8. 式

この章では、以下のとおり、Connectorプロジェクトに式を実装する方法について説明します。

8.1 概要
式とは何か、また、式がプロジェクトでどのように使用されるかについて説明します。
8.2 演算子
プロジェクト式でサポートされているさまざまな演算子を一覧表示して説明します。
8.3 関数
プロジェクト式でサポートされている関数を一覧表示して説明します。
8.4 式の評価と戻り値
式コンポーネントが式を評価するタイミングと返される値のタイプについて説明します。また、式の例も示します。
8.5 式の使用
式の評価値をイベントトリガとして使用すること、データ品質、再帰など、式の使用に関するさまざまな問題について説明します。
8.6 式の編集
新しい式を作成し、既存の式を編集する方法について説明します。
8.7 式のトラブルシューティング
プロジェクトからAxeda Connectorによって返されるエラーメッセージの場所を説明し、考えられるエラーメッセージと解決策の一覧表を示します。

8.1. 概要

式は、プロジェクトデータ(データアイテム、カウンタ、その他の式)、演算子、および数値(浮動小数点)で構成されるデータステートメントです。

例えば:
(DataItem1 + DataItem2) - 1

式は状態を維持せず、キューに入れられません。式ステートメントには、少なくとも1つのデータアイテム、カウンタ、または式を含める必要があります。

式は、[式の定義]ダイアログボックスを使用して作成および変更します(図8.1-1参照)。

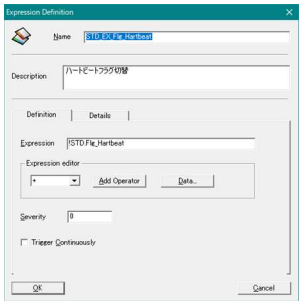

図8.1-1 式の定義ダイアログ

クライアント式は、多くの動的アクション用に定義できます。クライアント式はプロジェクト式と同じ形式ですが、実行時にクライアント側で評価されます。

動的アクションのクライアント式を定義する方法の詳細については、第10章「動的アクション」を参照してください。

注意

式は数値データ(アナログ・デジタル)に対してのみ利用可能です。文字列データ(String)には利用出来ません。

8.1.1 プロジェクトでの式の使用

式は、他のデータと同様にプロジェクトで利用できます。 式は、表示オブジェクト(動的アクション、プロセス値、アラームウィンドウなど)、アラームフィルタ(式がアラームで定義されている場合)、電子メールスタイルのコンテンツ、式ステートメント、ロガー定義、およびロジックスキーマで選択できます。

Builder全体で、式またはその他のプロジェクトデータ(データ項目、式、またはレシピ)を選択する必要がある場合は常に、データブラウザーが表示されます。 データブラウザの使用方法、およびビルダがカウンタ名に適用する構文については、84ページの「プロジェクト内のデータの選択」を参照してください。

8.1.2 式の表示

プロジェクトウィンドウで、[式]をクリックします。
定義された式はすべて、アプリケーションウィンドウの右側にある[式]ウィンドウに表示されます。

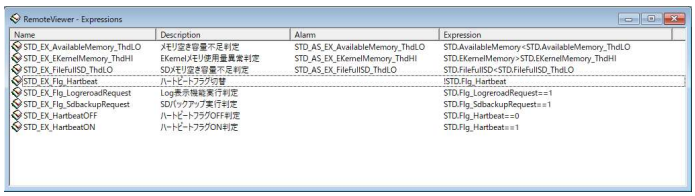

図8.1-2 式ウィンドウ

8.1.3 式のパラメータ

パラメータ	内容
Name	ユーザーが定義する式の名前。
Description	ユーザーが定義する式の説明文。
Alarm	式に関連付けられたアラームスタイル(アラームスタイルを関連付けした場合)。
Expression	データ、演算子、および浮動小数点数で設定されるデータステートメント。

