

Lab 5 – モデル生成

Created by M. Harada, July 2010

Updated by DevTech AEC WG

Last modified: 6/9/2017

<VB.NET>VB.NET バージョン</VB.NET>

目的:この実習では、Revit モデルを作成する方法を学習していきます。学習する項目は次のとおりです。

- 壁、ドア、窓、屋根のような建築要素のインスタンスを作成する

タスク:長方形のフットプリントに 4 つの壁と 1 つのドア、3 つの窓と屋根から構成されるシンプルな「家」を作成するコマンドを記述します。

1. 長方形のフットプリントで 4 つの壁を作成する
2. 最初の壁へドアを追加する
3. 窓を残りの壁に追加する
4. 壁の上に傾斜屋根を加える

図 1 は、この実習で定義するコマンドを実行した後の出力されるサンプル画像を示します:

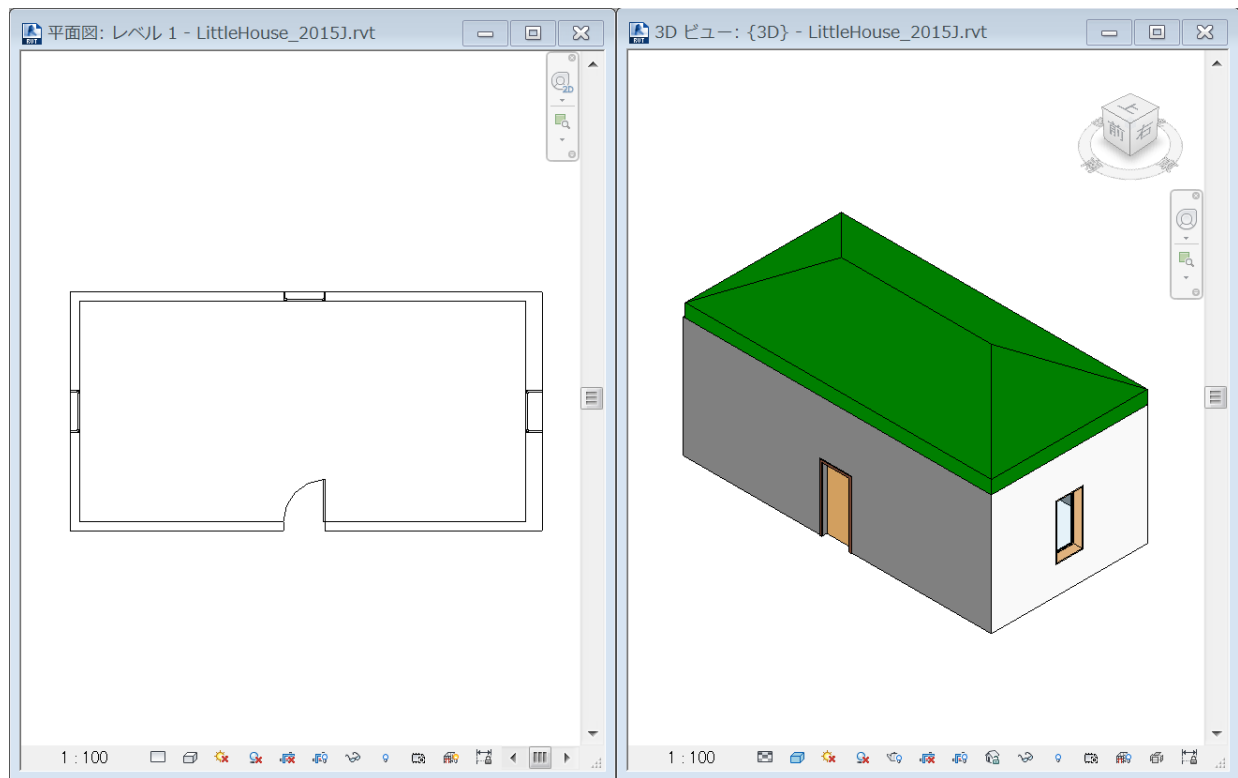


図 1. 4 つの壁とドア、窓と屋根から構成されるシンプルな家を作成

この実習の実装と確認の手順は、下記のとおりです:

