



Revit API & BIM セミナー 2018

Revit アドイン応用 ~ 具体例

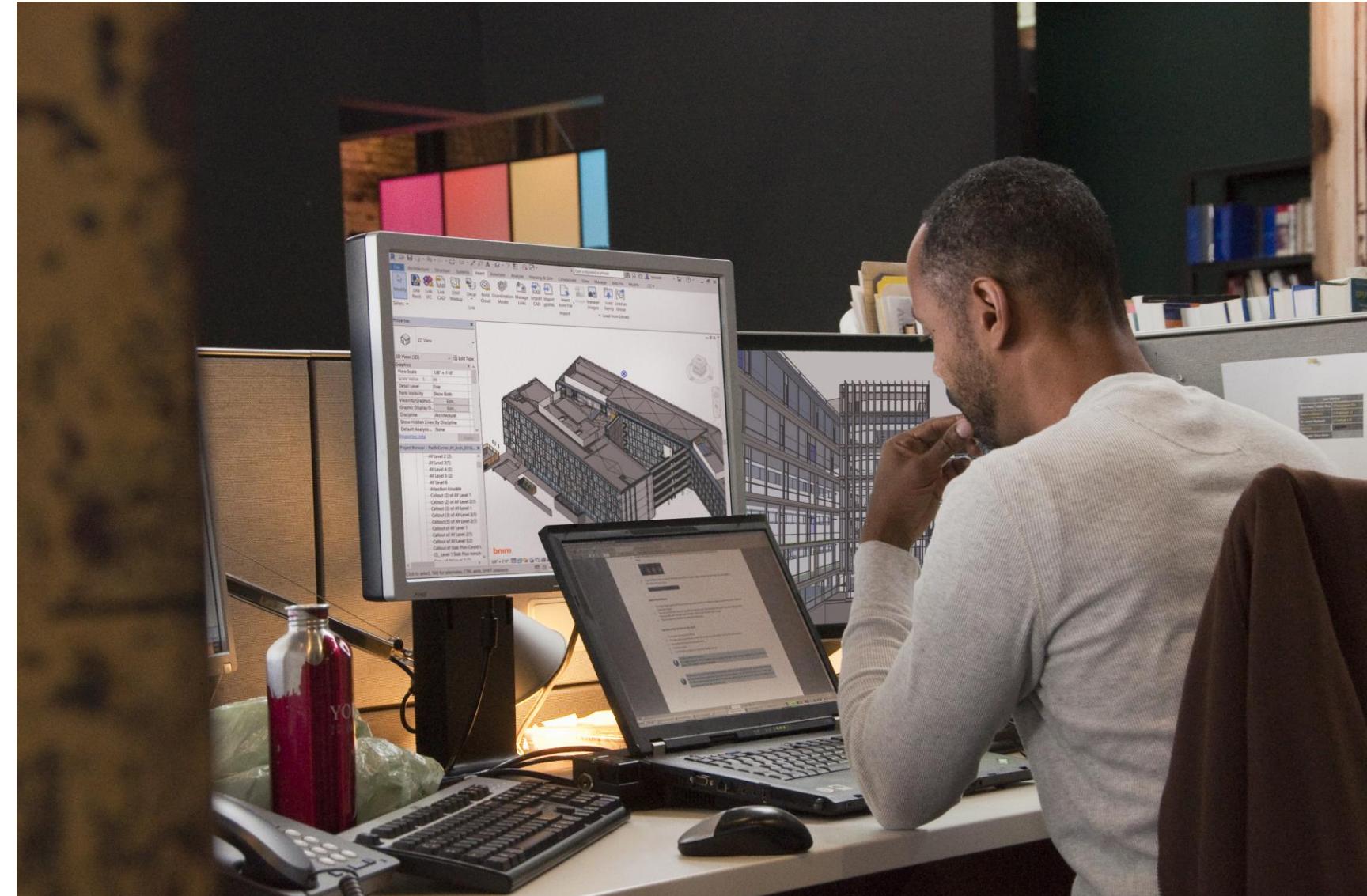
小笠原 龍司

Autodesk Developer Network, Forge Partner Development



アジェンダ

- マルチスレッドについて
- 一時トランザクション
- 単位と精度
- 通芯とレベル
- 参照と寸法
- スタイルとマテリアル
- ジオメトリ
- パフォーマンス
- API の調べ方

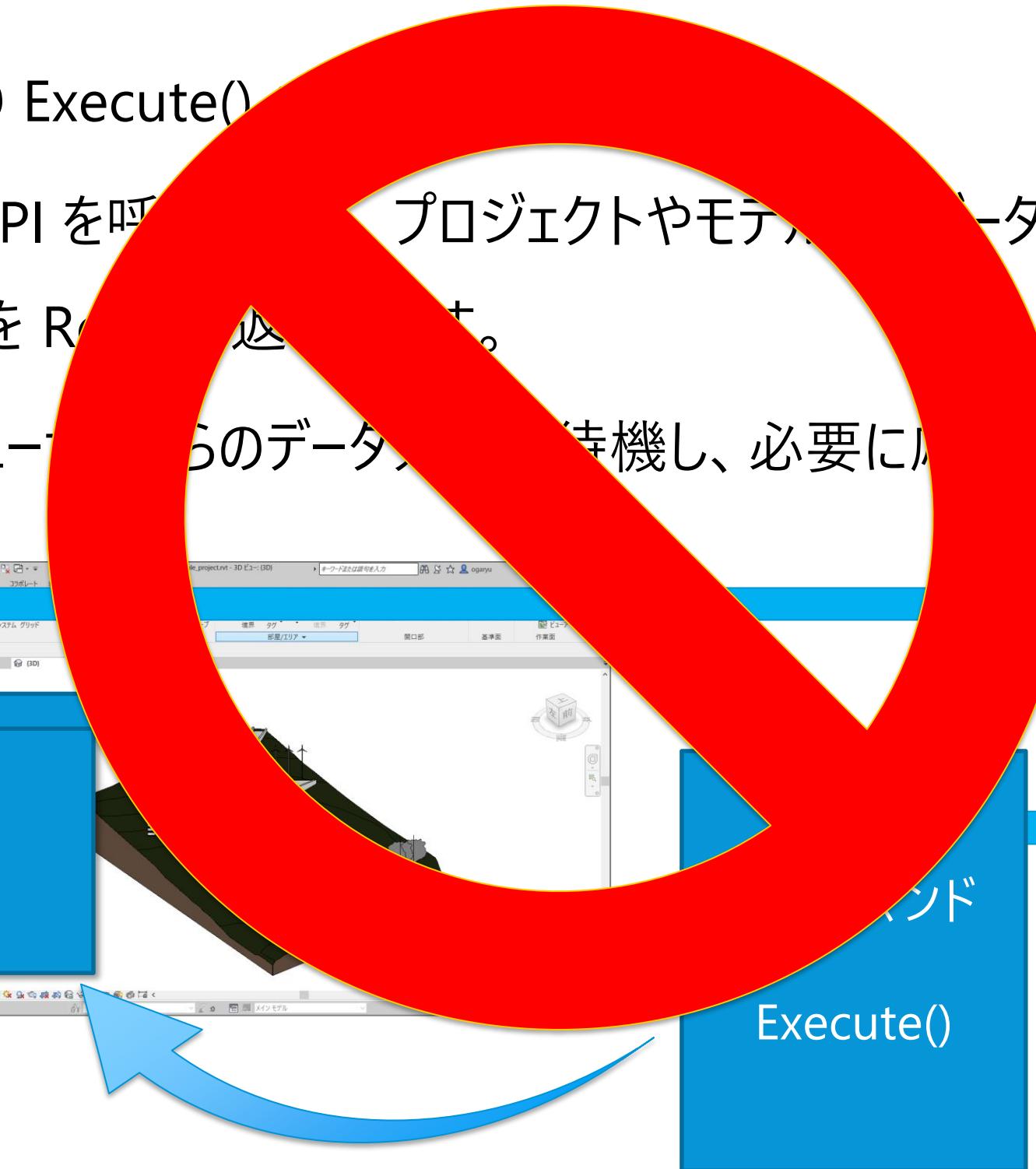
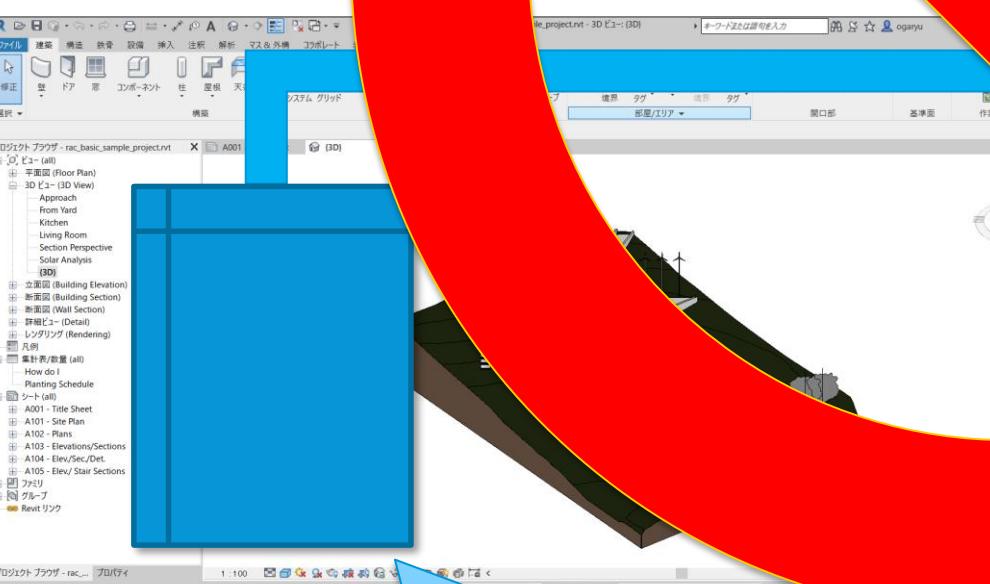


