

Lab4 – 可視性コントロールの追加

March 2010 by M. Harada
Updated by Ryuji Ogasawara
Last modified: 5/30/2024

<C#>C#バージョン</C#>

目的:この実習で学習する項目は次のとおりです。

- 可視性コントロールを加える

タスク:前の実習で定義したコマンドを拡張して、可視性コントロールを加えます。これによって簡略ビュー用の柱の表現が追加されます。

(0) Lab3 で定義したコマンド クラスを利用します。これは、この実習の開始点になります。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ます

(1) 柱用の線分表現を定義する:

- 平面図ビューでの単一のシンボル線分表現
- モデルビューでの単一のシンボル線分表現

(2) 柱インスタンスをプロジェクトに配置し、簡略ビューで単一の線分表現が使用されるように可視性コントロールを設定する

図 1 は、「詳細」/「標準」の詳細レベルと「簡略」詳細レベルでのグラフィカル表現を示すイメージです。

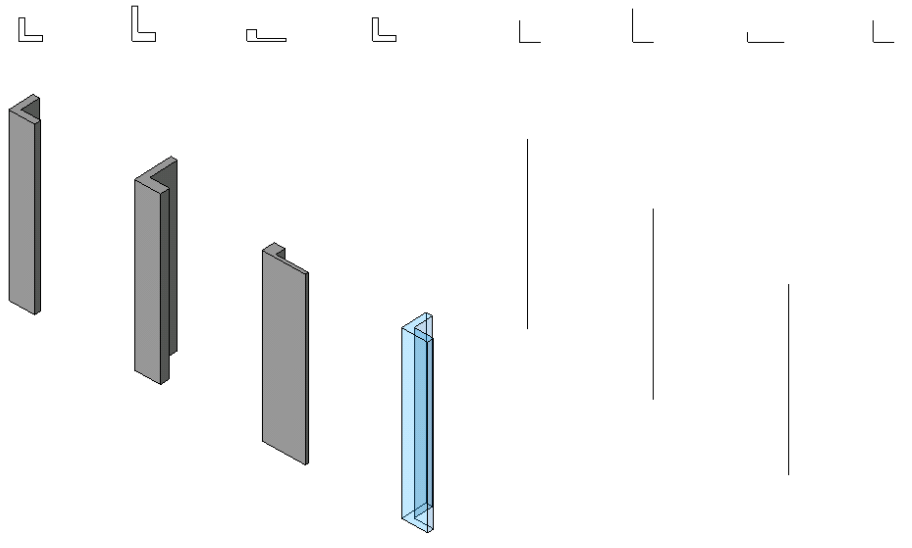


図 1. 「詳細」と「標準」の詳細レベルと「簡略」詳細レベルでのグラフィカル表現

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. 簡略レベルの線分オブジェクトを追加する
3. ソリッドの可視性を変更する
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

Lab3 で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 Lab1 をコピーするか、Lab1 自体を拡張し続けることができます(後者の場合、Lab1 の状態をバックアップすることをお勧めします)。

1.1 Lab3 からコマンド クラスをコピーし、Lab2 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: **4_ColumnVisibility.cs**
- コマンド クラス名: **RvtCmd_FamilyCreateColumnVisibility**

(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)

```
<C#>
[Autodesk.Revit.Attributes.Transaction(Autodesk.Revit.Attributes.TransactionMode.Manual)]
[Autodesk.Revit.Attributes.Regeneration(Autodesk.Revit.Attributes.RegenerationOption.Automatic)]
class RvtCmd_FamilyCreateColumnVisibility : IExternalCommand
{
    ...
}
</C#>
```

2. 簡略レベルの線分オブジェクトを追加する

柱用の単純な線分表現を加えて、可視性を簡略レベルに設定します:

- 平面図ビューでのみ表示される L 字形状内の 2 本のシンボル線分
- 3D ビューでのみ表示される垂直なモデル線分

2.1 下記の関数をクラスに追加してください:

```
<C#>
// =====
// (5.1.1) create simple line objects to be displayed in coarse level
// =====
public void addLineObjects()
{
    //
    // define a simple L-shape detail line object
    //
    // 0
    // +          h = height
    // |          (we also want to draw a vertical line here at point 1)
    // d |
    // +-----+
    // 1       2
    //      w
    //

    // sizes
    double w = mmToFeet( 600.0 ); // modified to match reference plane.
    otherwise, alignment won't work.
    double d = mmToFeet( 600.0 );
    double h = mmToFeet( 4000.0 ); // distance between Lower and Upper Ref
    Level.
```

```

double t = mmToFeet( 50.0 ); // slight offset for visibility

// define vertices
//
XYZ[] pts = new XYZ[] { new XYZ( ( -w / 2.0 ) + t, d / 2.0, 0.0 ), new
XYZ( ( -w / 2.0 ) + t, ( -d / 2.0 ) + t, 0.0 ), new XYZ( w / 2.0, ( -d /
2.0 ) + t, 0.0 ) };
XYZ ptH = new XYZ( ( -w / 2.0 ) + t, ( -d / 2.0 ) + t, h ); // this is
for vertical line.

//
// (2) create a sketch plane
//
// we need to know the template. If you look at the template (Metric
Column.rft) and "Front" view,
// you will see "Reference Plane" at "Lower Ref. Level". We are going
to create lines there.
// findElement() is a helper function that find an element of the given
type and name. see below.
// Note: we did the same in creating a profile.
//
ReferencePlane pRefPlane = findElement( typeof( ReferencePlane ), "参照
面" ) as ReferencePlane;
SketchPlane pSketchPlane = SketchPlane.Create(_rvtDoc,pRefPlane.Plane);

// for vertical line, we draw a straight vertical line at the point[1]
XYZ normal = new XYZ(1.0, 0.0, 0.0);
Plane pGeomPlaneH = Plane.CreateByNormalAndOrigin(normal, pts[1]);
SketchPlane pSketchPlaneH = SketchPlane.Create(_rvtDoc,pGeomPlaneH);

// (3) create line objects: two symbolic curves on a plan and one model
curve representing a column like a vertical stick.
//
Line geomLine1 = Line.CreateBound ( pts[0], pts[1];
Line geomLine2 = Line.CreateBound ( pts[1], pts[2];
Line geomLineH = Line.CreateBound ( pts[1], ptH);

SymbolicCurve pLine1 = _rvtDoc.FamilyCreate.NewSymbolicCurve( geomLine1,
pSketchPlane );
SymbolicCurve pLine2 = _rvtDoc.FamilyCreate.NewSymbolicCurve( geomLine2,
pSketchPlane );

ModelCurve pLineH = _rvtDoc.FamilyCreate.NewModelCurve( geomLineH,
pSketchPlaneH ); // this is vertical line

// set the visibilities of two lines to coarse only
//
FamilyElementVisibility pVis = new
FamilyElementVisibility( FamilyElementVisibilityType.ViewSpecific );
pVis.IsShownInFine = false;
pVis.IsShownInMedium = false;

pLine1.SetVisibility( pVis );
pLine2.SetVisibility( pVis );

