



Revit API & BIM セミナー 2018

# Revit データの理解

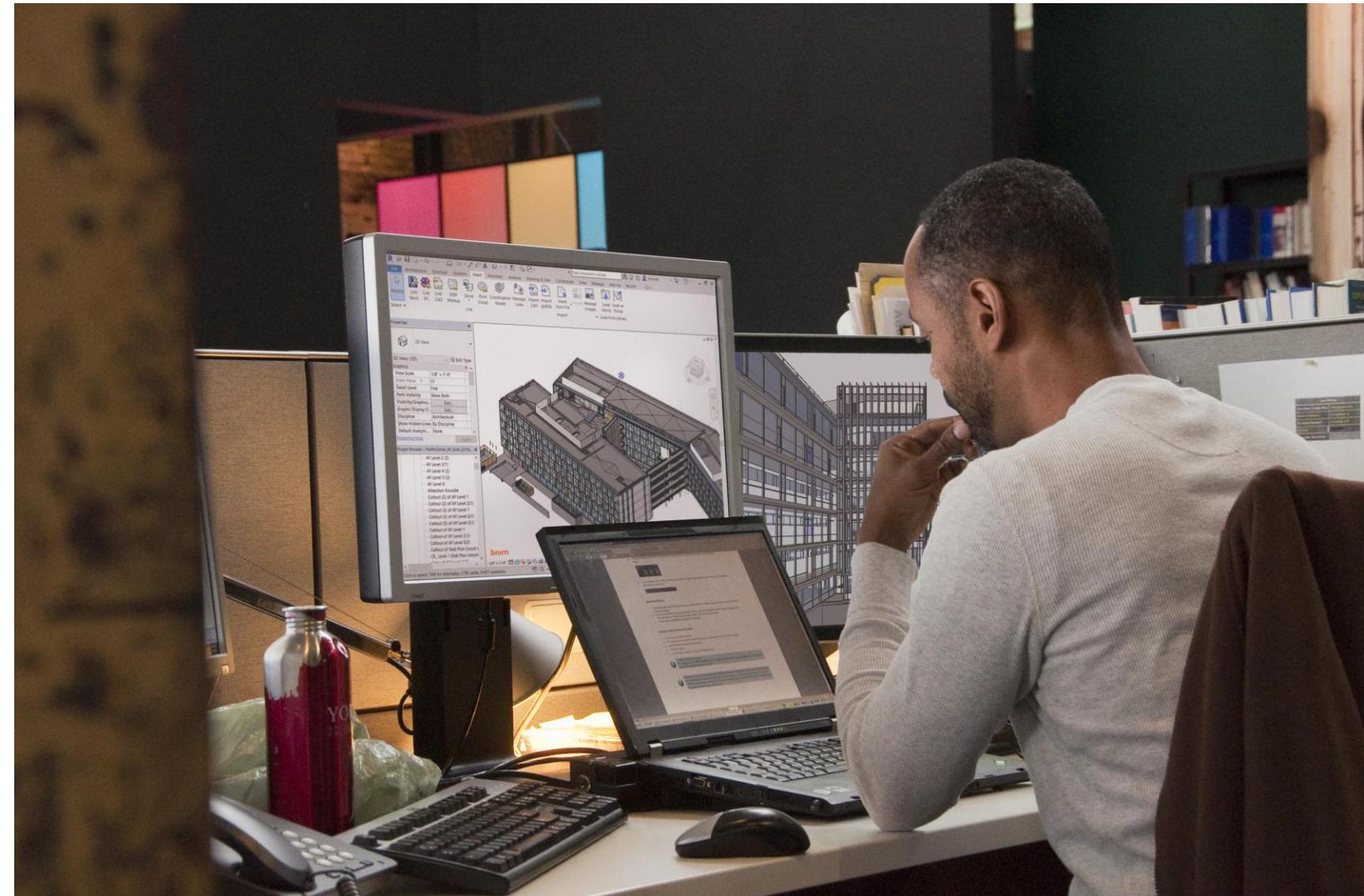
小笠原 龍司

Autodesk Developer Network, Forge Partner Development



# アジェンダ

- Revit API の構成
- Revit DB 要素
- 要素の基本クラス
- 要素のフィルタリング
- 要素の編集
- 要素の作成
- Revit API の学習コンテンツ



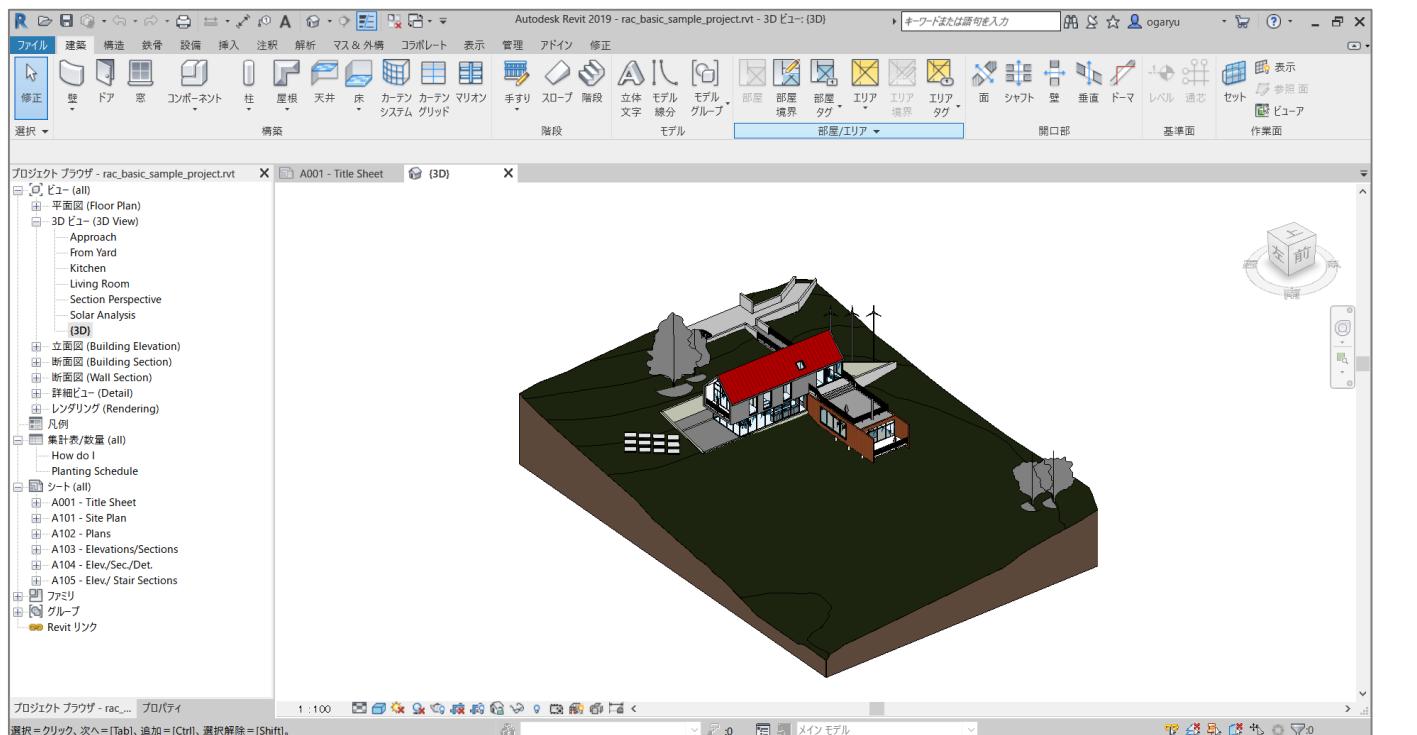
# Revit API の構成



# UI と DB の分離

UI

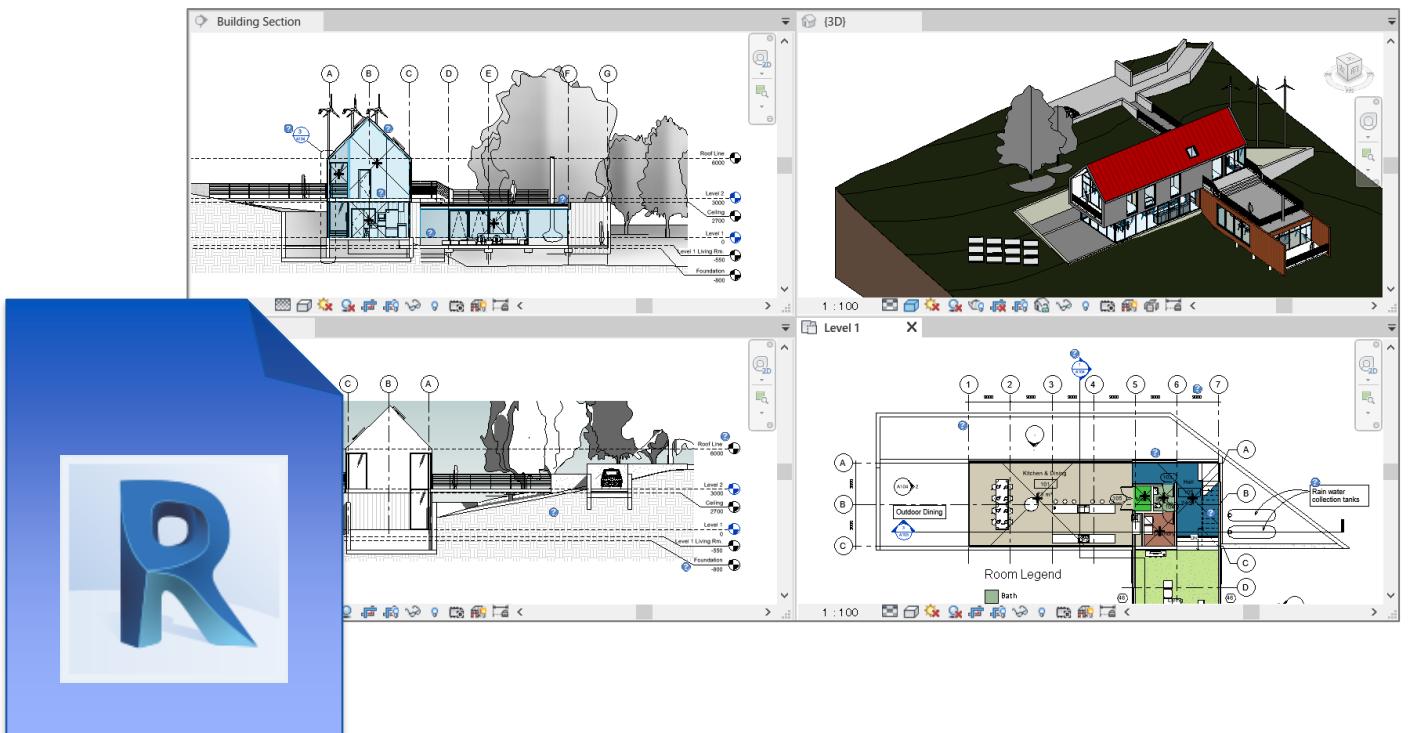
Autodesk.Revit.UI 名前空間



Revit のユーザインターフェースにアクセスするための API

DB

Autodesk.Revit.DB 名前空間



Revit プロジェクトのデータにアクセスするための API

# アプリケーションとドキュメント オブジェクトの位置づけ

UI

UI アプリケーション オブジェクト

Autodesk.Revit.UI.UIApplication

(Autodesk.Revit.UI.UIControlledApplication)

UI ドキュメント オブジェクト

Autodesk.Revit.UI.UIDocument

DB

アプリケーション オブジェクト

ApplicationServices.Application

(ApplicationServices.ControlledApplication)

ドキュメント オブジェクト

Autodesk.Revit.DB.Document

# アプリケーションとドキュメント オブジェクトの役割

# UI

Autodesk.Revit.UI.UIApplication

アプリケーションの UI レベルのインターフェースを提供

- リボンパネルの追加
- アクティブなドキュメントを取得
- メインウィンドウのハンドルを取得
- アプリケーションレベルのイベントにアクセス

# DB

Autodesk.Revit.ApplicationServices.Application

アプリケーションの DB レベルのプロパティにアクセス

- ファミリドキュメントの新規作成
- プロジェクトの新規作成
- ドキュメントの管理・イベントにアクセス
- セッション情報・ログイン情報の取得

Autodesk.Revit.UI.UIDocument

ドキュメント操作に関連するインターフェースを提供

- 選択中の要素を取得
- ユーザーに選択を要求するプロンプト
- ユーザーに要素を配置するプロンプト

Autodesk.Revit.DB.Document

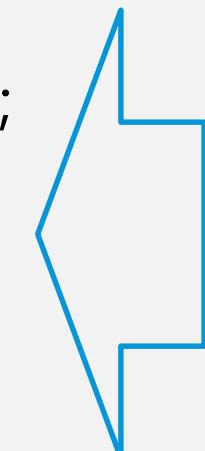
ドキュメントの DB レベルのプロパティにアクセス

- ファイルをインポート/エクスポート
- ファミリをロード
- ビューの管理
- 要素を取得

```
namespace RevitAddin
{
    [Transaction(TransactionMode.Manual)]
    public class Command : IExternalCommand
    {
        public Result Execute(
            ExternalCommandData commandData,
            ref string message,
            ElementSet elements)
        {
            UIApplication uiapp = commandData.Application;
            UIDocument uidoc = uiapp.ActiveUIDocument;
            Application app = uiapp.Application;
            Document doc = uidoc.Document;

            // 省略

            return Result.Succeeded;
        }
    }
}
```

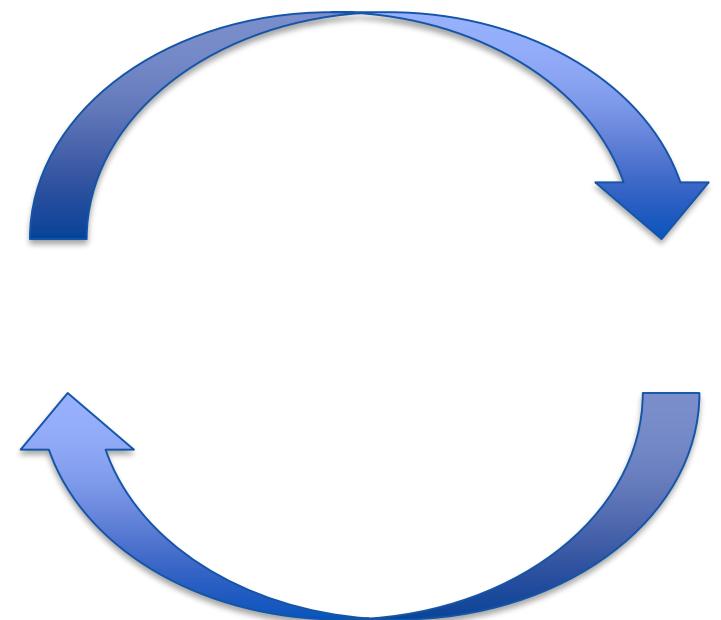
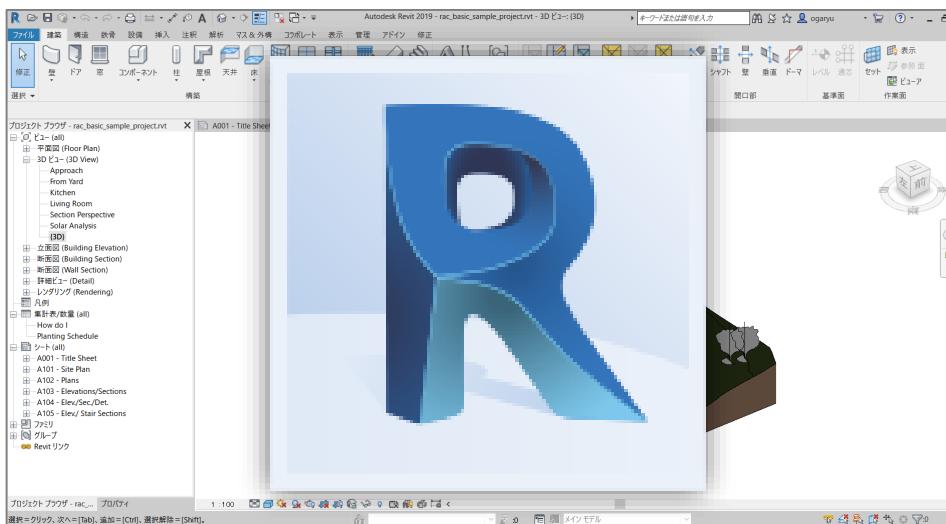


Execute() メソッドの引数  
ExternalCommandData オブジェクト  
を起点に、アプリケーションとドキュメン  
トオブジェクトを取得