マルチスレッド



シナリオ

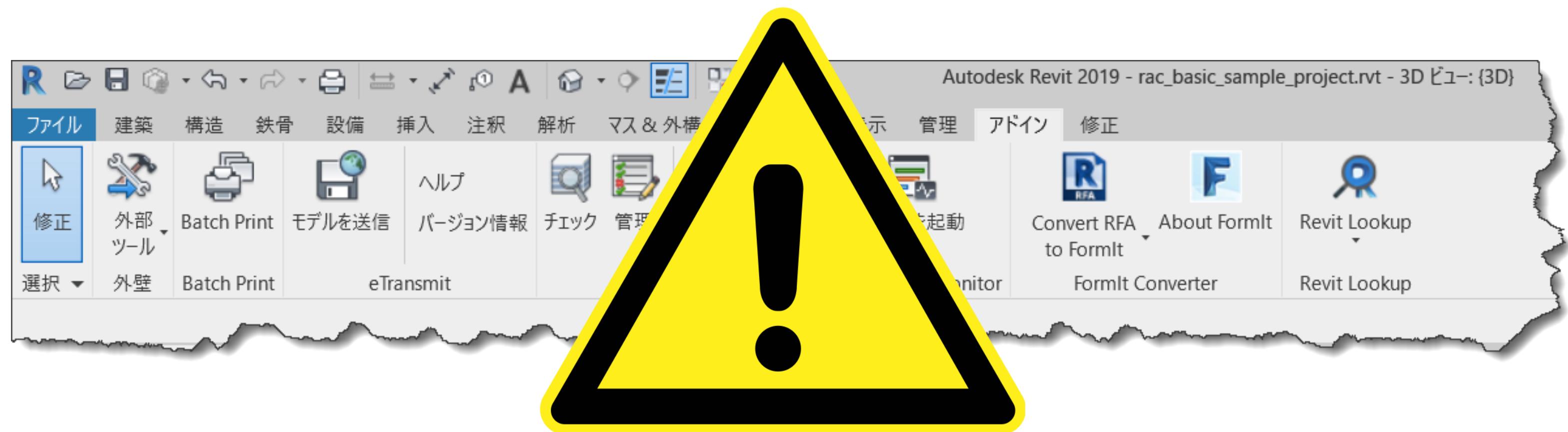
1. Revit が、外部コマンドの Execute()
2. 外部コマンドは、Revit API を呼び出し、データを抽出し、モードレスダイアログで表示して、コントロールを Revit に返します。
3. モードレスダイアログは、ユーザーからのデータ入力を待機し、必要に応じて Revit API を呼び出します。