1. 新しい外部コマンドを定義する
2. 壁を作成する
3. ドアを追加する
4. 窓を追加する
5. 傾斜屋根を加える
6. サマリ

1. 新しい外部コマンドを定義する

現在のプロジェクトに新しい外部コマンドを追加します。

1.1 新しいファイルを追加して、プロジェクトに新しい外部コマンドを定義します。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: **5_ModelCreation.vb**
- コマンドクラス名: **ModelCreation**

追加する名前空間:

- Autodesk.Revit.DB.Structure(StructuralType のファミリ インスタンスの作成用)

追加考察:

ElementFiltering クラスから次の関数を使用します:

- ElementFiltering.FindFamilyType ()
- ElementFiltering.FindElement ()

1.2 前の実習で行ったように、メンバ変数を定義します。DB レベルのアプリケーションとドキュメントをそれぞれ保持する m_rvtApp と m_rvtDoc を定義します。下記はその例です:

```
<VB.NET>

'' Model Creation - learn how to create elements

<Transaction(TransactionMode.Manual)> _
Public Class ModelCreation
    Implements IExternalCommand

    '' member variables
    Dim m_rvtApp As Application
    Dim m_rvtDoc As Document

    Public Function Execute(ByVal commandData As ExternalCommandData, _
                           ByRef message As String, _
                           ByVal elements As ElementSet) _
        As Result _
```

```

        Implements IExternalCommand.Execute

        ''' Get the access to the top most objects.

        Dim rvtUIApp As UIApplication = commandData.Application
        Dim rvtUIDoc As UIDocument = rvtUIApp.ActiveUIDocument
        m_rvtApp = rvtUIApp.Application
        m_rvtDoc = rvtUIDoc.Document

        ''' ...

        Return Result.Succeeded

    End Function

End Class
</VB.NET>

```

2. 壁を作成する

前の実習では、新しいジオメトリ要素を作成するためにアプリケーション 伊部ジェクトを使用しました:

- Line.CreateBound(point1, point2)

壁の新しいインスタンスを作成するために、Wall クラスの静的な “Create” メソッドを使用することができます。また、最初の引数としてドキュメントを渡すことができます。壁を作成する5つのオーバーロード メソッドが存在します (Revit API 開発者用ガイドの[「壁」項目](#)を参照して下さい)。例えば、下記は与えられたレベルに既定の壁スタイルで新しい長方形のプロファイル(外形線)を作成します:

- Wall.Create(doc, baseCurve, level, isStructural)

ベース カーブは2つのエンドポイントを使用して、Line.CreateBoundで定義することができます。壁を配置したい場所で、レベルを見つけるために ElementFiltering.FindElement() を使用することができます (例えば “レベル 1”)。

上部の壁を上レベルに拘束したい場合、もしくは“上部の拘束”を”レベル 2” に設定したい場合には、新しい壁を作成した後で、パラメータを追加で設定することができます。; “上部の拘束” のBuiltInParameter に対応するのはWALL_HEIGHT_TYPEです:

- aWall.Parameter(BuiltInParameter.WALL_HEIGHT_TYPE).Set(level2.Id)

```

<VB.NET>
    ''' create four walls

    Function CreateWalls() As List(Of Wall)

        ''' hard coding the size of the house for simplicity

```

```

Dim width As Double = mmToFeet(10000.0)
Dim depth As Double = mmToFeet(5000.0)

'' get the levels we want to work on.
'' Note: hard coding for simplicity. Modify here you use a different
'' template.

Dim level1 As Level = _
    ElementFiltering.FindElement(m_rvtDoc, GetType(Level), "レベル 1")
If level1 Is Nothing Then
    TaskDialog.Show("Revit Intro Lab", "Cannot find (Level 1). Maybe
you use a different template? Try with DefaultMetric.rte.")
    Return Nothing
End If

Dim level2 As Level = _
    ElementFiltering.FindElement(m_rvtDoc, GetType(Level), "レベル 2")
If level2 Is Nothing Then
    TaskDialog.Show("Revit Intro Lab", "Cannot find (Level 2). Maybe
you use a different template? Try with DefaultMetric.rte.")
    Return Nothing
End If

'' set four corner of walls.
'' 5th point is for convenience to loop through.

Dim dx As Double = width / 2.0
Dim dy As Double = depth / 2.0

Dim pts As New List(Of XYZ)(5)
pts.Add(New XYZ(-dx, -dy, 0.0))
pts.Add(New XYZ(dx, -dy, 0.0))
pts.Add(New XYZ(dx, dy, 0.0))
pts.Add(New XYZ(-dx, dy, 0.0))
pts.Add(pts(0))

'' flag for structural wall or not.
Dim isStructural As Boolean = False

'' save walls we create.
Dim walls As New List(Of Wall)(4)

'' loop through list of points and define four walls.
For i As Integer = 0 To 3
    '' define a base curve from two points.
    Dim baseCurve As Line = _
        Line.CreateBound(pts(i), pts(i + 1))
    '' create a wall using the one of overloaded methods.