8.1.4 式をディスプレイに表示する

実行時にディスプレイにデータの値を表示するために、プロジェクトデータ(カウンタ、データアイテム、および式)をディスプレイにコピーできます。 データはプロセス値オブジェクトとして挿入されます。
手順と詳細については、142ページの「プロセス値」を参照してください。

8.1.5 式のインポートとエクスポート

データアイテムをインポートおよびエクスポートすると、カウンタレコードがインポートおよびエクスポートされます。
詳細については、86ページの「プロジェクトデータのインポートとエクスポート」を参照してください。

8.2. 演算子

式には、任意の深さの括弧と任意のサイズの文字列が含まれます。
式全体は260文字未満である必要があります。

Axedaエージェントは、次のような式のさまざまなタイプの演算子をサポートしています。
・算術演算子
・比較(関係および等価)演算子
・論理(ブールおよび条件付き)演算子
・ビット演算子

8.2.1 算術演算子

算術演算子は左から右にグループ化します。それらは最高の優先順位を持ち、浮動小数点数を返します。
+演算子と-演算子の優先順位は同じであり、*、/、および%の優先順位よりも低くなります。

表8-2に、サポートされている算術演算子を優先順に示します。

算術演算子一覧

演算子	説明	例
+	加算	Data1 + Data2
-	減算	Data3 - Data2
*	乗算	DataItem4 * DataItem5 * DataItem6
/	除算	(DataItem3 * DataItem4) / 2
%	余り 第1オペランドを第2オペランドで整数回除算したときの余りを返します。	Data8 % Data2 12 % 5 = 2 7 % 3 = 1 6.5 % 2.1 = 0.2
^	べき乗 注:AxedaConnectorはブールXOR演算子をサポートしていません。	Data6 ^ 2 5 ^ 3 = 125 12 ^ 2 = 144

8.2.2 比較演算子

比較演算子には、関係演算子(>、<、<=、>=)と等価演算子(==、!=)が含まれます。比較演算子は左から右に評価され、1または0の値を返します。

関係演算子はすべて同じ優先順位を持ち、算術演算子の優先順位よりもわずかに低くなります。等価演算子の優先順位は、関係演算子の優先順位のすぐ下です。

表8-3に、これらの演算子を優先順にリストして説明します。

比較演算子一覧

演算子	説明	例
>	式の左辺の値が右辺の値よりも大きい。	Data4 > Data5 Data5 > 2
<	式の左辺の値が右辺の値よりも小さい。	Data3 < Data10 Data6 < 10
>=	式の左辺の値が右辺の値よりも大きいまたは同じ。	Data2 >= 11.5 Data6 >= 22
<=	式の左辺の値が右辺の値よりも小さいまたは同じ。	Data1 <= Data2 Data5 <= 100
==	式の左辺の値と右辺の値が同じ。(一致)	Data4 == 7
!=	式の左辺の値と右辺の値が異なる。(不一致)	Data4 != 5

8.2.3 論理演算子

論理演算子には、ブール演算子AND、OR、およびNOTと、条件演算子?、:が含まれます。これは、If … Then … Elseステートメントの省略形として用います。
論理式は左から右に評価され、trueまたはfalseを返します。
ブール演算子を使用すると、評価のために複数の条件を指定できます。それらは0または1の値を返します。

サポートされている演算子は以下のとおりです。(優先順位順)

・AND(&&)は、(ANDで関連付けされた)すべての条件がtrueのとき、trueを返します。
この演算子の優先順位はブール演算子の中で最も高くなりますが、すべての比較演算子の優先順位よりも低くなります。
・OR(||)は、(ORによって関連付けされた)任意の条件がtrueのとき、trueを返します。
・NOT(!)を使用すると、直後の条件がfalseであるかどうかを評価します。
NOTの優先順位は、不一致演算子(!=)の優先順位よりも低いため、!=演算子を使用することをお勧めします。

条件演算子を使用すると、1つ以上の条件を評価し、評価の結果に基づいてアクションを実行できます。
条件演算子を理解するための例として、次の式について考えてみます。

ExpA = (Data2 > 100) ? Data3 : (Data4 * 1000)