ダイアログ表示

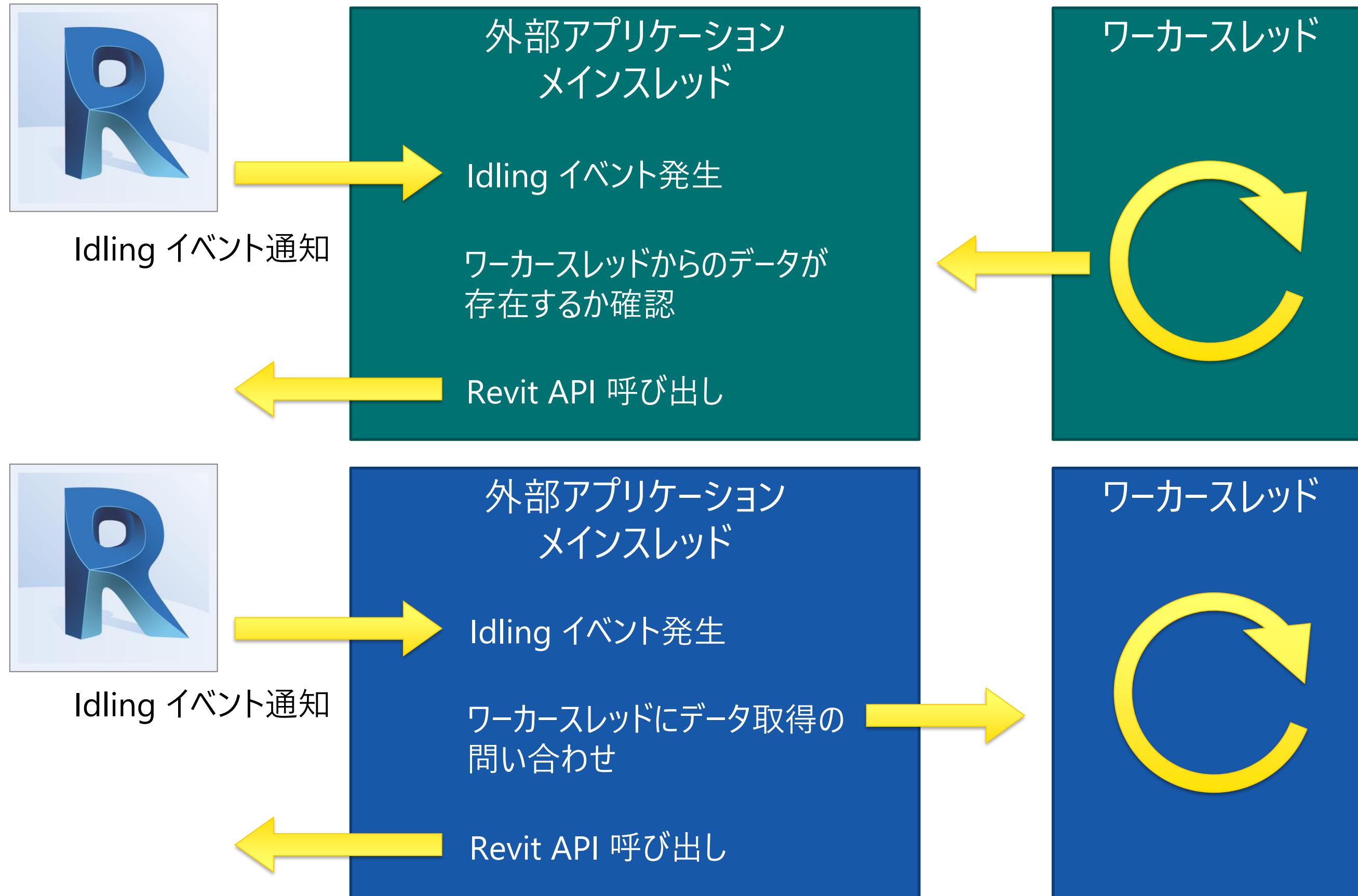
Revit API はマルチスレッド非対応

- Revit API は、**单ースレッド**からのアクセスのみをサポートします。
- アプリケーションは、すべての Revit API の呼び出しを**メインスレッド**で実行する**必要**があります。
- 他のスレッドから API にアクセスすることは可能ですが、予期しないエラーが発生する可能性があるため、推奨しておりません。



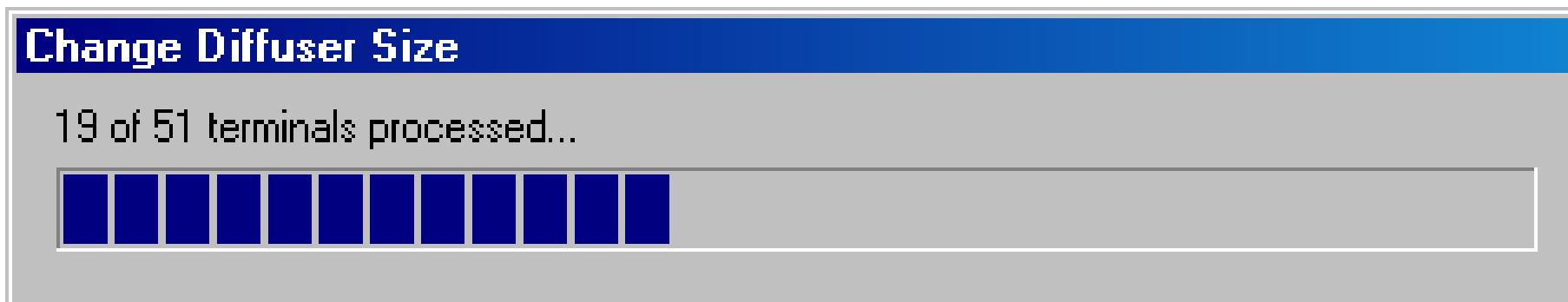
回避策

- 外部アプリケーションが、マルチスレッドを実装すること自体は問題ありません。
 1. 外部アプリケーションとその外部コマンドを Revit に登録します。
 2. 外部コマンドは、ワーカースレッド（またはモードレスダイアログ）を起動し、動作中（または待機中）のままにします。
 3. スレッドの起動が成功した場合、外部コマンドは Idling イベントのハンドラを登録します。
 4. Revit はそのまま実行中です。
 5. Revit は、可能な限り適切なタイミングで Idling イベントを発生させます。
 6. ここで、外部アプリケーションが動作中のスレッド（またはモードレスダイアログ）と通信する方法は次のようなパターンが考えられます。



プログレスバーの実装

- Application.DoEventsを使用する方法
- The ADN MEP Sample AdnRme for Revit MEP 2013
- <http://thebuildingcoder.typepad.com/blog/2012/05/the-adn-mep-sample-adnrme-for-revit-mep-2013.html>
-
- Github AdnRme サンプルアプリケーション
- <https://github.com/jeremytammik/AdnRme>