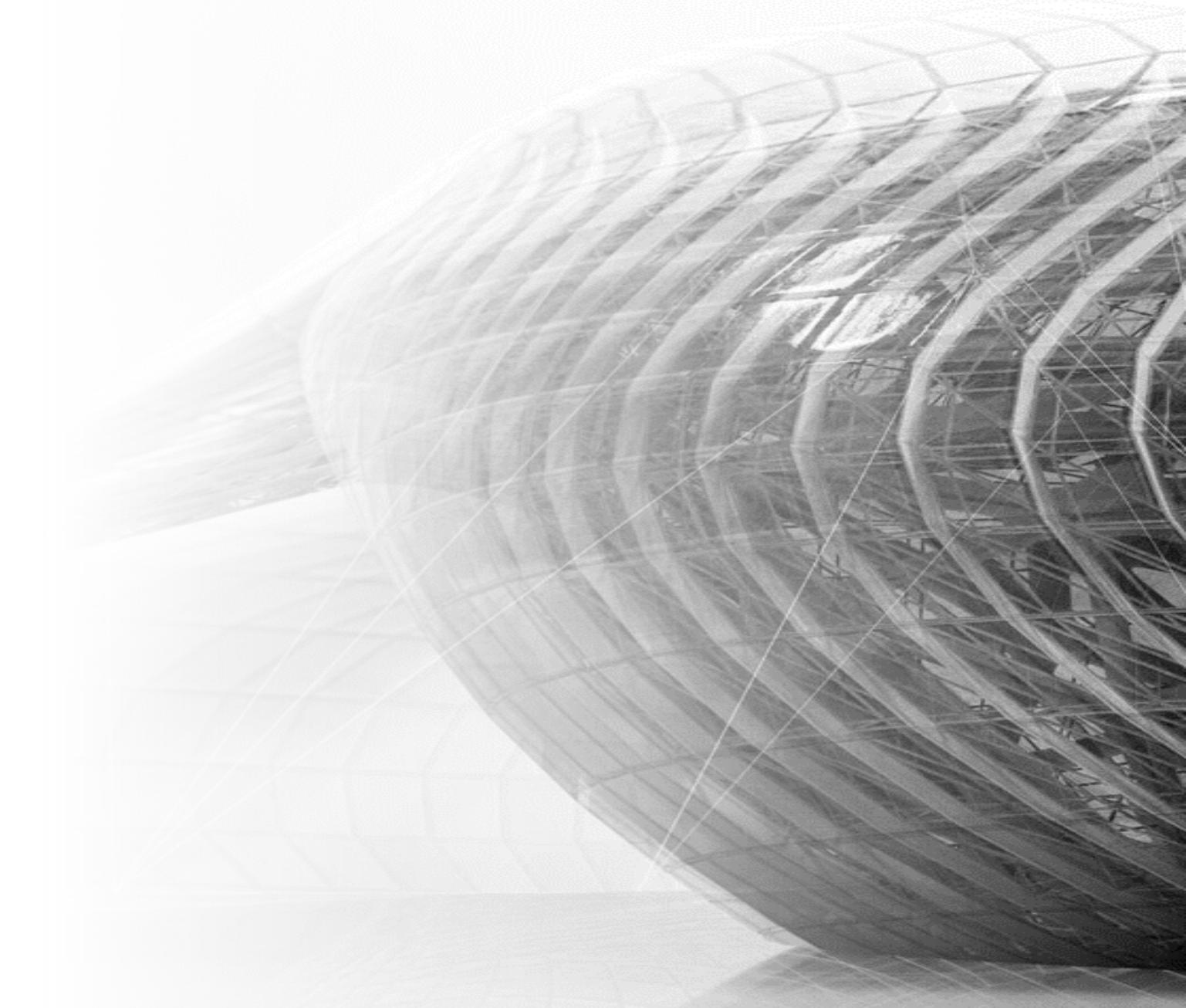
# UI レベルの API と DB レベルの API を使いこなす

UI

DB

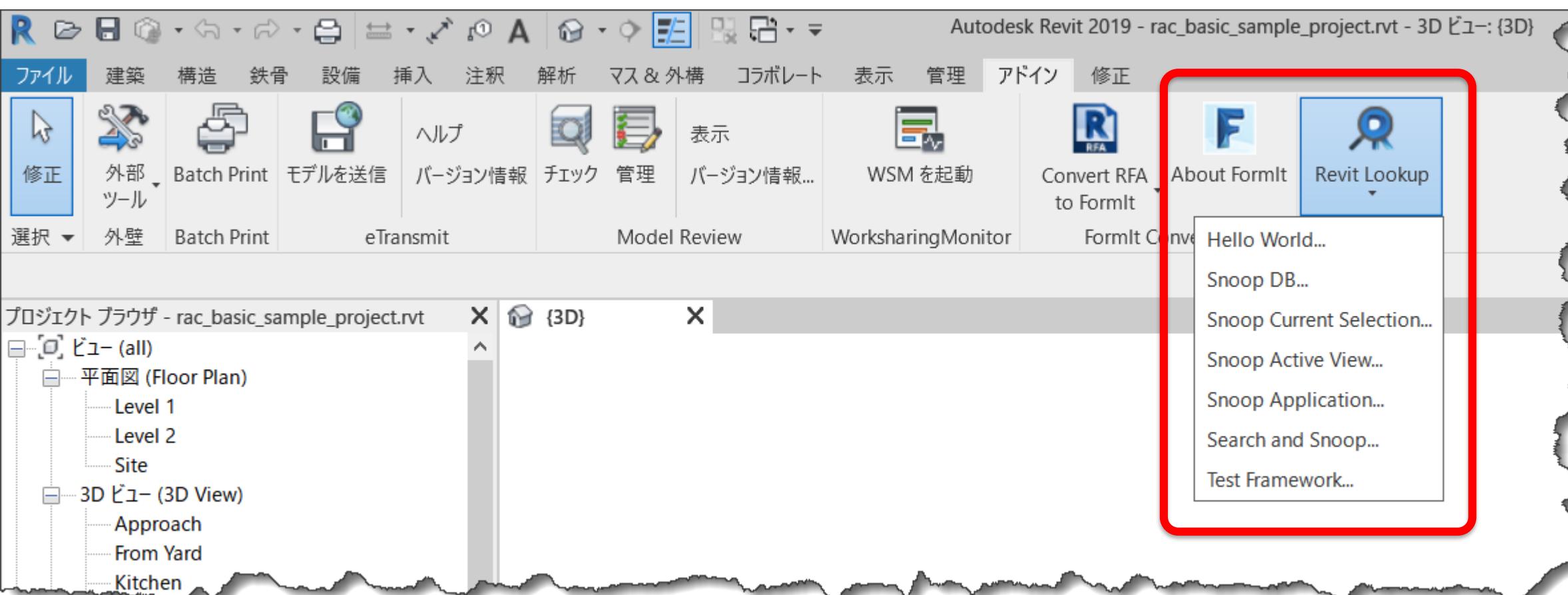


# Revit DB 要素



# DB 要素のデータ構造を調べるツール – Revit LookUp ツール

- RevitLookup ツールは、Revit の UI 上から、アプリケーション、ドキュメント、ファミリ、ジオメトリ等の、各オブジェクトのデータベース構造を確認するための無償のアドインです。
- <https://github.com/jeremytammik/RevitLookup>



- Snoop DB
- Snoop Current Selection
- Snoop Active View
- Snoop Application

## Snoop Objects

Application < Application >

Field	Value
--- Application ---	
--- Properties ---	
ActiveAddInId	< AddInId >
AllowNavigationDuringRedraw	True
AllUsersAddinsLocation	C:\ProgramData\Autodesk\Revit\Addins\2019
AngleTolerance	0.0017453295199433
Assets	パラメーター カウントが一致しません。
BackgroundColor	< Color >
Cities	< CitySet >
Create	< Application >
CurrentRevitServerAccelerator	
CurrentUserAddinsLocation	C:\Users\yogasawr\AppData\Roaming\Autodesk\Revit\Addins\2019
CurrentUsersAddinsDataFolder	C:\Users\yogasawr\AppData\Roaming\Autodesk\Revit\Autodesk Revit 2019\Addins\2019
CurrentUsersDataFolderPath	C:\Users\yogasawr\AppData\Roaming\Autodesk\Revit\Autodesk Revit 2019
DefaultIFCProjectTemplate	
DefaultProjectTemplate	C:\ProgramData\Autodesk\RVT 2019\Templates\Japan\Construction-Default.rvt
DefaultViewDiscipline	Structural
Documents	< DocumentSet >
ExportIFCAsTable	C:\ProgramData\Autodesk\RVT 2019\exportlayers-ifc-TAT.txt
FamilyTemplatePath	C:\ProgramData\Autodesk\RVT 2019\
ImportIFCCategoryTable	
IsArchitectureEnabled	True
IsElectricalAnalysisEnabled	True
IsElectricalEnabled	True
IsEnergyAnalysisEnabled	True

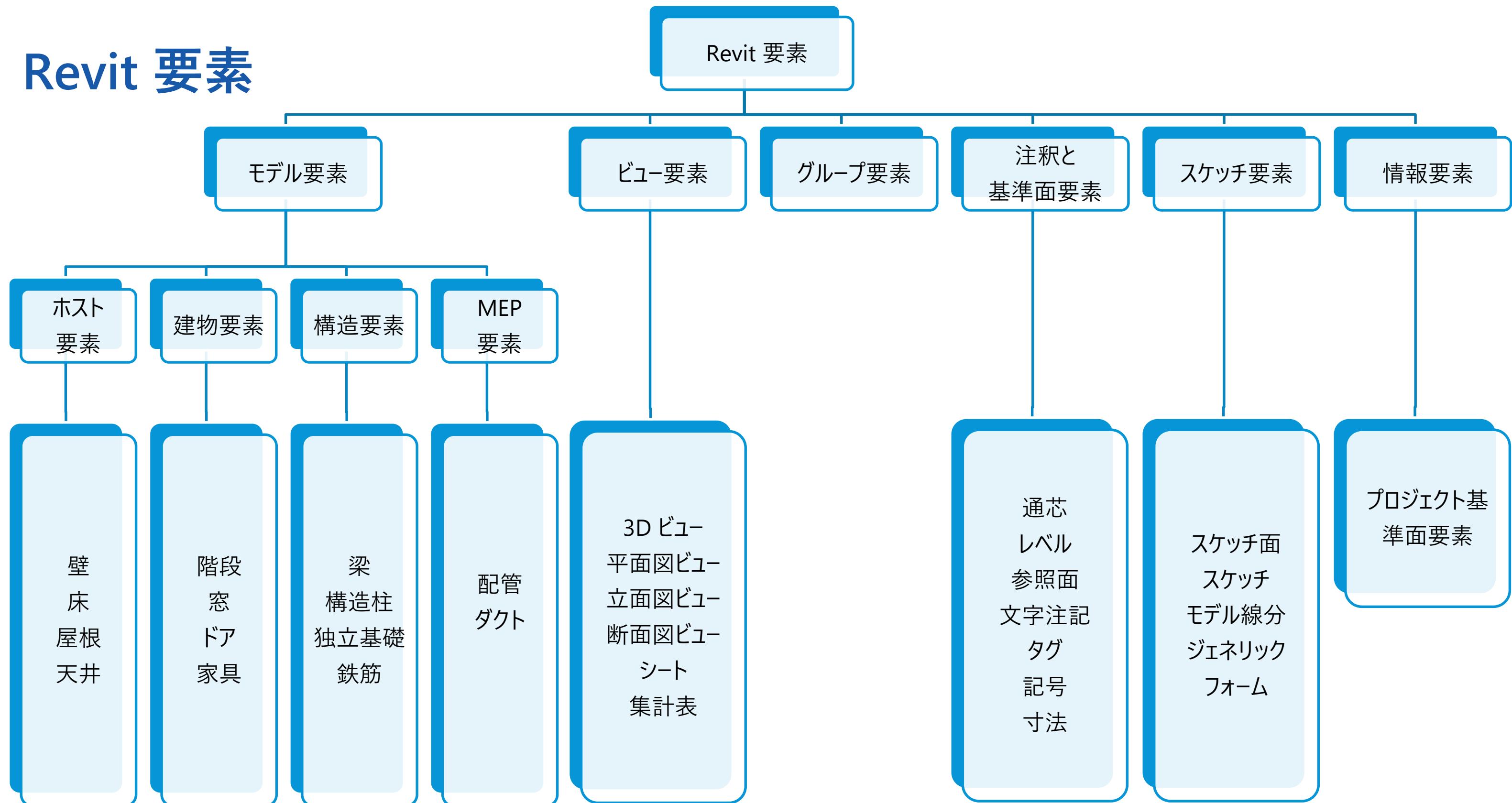
OK

Field	Value
--- Application ---	
--- Properties ---	
ActiveAddInId	< AddInId >
AllowNavigationDuringRedraw	True
AllUsersAddinsLocation	C:\ProgramData\Autodesk\Revit\Addins\2019
AngleTolerance	0.0017453295199433
Assets	パラメーター カウントが一致しません。
BackgroundColor	< Color >
Cities	< CitySet >
Create	< Application >
CurrentRevitServerAccelerator	
CurrentUserAddinsLocation	C:\Users\yogasawr\AppData\Roaming\Autodesk\Revit\Addins\2019
CurrentUsersAddinsDataFolder	C:\Users\yogasawr\AppData\Roaming\Autodesk\Revit\Autodesk Revit 2019\Addins\2019
CurrentUsersDataFolderPath	C:\Users\yogasawr\AppData\Roaming\Autodesk\Revit\Autodesk Revit 2019
DefaultIFCProjectTemplate	
DefaultProjectTemplate	C:\ProgramData\Autodesk\RVT 2019\Templates\Japan\Construction-Default.rvt
DefaultViewDiscipline	Structural
Documents	< DocumentSet >
ExportIFCAsTable	C:\ProgramData\Autodesk\RVT 2019\exportlayers-ifc-TAT.txt
FamilyTemplatePath	C:\ProgramData\Autodesk\RVT 2019\
ImportIFCCategoryTable	
IsArchitectureEnabled	True
IsElectricalAnalysisEnabled	True
IsElectricalEnabled	True
IsEnergyAnalysisEnabled	True

OK

Field	Value
--- Document ---	
--- Properties ---	
ActiveProjectLocation	< ProjectLocation Internal 21748 >
ActiveView	< View3D {3D} 960621 >
Application	< Application >
Create	< Document >
DisplayUnitSystem	METRIC
FamilyCreate	呼び出しのターゲットが例外をスローしました。
FamilyManager	呼び出しのターゲットが例外をスローしました。
IsDetached	False
IsFamilyDocument	False
IsLinked	False
IsModifiable	False
IsModified	False
IsReadOnly	False
IsReadOnlyFile	True
IsValidObject	True
IsWorkshared	False
MassDisplayTemporaryOverrides	ShowMassByViewSettings
MullionTypes	< MullionTypeSet >
OwnerFamily	< null >
PanelTypes	< PanelTypeSet >
ParameterBindings	< BindingMap >
PathName	C:\Program Files\Autodesk\Revit 2019\Samples\rac_basic_sample_project.rvt
Phases	< PhaseArray >
PlanTopologies	パラメーター カウントが一致しません。

# Revit 要素



# Revit ファミリ

- Revit プロジェクトに追加するほとんどの要素は、ファミリを使用して作成されます。
- パラメータと呼ばれる共通のプロパティセットおよび関連するグラフィックス表現を持つ要素のグループです。

ファミリの種類	定義	API 対応
システム ファミリ	Revit にあらかじめ組み込まれている 壁、屋根、床などのホスト要素。 また、レベル、通芯、シート、およびビューポートなどの要素も含まれます。	○
コンポーネント ファミリ	モデルとは独立して作成され、必要に応じてモデルにロードされます。 <ul style="list-style-type: none"><li>ドアや器具などの建物コンポーネントや、注釈要素。</li><li>通常は、システム ファミリによってホストされます。たとえば、ドアや窓は壁によってホストされます。</li></ul>	○
インプレイス ファミリ	モデルのコンテキスト内で作成するカスタム要素です。 再利用を想定しない独自のジオメトリの場合は、インプレイス ファミリを作成します。	×

# ファミリ タイプとファミリ インスタンス

- ファミリは、1つ以上の複数のファミリ タイプを保持します。
- ファミリ タイプとは、ある特定のファミリのバリエーションです。
- ファミリ インスタンスとは、プロジェクトの中のモデル空間やビュー、シート上で、任意の位置に配置された、1つの要素（インスタンス）です。1つのファミリ タイプを雛型とします。