この例では、式の名前はExpAです。評価される条件(If)は、Data2の値が100より大きいかどうかです。
疑問符(?)はThenに対応しているため、条件がtrueの場合、Data3が返されます。コロン(:)はElseに対応するため、条件がfalseの場合、(Data4 * 1000)の結果が返されます。

多くの場合、ブール演算子を条件演算子と組み合わせて使用する必要があります。
次の例を考えてみましょう。

ExpB = ((Data8 <= 4) && !(Data2 = 50)) ? Data4 : (Data5 * 10)

この例では、式の名前はExpBです。2つの条件が評価されます。

・Data8の値が4以下であるかどうか AND
・Data2の値が50でないかどうか

両方の条件がtrueの場合、Data4が返されます。一方または両方の条件がfalseの場合、(Data5 * 10)の結果が返されます。
表8-4に、ブール演算子の一覧と説明を示します。 例では、説明されているブール演算子とともに条件演算子を使用しています。

論理演算子一覧

演算子	説明	例
&&	論理積(AND)	((Data2 = 22) && (Data3 > 50)) ? Data5 : Data7
||	論理和(OR)	((Data4 = 100) OR (Data5 >= 100)) ? Data23 : Data1
!	論理否定(NOT)	(!(Data9 > 1000)) ? (Data9 * 10) : (Data9 / 10)

8.2.4 ビット演算子

ビット演算子を使用すると、整数を表すビットを操作できます。

ビット演算子のより一般的な使用法には、条件のテストとフラグ付けが含まれます。
AxedaConnectorは、ビット演算子AND(&)、OR(|)、!&、^(の累乗)、およびNOT(~)をサポートします。

ビットごとの演算子の使用例として、Connectorプロジェクトが同一メーカー・同一モデルの高機能プリンタを監視している状況を考えてみます。

すべてのプリンタには4つの用紙トレイがあり、空のままにすると、プリンタは印刷ジョブを停止します。
空の用紙トレイを検出すると、プリンタはプロジェクトに値を送信できます。
その値のビットは、どのトレイに用紙が必要かを示します。

次のように、最初のビット(0)は一番上のトレイ、2番目のビット(1)は側面のトレイ、3番目(2)と4番目(3)のビットは、AとBのラベルが付いた特大用紙の下部トレイを示します。

```
値     :  1     2     4     8
ビット :  0     1     2     4
トレイ :  上面  側面  下部A 下部B
```

どのビットがアクティブであるかを判別するために、プロジェクト作成者はビット単位の演算子を使用して4つの式を定義し、ビットで表されるトレイが空であるかどうかを次のように確認します。

```
式の名称        式                  内容
TopEmpty        ((PDData & 1) == 1))
SideEmpty       ((PDData & 2) == 2))
LowerAEmpty     ((PDData & 4) == 4))
LowerBEmpty     ((PDData & 8) == 8))
```

それぞれの式は適切なビットに移動し、その値を評価します。
式がtrueと評価された場合、指定されたトレイは空です。それぞれの式は、適切な引き出しに用紙をセットするようにプリンタの保守担当者に通知するアクションのトリガをかけることができます。
または、アクションのトリガをかけるために、どのトレイが空であるかに関係なく、プロジェクト作成者は、4つの個別の式のそれぞれを評価するLoadPaperと呼ばれる最上位の式を追加する場合があります。

```
式の名称     内容
LoadPaper    TopEmpty || SideEmpty || LowerAEmpty || LowerBEmpty
```