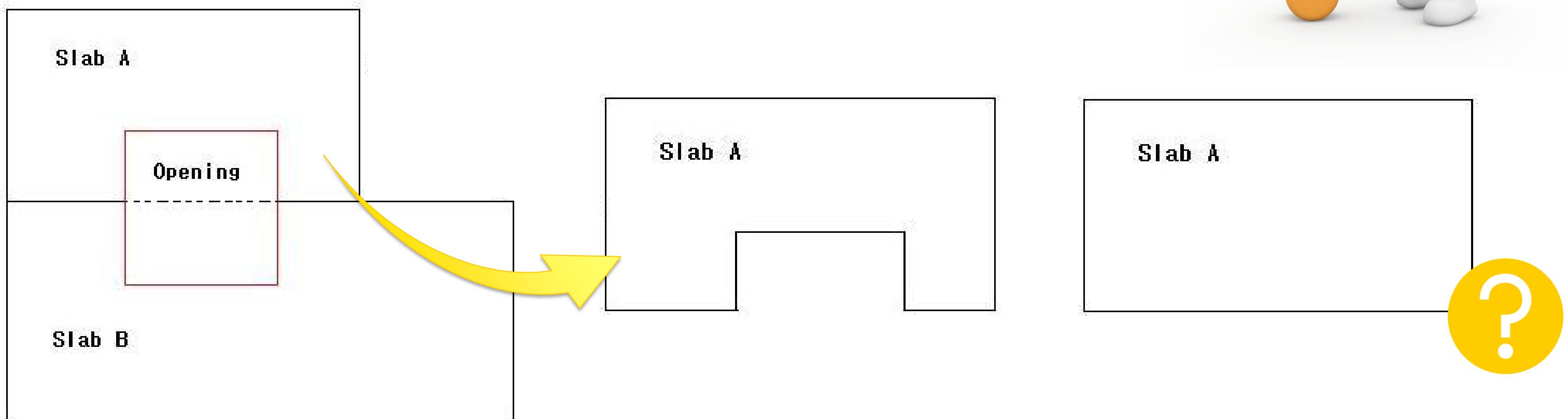


一時トランザクション



シナリオ

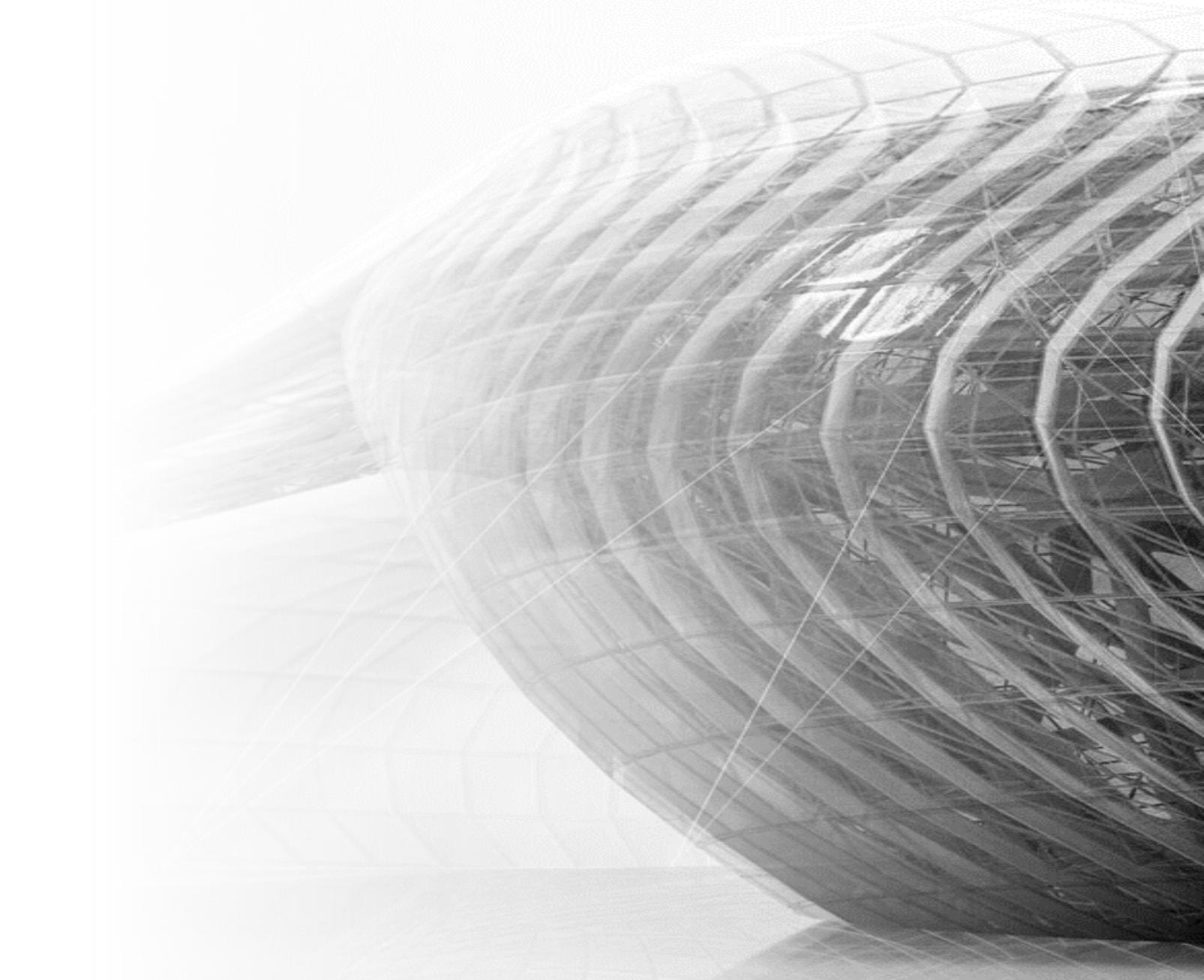
- スラブに開口部が存在している場合、単純にスラブのジオメトリを取得すると、一部が切断されたジオメトリが取得できます。
- 元の切断される前のスラブの形状を取得するには、どうすればよいでしょうか。



一時トランザクション

- トランザクションは、必ずしもコミットする必要はありません。
 - ロールバックはトランザクションの処理中にエラーがある場合に便利ですが、それ以外にも活用できます。
1. トランザクションのコンストラクタを使用してトランザクションをインスタンス化し、名前を割り当てます。
 2. Transaction.Start() を呼び出します。
 3. ドキュメントには、一時的な変更(要素の修正、削除、または作成)を加えます。
 4. Document.Regenerate() を呼び出します。
 5. 目的のジオメトリとプロパティを抽出します。
 6. Transaction.Rollback() を呼び出して、ドキュメントを元の状態に復元します。

単位と精度



Revit の単位

- Revit には **7 つの基本量**があり、それぞれに内部単位を持っています。
- これらの内部単位を次の表に示します。
- Revit は**長さをフィート単位で、他の基本量をメートル単位で格納する**ため、長さに関する派生単位は、インチ/フィート単位とメートル単位の両方に基づく非標準の単位になります。

基本単位	Revit の単位	単位系
長さ	フィート(ft)	インチ/フィート単位
角度	ラジアン	メートル単位
マス	キログラム(kg)	メートル単位
時刻	秒(s)	メートル単位
電流	アンペア(A)	メートル単位
温度	ケルビン(K)	メートル単位
光度	カンデラ(cd)	メートル単位

UnitUtils クラス

- UnitUtils クラスを使用すると、平方フィートから平方メートルなど、ある単位タイプから別の単位タイプに値を変換できます。

Convert()	値の表示単位(DisplayUnitType)を、別の表示単位に変換します。
ConvertFromInternalUnits()	Revit の内部単位の値を、指定の表示単位の値に変換します。
ConvertToInternalUnits()	指定の表示単位の値を、Revit の内部単位の値に変換します。

```
void SetTopOffset(Wall wall, double dOffsetInches)
{
    // convert user-defined offset value to feet from inches prior to setting
    double dOffsetFeet = UnitUtils.Convert(dOffsetInches,
        DisplayUnitType.DUT_DECIMAL_INCHES,
        DisplayUnitType.DUT_DECIMAL_FEET);

    Parameter paramTopOffset = wall.get_Parameter(BuiltInParameter.WALL_TOP_OFFSET);
    paramTopOffset.Set(dOffsetFeet);
}
```

UnitFormatUtils クラス

- UnitFormatUtils クラスは、データを書式設定したり、**書式設定された単位データを解析**できます。
 - Format()メソッドを使用すると、[書式]オプションに基づき、値を文字列に書式設定できます。
 - TryParse()メソッドは、指定された単位タイプの Revit の内部単位を使用して、単位などの書式設定された文字列を、可能であれば値に解析します。