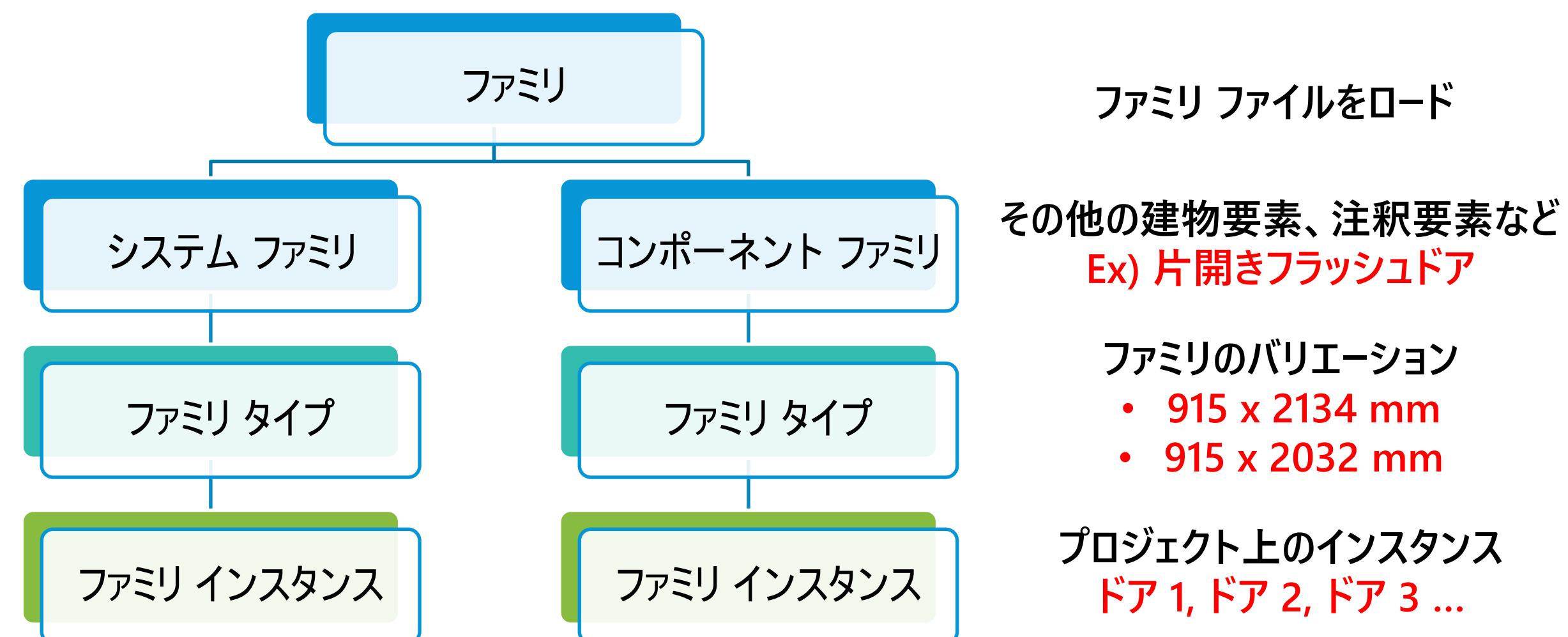
Revit に組み込まれている

壁、屋根、床などのホスト要素  
Ex) 標準壁

ファミリのバリエーション

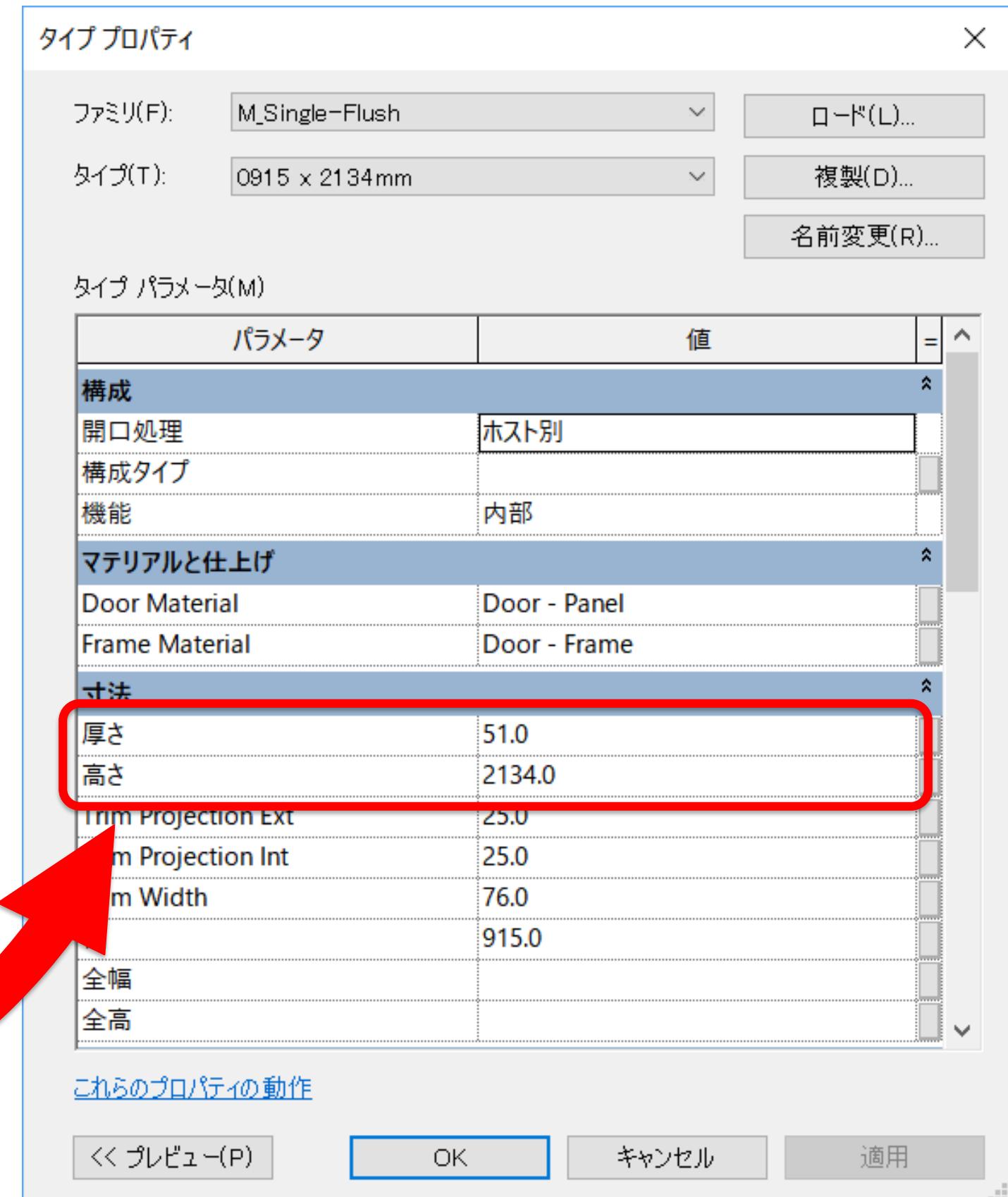
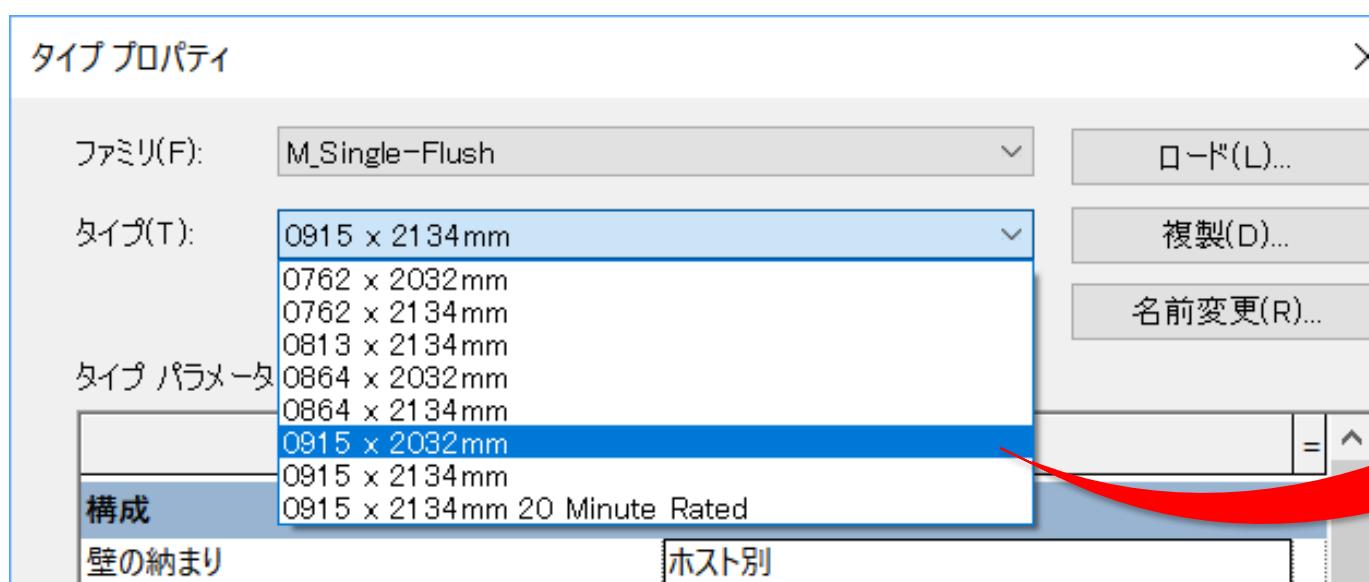
- Generic 140mm
- Generic 200mm

プロジェクト上のインスタンス  
標準壁 1, 標準壁 2 ...



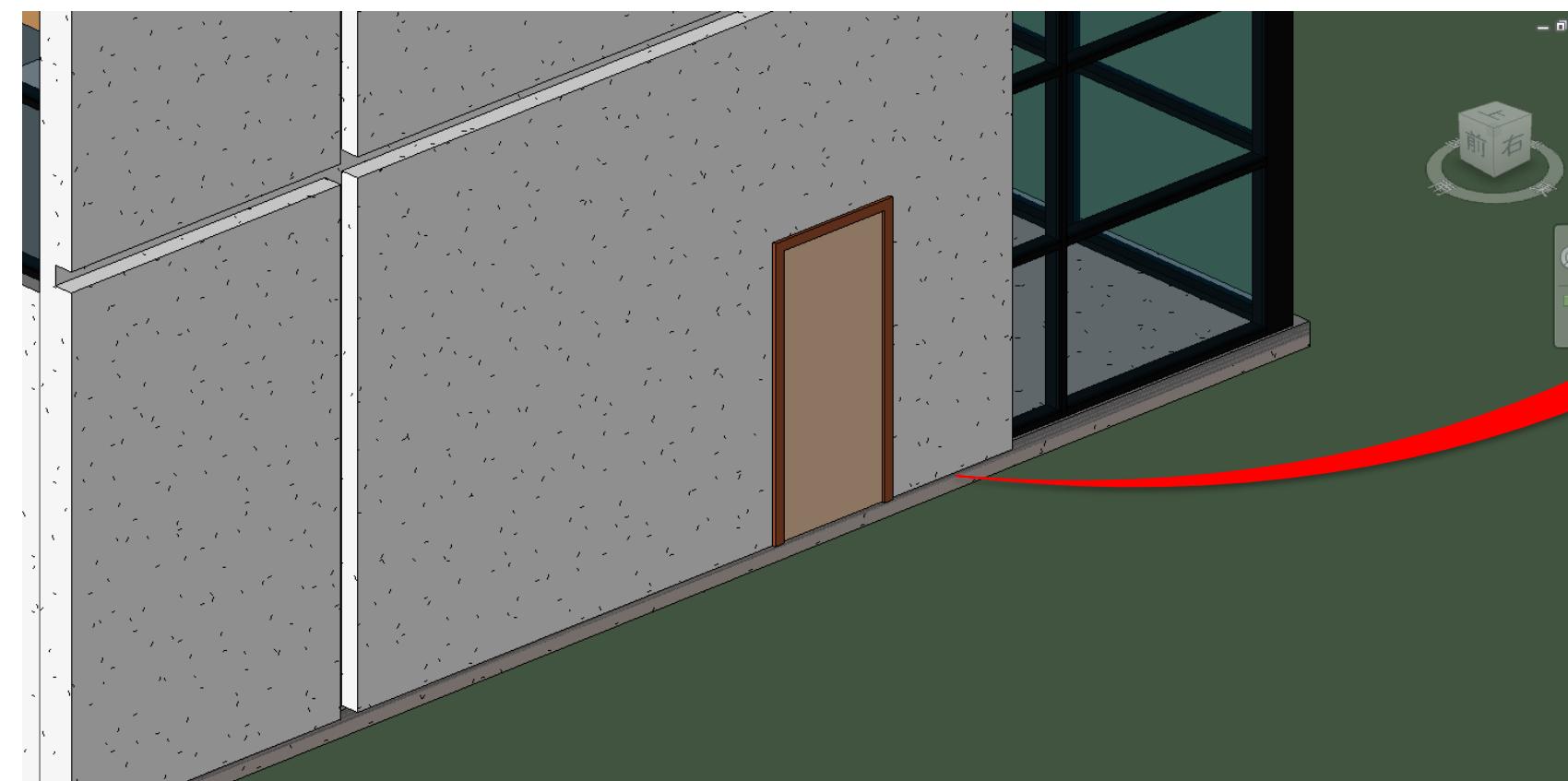
# タイプ プロパティ

- タイプ プロパティは、ファミリ内のすべての要素に共通。
- 各プロパティの値は、特定のファミリ タイプのすべてのインスタンスで同じ値を持ちます。
- タイプ プロパティの値を変更すると、その値は既に配置されている、あるいはこれから配置されるそのファミリ タイプのすべてのインスタンスに反映されます。



# インスタンス プロパティ

- インスタンス プロパティの属性項目は、特定のファミリ タイプに属するすべての要素で共通。
- ただし、プロパティの値は、建物またはプロジェクト内の要素の位置によって異なります。



Properties

M\_Single-Flush  
0915 x 2134mm

Door (1) Type edit

Constraint

Level: 01 - Entry Level (highlighted with a red box)  
Bottom height: 0.0

Frame type

Material and finish

Frame material

Finish

Category information

Hardware group: (none)