```

```

        Dim aWall As Wall = _
            Wall.Create(m_rvtDoc, baseCurve, level1.Id, isStructural)
        ' set the Top Constraint to Level 2
        aWall.Parameter(BuiltInParameter.WALL_HEIGHT_TYPE).Set(level2.Id)
        ' save the wall.
        walls.Add(aWall)
    Next

    ' This is important. we need these lines to have shrinkwrap working.
    m_rvtDoc.Regenerate()
    m_rvtDoc.AutoJoinElements()

    Return walls

End Function
</VB.NET>

```

グラフィックスの再描画

この関数の終わりの2行には注意が必要です:

```

m_rvtDoc.Regenerate ()
m_rvtDoc.AutoJoinElements ()

```

これら2行を加えることによって、Revitは最初に個々の壁のグラフィックス情報を更新して、その後、壁のコーナーの自動調整を実行します。

実習:

- 長方形のフットプリントを形成する4つの壁を作成して、作成された壁のリストを返す関数を実装してください。
この実習では、壁を任意の位置に配置できます

3. ドアを追加する

ドアや窓のようなコンポーネント ファミリのインスタンスを作成する際には、NewFamilyInstance() メソッドを使用する必要があります。NewfamilyInstance() には、12のオーバーロード メソッドがあります。どのメソッドを使用するかは、どの種類のファミリ インスタンスを作成したいかに依って変わります (例えば、参照に拘束されるか、または、自由に配置できるかなど)。Revit API 開発者用ガイド の「[ファミリインスタンス](#)」項目には、どのメソッド形式をどのような状況で利用するかというコメントと共に、要素カテゴリ毎に適用可能なオーバーロードメソッドの一覧を確認できます。より詳細な情報に関しては、そちらを参照してください。

下記は、ドアを追加する例です。ここでは、NewFamilyInstance() の次の形式を使用します:

- m_rvtDoc.Create.NewFamilyInstance(xyzLocation, aFamilySymbol, hostObject, level, structuralType)

下記は、与えられた壁にドアを追加する関数の例です。この例では、ホストとして与えられた壁の中心にドアを配置しています。ドアは壁の下部に拘束されます(日本用のテンプレートを使ったプロジェクト上で実行する際には、ハードコードされたファミリ名を日本語版に適合させる必要があります):

```
<VB.NET>
    ' add a door to the center of the given wall.
    ' cf. Developer Guide p140. NewFamilyInstance() for Doors and Window.

    Sub AddDoor(ByVal hostWall As Wall)

        ' hard coding the door type we will use.
        ' e.g., "M_Single-Flush: 0915 x 2134mm"

        Const doorFamilyName As String = "片側フラッシュ"
        Const doorTypeName As String = "0915 x 2134mm"
        Const doorFamilyAndTypeName As String = _
            doorFamilyName + ": " + doorTypeName

        ' get the door type to use.

        Dim doorType As FamilySymbol = _
            ElementFiltering.FindFamilyType(m_rvtDoc, GetType(FamilySymbol), _
            doorFamilyName, doorTypeName, BuiltInCategory.OST_Doors)
        If doorType Is Nothing Then
            TaskDialog.Show("Revit Intro Lab", "Cannot find (" + _
            doorFamilyAndTypeName + "). Maybe you use a different template?
Try with DefaultMetric.rte.")
        End If

        ' get the start and end points of the wall.

        Dim locCurve As LocationCurve = hostWall.Location
        Dim pt1 As XYZ = locCurve.Curve.GetEndPoint(0)
        Dim pt2 As XYZ = locCurve.Curve.GetEndPoint(1)
        ' calculate the mid point.
        Dim pt As XYZ = (pt1 + pt2) / 2.0

        ' we want to set the reference as a bottom of the wall or level1.

        Dim idLevel1 As ElementId = _
            hostWall.Parameter(BuiltInParameter.WALL_BASE_CONSTRAINT).AsElementId
        Dim level1 As Level = m_rvtDoc.GetElement(idLevel1)

        ' finally, create a door.

        Dim aDoor As FamilyInstance =
            m_rvtDoc.Create.NewFamilyInstance( _
            pt, doorType, hostWall, level1, StructuralType.NonStructural)

    End Sub
</VB.NET>
```

