

Lab3 – 計算式とマテリアルの追加

March 2010 by M. Harada
Updated by DevTech AEC WG
Last modified: 6/9/2017

<VB.NET>VB.NET バージョン</VB.NET>

目的:この実習で学習する項目は次のとおりです。

- 計算式を追加する
- マテリアルを追加する

タスク:前の実習で定義されたコマンドを拡張して、単純な計算式とマテリアル L 字形 柱ファミリに加えます。

(0) この実習の開始点として、Lab2 に定義したコマンド クラスを使用するものとし、ファミリ エディタと「柱(メートル単位).rft」テンプレートを使用し続ける

(1) 2 つのパラメータを計算式として定義する:

- $Tw = \text{幅} / 4.0$
- $Td = \text{奥行} / 4.0$

(2) ファミリにマテリアル グループのインスタンス パラメータを追加して、ソリッドのマテリアル パラメータに関連付けることでマテリアルを割り当てる

図 1 は、マテリアルが適用された柱(左側)と、単純な計算式が適用されたファミリ タイプ ダイアログです(右側)。



図 1. マテリアルが適用された柱(左側)と、計算式とマテリアルが適用されたファミリ タイプ ダイアログ(右側)

この実習の実装と確認の手順は、下記のとおりです:

1. 別の外部コマンドを定義する
2. 計算式を追加する
3. マテリアルを追加する
4. 作成した柱をテストする

1. 別の外部コマンドを定義する

Lab2 に定義したコマンドを拡張します。新しいクラスを定義するために Lab2 をコピーするか、Lab2 自体を拡張することができます(後者の場合、Lab2 の完了状態のプロジェクトをバックアップすることをお勧めします)。

1.1 Lab2 からコマンド クラスをコピーし、Lab3 で作業するための新しいクラスを定義してください。ファイル名とクラス名は、下記のようにしてください:

- ファイル名: **3_ColumnFormulaMaterial.vb**
- コマンド クラス名: **RvtCmd_FamilyCreateColumnFormulaMaterial**

(繰り返しになりますが、ここで希望する名前を使用しても構いません。ただし、その場合、プロジェクト名など、このドキュメント内では記述されている名称は、自分でつけた名称で代替して参照してください)

```
<VB.NET>
<Autodesk.Revit.Attributes.Transaction(Autodesk.Revit.Attributes.TransactionMode.Manual)> _
<Autodesk.Revit.Attributes.Regeneration(Autodesk.Revit.Attributes.RegenerationOption.Automatic)> _
Public Class RvtCmd_FamilyCreateColumnFormulaMaterial

    ...

End Class
</VB.NET>
```

2. 計算式を追加する

単純な 2 つの計算式を、パラメータ Tw および Td に加えます。それらが幅と奥行き の 4 分の 1 になるように、各々のパラメータを設定します:

- Tw=幅/4.0
- Td=深さ/4.0

2.1 下記の関数をクラスに加えてください:

```
<VB.NET>
    ' =====
    ' (4.1) add formula
    ' =====
    Sub addFormulas()

        ' we will add the following fomulas
        '     Tw = Width / 4.0
        '     Td = Depth / 4.0
        '

        ' first get the parameter
        Dim pFamilyMgr As FamilyManager = _rvtDoc.FamilyManager

        Dim paramTw As FamilyParameter = pFamilyMgr.Parameter("Tw")
        Dim paramTd As FamilyParameter = pFamilyMgr.Parameter("Td")

        ' set the formula
        pFamilyMgr.SetFormula(paramTw, "幅 / 4.0")
        pFamilyMgr.SetFormula(paramTd, "奥行 / 4.0")

    End Sub
</VB.NET>
```

ここでは以下のメソッドが重要です:

pFamilyMgr.SetFormula(paramTw, "幅 / 4.0")

計算式の実際の定義は、UI と同じく、直感的に文字列として設定することができます。

2.2 メイン コマンドの Execute() 関数から addFormulas() を呼び出します:

```
<VB.NET>
...
    ' (3.3) add types
    addTypes()
```

```

    '' (4.1) add formula
    addFormulas()

    ''
</VB.NET>

```

2.3 コードをビルドし、実行して確認することができます。この時点で、計算式をテストしてみましょう。

- ファミリタイプ ダイアログをチェックしてください。計算式が Tw と Td が追加されていますか？
- タイプを変更した際に、Tw と Td が幅と奥行き 1/4 になっていますか？

3. マテリアルを追加する

続いて、マテリアルを追加していきます。このセクションでは、ソリッドにマテリアルを割り当てます。ファミリ テンプレート内に対象のマテリアルが存在していると仮定します。

注意: 現在、ソリッドの表現にとって重要な「レンダリングの外観」のプロパティに API からアクセスして編集することが出来ません([SPR#155053])。今後、新しいマテリアルを作成するために、この機能を拡張したいと考えています。今のところ、UI によってテンプレートにマテリアルを埋め込んで使用するようになっています。

この実習については、既にテンプレートにある「ガラス」マテリアルを割り当てます。

3.1 次の関数をクラスに加えてください:

```

<VB.NET>
    '' =====
    '' (4.2) add materials
    '' =====
    ''
    Sub addMaterials(ByVal pSolid As Extrusion)

        '' We assume Material type "Glass" exists. Template "Metric
Column.rft" include "Glass",
        '' which in fact is the only interesting one to see the effect.
        '' In practice, you will want to include in your template.
        ''
        '' To Do: For the exercise, create it with more appropriate ones in
UI, then use the name here.
        ''

        '' (1) get the materials id that we are intersted in (e.g.,
"Glass")
        ''