Image

Comment

Mark: 132B

Phase

Built phase: New Construction

Demolished phase: none

Others

Top height: 2134.0

Hinge Set

Lock Function

Egress Hardware: checked

Closer: checked

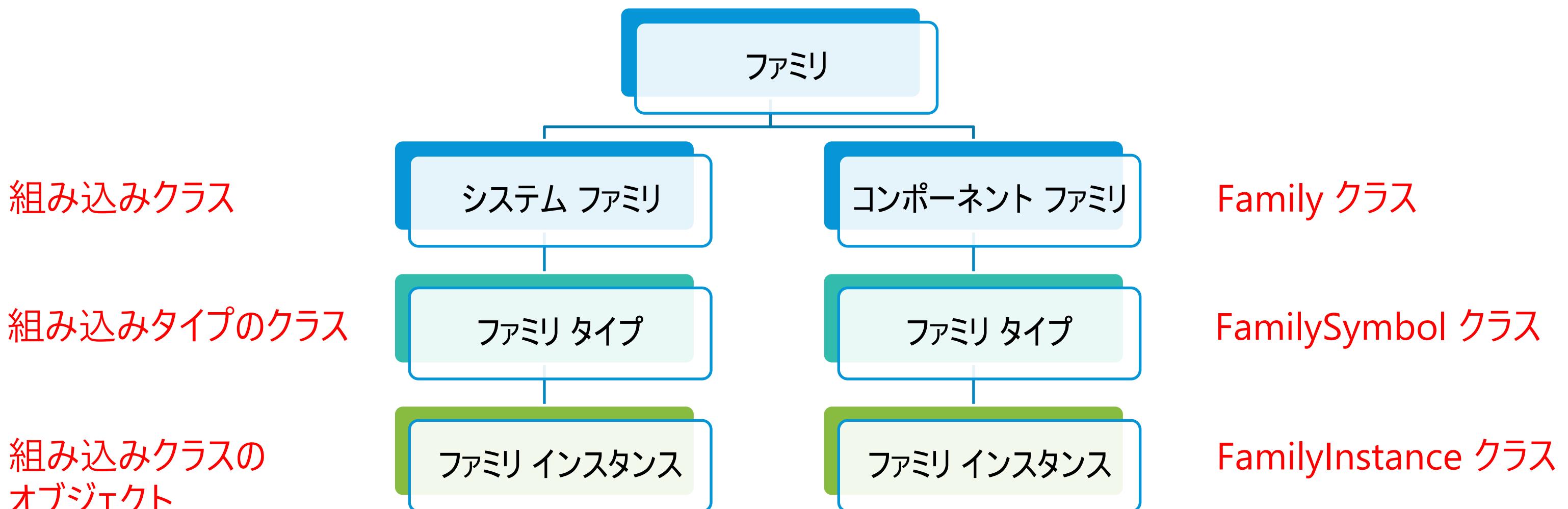
Properties Help

Apply

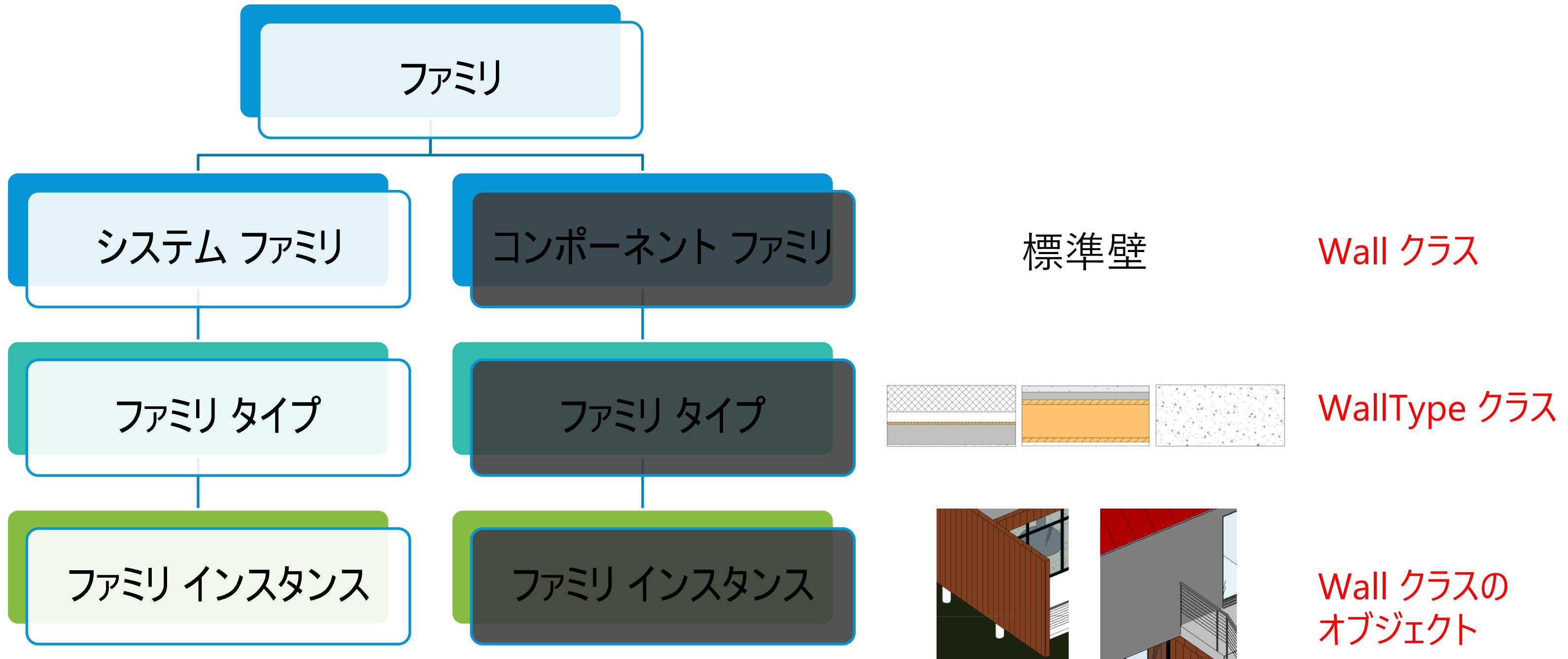
Project Browser - rac\_advanced\_sample\_project... Properties

# API との対応関係

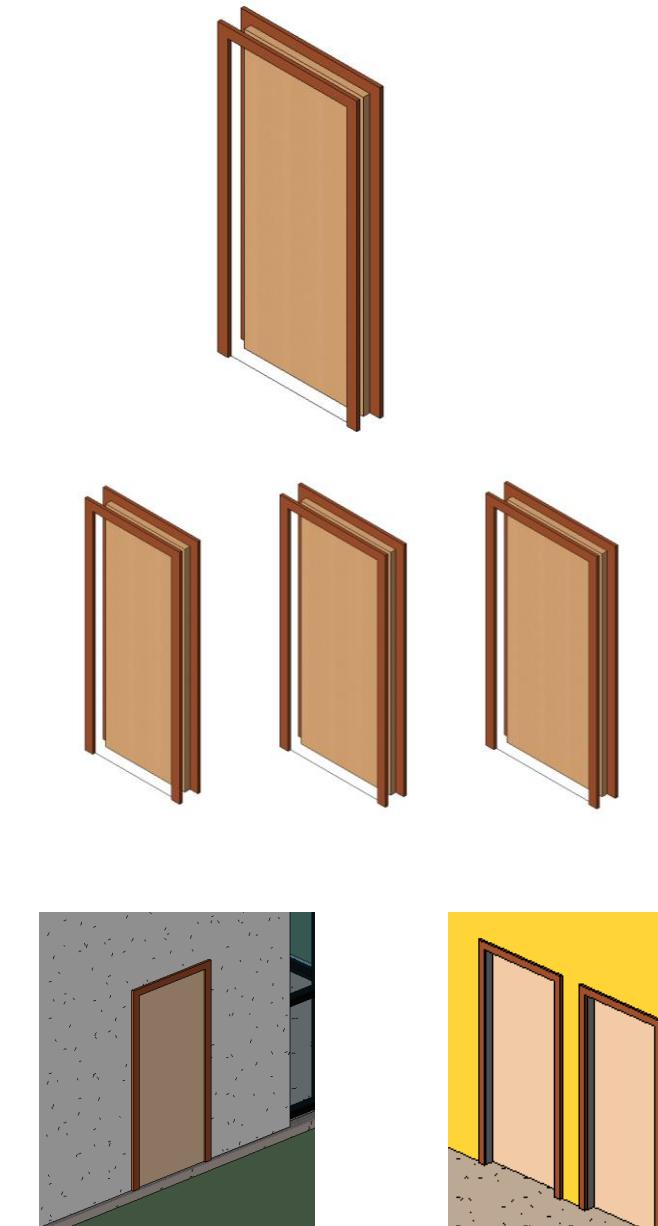
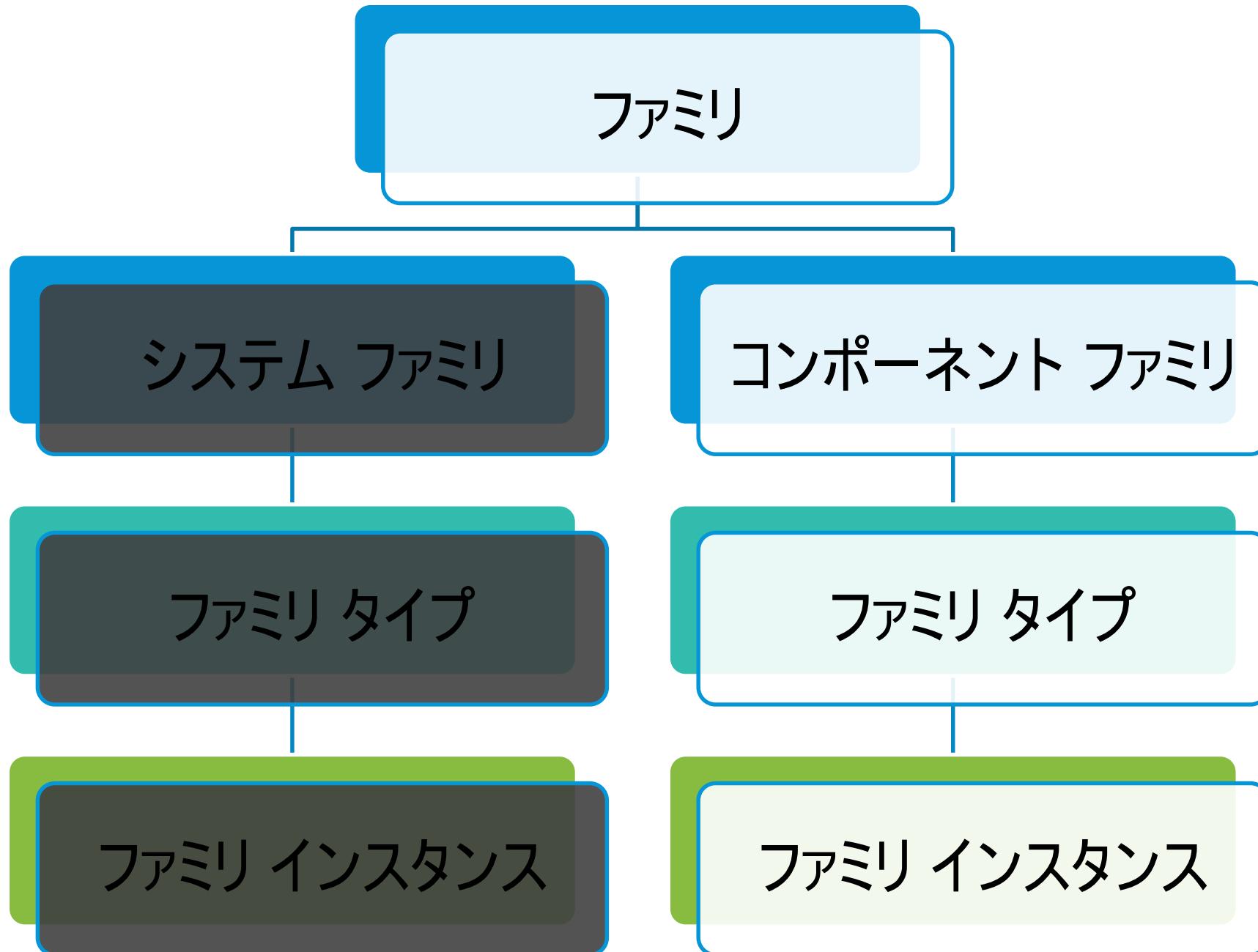
- システム ファミリは、各オブジェクト用のクラスが用意されています。
  - Wall, Roof, Ceiling, Floor, Grid, Level, Dimension...
- コンポーネント ファミリは、Family クラス、FamilySymbol クラス、FamilyInstance クラスで表されます。



# 壁 ファミリの場合



# ドア ファミリの場合



Family クラス

FamilySymbol クラス

FamilyInstance クラス

## 要素

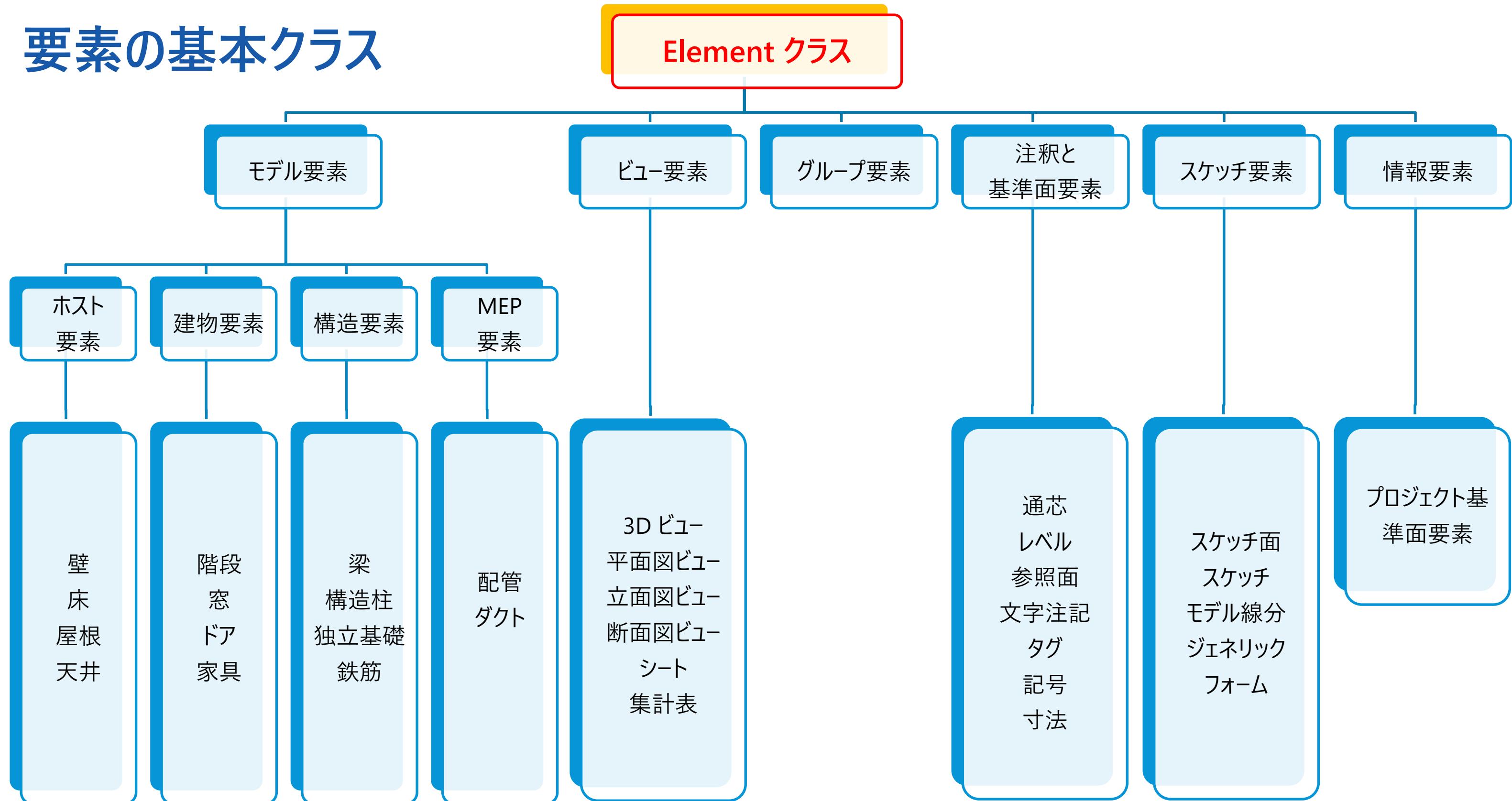
## シンボル

UI の要素種別	要素/タイプ別から派生	カテゴリ	要素/タイプ別から派生	カテゴリ
壁	HostObject/Wall	壁	HostObjAttributes/WallType	壁
ドア	Instance/InsertableInstance/ <a href="#">FamilyInstance</a>	ドア	InsertableObject / <a href="#">FamilySymbol</a>	ドア
ドアタグ	IndependentTag	ドアタグ	InsertableObject / <a href="#">FamilySymbol</a>	ドアタグ
窓	Instance/InsertableInstance/ <a href="#">FamilyInstance</a>	窓	InsertableObject / <a href="#">FamilySymbol</a>	窓
窓タグ	IndependentTag	窓タグ	InsertableObject / <a href="#">FamilySymbol</a>	窓タグ
開口部	Opening	開口部	< null >	---
床	HostObject/CeilingAndFloor /Floor	床	HostObjAttributes/FloorType	床
天井	HostObject/CeilingAndFloor /Ceiling	天井	HostObjAttributes/CeilingType	天井
屋根	HostObject/RoofBase/FootPrintRoof,ExtrusionRo of	屋根	HostObjAttributes/RoofType	屋根
柱	Instance/InsertableInstance/ <a href="#">FamilyInstance</a>	柱	InsertableObject / <a href="#">FamilySymbol</a>	柱
机	Instance/InsertableInstance/ <a href="#">FamilyInstance</a>	家具	InsertableObject / <a href="#">FamilySymbol</a>	家具
木	Instance/InsertableInstance/ <a href="#">FamilyInstance</a>	植栽	InsertableObject / <a href="#">FamilySymbol</a>	植栽
階段	Stairs	階段	< Symbol >	階段
手すり	Railing	手すり	< Symbol >	手すり
部屋	Room	部屋	< null >	---
部屋タグ	RoomTag	部屋タグ	< Symbol >	部屋タグ
グリッド	Grid	グリッド	LineAndTextAttrSymbol/GridType	< null >
モデル線分	ModelCurve/ModelLine	線分	< null >	---
参照面	ReferencePlane	参照面	< null >	---
寸法	Dimension	寸法	DimensionType	< null >
断面	< Element >	ビュー	< Symbol >	< null >
文字	TextElement/TextNote	文字	TextElementType/TextNoteType	< null >
レベル	Level	レベル	LevelType	レベル
モデルグループ	Group	モデルグループ	GroupType	モデルグループ