8.3. 関数

実行時に、AxedaConnectorのExpressionsコンポーネントは多くの標準関数を処理できます。

補足

関数名では大文字と小文字が区別されることに注意してください。

関数一覧

関数	説明
NEG(exp)	ネゲート関数。expの+/-符号を切り替えます。
!(exp)	NOT関数。exp = 0の場合は1を返します。exp ！= 0の場合は0を返します。 !(exp)と!expはどちらも式として有効で、同等の意味です。
~(exp)	ビット単位の補数関数。expのすべてのビットを切り替えます。この操作は、expのビットパターンを知っている場合にのみ意味があります。 ~(exp)と~expはどちらも式として有効で、同等の意味です。 ~(0x00000000) = 0xFFFFFFFF つまり ~0 = -1 ~(0x00000001) = 0xFFFFFFFE つまり ~1 = -2
LN(exp)	自然対数。底 e^a2.718282
LOG(exp)	常用対数。底 10
EXP(exp)	指数。e exp
ABS(exp)	絶対値。
SQRT(exp)	ルート。
CEIL(exp)	天井関数。パラメータ値以上の最も近い整数を返します。 たとえば、CEIL(DataItemA + DataItemB)において、DataItemAの値が2.5、DataItemBの値が2.6の場合、合計(5.1)以上の最も近い整数は6です。
FLOOR(exp)	床関数。パラメータ値以下の最も近い整数を返します。 たとえば、FLOOR(DataItemA + DataItemB)において、DataItemAの値が2.5、DataItemBの値が2.6の場合、合計(5.1)以下の最も近い整数は5です。

これらの三角関数はラジアンでαを取ります。3600 =2πラジアン

SIN(α) 正弦。
COS(α) コサイン。
TAN(α) タンジェント。
SEC(α) 割線 = 1 / COS(α)
CSC(α) 余割 = 1 / SIN(α)
COT(α) Cotangent = 1 / TAN(α)

これらの6つの三角関数は、expをdoubleとして取り、ラジアンで返します。

ASIN(exp)	ArcSineInverseまたはSIN()
ACOS(exp)	ArcCosineInverse of COS()
ATAN(exp)	ArcTangentInverse of TAN()
ASEC(exp)	ArcSecantInverse of SEC()
ACSC(exp)	ArcCosecantInverse of CSC()
ACOT(exp)	ArcCotangentInverse of COT()

これらの6つの双曲線関数は、ラジアンでαを取ります。

SINH(α)	双曲線正弦 = (eα - e- α) / 2
COSH(α)	双曲線余弦 = (eα + e- α) / 2
TANH(α)	双曲線タンジェント = SINH(α) / COSH(α)
SECH(α)	双曲線セカント = 1 / COSH(α)
CSCH(α)	双曲線コセカント = 1 / SINH(α)
COTH(α)	双曲線コタンジェント = COSH(α) / SINH(α)

8.4. 式の評価と戻り値

式には文字列が含まれ、データアイテムの名前を含めることができます。
式は、1つ以上のデータアイテムの値が変更されるたびに評価されます。
一般に、式は浮動小数点値を返します。
式がtrueと評価された場合、浮動小数点1.0を返します。falseと評価された場合、0を返します。算術演算で返される浮動小数点値は、「アナログ値」と呼ばれることもあります。

演算式	演算結果
(Data1 + Data2) / 2	(アナログ値)
Data1 > 70	(0 or 1)
((Data1 & 4) == 4) && (Data2 <= 100)	(0 or 1)
((Data1 > 7) ? Data2 : (Data3 *100)	(アナログ値)
Exp1 > 50	(0 or 1)

8.5. 式の使用

式の値は、式が評価されるときに設定されます。

8.5.1 イベントのトリガとして使用される式

ロガー定義およびロジックスキーマでは、任意の式をイベントトリガとして使用できます。
値がゼロ以外の場合、イベントが発生します。式がトリガとして使用され、データアイテム値が含まれている場合、データアイテム値がゼロ以外の異なる値に変更されるたびに、イベントがトリガされます。

8.5.2 式の品質

式の値には品質特性があります。品質はGood、Bad、またはUnknownのいずれかです。
品質は、評価されるデータ項目の品質によって決定されます。
評価されたデータ項目の品質がBadの場合、式の品質はBadです。評価されたデータ項目の品質がUnknownであり、他の項目のいずれもBadでない場合、式の品質はUnknownです。式の品質は、サブスクライバーに送信されるデータ値の一部です。
品質がBadの場合、式はイベントをトリガしません。