4. 窓を追加する

窓のインスタンスの作成は、基本的にドアと同じです。同じ `NewFamilyInstance()` メソッドを使用してもかまいません:

- `m_rvtDoc.Create.NewFamilyInstance(xyzLocation, aFamilySymbol, hostObject, level, structuralType)`

検討すべき追加の項目は、敷居の高さを窓に加える点です。そうしないと、窓は壁の底に配置されてしまいます。対応するパラメータのセットによりそれをセットすることができます:

- `aWindow.Parameter(BuiltInParameter.INSTANCE_SILL_HEIGHT_PARAM).Set(sillHeight)`

下記は、与えられた壁に窓を追加する関数の例です。この例では、ホストとして与えられた壁の中心に窓を配置しています。窓は敷居高さ915mmで壁の最下段に拘束されます。

```
<VB.NET>
''' add a window to the center of the wall given.

Sub AddWindow(ByVal hostWall As Wall)

    ''' hard coding the window type we will use.
    ''' e.g., "M Fixed: 0915 x 1830mm

    Const windowFamilyName As String = "固定"
    Const windowTypeName As String = "0915 x 1830 mm"
    Const windowFamilyAndTypeName As String = _
        windowFamilyName + ": " + windowTypeName
    Dim sillHeight As Double = mmToFeet(915)

    ''' get the door type to use.

    Dim windowType As FamilySymbol = _
        ElementFiltering.FindFamilyType(m_rvtDoc, GetType(FamilySymbol),
            windowFamilyName, windowTypeName, BuiltInCategory.OST_Windows)
    If windowType Is Nothing Then
        TaskDialog.Show("Revit Intro Lab", "Cannot find (" + _
            windowFamilyAndTypeName + "). Please use DefaultMetric.rte?")
    End If

    ''' get the start and end points of the wall.

    Dim locCurve As LocationCurve = hostWall.Location
    Dim pt1 As XYZ = locCurve.Curve.GetEndPoint(0)
    Dim pt2 As XYZ = locCurve.Curve.GetEndPoint(1)
    ''' calculate the mid point.
    Dim pt As XYZ = (pt1 + pt2) / 2.0

    ''' we want to set the reference as a bottom of the wall or level1.

    Dim idLevel1 As ElementId = _
        hostWall.Parameter(BuiltInParameter.WALL_BASE_CONSTRAINT).AsElementId
    Dim level1 As Level = m_rvtDoc.GetElement(idLevel1)
```



```

    '' create a window.

    Dim aWindow As FamilyInstance = m_rvtDoc.Create.NewFamilyInstance( _
        pt, windowType, hostWall, level1, StructuralType.NonStructural)

    '' set the sill height

    aWindow.Parameter(BuiltInParameter.INSTANCE_SILL_HEIGHT_PARAM). _
        Set(sillHeight)

End Sub
</VB.NET>

```

実習:

- 引数として壁をとり、ドアまたは窓を与えられた壁に追加する関数を実装してください。
ドアや窓のファミリ タイプはハードコードして結構です

5. 屋根を加える

システム ファミリとコンポーネント ファミリのインスタンスの作成方法を学習しました。屋根はシステム・ファミリであるため、指定のメソッドを使用する必要があります。屋根には2種類のメソッド、NewFootPrintRoof() と NewExtrusionRoof() が用意されています。Revit API 開発者用ガイド の「[屋根](#)」項目では、使用方法に関する詳細な記述があります。「NewRoof」SDKサンプルは、フットプリントと押し出し屋根の両方の使用方法を実証しています。

今回は、屋根のフットプリント の作成に次のメソッドを使用します。

- m_rvtDoc.Create.NewFootPrintRoof(footprintCurve, level, roofType, curveMapping)

最後の引数 curveMapping は、ModelCurveArray データ型です。空のモデル カーブ配列を渡すと、関数が作成された屋根のカーブをそれに代入します。勾配角度のようなカーブのプロパティを設定するために、このカーブを後で使用する場合があります。