```

```

        FamilyElementVisibility pVisH = new
FamilyElementVisibility( FamilyElementVisibilityType.Model );
        pVisH.IsShownInFine = false;
        pVisH.IsShownInMedium = false;

        pLineH.SetVisibility( pVisH );
    }
</C#>

```

線分を作図するために、頂点の初期値を定義するところから始めます。コードを単純化するためにハードコーディングしています。

シンボル線分やモデル線分を作図する際、Document.FamilyCreate を使用することができます:

- ☐ _rvtDoc.FamilyCreate.NewSymbolicCurve(geomLine1, pSketchPlane)
- ☐ _rvtDoc.FamilyCreate.NewModelCurve(geomLineH, pSketchPlaneH)

可視性を設定する箇所では重要なのは、次の部分です:

```

FamilyElementVisibility pVis = new
    FamilyElementVisibility( FamilyElementVisibilityType.ViewSpecific );
pVis.IsShownInFine = false;
pVis.IsShownInMedium = false;
pLine1.SetVisibility( pVis );

```

FamilyElementVisibility は、特定の要素の可視性に関する情報を追跡するヘルパー クラスです。コンストラクターに、ビュー固有なのかモデルなのかを指定するための FamilyElementVisibilityType オブジェクトを引数に渡します:

- ViewSpecific – 要素は要素が作成されるビューでのみ表示されます。これは、詳細コンポーネント、注釈、ビュー固有のインポートのような項目に適用します。
- Model - 要素は 3D ビューと他のモデルビューで表示されます。

既定値では、可視性はすべて True に設定されています。各ビューの状況に合わせて、それらを False に設定することで、表示をオフにすることができます。ここでは、それを各要素に設定することにします。

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

```

<C#>
    ...

    // (4.2) add materials
    addMaterials(pSolid);

    // (5.1) add visibilities
    addLineObjects();

```

```
</c#>
```

2.3 この時点でコードをビルドし、実行して確認することができます。

3. ソリッドの可視性を変更します

最後に、簡略ビューにおいて、L 字形ソリッドの押し出しの可視性をオフにする必要があります。

3.1 次の関数をクラスに追記してください:

```
<C#>
// =====
// (5.1.2) set the visibility of the solid not to show in coarse
// =====
public void changeVisibility( Extrusion pSolid )
{
    // set the visibility of the model not to shown in coarse.
    //
    FamilyElementVisibility pVis = new
FamilyElementVisibility( FamilyElementVisibilityType.Model );
    pVis.IsShownInCoarse = false;

    pSolid.SetVisibility( pVis );
}
</c#>
```

これは、以前に線分オブジェクトに行ったものと同じ手法です。FamilyElementVisibilityType.Model を使用して、FamilyElementVisibility クラスを作成してください。IsShownInCoarse プロパティを False にして、ソリッドに設定してください。

4. 作成した柱をテストする

コードはテスト用にビルドし、実行して確認することができます。

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、Lab3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>FamilyCs.dll</Assembly>
    <AddInId>943A6BDC-2D64-4033-A3B0-E18BF3598006</AddInId>
    <FullClassName>FamilyCs.RvtCmd_FamilyCreateColumnVisibility</FullClassNam
```

```
e>
  <Text>Family Labs Define Visibility</Text>
  <Description>Family API lab 4 to create L-
shaped column with visibility settings</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。

コマンドを実行して、以下の点について確認してみましょう:

- 平面図でシンボル線分オブジェクトが見えますか? 3D ビューでモデル ラインが見えますか?
- 可視性レベルを変更して、ビューの中でグレイアウトされた線分を見ることができますか?
- 線分は、タイプやレベル、高さを変更すると変更されますか?
- 新しいプロジェクトに柱ファミリをロードして柱を配置してください。ビューの詳細レベルを簡略に変更して、様々なレベルと方向から比較してください。

おめでとうございます!!

ファミリ API の実習が完了しました。

Autodesk Developer Network