

Lab 2 – 選択

May, 2011 by A. Nagy
Updated by DevTech AEC WG
Last modified: 8/4/2015

<VB.NET>VB.NET バージョン</VB.NET>

目的:この実習では、様々な選択機能を使用する方法を学習します。学習する項目は次のとおりです。

- オブジェクト/オブジェクト群/点(座標)を選択する
- 要素ジオメトリを選択する
- 選択を制限する

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作成する
2. 現在の選択内容を示す
3. オブジェクトを選択する
4. オブジェクト群を選択する
5. 点を選択する
6. 面/エッジを選択する
7. 選択フィルタ
8. 対話的に家を作成する
9. サマリ

1. 新しい外部コマンドを作成する

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: **2_Selection.vb**
- コマンド クラス名: **UISelection**

(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)

要求される名前空間:

この実習に必要とされる名前空間は次のとおりです:

- Autodesk.Revit.DB
- Autodesk.Revit.UI
- Autodesk.Revit.DB
- Autodesk.Revit.UI
- Autodesk.Revit.ApplicationServices
- Autodesk.Revit.Attributes
- Autodesk.Revit.UI.Selection(選択処理用)
-

注意: (VB.NET のみ):VB.NET でコードを記述している場合、プロジェクト・レベルで名前空間をインポートします(つまり、プロジェクトプロパティでは、各ファイル内で明示的にインポートする必要はありません)。

クラス内で UIApplication とアクティブな UIDocument を参照する変数を宣言します。

<VB.NET>

```
<Transaction(TransactionMode.ReadOnly)> _  
Public Class UISelection  
    Implements IExternalCommand  
  
    ' Member variables  
    Dim _uiApp As UIApplication  
    Dim _uiDoc As UIDocument
```

```

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.
    ' (we may not use them all in this specific lab.)
    _uiApp = commandData.Application
    _uiDoc = _uiApp.ActiveUIDocument

    ' ...
    Return Result.Succeeded

End Function

End Class
</VB.NET>

```

2. 現在の選択内容を示す

現在選択されているオブジェクトの内容に関する情報を示すために、いくつかのヘルパー関数を作成しましょう。これらのヘルパー関数は、他の関数から呼び出して使用します。

```

<VB.NET>
Sub ShowElementList(ByVal elems As IEnumerable, ByVal header As String)

    Dim s As String = vbCr + vbCr + _
        " - Class - Category - Name (or Family: Type Name) - Id - " + vbCr

    Dim count As Integer = 0
    For Each e As Element In elems
        count += 1
        s += ElementToString(e)
    Next

    s = header + "(" + count.ToString() + ")" + s

    TaskDialog.Show("Revit UI Lab", s)

End Sub

Function ElementToString(ByVal e As Element) As String

    If e Is Nothing Then
        Return "none"
    End If

    Dim name As String = ""

    If TypeOf e Is ElementType Then

```

```

    Dim param As Parameter = _
        e.Parameter(BuiltInParameter.SYMBOL_FAMILY_AND_TYPE_NAMES_PARAM)
    If param IsNot Nothing Then
        name = param.AsString
    End If
Else
    name = e.Name
End If

Return e.GetType.Name + "; " + e.Category.Name + "; " _
    + name + "; " + e.Id.IntegerValue.ToString + vbCrLf

End Function

Public Shared Function PointToString(ByVal pt As XYZ) As String

    If pt Is Nothing Then
        Return ""
    End If

    Return "(" + pt.X.ToString("F2") + ", " + pt.Y.ToString("F2") + ", " _
        + pt.Z.ToString("F2") + ")"

End Function
</VB.NET>

```

UIDocument.Selection.GetElementIds() から、現在選択された要素 ID のリストを取得できます。(注意: Revit 2015 から UIDocument.Selection.Elements は非推奨です)