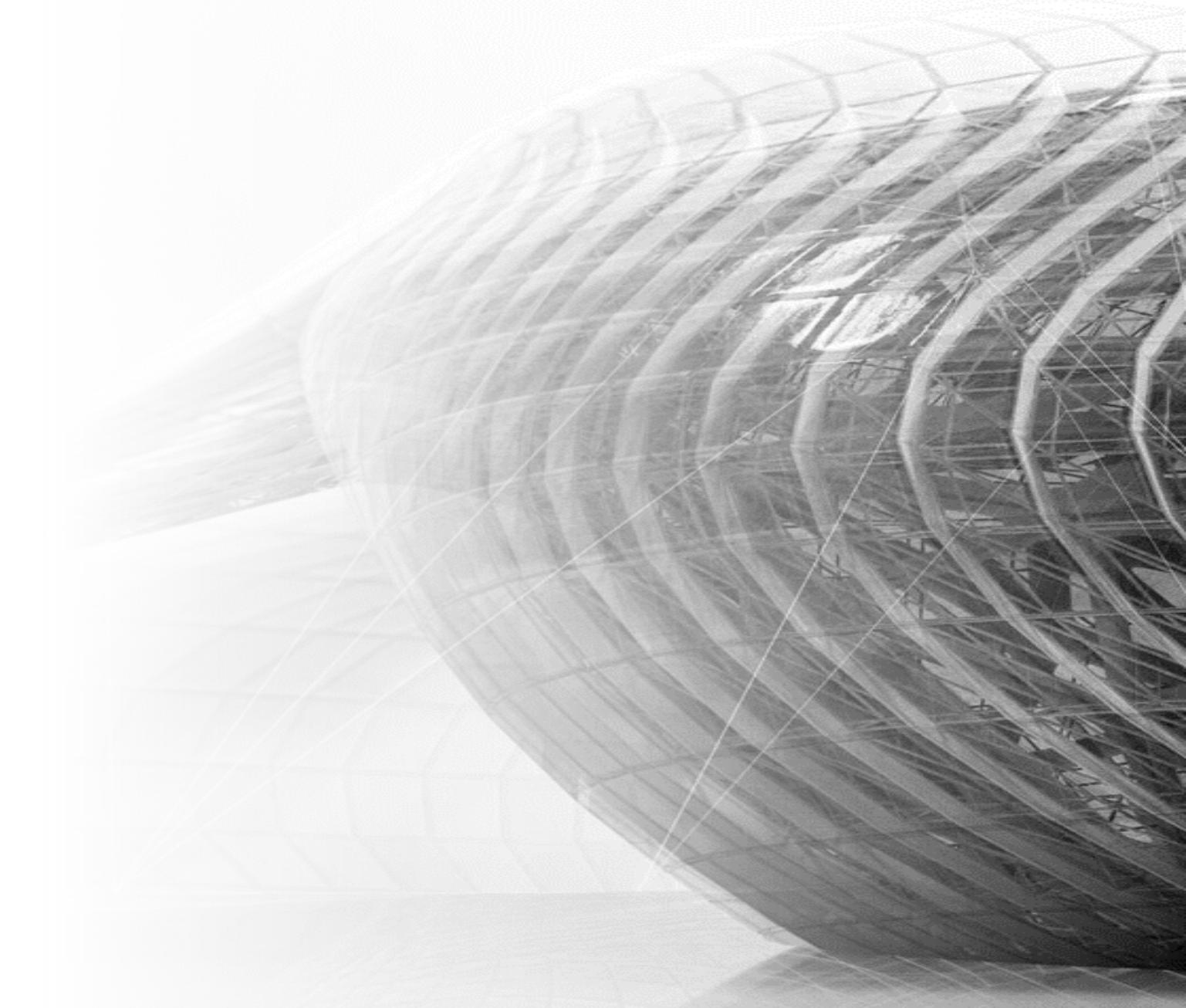
```
double GetLengthInput(Document document, String userInputLength)
{
    double dParsedLength = 0;
    Units units = document.GetUnits();
    // try to parse a user entered string (i.e. 100 mm, 1'6")
    bool parsed = UnitFormatUtils.TryParse(units, UnitType.UT_Length, userInputLength, out dParsedLength);
    if (parsed == true)
    {
        string msg = string.Format("User Input: {0}`r`nParsed value: {1}", userInputLength, dParsedLength);
        TaskDialog.Show("Parsed Data", msg);
    }

    return dParsedLength;
}
```

精度

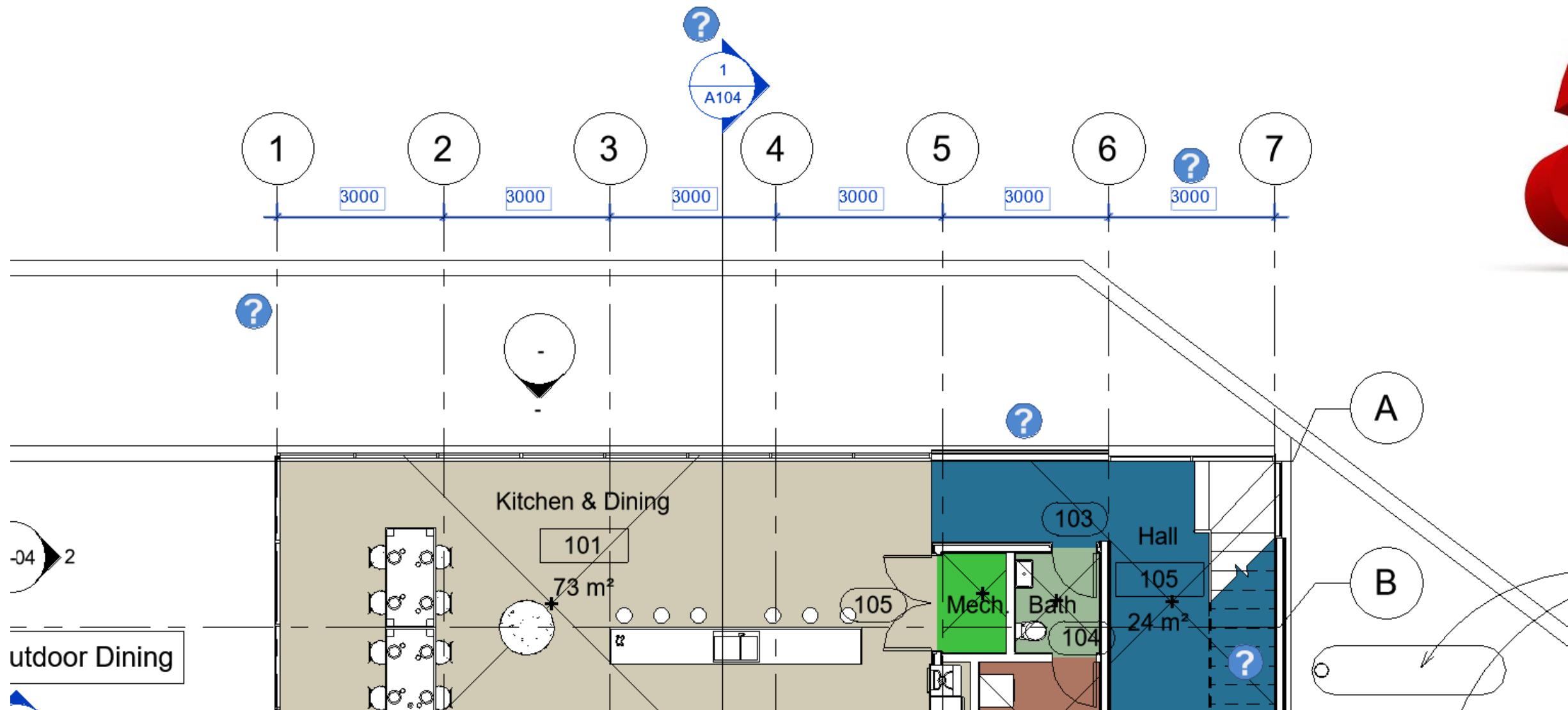
- `double` 型(倍精度浮動小数点実数)の計算処理は、誤差が生じる可能性があるため、`UnitUtils.convert()` メソッドでは保証していない。
- `UnitUtils.convert()` メソッドでは、Revit は内部で $1\text{mm} = 0.00328 \text{ feet}$ を換算値としている点で、 0.00328084 での計算結果とは異なります。
- Revit では長さの最小値は $1/32"$ (0.79 mm)として設計されています。そのため、モデリング、寸法、要素の修正などの処理で、これより小さい値で処理しても想定されている結果が得られない可能性があります。