傾斜屋根を作成するために、屋根用に作成されたフットプリント モデル カーブの各エッジで角度を設定する必要があります。屋根を作成したら、NewFootPrintRoof()メソッドから返されたカーブマッピングをループすることで、適切な値を DefineSlope() と SlopeAngle() に設定します:

- aRoof.DefineSlope(modelCurve) = True
- aRoof.SlopeAngle(modelCurve) = <some angular value>

下記は、フットプリント屋根を作成して、傾斜を設定するスケルトン コードです:

```

<VB.NET>
    '' create a roof.
    Dim aRoof As FootPrintRoof = m_rvtDoc.Create.NewFootPrintRoof( _
        footprint, level2, roofType, mapping)

    '' set the slope
    For Each modelCurve As ModelCurve In mapping

```

```

aRoof.DefinesSlope(modelCurve) = True
aRoof.SlopeAngle(modelCurve) = 0.5
Next
</VB.NET>

```

下記は、渡された4つの壁の上にフットプリント屋根を作成するサンプルコードです (注意:壁の厚さで屋根の4つのコーナーを定義しています。そうでないと、屋根は壁の中心線に基づいて配置されます)。

```

<VB.NET>
    ' add a roof over the rectangular profile of the walls we created.

    Sub addRoof(ByVal walls As List(Of Wall))

        ' hard coding the roof type we will use.
        ' e.g., "Basic Roof: Generic - 400mm"

        Const roofFamilyName As String = "標準屋根"
        Const roofTypeName As String = "一般 - 400 mm"
        Const roofFamilyAndTypeName As String = _
            roofFamilyName + ": " + roofTypeName

        ' find the roof type
        Dim roofType As RoofType = _
            ElementFiltering.FindFamilyType( _
                m_rvtDoc, GetType(RoofType), roofFamilyName, roofTypeName)

        If roofType Is Nothing Then
            TaskDialog.Show("Revit Intro Lab", "Cannot find (" + _
                roofFamilyAndTypeName + "). Please use DefaultMetric.rte.")
        End If

        ' wall thickness to adjust the footprint of the walls
        ' to the outer most lines.
        ' Note: this may not be the best way.
        ' but we will live with this for this exercise.

        Dim wallThickness As Double = walls(0).Width
        Dim dt As Double = wallThickness / 2.0
        Dim dts As New List(Of XYZ) (5)
        dts.Add(New XYZ(-dt, -dt, 0.0))
        dts.Add(New XYZ(dt, -dt, 0.0))
        dts.Add(New XYZ(dt, dt, 0.0))
        dts.Add(New XYZ(-dt, dt, 0.0))
        dts.Add(dts(0))

        ' set the profile from four walls

        Dim footPrint As New CurveArray()
        For i As Integer = 0 To 3
            Dim locCurve As LocationCurve = walls(i).Location
            Dim pt1 As XYZ = locCurve.Curve.GetEndPoint(0) + dts(i)
            Dim pt2 As XYZ = locCurve.Curve.GetEndPoint(1) + dts(i + 1)
            Dim line As Line = Line.CreateBound(pt1, pt2)
            footPrint.Append(line)
        End For
    End Sub

```

```

Next

    ' get the level2 from the wall

    Dim idLevel2 As ElementId = walls(0).Parameter( _
        BuiltInParameter.WALL_HEIGHT_TYPE).AsElementId
    Dim level2 As Level = m_rvtDoc.GetElement(idLevel2)

    ' footprint to morel curve mapping

    Dim mapping As New ModelCurveArray

    ' create a roof.

    Dim aRoof As FootPrintRoof = _
        m_rvtDoc.Create.NewFootPrintRoof(footPrint, level2, roofType, _
        mapping)

    ' setting the slope

    For Each modelCurve As ModelCurve In mapping
        aRoof.DefinesSlope(modelCurve) = True
        aRoof.SlopeAngle(modelCurve) = 0.5
    Next

End Sub
</VB.NET>

```

実習:

- 壁のリストをとって、その上の屋根を配置する関数を実装します。ここでは、長方形のフットプリントを形成する4つの壁が渡されるものとします。
- 壁、ウィンドウ、ドア、屋根をすべて一緒に作成して、シンプルな家を作成するコマンドを定義してみましょう。

6. サマリ

この実習では、Revit モデルを作成する方法を学習しました。学習した項目は次のとおりです。

- 壁、ドア、窓、屋根のような建築要素のインスタンスを作成する