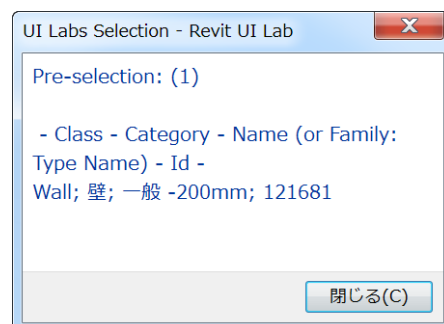
```

<VB.NET>
    Dim selSet As ICollection(Of ElementId) = _uiDoc.Selection.GetElementIds(
    )

    ShowElementList(selSet, "Pre-selection: ")
</VB.NET>

```

3. オブジェクトを選択する



様々な選択関数は、すべて UIDocument.Selection 下で見つけることができます。まず、ユーザに単一のオブジェクトの選択を促す PickObject() を最初に見ていきます。この関数には、多くのオーバーロードが存在します。選択を許可するオブジェクトのタイプを指定することも出来ます。今回は、選択対象を要素として、ユーザにプロンプトを提供します。さらに、選択フィルタ クラスを指定することもできます(後述)。

<VB.NET>

```
Sub PickMethod_PickObject()  
  
    Dim r As Reference = _uiDoc.Selection.PickObject(ObjectType.Element, _  
        "Select one element")  
    Dim e As Element = _uiDoc.Document.GetElement(r)  
    ShowBasicElementInfo(e) 'Defined in DBElelemt Lab in the Intro Labs  
  
End Sub
```

</VB.NET>

4. オブジェクト群を選択する

複数のオブジェクトを選択するようにユーザに促したい場合には、選択したオブジェクトの参照のリストを返す PickObjects() を呼び出してください。先に作成した、選択要素に関する情報を示すユーティリティ関数を使用するために、参照のリストを要素のリストに変換する必要があります。

<VB.NET>

```
Sub PickMethod_PickObjects()  
  
    Dim refs As IList(Of Reference) = _  
        _uiDoc.Selection.PickObjects(ObjectType.Element, _  
            "Select multiple elemens")  
  
    ' put it in a List form.  
    Dim elems As IList(Of Element) = New List(Of Element)  
    For Each r As Reference In refs  
        elems.Add(_uiDoc.Document.GetElement(r))  
    Next  
    ShowElementList(elems, "Pick Objects: ")  
  
End Sub
```

</VB.NET>

さらに、矩形内にある要素を選択するようにユーザに促すことも出来ます。

<VB.NET>

```
Sub PickMethod_PickElementByRectangle()  
  
    ' Note: PickElementByRectangle returns the list of element. not reference.  
    Dim elems As IList(Of Element) = _  
        _uiDoc.Selection.PickElementsByRectangle("Select by rectangle")  
  
    ' Show it
```

```
ShowElementList (elems, "Pick By Rectangle: ")  
  
End Sub  
</VB.NET>
```

5. 点を選択する

空間上の点(座標)を選択したい場合には、PickPoint() を使用することが出来ます。

```
<VB.NET>  
Sub PickMethod_PickPoint()  
  
    Dim pt As XYZ = _uiDoc.Selection.PickPoint("Pick a point")  
  
    Dim msg As String = "Pick Point: "  
    msg += PointToString(pt)  
    TaskDialog.Show("PickPoint", msg)  
  
End Sub  
</VB.NET>
```

要素上の点(座標)を選択したければ、PickObject() に ObjectType.PointOnElement を渡します。

```
<VB.NET>  
Sub PickPointOnElement()  
  
    Dim r As Reference = _  
        _uiDoc.Selection.PickObject( _  
            ObjectType.PointOnElement, _  
            "Select a point on element")  
  
    Dim e As Element = _uiDoc.Document.GetElement(r)  
    Dim pt As XYZ = r.GlobalPoint  
  
    Dim msg As String  
    If pt Is Nothing Then  
        msg = "No point picked."  
    Else  
        msg = "You picked the point " + PointToString(pt) _  
            + " on element " + e.Id.ToString  
    End If  
    TaskDialog.Show("PickPointOnElement", msg)  
  
End Sub  
</VB.NET>
```