通芯とレベル



通芯の線分

- 通芯の線分を取得する **Grid.Curve** プロパティは読み取り専用。
- ではどうやって、通芯の長さを変更できるでしょうか。



DatumPlane クラス

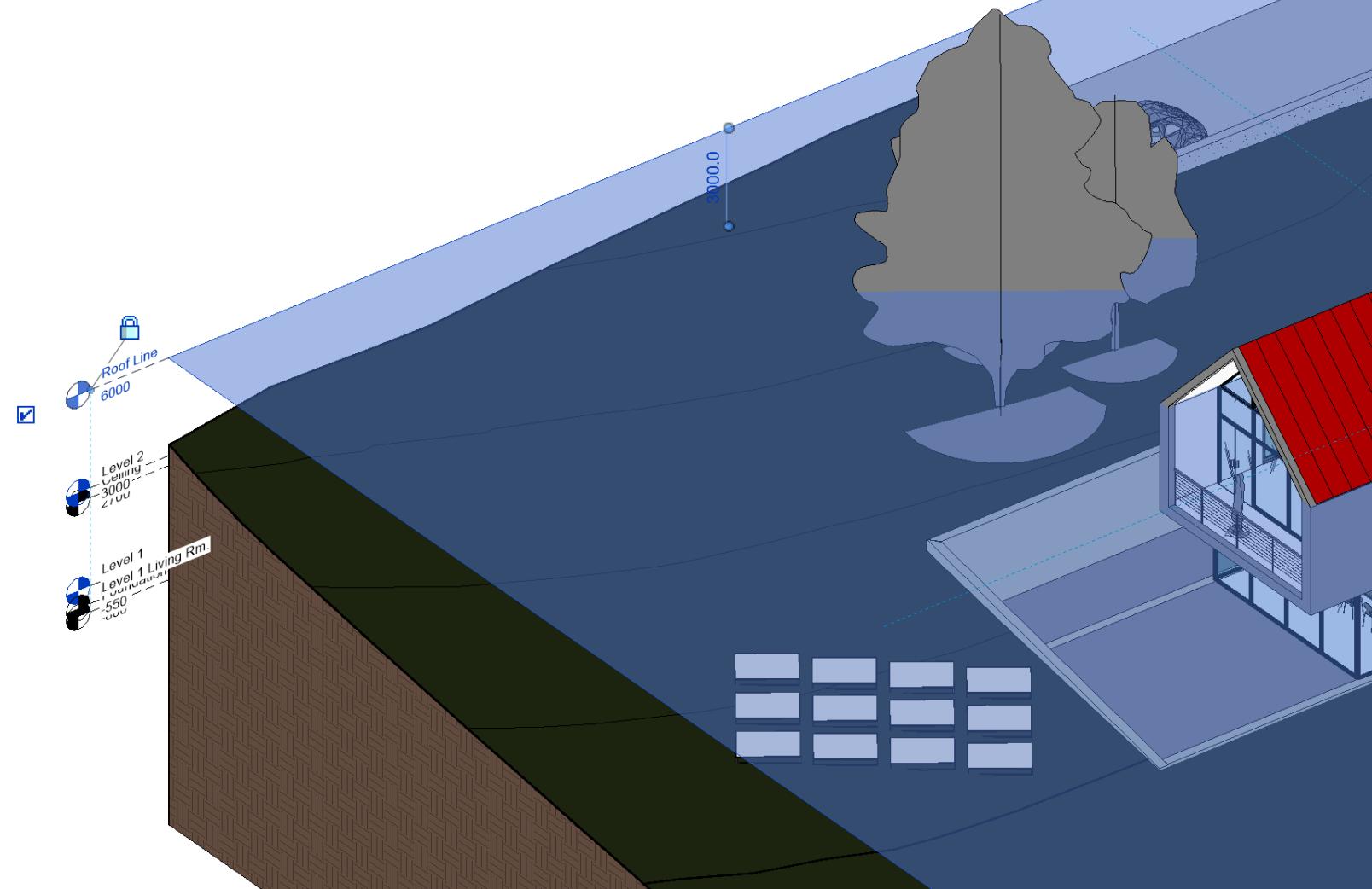
- Grid クラス、Level クラスは、DatumPlane クラスから派生。DatumPlane クラスは Element クラスから派生。
- DatumPlane.GetCurvesInView() メソッドを利用して通芯の線分を取得すると、DatumPlane. SetCurvesInView() で修正した線分を再設定することができます。
- 引数の DatumExtentType で、3D範囲（すべてのビュー）か 2D 範囲（特定のビュー）かを指定できます。

```
public IList<Curve> GetCurvesInView(  
    DatumExtentType extentMode,  
    View view  
)
```

```
public void SetCurveInView(  
    DatumExtentType extentMode,  
    View view,  
    Curve curve  
)
```

レベルの線分

- レベルは**有限の水平面**です。
- 立面図ビューで表示した時には線分として表示されておりますが、内部的には水平面として処理されています。
- そのため、レベルのXYZ 座標も、**レベルの面を上から見下ろした状態**となります。



```
Level level = elem as Level;
```