8.5.3 再帰

式の定義に別の式の名前が含まれている場合、その式は最終的に呼び出した式自身を呼び出す可能性があります。プロジェクトで定義されているすべての式には、常に一意の名前を割り当ててください。
さらに、各式が評価され、最終的に終了するように値を返すことができることを確認してください。

8.6. 式の編集

[式]ウィンドウを使用して、式を表示、作成、変更、および削除できます。

・新しい式の作成手順

1. プロジェクトウィンドウで、[式]を右クリックし、[新しい式の追加]を選択します。

2. [式の定義]ダイアログボックス(図8.1-1参照)で、式の一意の半角英数字の名前を入力します。
   式名では大文字と小文字が区別され、関連する式グループ内で一意である必要があり、次の文字を含めることはできません。

   + - = / % ^ & | ! $ ~ * ? : > < ( ) [ ] “ , .

3. 必要に応じて式の説明を入力します)。

4. [定義]タブで、式を次のように定義します。
   ・有効なデータ名または演算子を入力し、式に含める数値を入力します。
   ・[データ]をクリックして、式に含めるプロジェクトデータを選択します。
   ・ドロップダウンメニューから演算子を選択し、[演算子の追加]をクリックして式に追加します。

5. 必要に応じて[重大度]フィールドに、式の重大度レベルを入力します。

6. 式がゼロ以外の値(および前の値と等しくない)と評価されるたびにイベント(ロジックスキーマなど)をトリガーする場合は、[連続的トリガ]を選択します。

   [連続トリガ]を選択しない場合、イベントは、式の評価の結果、ゼロからゼロ以外の値に遷移した場合にのみトリガされます。

7. この式のアラームを有効にする場合は、[詳細]タブの[アラーム]で[有効にする]チェックボックスをオンにします。次に、リストからアラームスタイルを選択します。

   注:アラームスタイルをまだ定義していない場合、アラームスタイルのリストは空です。

   アラームスタイルの定義方法については、211ページの「アラームスタイル」を参照してください。

8. [データ表現]で、データに使用するオプションのエイリアスを入力します。データがAxedaアプリケーションに表示されると、データ名の代わりにエイリアスが表示されます。

9. 度、ページ、インチなど、このデータで表される単位を入力します。

10. データを表示する形式を選択します。選択肢は、10進数(デフォルト)、16進数、8進数、および2進数です。

11. [初期化]で、生のカウントではなく、工学単位でデータの初期値を入力します。デフォルトはゼロ(0)です。

12. [OK]をクリックします。

新しい式を保存すると、Builderは式の構文を検証します。エラーが見つかった場合は、メッセージが表示され、[式の定義]ダイアログボックスが開いたままになるため、式を修正できます。エラーが見つからない場合は、式が保存され、[式の定義]ダイアログボックスが終了します。

・作成済みの式の編集手順

1. [式]ウィンドウを開きます。

2. [式]ウィンドウで、編集する式をダブルクリックします。
[式の定義]ダイアログボックスが表示されます(図8.1-1参照)。

3. 選択した式を編集します。

4. [OK]をクリックします。

・式の削除手順

1. [式]ウィンドウを開きます。

2. 削除する式を右クリックして、[削除]を選択します。

8.7. 式のトラブルシューティング

Axeda Builderは、式を定義するときに式の構文をチェックします。構文が無効な場合、エラーメッセージが表示され、式を保存する前に式を修正するように求められます。

Axeda Connectorは、式の構文もチェックします。Connectorはエラーを検出すると、ログファイル(EKernel.log)にエラーを出力します。

Axeda Connectorからプロジェクトを監視している場合、プロジェクトの実行時にエラー情報がコマンドプロンプトウィンドウに出力されまれます。

リモートデバイスからプロジェクトを実行している場合、エラーメッセージを表示するには、Axeda Connectorを実行しているデバイスのログファイルにアクセスする必要があります。