# 要素の基本クラス

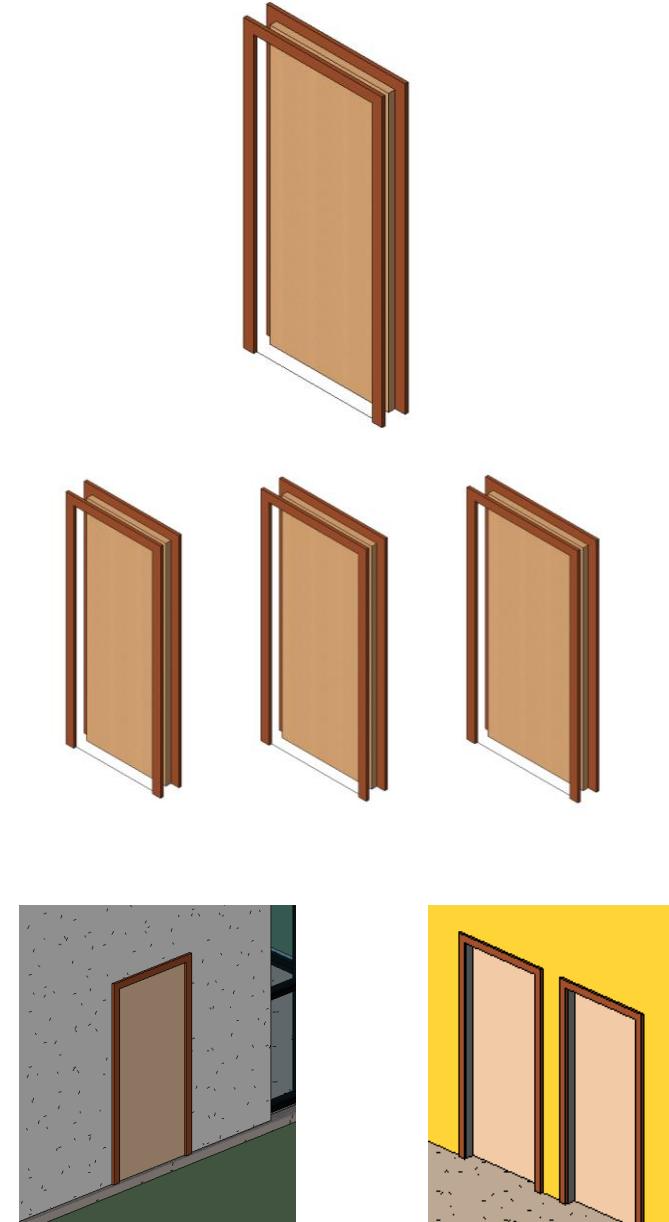


# 要素の基本クラス



# Element クラスのプロパティ

- 要素 ID
- ユニーク ID
- 名称
- カテゴリ
- タイプ
- パラメータ
- ジオメトリ
- バウンディングボックス
- ロケーション
- レベル ID
- グループ ID



Element クラスを継承した Family クラス

Element クラスを継承した FamilySymbol クラス

Element クラスを継承した FamilyInstance クラス

# 要素 ID

- ElementId は、**単一のプロジェクト内での一意の ID** であり、プロジェクト間では一意ではありません。
- Revit データベースから特定の要素を取得する際に利用します。
- ただし、ID は、**Autodesk Revit セッション中に変更される可能性があり**、そのため、外部コマンドを繰り返し呼び出すときに保持して使用しないでください。

```
// Get the id of the element
Autodesk.Revit.DB.ElementId selectedId = element.Id;
int idInteger = selectedId.IntegerValue;

// create a new id and set the value
Autodesk.Revit.DB.ElementId id = new Autodesk.Revit.DB.ElementId(idInteger);

// Get the element
Autodesk.Revit.DB.Element first = document.GetElement(id);
```

## ユニーク ID

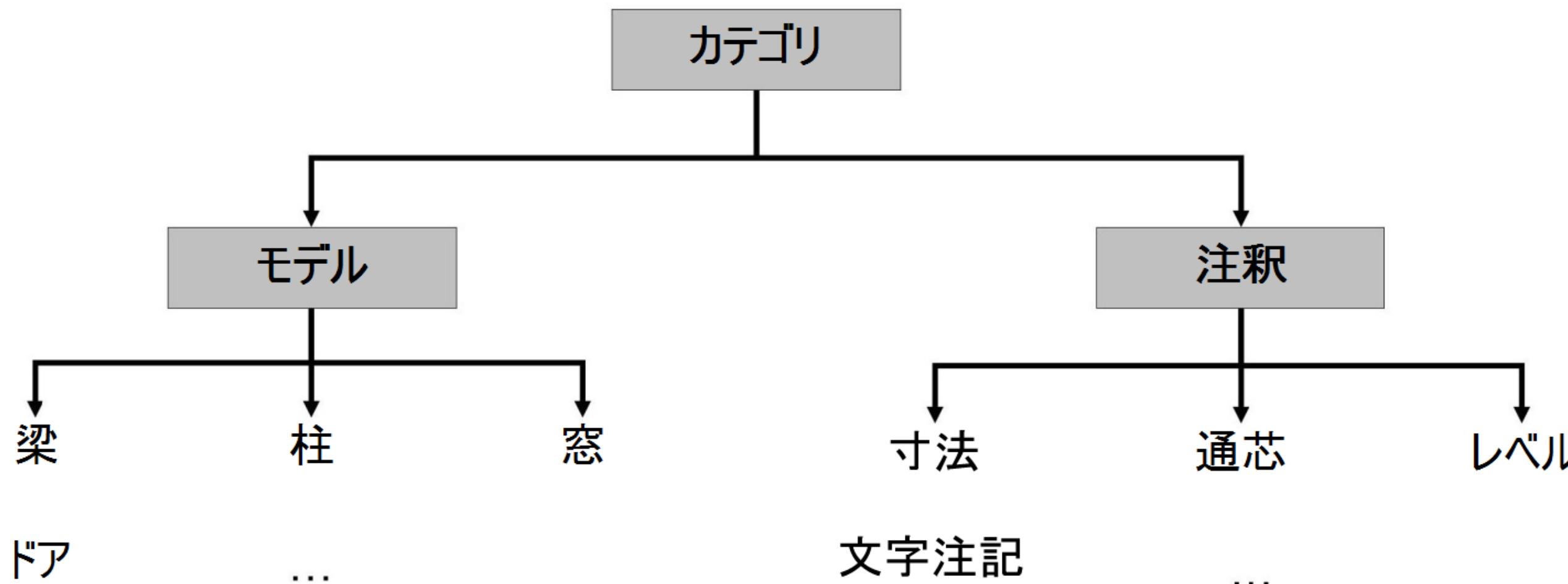
- 要素は、文字列ストレージ タイプで表される UniqueId を持ります。
- UniqueId は ElementId に対応します。ただし、ElementId とは異なり、UniqueId は GUID (Globally Unique Identifier)と同様の機能を持ち、別々の Revit プロジェクト間でユニークとなります。
- UniqueId を使用すると、Revit プロジェクト ファイルを他の形式に書き出すときに要素を追跡することができます。

例えば、

6cbabf1d-e8d0-47f0-ac4d-9a7923128d37-0006fb46

# 要素のカテゴリ

- Element.Category プロパティは、要素が属するカテゴリまたはサブカテゴリを表します。
- 要素のタイプを識別するために使用されます。
- Document.Settings.Categories プロパティは、ドキュメント内のすべてのカテゴリ オブジェクトが含まれるマップであり、次のように再分類されます。



# カテゴリによる要素の識別

- 要素のクラスとカテゴリから、そのファミリがモデル要素の何を表しているか識別することができます。

	システム ファミリ	コンポーネント ファミリ
ファミリ タイプ	WallType FloorType	FamilySymbol & カテゴリードア、窓
インスタンス	Wall Floor	FamilyInstance & カテゴリードア、窓

説明:  
左側の列は「ファミリ タイプ」で、右側の列は「インスタンス」です。  
「システム ファミリ」欄には、Revit の内部名前である WallType, FloorType が示されています。  
「コンポーネント ファミリ」欄には、Revit の内部名前である FamilySymbol & FamilyInstance が示されています。  
「カテゴリードア、窓」の部分は赤色で強調されています。これは、これらの要素が Revit の内部で「ドア」や「窓」として認識されるためです。  
各要素の例として、壁面構成図 (WallType)、床面構成図 (FloorType)、実際の壁面 (Wall)、実際の床面 (Floor) が示されています。また、ドアと窓の例として、ドア開口部の詳細図 (FamilySymbol) と、ドアと窓の実際のインスタンス (FamilyInstance) が示されています。

- 要素カテゴリは特定の振る舞いを定義します。
- 要素にはそのカテゴリに基づくパラメータが割り当てられます。
- カテゴリは Revit での表示やグラフィカルな表現をコントロールするためにも使用されます。

# カテゴリの API 操作

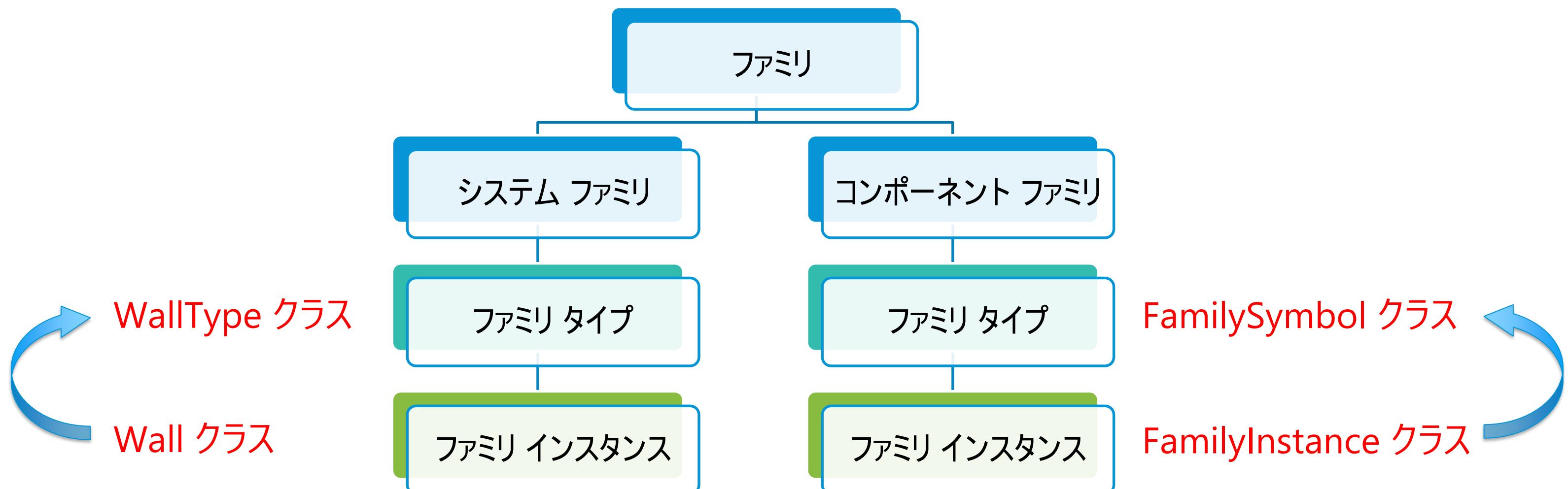
- カテゴリは、それぞれカテゴリ ID を保持し、ElementId クラスによって表されます。
- ほとんどのカテゴリは、組み込みのカテゴリであり、ElementId の定数が割り当てられます。
- 各組み込みカテゴリ ID は BuiltInCategory 列挙値に対応する値を持ちます。
- カテゴリIDは、ElementId であり、BuiltInCategory 列挙型に変換することができます。

```
Element selectedElement = null;  
  
Category category = selectedElement.Category;  
  
ElementId categoryId = category.Id;  
  
BuiltInCategory enumCategory = (BuiltInCategory) category.Id.IntegerValue;
```

BuiltInCategory.OST\_Roofs  
BuiltInCategory.OST\_Floors  
BuiltInCategory.OST\_Doors  
BuiltInCategory.OST\_Windows  
BuiltInCategory.OST\_Walls  
...

# 要素タイプ

- 要素タイプは、インスタンスを設定するために使用される非表示の要素です。
- システム ファミリのインスタンスから要素タイプを取得すると、そのインスタンスの雛型となる組み込みタイプが返却されます。
- コンポーネント ファミリのインスタンスから要素タイプを取得すると、FamilySymbol を取得できます。



# インスタンスの要素タイプを変更

- ホスト要素のインスタンスの場合

```
Wall aWall = (Wall)e;  
  
Element newWallType = ElementFiltering.FindFamilyType(_doc, typeof(WallType), wallFamilyName, wallTypeName, null);  
  
if (newWallType != null)  
{  
    aWall.WallType = (WallType) newWallType;  
}
```

- ファミリ インスタンスの場合

```
FamilyInstance aDoor = (FamilyInstance) doorElem;  
  
Element newDoorType = ElementFiltering.FindFamilyType(_doc, typeof(FamilySymbol), doorFamilyName, doorTypeName,  
BuiltInCategory.OST_Doors);  
  
if (newDoorType != null)  
{  
    aDoor.Symbol = (FamilySymbol) newDoorType; // 変更方法1  
    aDoor.ChangeTypeId(newDoorType.Id); // 変更方法2  
}
```

# 要素パラメータへのアクセス

- すべての要素には、ユーザが Revit で表示、編集できる一連のパラメータがあります。
- Revit UI では、**プロパティパレット**には一部の要素パラメータが表示されます。
- 各 Element オブジェクトには、**要素にアタッチされたすべてのプロパティを取得できる Parameters プロパティ**があります。

Element.Parameters	Element のすべてのパラメータのセットを取得する。
Element.GetOrderedParameters()	プロパティパレットで表示されるパラメータのみを取得する。
Element.GetParameters(string name)	指定された名前に一致するすべてのパラメータを取得する。
Element.LookupParameter(string name)	指定された名前に一致するすべてのパラメータのうち、 <b>最初に一致したパラメータ</b> を取得する。（※非推奨）
Element.Parameter オーバーロード	指定された引数に一致するパラメータを個別に取得する。
Element.ParametersMap	名前によりパラメータにアクセスする方法。

# 要素パラメータの種類

- 要素パラメータを「**名前**」をキーにして取得する際は、既に同じ名前の組み込みパラメータがある場合でも、共有パラメータやプロジェクト パラメータを要素カテゴリにバインドすることができるため、**同じ名前を持つ複数のパラメータの一一致が発生することがあります。**

Element.Parameters

組み込みパラメータ	Revit の要素パラメータのほとんどは組み込みパラメータとして定義されています。 <a href="#">Autodesk.Revit.Parameters.BuiltInParameter</a> 列挙値で識別することができます。
共有パラメータ	外部テキスト ファイルに保存されているパラメータの定義です。ファミリに追加したり、カテゴリ毎にファミリ タイプやファミリ インスタンスに追加することができます。
プロジェクト パラメータ	プロジェクトで設定し複数の要素カテゴリに追加する情報のコンテナです。プロジェクト固有のもので、他のプロジェクトと共有することはできません。
グローバルパラメータ	1つのプロジェクト ファイルに固有のものですが、カテゴリには割り当てられません。

# BuiltInParameter 列挙値の利用を推奨

- Parameter(BuiltInParameter)
  - 組み込みパラメータの ID を使ってパラメータを取得
  - Revit LookUp ツールを使用して、パラメータ名に対応する BuiltInParameter の調査が可能
- Parameter(Definition)
  - パラメータの定義からパラメータを取得
- Parameter(GUID)
  - GUID を使って共有パラメータを取得

```
public Parameter FindWithBuiltInParameterID(Wall wall)
{
    BuiltInParameter paraIndex = BuiltInParameter.WALL_BASE_OFFSET;
    Parameter parameter = wall.get_Parameter(paraIndex);

    return parameter;
}
```

# パラメータの値を変更

- パラメータに含まれるデータは、Double、Integer、String、ElementId のいずれかになります。パラメータの **StorageType** プロパティで確認できます。
- 要素の内部状態に依存する一部のパラメータは、常に読み取り専用になります。
- UI 操作でそのパラメータが編集可能かどうか確認したり、Revit Lookup ツールを活用しましょう。

```
public bool SetParameter(Parameter parameter, double value)
{
    bool result = false;

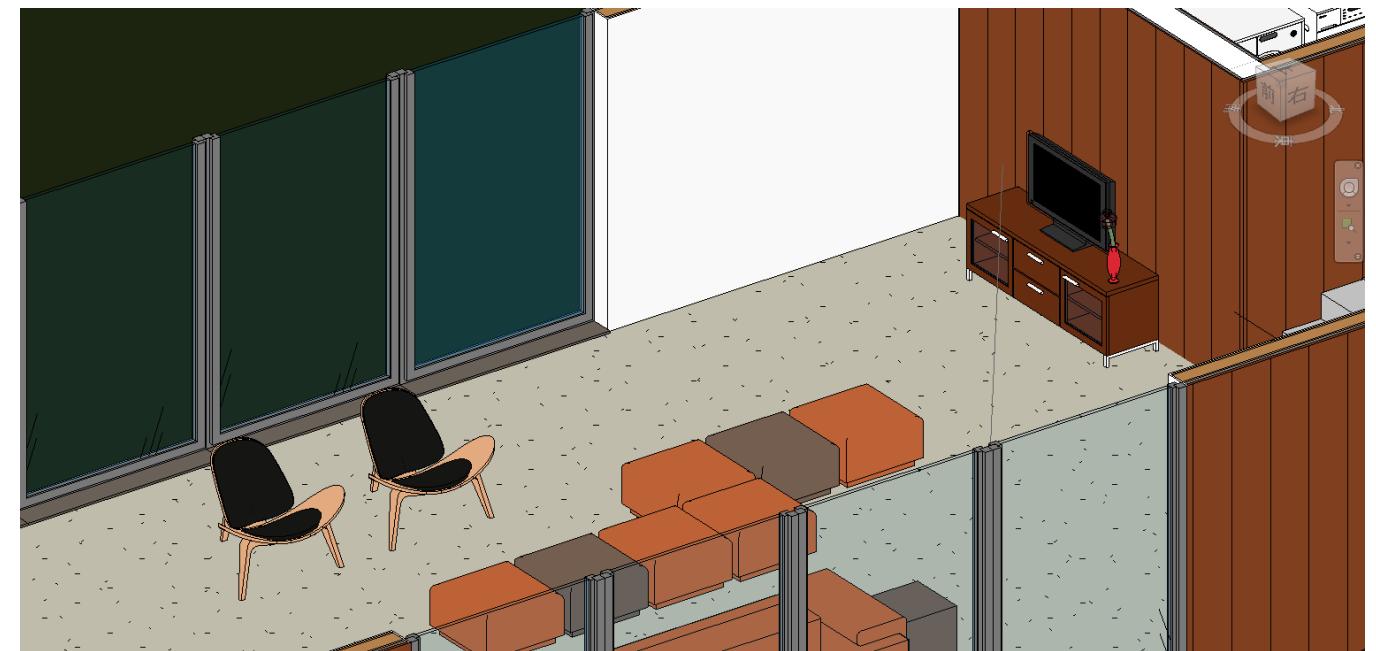
    if (null != parameter && !parameter.IsReadOnly)
    {
        StorageType parameterType = parameter.StorageType;
        if (StorageType.Double != parameterType)
        {
            throw new Exception("The storagetypes of value and parameter are different!");
        }

        result = parameter.Set(value);
    }

    return result;
}
```