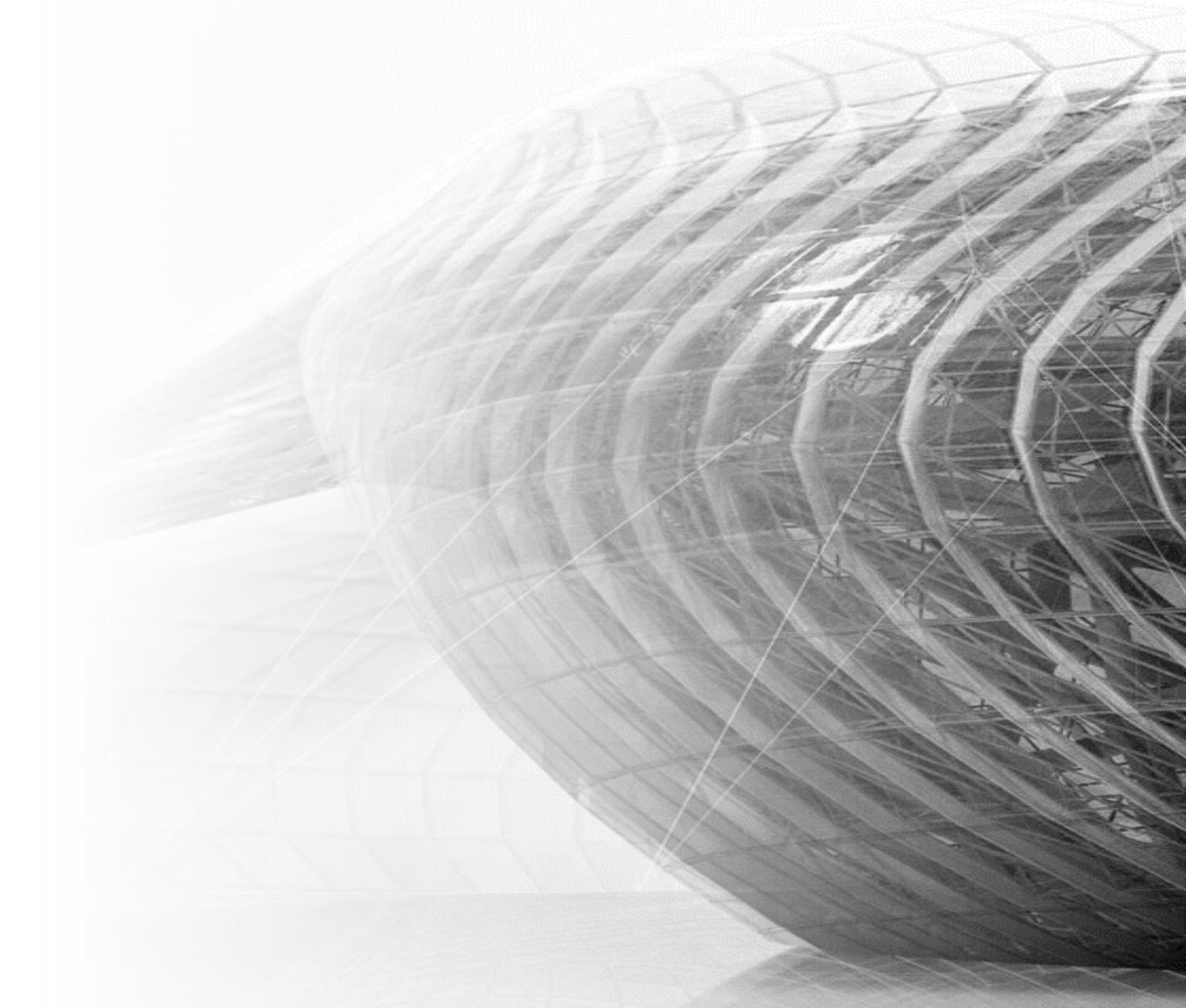
```
IList<Curve> curves = level.GetCurvesInView(DatumExtentType.ViewSpecific, doc.ActiveView);  
Curve curve = curves[0];
```

```
XYZ end0 = new XYZ(curve.GetEndPoint(0).X, curve.GetEndPoint(0).Y - 10, curve.GetEndPoint(0).Z);  
XYZ end1 = new XYZ(curve.GetEndPoint(1).X, curve.GetEndPoint(1).Y + 10, curve.GetEndPoint(1).Z);
```

```
Curve newCurve = Line.CreateBound(end0, end1);
```

```
level.SetCurveInView(DatumExtentType.ViewSpecific, doc.ActiveView, newCurve);
```

参照と寸法



シナリオ

- ユーザーが、ファミリエディタでファミリを作成するときに、ある参照面に「任意の場所 A」という名前を付けたいと考えています。
- そして、このファミリをプロジェクトにロードして、そのファミリインスタンスを配置し、プロジェクト上で、参照面「任意の場所 A」を見つけて、その場所のデータを別の処理に使いたいと考えています。
- ファミリエディタ (Document.EditFamily()) を使用せずに、ファミリで定義されている名前付きの参照を取得することは可能ですか？



ファミリで定義されている名前付きの参照へのアクセス

- Revit 2018 で、FamilyInstance オブジェクトに追加された新しいメソッドで、ファミリの参照面と参照線に対応する参照に簡単にアクセスできるようになりました。
- FamilyInstanceReferenceType 列挙型のオプションを引数として使用して、「強」または「弱」参照、または3つの座標方向の特定の位置にある参照を取得します。
 - FamilyInstance.GetReferences()
 - FamilyInstance.GetReferenceByName()
 - FamilyInstance.GetReferenceType()
 - FamilyInstance.GetReferenceName()

通芯に寸法を作成する

- あるデベロッパー様が通芯の間に寸法を作成しようとしましたが、Invalid Number of references という例外エラーが発生しました。どうすれば、寸法を作成できるでしょうか。

```
public static void createGridDimension(View view, Grid grid1, Grid grid2)
{
    Curve curve1 = grid1.Curve;
    Curve curve2 = grid2.Curve;

    Line line = Line.CreateBound(curve1.GetEndPoint(0), curve2.GetEndPoint(0));

    ReferenceArray references = new ReferenceArray();
    references.Append(curve1.Reference);
    references.Append(curve2.Reference);

    view.Document.Create.NewDimension(view, line, references);
}
```



Exception User-Unhandled

Autodesk.Revit.Exceptions.InvalidOperationException: 'Invalid number of references.'

[View Details](#) | [Copy Details](#)

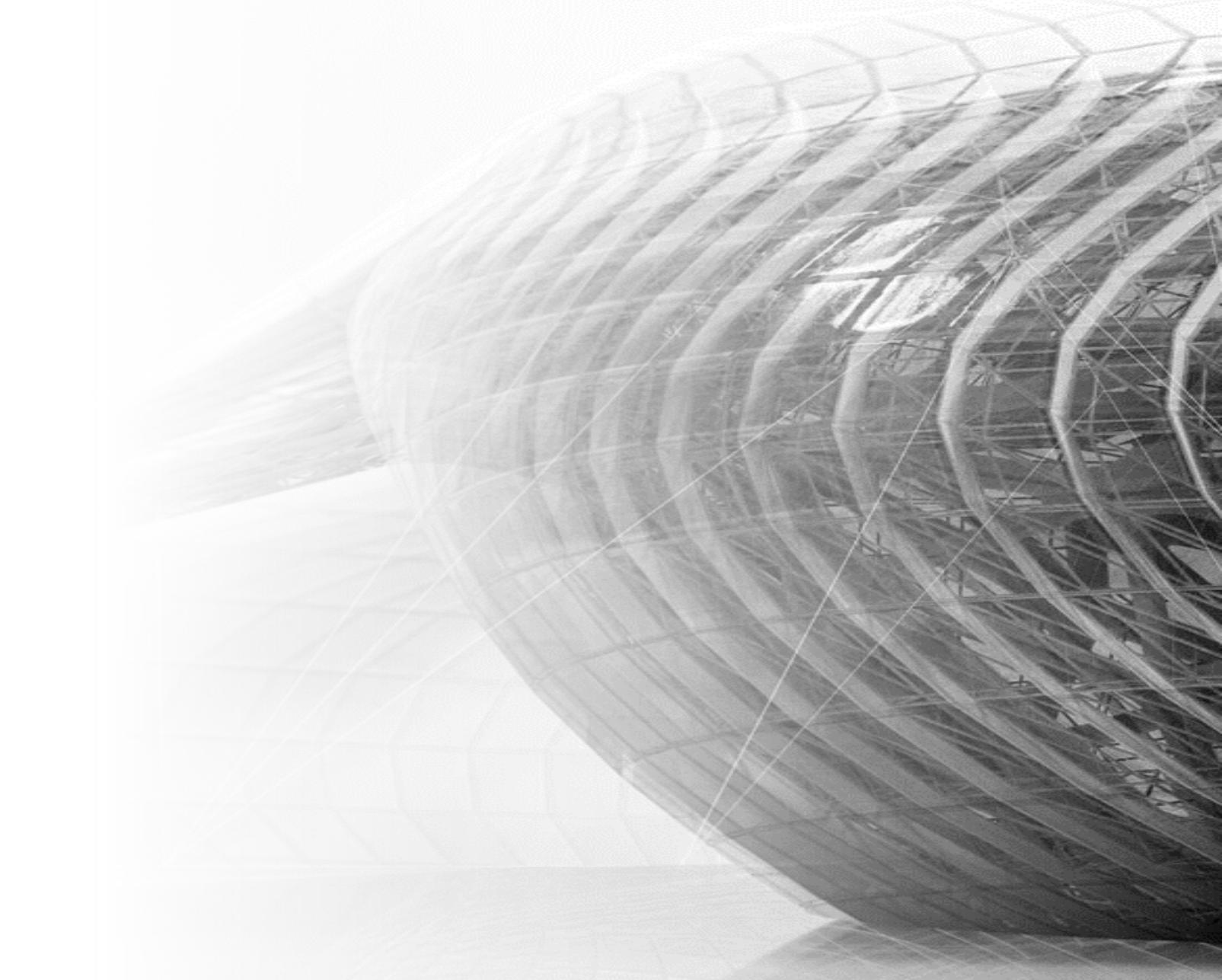
► [Exception Settings](#)