```

```

        Dim pMat As Elements.Material =
findElement(GetType(Elements.Material), "ガラス") ' hard coded fot simplicity.
        If pMat Is Nothing Then
            '' no material with the given name.
            Return
        End If
        Dim idMat As ElementId = pMat.Id

        '' (2a) this add a material to the solid base. but then, we cannot
change it for each column.
        ''
        'pSolid.Parameter("Material").Set(idMat)

        '' (2b) add a parameter for material finish
        ''
        '' this time we use instance parameter so that we can change it at
instance level.
        ''
        Dim pFamilyMgr As FamilyManager = _rvtDoc.FamilyManager
        Dim famParamFinish As FamilyParameter =
pFamilyMgr.AddParameter("ColumnFinish", BuiltInParameterGroup.PG_MATERIALS,
ParameterType.Material, True)

        '' (2b.1) associate material parameter to the family parameter we
just added
        ''
        Dim paramMat As Parameter = pSolid.LookupParameter("マテリアル") '' Revit
2015
        pFamilyMgr.AssociateElementParameterToFamilyParameter(paramMat,
famParamFinish)

        '' (2b.2) for our combeniencem, let's add another type with Glass
finish
        ''
        addType("ガラス", 600.0, 600.0)
        pFamilyMgr.Set(famParamFinish, idMat)

    End Sub
</VB.NET>

```

この関数は下記の処理を実行します:

- (1) 「ガラス」マテリアルの要素 id を取得する
- (2) 「マテリアル」パラメータ グループのインスタンス パラメータを追加する(例、「ColumnFinish」)
- (3) パラメータ「ColumnFinish」にソリッドのマテリアル パラメータを関連付ける
- (4) 「ガラス」に設定する「ガラス」マテリアルと呼ぶ新しいタイプを追加する

対象のマテリアルを見つけるために。ヘルパー関数 findElement() を使用することができます。

インスタンス パラメータを加えるために、第引数に True を設定します。今回は、Materials タイプのパラメータです。Materials パラメータ グループの後ろに加えます:

```
pFamilyMgr.AddParameter("ColumnFinish", BuiltInParameterGroup.PG_MATERIALS,
    ParameterType.Material, true)
```

ここでは以下の関数が重要です:

```
pFamilyMgr.AssociateElementParameterToFamilyParameter(paramMat,
    famParamFinish)
```

定義したファミリ パラメータにマテリアル パラメータを関連付けます。これによって、柱ファミリをプロジェクトにロードする際に、インスタンス パラメータとしてマテリアルを変更する機能が加わります。

最後に、「ガラス」マテリアルにもう 1 つのタイプを追加します。

3.2 メイン コマンドの Execute() 関数から addMaterials() を呼び出します:

```
<VB.NET>
...
'' (4.1) add formula
addFormulas()

'' (4.2) add materials
addMaterials(pSolid)
...
</VB.NET>
```

4. 作成した柱をテストする

ビルドしてテスト用に実行する準備が整いました。

テスト用に Revit のアドイン マニフェスト ファイルに作成したコマンドを付け加えることができます (新しいコマンドを追加するか、Lab2から置き換えることができます)。もちろん、お使いの環境と一致するよう、必要な調節を行ってください。

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<RevitAddIns>

  <AddIn Type="Command">
    <Assembly>FamilyVb.dll</Assembly>
    <AddInId>74FEF9E3-81E2-44c8-B137-0C9EE48F68A4</AddInId>
    <FullClassName>FamilyVb.RvtCmd_FamilyCreateColumnFormulaMaterial</FullClassName>
    <Text>Family Labs Define Formula and Material</Text>
```

```
<Description>Family API lab 3 to create L-  
shaped column with formula and material</Description>  
<VisibilityMode>NotVisibleInProject</VisibilityMode>  
<AccessibilityClassName>Revit.Samples.SampleAccessibilityCheck </Accessib  
ilityClassName>  
<VendorId>ADNP</VendorId>  
<VendorDescription>Autodesk, Inc. www.autodesk.com</VendorDescription>  
</AddIn>  
  
</RevitAddIns>
```

「柱(メートル単位).rft」テンプレートを使用して、ファミリ エディタを起動してください。

コマンドを実行した後で、次の点を確認してみましょう:

- ファミリ タイプ ダイアログで、計算式が Tw と Td に追加されていますか?
- タイプを変更した際に、Tw と Td が幅と奥行き 1/4 になっていますか?
- ファミリ タイプ ダイアログで、新しいパラメータ「ColumnFinish」が「マテリアルと仕上げ」グループの下に定義されましたか?
- 新しいタイプ「ガラス」が作成されましたか?
- 3Dビューに始点を変更して、グラフィックス表示オプション ダイアログで表示スタイルを「シェーディング」に設定してください。「ガラス」タイプを適用した外観の変化を確認できますか?
- プロジェクトに柱ファミリをロードして、インスタンス レベルでマテリアルを変更できますか?
また、全体として柱は正しく振る舞いますか?

次の実習では、柱ファミリに可視性コントロールを追加します。