# 位置

- Location プロパティ
- Location は2つの形式で派生:
  - LocationPoint – 点ベースの場所（例、家具）
  - LocationCurve – 線分ベースの場所（例、壁）
- Location 要素を LocationCurve オブジェクトや LocationPoint オブジェクトにダウンキャストすると、**曲線や点を新しい場所に直接移動します。**

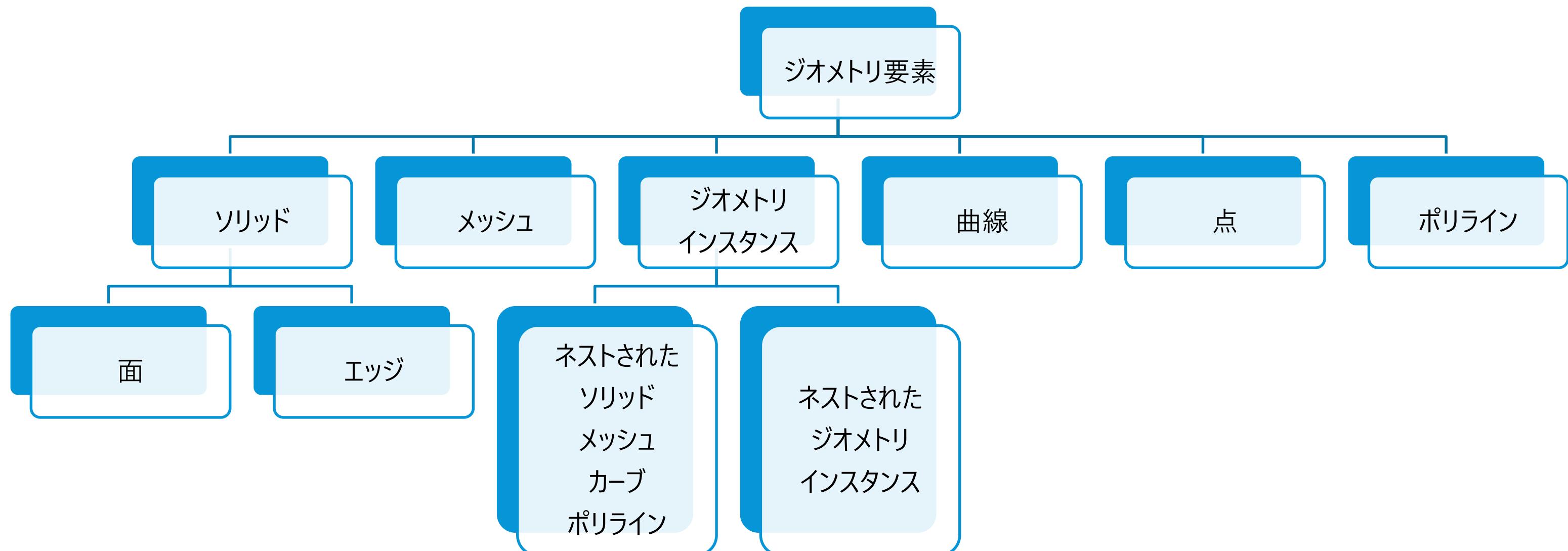


```
void MoveUsingCurveParam(Autodesk.Revit.ApplicationServices.Application application, Wall wall)
{
    LocationCurve wallLine = wall.Location as LocationCurve;
    XYZ p1 = XYZ.Zero;
    XYZ p2 = new XYZ(10, 20, 0);
    Line newWallLine = Line.CreateBound(p1, p2);

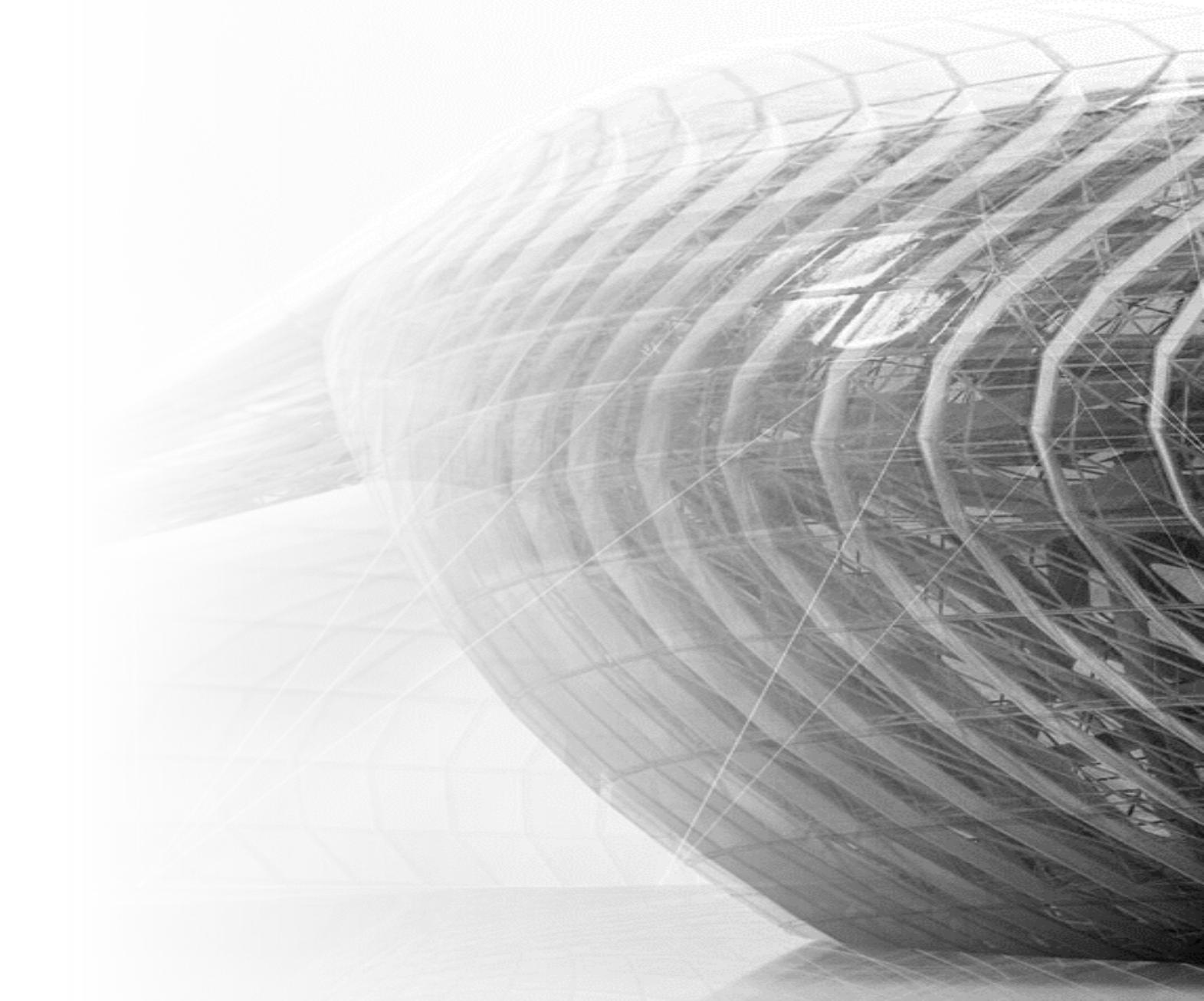
    wallLine.Curve = newWallLine;
}
```

# ジオメトリ

- Geometry プロパティは、Autodesk.Revit.DB.GeometryElement を返します。
- GetEnumerator() メソッドを使用すると、該当する要素のジオメトリ メンバーを反復することができます。
- GeometryInstance とは、要素内に配置されたジオメトリ要素のインスタンスです。



# 要素のフィルタリング



# 要素のフィルタリング

- Revit API は Revit ドキュメントの要素をフィルタリング、走査するためのメカニズムが用意されています。
- 例えば、次のような目的で要素を検索・取得します。

1. ファミリ タイプのリストを取得

(例、壁タイプ、ドア タイプ)

2. 特定のオブジェクト クラスのインスタンスを取得

(例、すべての壁、すべてのドア)

3. 与えられた名前で特定のファミリ タイプを検索

(例、"標準壁: 標準-200mm"、"片開きフラッシュ: 0915 x 2134mm")

4. 特定のインスタンスを検索

(例、"レベル 1"、"平面図 1")

# FilteredElementCollector クラス

1. 新しい FilteredElementCollector オブジェクトを作成します。コンストラクタの引数に応じて、検索対象の条件を設定することができます。
  - ドキュメントから - ドキュメントで要素のセットを検索してフィルタします。
  - ドキュメントと ElementIds のセットから - 指定したセットの要素を検索してフィルタします。
  - ドキュメントとビューから - ビューに表示可能な要素を検索してフィルタします。
2. そのオブジェクトに 1 つまたは複数のフィルタ（条件）を適用する。
3. 必要に応じて LINQ クエリを使用。
  - 可能な限りネイティブフィルタを利用してパフォーマンスが向上。
4. フィルタされた要素または要素 ID を取得する(複数のメソッドのいずれかを使用)。

# サンプルコード

- 要素のフィルタリングを使用してドキュメント内のすべての壁のインスタンスを取得

```
FilteredElementCollector collector = new FilteredElementCollector(document); ← インスタンス生成  
ElementCategoryFilter filter = new ElementCategoryFilter(BuiltInCategory.OST_Walls); ← フィルタを作成  
IList walls =  
collector.WherePasses(filter).WhereElementIsNotElementType().ToElements(); ← コレクタを実行して要素抽出  
String prompt = "The walls in the current document are:¥n";  
foreach (Element e in walls)  
{  
    prompt += e.Name + "¥n";  
}  
TaskDialog.Show("Revit", prompt);
```

# 要素の編集



# ElementTransformUtils クラス

要素を移動する	
要素をコピーする	ElementTransformUtils クラスでサポート
要素を回転する	
要素を鏡像化する	
要素に位置合わせする	Creation.Document.NewAlignment()メソッドを使用すると、2つの参照間にロックされた位置合わせを新たに作成できます。
要素をグループ化する	Creation.Document.NewGroup()メソッドを使用して、1つまたは複数の要素やグループを選択して結合します。
要素の配列を作成する	プロジェクトに1つまたは複数の要素を配列するための LinearArray と RadialArray の2つのクラスがあります。
要素を削除する	プロジェクト内の1つまたは複数の要素を削除するための Document.Delete()メソッドがあります。

# 要素の再生成と自動結合

- 新しい要素を作成したり、要素を修正した後には、モデル全体にわたって変更を適用するために、**要素の再生成と自動結合**が必要です。
- 再生成がない場合、要素のジオメトリプロパティが取得不能になったり、無効となる場合があります。
- モデルを修正するトランザクションが正しくコミットされた場合は、自動的に再生成と自動結合が実行されます。
- トランザクション中に、要素の再生成を行いたい場合は、**Document.Regenerate()**や**Document.AutoJoinElements()**メソッドを呼び出します。

```
using (Transaction tx = new Transaction(doc))
{
    tx.Start("Transaction Test");

    ... 要素の作成や編集処理 ...

    tx.Commit(); // このタイミングで再生成と自動結合が実行
}
```

```
using (Transaction tx = new Transaction(doc))
{
    tx.Start("Transaction Test");

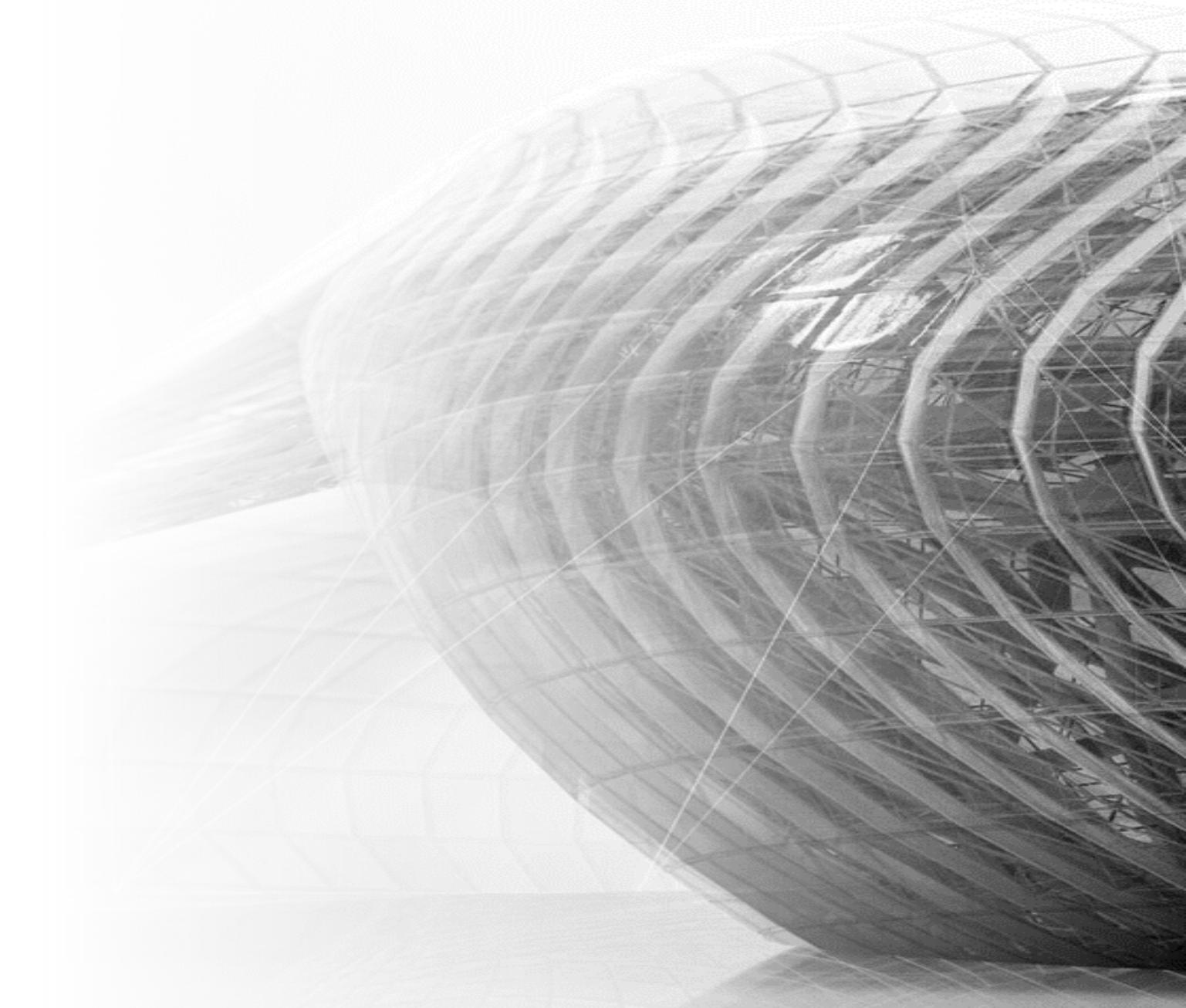
    ... 要素の作成や編集処理 ...

    doc.Regenerate(); // 再生成と自動結合を強制実行

    ... その要素をもとにして、さらに別の処理に利用

    tx.Commit();
}
```

# 要素の作成



# システム ファミリのインスタンスの作成

オブジェクト	要素クラス	要素タイプ	要素の作成方法	その他
壁	Wall	WallType	Wall.Create()	Category = OST_Walls
床	Floor	FloorType	NewFloor()/NewSlab()	Category = OST_Floors FloorType.IsFoundationSlab = false
スラブ	Floor	FloorType	NewSlab()	Category = OST_Floors FloorType.IsFoundationSlab = false
基礎スラブ	Floor	FloorType	NewFoundationSlab()	Category = OST_Floors FloorType.IsFoundationSlab = true
天井	Ceiling	CeilingType	なし	Category = OST_Ceilings
布基礎	WallFoundation	WallFoundationType	WallFoundation.Create()	Category = OST_StructuralFoundation
独立基礎	FamilyInstance	FamilySymbol	NewFamilyInstance()	Category = OST_StructuralFoundation
べた基礎	Floor	FloorType	NewFloor()	Category = OST_StructuralFoundation FloorType.IsFoundationSlab = true
屋根	FootPrintRoof ExtrusionRoof	RoofType	NewFootPrintRoof()	Category = OST_Roofs