補足

Axeda Builderのリモートビューアーツールを使用して、エージェントからの出力を表示することもできます。p.347の「出力の表示」を参照してください。

エラーメッセージ一覧

16進表記	メッセージ	説明/トラブルシューティング
0xC0000A00L	KE_INVALID_EXPRESSION	式に、サポートされていない演算子または無効なデータが含まれています。 構文をチェックして、110ページの「式演算子」で説明されている演算子のみを使用していることを確認してください。 データの品質を確認し、監視対象のデバイスに対して適切なアクションを実行します。
0xC0000A01L	KE_INVALID_EXP_ITEM_NAME	データアイテムの名前が無効です。 式で使用されているデータアイテムの名前で入力ミスを確認してください。
0xC0000A02L	KE_DIVISION_BY_ZERO	ゼロと評価されたデータアイテムの値。ゼロによる除算は許可されていません。 データアイテムの定義と式を確認し、データアイテムが0を返している場合は式を修正してください。

エラーメッセージ一覧

16進表記	メッセージ	説明/トラブルシューティング
0xC0000A03L	KE_UNKNOWN_VARIABLE	式は、システムが認識しない変数名を使用します。 構文を確認してください。
0xC0000A04L	KE_EMPTY_EXPRESSION	システムは式に文字列を予期していましたが、文字列はありません。構文を確認してください。
0xC0000A05L	KE_MISSING_OPERATOR	システムはオペレーターを期待していましたが、誰もいません。構文を確認してください。
0xC0000A06L	KE_SYNTAX_ERROR	式の解析中にシステムが構文エラーを検出しました。 構文を確認してください。
0xC0000A07L	KE_UNBALANCED_PAREN	式に開き括弧または閉じ括弧がありません。 構文を確認してください。
0xC0000A08L	KE_UNKNOWN_FUNCTION	式に、システムでサポートされていない関数が含まれています。 式でサポートされるすべての操作については、110ページの「式演算子」で説明しています。 式で他の関数を使用することはできません。構文を確認してください。
0xC0000A09L	KE_UNKNOWN_OPERATOR	式に、システムでサポートされていない演算子が含まれています。 構文を確認してください。
0xC0000A0AL	KE_MISSING_IF	式は条件付き構文(？:)を使用しますが、システムは疑問符(？)の前に評価される条件を検出しません。 構文を確認してください。
0xC0000A0BL	KE_MISSING_THEN_ELSE	式は評価する条件を提供しますが、疑問符またはコロンがありません。 構文を確認してください。
0xC0000A0CL	KE_EXPRESSION_TOO_LONG	式には260文字以上が含まれています。 式によって評価されるデータ項目またはその他の式の名前を短くします。

エラーメッセージ一覧

16進表記	メッセージ	説明/トラブルシューティング
0xC0000A0DL	KE_RECURSIVE_EXPRESSION	式に別の式の名前(再帰的)が含まれているため、無限ループが発生します。 例えば、 Exp2 = Data1 + Exp4 Exp4 = Data2 + Exp2 この例では、Data1が変更されるたびに、Exp2が計算されます。これによりExp4が計算され、Exp2が再度計算されます。 パーサーはこの状態をチェックし、それを検出するとこのエラーメッセージを返します。 可能な限り、式内で式を使用することは避けてください。 「8.5.3 再帰」も参照してください。

Axeda Agentは、式オブジェクトの作成中に式を解析します。
パーサーは構文エラーを見つけることができますが、データ項目のエラーを見つけることはできません。
したがって、さまざまなタイプのエラー、定義エラー(ビルド時のパーサーから)、およびランタイムエラー(データ項目が検証されたとき)が表示されます。

たとえば、データアイテムの値で除算していて、実行時の値が0の場合、ゼロによる除算が許可されていないことを示すエラーメッセージが表示されます。