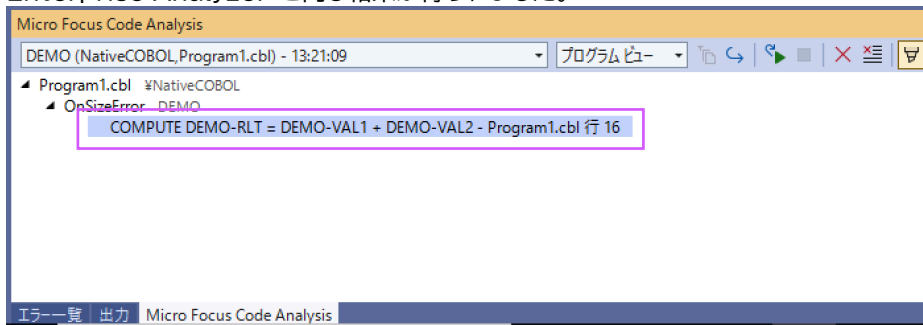
3.6.5 「更新」ボタンを押すと Enterprise Analyzer で作成したルールが表示されます。チェックボックスの「有効」項目と「ショートカット」項目にチェックを入れて「OK」ボタンをクリックします。



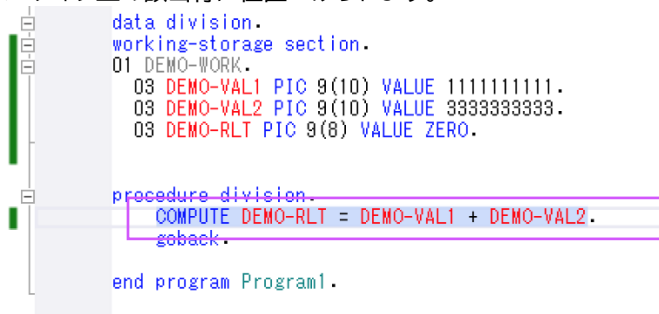
3.6.6 用意した COBOL ソースコードを右クリックして [Code Analysis] > [DEMO] を選択すると分析が開始されます。



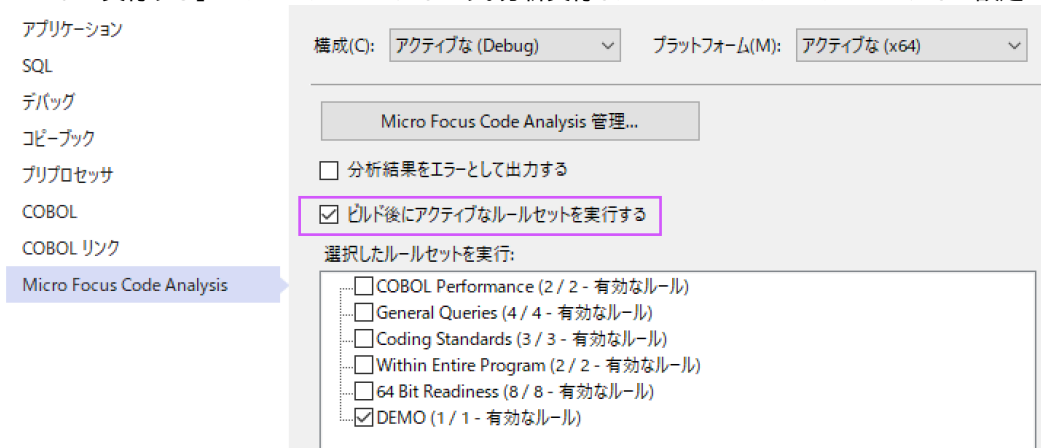
- 3.6.7 分析が終了すると Visual Studio の [Micro Focus Code Analysis] タブに結果が表示されます。Enterprise Analyzer と同じ結果が得られました。



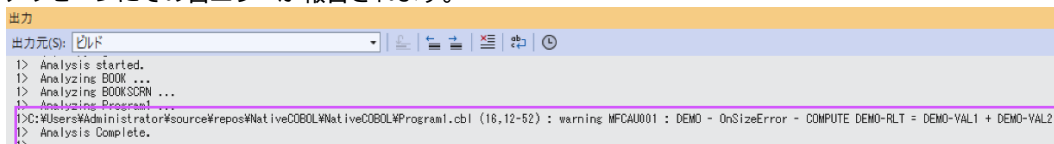
- 3.6.8 [Micro Focus Code Analysis] タブの指定されている COMPUTE 命令の行をダブルクリックするとソースエディタ上の該当行に位置づけられます。



- 3.6.9 また、Visual Studio のプロジェクト プロパティより、ビルド後に自動的にコード分析を行うことができます。プロジェクトプロパティウィンドウの [Micro Focus Code Analysis] ペインにて [ビルド後にアクティブなルールセットを実行する] のチェックをオンにします。分析実行したいルールのチェックをオンにして設定を保存します。



- 3.6.10 Visual Studio の [ビルド] メニューから [リビルド] を選択して、コンパイルを行うと自動的に解析され、出力メッセージにその旨エラーが報告されます。



3.7 最後に

Enterprise Analyzer には既存ルールを多数用意してありますが、これらをベースにオリジナルルールを作成し、分析することが可能です。加えて、ルールをコーディング標準として開発環境製品に取り込み、ビルドと同時に分析すればルールに反するコーディングを即時に知ることができます。これにより、管理者や開発者がルールを目視確認する作業工数の削減が見込めます。

Enterprise Analyzer のドキュメントは下記 URL をご参照ください。この時、ブラウザ右上の言語設定を「English」に変更してください。

https://docs.rocketsoftware.com/bundle?labelkey=enterprise_analyzer_11.0

4 免責事項

本チュートリアル of 例題ソースコードは機能説明を目的としたサンプルであり、無謬性を保証するものではありません。例題ソースコードは弊社に断りなくご利用いただけますが、本チュートリアルに関わるすべてを対象として、二次的著作物に引用する場合は著作権法に基づき適切な扱いを行ってください。

本チュートリアルで学習した技術の詳細については製品マニュアルをご参照ください。