# ファミリ インスタンスの作成

- Document.Create プロパティから取得できるAutodesk.Revit.Creation.Document オブジェクトの **NewFamilyInstance()** メソッドを使用します。
- 引数の異なる 12 のオーバーロード メソッド。
- インスタンスのカテゴリだけでなく、ホストされるかどうか、参照レベルを基準にして配置されるかどうか、特定の面に直接配置するかどうかといった、配置の特性によって使用するオーバーロードを選択。

Revit API 開発者用ガイドから一部抜粋

カテゴリ	NewFamilyInstance() パラメータ	コメント
柱 構造柱	XYZ, FamilySymbol, Level, StructuralType	基部が参照レベルに配置されるように柱を作成します。柱はモデル内の次に使用可能なレベルまで延長するか、参照レベルの上に適したレベルがない場合は既定の柱の高さまで延長します。
ドア 窓	XYZ, FamilySymbol, Element, StructuralType	ドアと窓は壁でホストする必要があります。既定の向きに配置できる場合にこのメソッドを使用します。
	XYZ, FamilySymbol, XYZ, Element, StructuralType	作成したインスタンスを既定以外の方向に設定する必要がある場合
	XYZ, FamilySymbol, Element, Level, StructuralType	インスタンスを参照レベルに関連付ける必要がある場合

# ファミリの作成と編集

- ファミリもプロジェクトと同じくドキュメントで表されます。
- 新しいファミリドキュメントを作成するには、Application.NewFamilyDocument() メソッドを使用します。

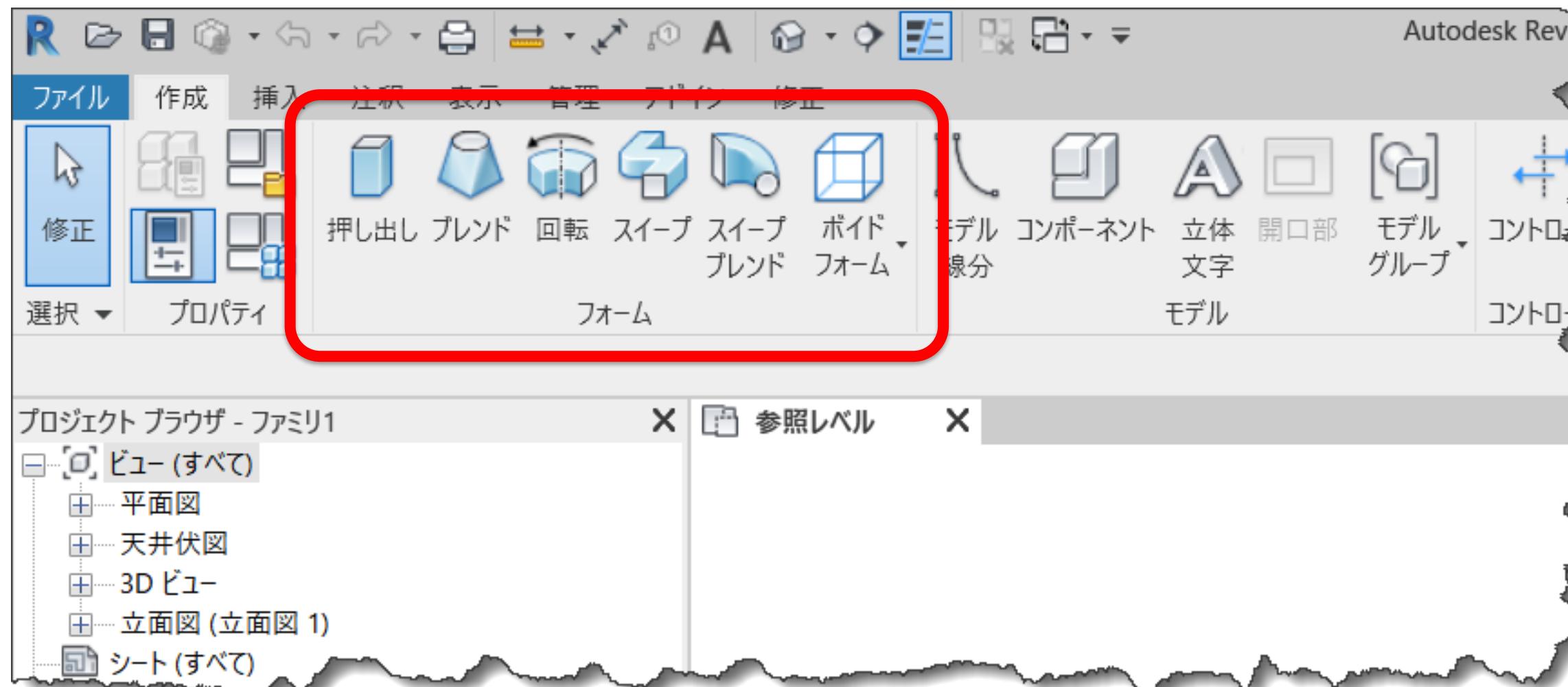
```
string templateFileName = @"C:\Documents and Settings\All Users\Application Data\Autodesk\RST 2011\Imperial  
Templates\Generic Model.rft";
```

```
Document familyDocument = application.NewFamilyDocument(templateFileName);  
if (null == familyDocument)  
{  
    throw new Exception("Cannot open family document");  
}
```

- プロジェクトドキュメントでの作業中に既存のファミリを編集するには、Document クラスから使用できる EditFamily() メソッドを使用します。
- その後、編集が終わったら LoadFamily() メソッドを使用してプロジェクトドキュメントにファミリを再ロードします。

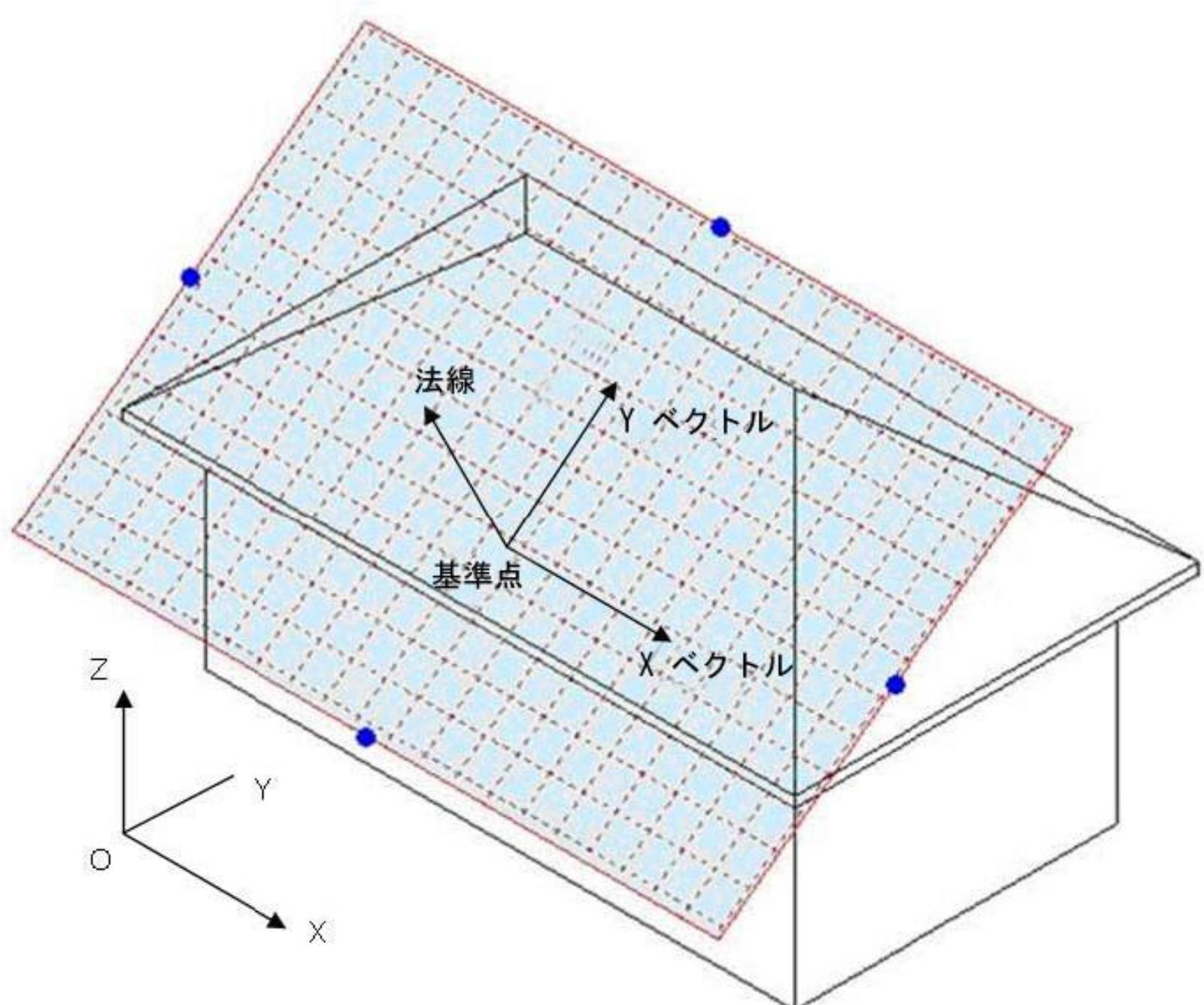
# ファミリ内のフォーム要素を作成

- Document.FamilyCreate プロパティから取得できる Autodesk.Revit.Creation.FamilyItemFactory オブジェクトのメソッドを使用します。
- FamilyItemFactory クラスには、ファミリ内にフォーム要素(押し出し、回転、スイープ、ブレンドなど)を作成するメソッドがあります。



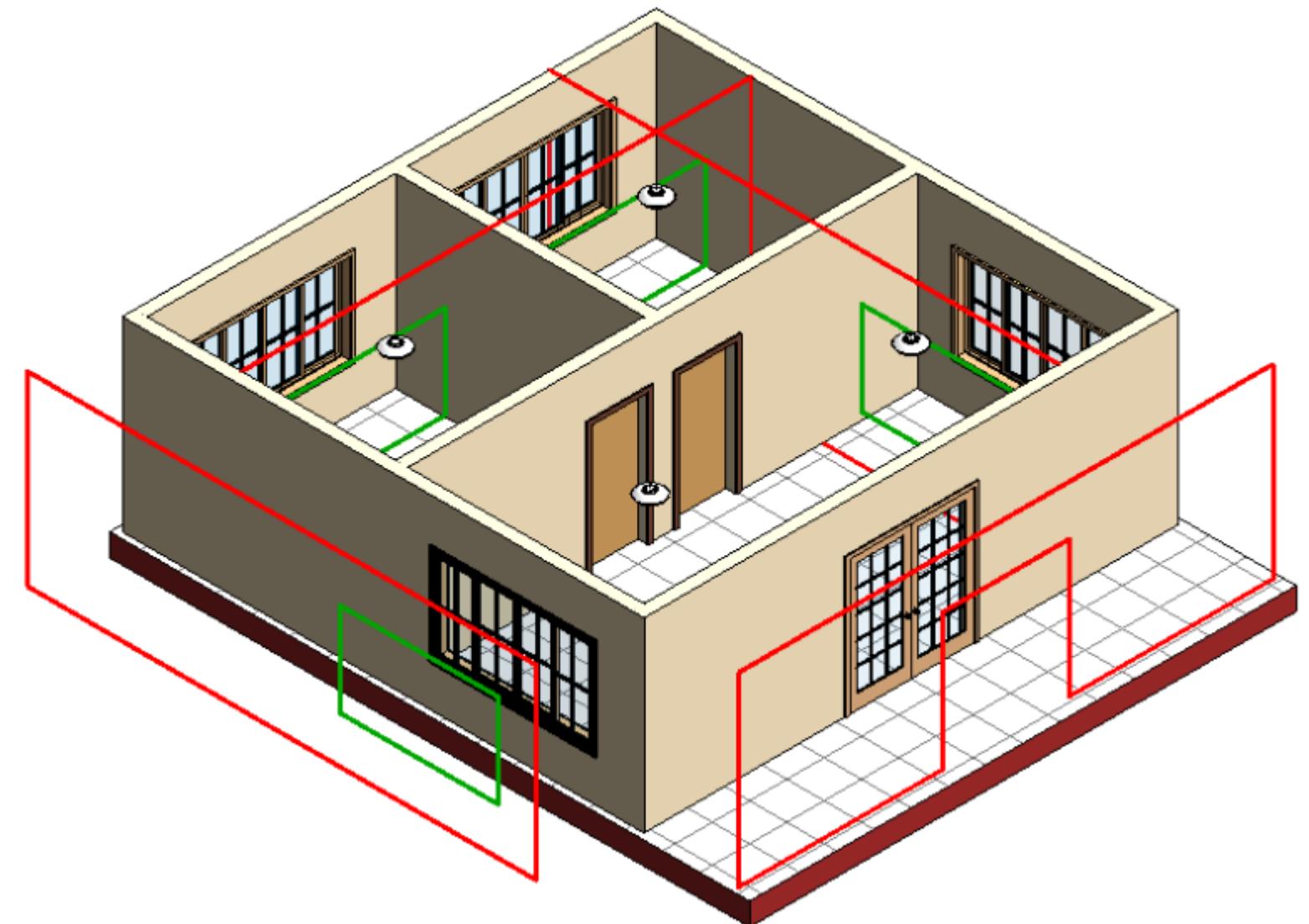
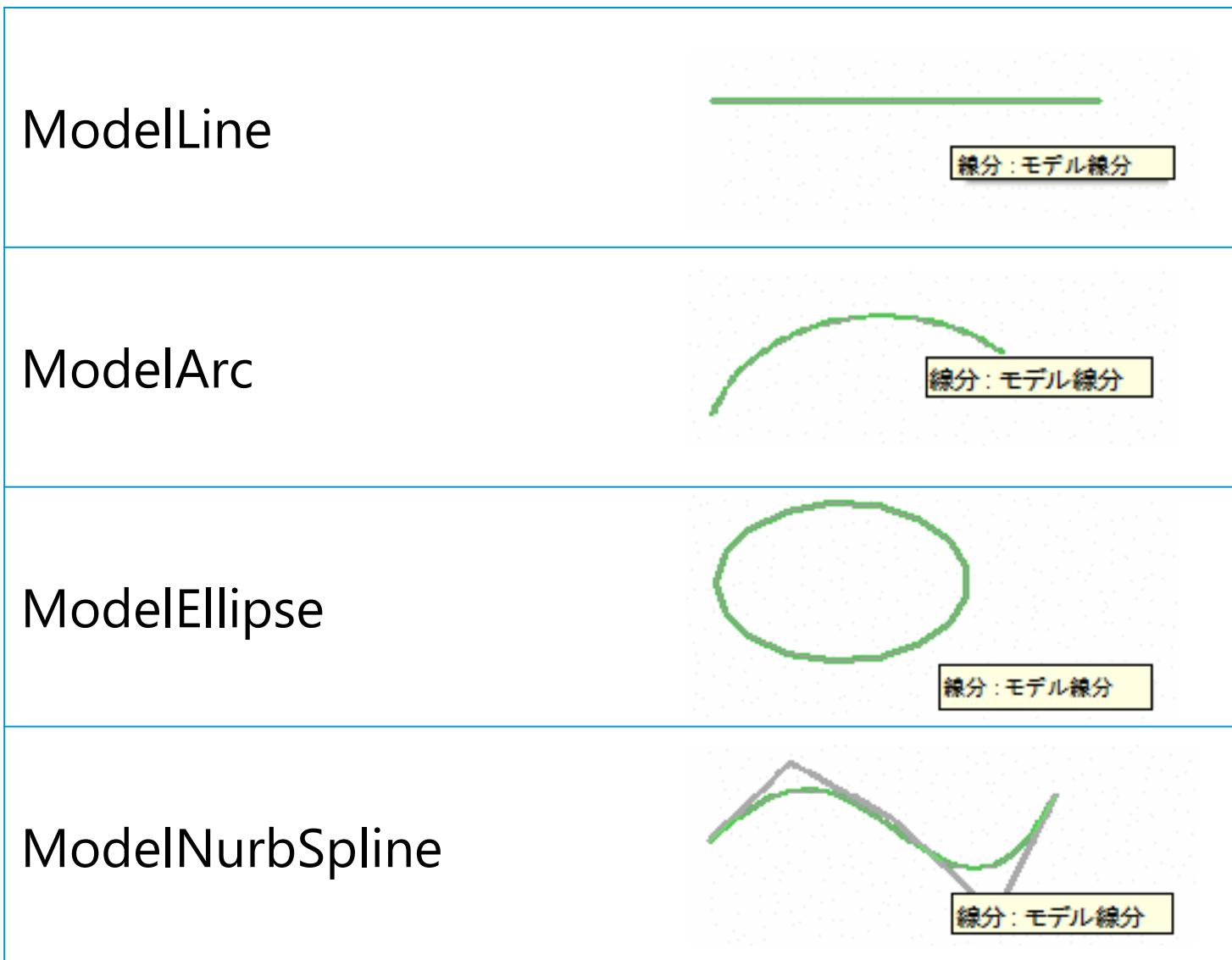
# SketchPlane と作業面