6. 面/エッジを選択する

PickObject() に、ObjectType.Face または ObjectType.Edge をそれぞれ渡すことで、要素の面やエッジを選択することができます。これによって、要素の参照が返されます。選択した要素の面やエッジを得るためには、GetGeometryObjectFromReference() を使用します。

```
<VB.NET>
Sub PickFace()

    Dim r As Reference = _
        _uiDoc.Selection.PickObject(ObjectType.Face, "Select a face")
    Dim e As Element = _uiDoc.Document.GetElement(r)

    Dim oFace As Face = e.GetGeometryObjectFromReference(r)

    ' show a message to the user.

    Dim msg As String = ""
    If oFace IsNot Nothing Then
        msg = "You picked the face of element " + e.Id.ToString + vbCr
    Else
        msg = "no Face picked" + vbCr
    End If
    TaskDialog.Show("PickFace", msg)

End Sub
</VB.NET>
```

エッジを選択するために、同様の関数を実装してみてください。

7. 選択フィルタ

PickObject()/PickObjects() を使用してオブジェクトを選択した場合、希望する方法で選択をフィルタリングするために、ISelectionFilter を実装するクラスのインスタンスを指定することもできます。例えば、ユーザが壁だけを選択するように、選択操作をフィルタリングできます。

最初に、ISelectionFilter インターフェースを実装するクラスを作成します。このインターフェースは、実装可能な2つのメソッド、AllowReference () と AllowElement() があります。ObjectType.Element を PickObject() に指定すれば、AllowElement() だけが呼び出されるようになります。もし、ObjectType.Face/Edge/PointOnElement を指定すれば、AllowReference() が呼び出されます。

```
<VB.NET>
Class SelectionFilterWall
    Implements ISelectionFilter

    Public Function AllowElement(ByVal e As Element)
        As Boolean Implements ISelectionFilter.AllowElement

        If e.Category Is Nothing Then Return False
        If e.Category.Id.IntegerValue.Equals(BuiltInCategory.OST_Walls) Then _
            Return True
        Return False
    End Function
End Class
```

```

End Function

Public Function AllowReference( _
    ByVal reference As Reference, _
    ByVal position As XYZ) _
    As Boolean Implements ISelectionFilter.AllowReference

    Return True

End Function

End Class
</VB.NET>

```

このクラスを使用して、選択をフィルタリングしてみましょう:

```

<VB.NET>
Sub PickWall()

    Dim selFilterWall As New SelectionFilterWall
    Dim r As Reference = _
        _uiDoc.Selection.PickObject(ObjectType.Element, selFilterWall, _
            "Select a wall")

    ' Show it
    Dim e As Element = _uiDoc.Document.GetElement(r)
    ShowBasicElementInfo(e) 'Defined in DBElelemt Lab in the Intro Labs

End Sub
</VB.NET>

```

面オブジェクトを選択する場合には、PlanarFace で選択をフィルタリングしてみてください。

8. 対話的に家を作成する

入門実習には、家を作成する関数をエクスポートする部分が含まれています。入門実習を開いてビルドして、現在のプロジェクトに、作成されたアドインの実行ファイル dll への参照を追加してください。参照される入門実習 dll の ModelCreationExport クラスの下には、対話的に家を作る手助けとなる様々な関数があります。それらの関数は、次のとおりです:

- CreateWalls()
- AddDoor()
- AddWindow()
- AddRoof()

下記の処理を実行するプログラムを作成してください:

- 新しい外部コマンドを 2_Selection.vb ファイルに加える

- コマンドプロンプト内で家の 2 つの対角点を選択して、それらの点に基づく 4 つの壁を作成する
- 壁のうちの 1 つを選択するようにユーザに促して、そこにドアを追加し、窓を他の 3 つの壁の各々に加える
- 家の屋根を作成する

9. サマリ

この実習では、様々な選択機能を使用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。

- オブジェクト/オブジェクト/ポイントを選択する
- 要素ジオメトリを選択
- 選択を制限する