- フォーム要素やモデル線分などを作成する際には、SketchPlane が必要になります。
- SketchPlane は、Revit UI では常時非表示となっており、**作業面**とは厳密には異なります。
- 作業面は API で対応するクラスはありません。ただし、作業面は SketchPlane と対応関係にあります。
- 2D ビューには、既定では、SketchPlane が自動的に作成されます。

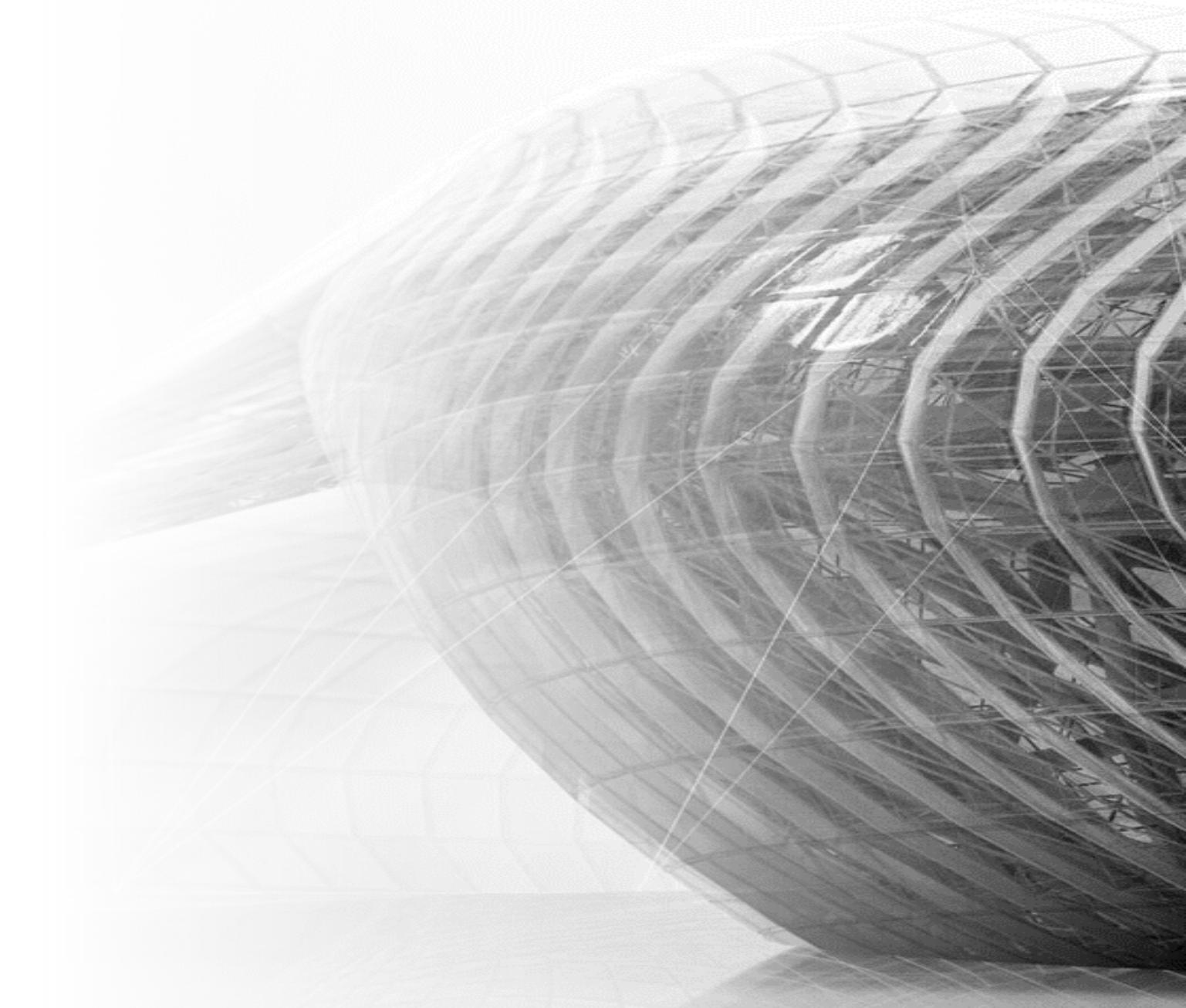


# ModelCurve

- ModelCurve は、プロジェクトのモデル線分を表します。
- 3D 空間に存在し、すべてのビューに表示されます。
- デバッグ時に、線分が想定通り取得・作成できているか確認する際に使用すると便利です。

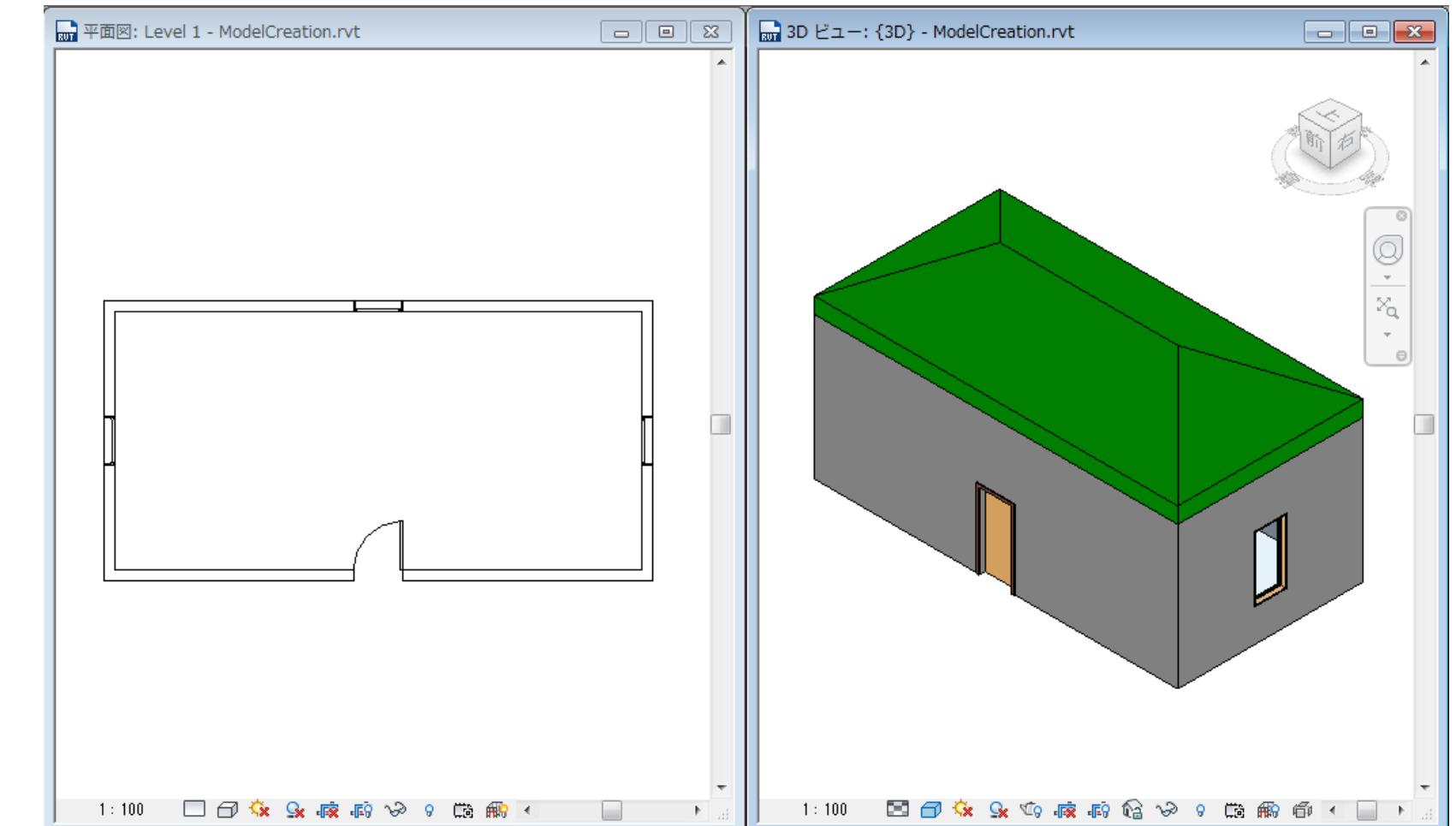
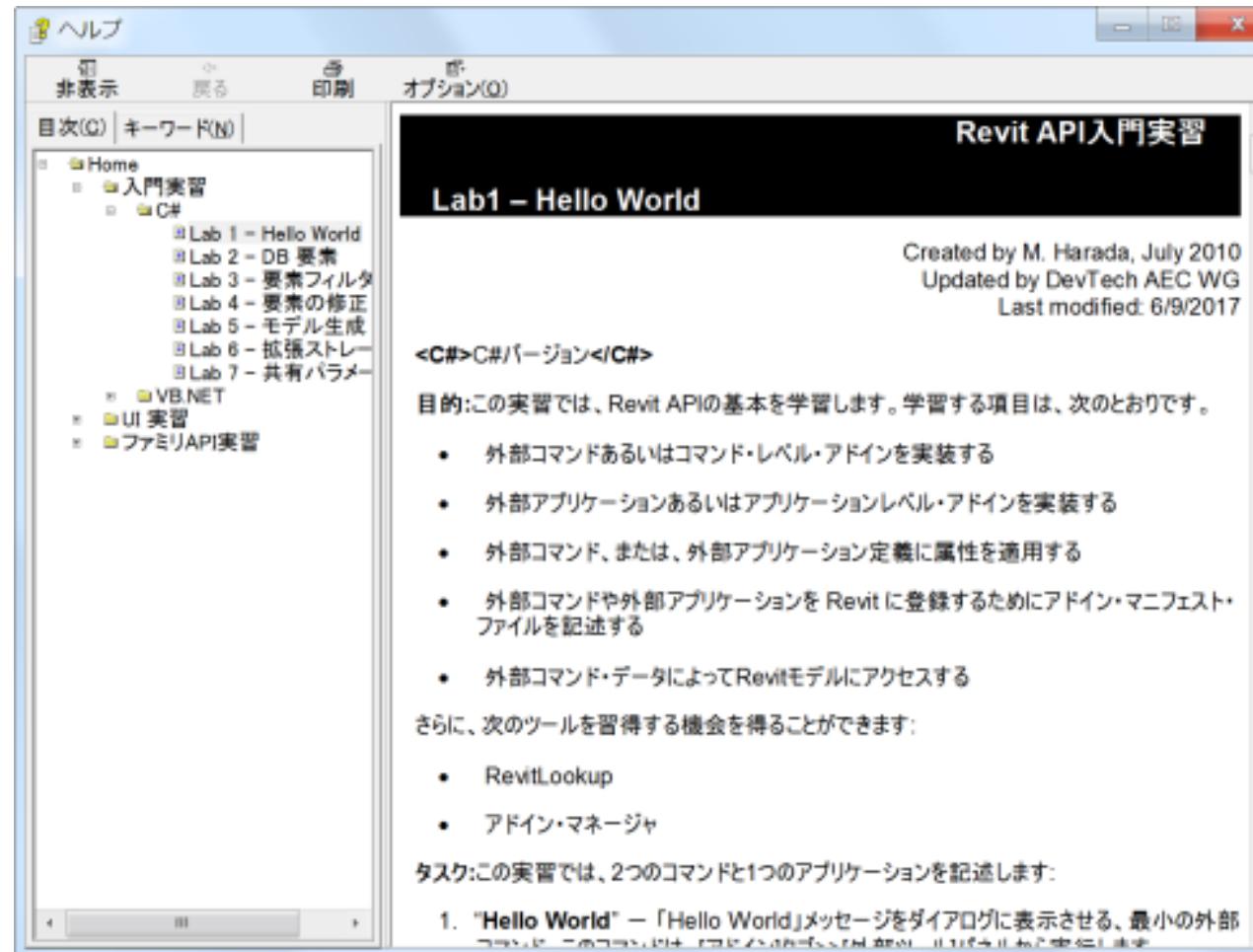


# Revit API の学習コンテンツ



# Revit API トレーニングマテリアル

- プрезентーションスライドとサンプルコードが同梱されているチュートリアル教材
- C# と VB.NET それぞれのコンテンツ
- Revit DB, UI, ファミリ作成



# Revit API 開発者用ガイド

- Revit API の概要を網羅的に解説
- 各ページでは、クラスやメソッドの解説や、サンプルコードが掲載
- 日本語で閲覧可能なコンテンツで最も情報量の多いコンテンツ

The screenshot shows the 'Revit API Developers Guide' page from the Autodesk Revit 2019 documentation. The top navigation bar includes links for 'Help Home Page', user profile 'Ryuji Ogasawara', language 'Japanese', and a search bar. The main content area features the title 'Revit API Developers Guide' and a brief introduction stating it's for the Autodesk Revit Application Programming Interface (API). It lists related concepts like 'Dynamo'. Below this is a section titled 'このセクションのページ' (Pages in this section) containing links to various Revit API topics such as 'はじめに' (Getting Started), 'Revit エлементとの基本的なやりとり' (Basic interaction with Revit elements), and 'FAQ'. At the bottom, there's a survey question 'この情報は役に立ちましたか?' (Did this information help you?) with two options: 'はい' (Yes) and 'いいえ' (No).

ヘルプのホームページ Ryuji Ogasawara 日本語

AUTODESK® REVIT® 2019 キーワードを入力

Revit API 開発者用ガイド

この API 開発者用ガイドは、Autodesk Revit のアプリケーション プログラミング インタフェース(API)の使用方法について説明します。

関連する概念

- Dynamo

このセクションのページ

- はじめに
- Revit エлементとの基本的なやりとり
- Revit のジオメトリ要素
- 専門分野固有の機能
- 高度なトピック
- 付録
- FAQ

この情報は役に立ちましたか?

はい |  いいえ

AUTODESK Knowledge Network この著作物は、特に断りのない限り、クリエイティブ コモンズ ライセンス(表示 - 非営利 - 継承 3.0 非移植)が適用されます。 詳細については、Autodesk Creative

# Revit SDK

- API リファレンスでは、メソッドの引数の詳細やオーバーロード、プロパティ、API を使用するにあたっての注意事項、仕様などを確認できます。
- 多数のサンプルコードを同梱。

The screenshot shows the Revit 2019 API documentation window. The left sidebar lists namespaces, and the main content area displays the following information:

## Major changes and renovations to the Revit API

### API changes

**.NET 4.7**

Revit's API assemblies are built using .NET 4.7. At a minimum, add-ons will need to target .NET 4.7 for Revit 2019.

### View Filters support OR operators and nesting

#### ParameterFilterElement API changes

View filters now support multiple nested levels of combinations of criteria joined with either "AND" or "OR". In the API, an ElementFilter hierarchy is replacing the use of lists of FilterRules previously obtained from ParameterFilterElement. Several methods are deprecated and replaced. Note that the deprecated methods will still function as before when accessing the criteria contained within a filter joined by a single AND, but cannot function as expected when accessing more complicated sets of criteria. The new method:

- ParameterFilterElement.UsesConjunctionOfFilterRules()

can be used to check if a ParameterFilterElement uses a conjunction of filter rules (without any logical OR operations), as was always the case before Revit release 2019. This function is also deprecated, as are all functions related to the use of FilterRules in ParameterFilterElement.

Deprecated function	Replacement function	Notes
ParameterFilterElement.Create(Document, String, ICollection<ElementId>, IList<FilterRule>)	ParameterFilterElement.Create(Document, String, ICollection<ElementId>, ElementFilter)	The ElementFilter input now specifies the filtering rules. This allows combinations of filter rules using logical AND and OR operations. The ElementFilter must be either an ElementParameterFilter or an ElementLogicalFilter containing only ElementParameterFilters and other ElementLogicalFilters.
ParameterFilterElement.GetRules()	ParameterFilterElement.GetElementFilter()	The new function returns an ElementFilter representing the combination of filter rules used by the ParameterFilterElement. Note that logical combinations using both AND and OR operations are possible. GetRules() is available only to filters with rules.



AUTODESK®

Make anything.

Autodesk and the Autodesk logo are registered trademarks or trademarks of Autodesk, Inc., and/or its subsidiaries and/or affiliates in the USA and/or other countries. All other brand names, product names, or trademarks belong to their respective holders. Autodesk reserves the right to alter product offerings and specifications at any time without notice, and is not responsible for typographical or graphical errors that may appear in this document.

© 2017 Autodesk. All rights reserved.