通芯の参照

- 通芯は、レベルと同じ DatumPlane クラスの派生クラスです。
- つまり、通芯は、3次元のサーフェスです。
- 通芯の線分 Curve オブジェクトの参照から寸法を作成することはできません。
- 解決策は、シンプルに **New Reference(Grid)** を実行して参照オブジェクトを作成することです。

```
Reference GetGridRef(Document doc)
{
    ElementId idGrid = new ElementId(397028);
    Element eGrid = doc.GetElement(idGrid);
    return new Reference(eGrid);
}
```

スタイルとマテリアル



オブジェクトスタイル

- プロジェクト内のモデル要素、注釈要素、および読み込まれたオブジェクトの各種カテゴリやサブカテゴリについて、線分の太さ、線分の色、線種パターン、マテリアルを指定することができます。

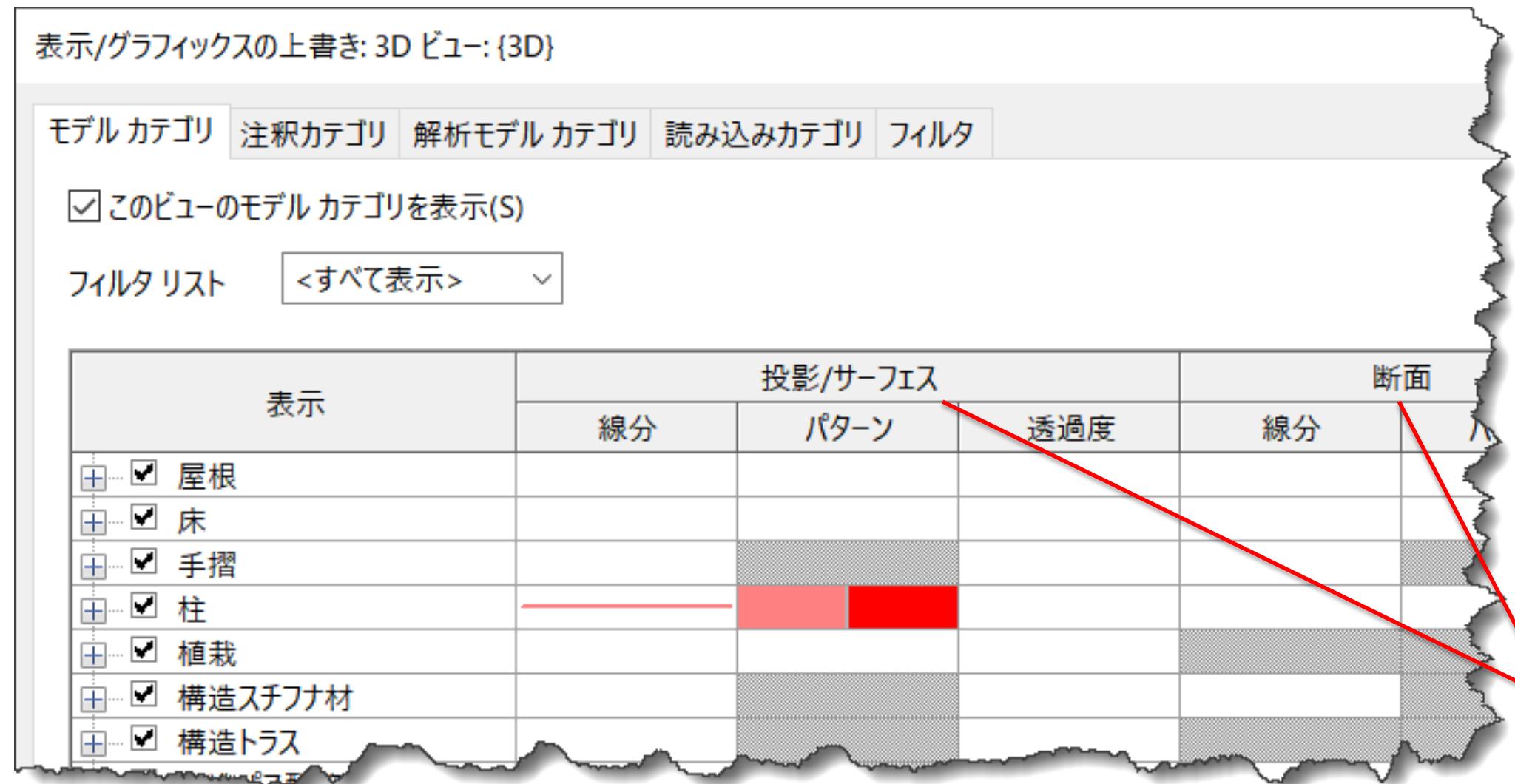


Category クラス メンバ	
線の太さ	Get/SetLineWeight()
線の色	LineColor
線種パターン	Get/SetLinePatternId()
マテリアル	Material

LinePatternElement クラスの Create() メソッドで新規作成もサポート

表示/グラフィックスの上書き

- カテゴリに既に適用されている上書きを表示することができます。
- あるカテゴリのグラフィック表示が上書きされていると、セルにはそのグラフィックのプレビューが表示されます。



View クラス メンバ	
カテゴリ	Get/SetCategoryOverrides()
フィルタ	Get/SetFilterOverrides()
ビュー固有 の要素	Get/SetElementOverrides()

OverrideGraphicSettings オブジェクトで各種パラメータを設定

線種の作成・修正

- 線種は、線分(Line カテゴリ)のサブカテゴリです。
- BuiltInCategory.OST_Lines のサブカテゴリを作成し、線の太さ、線の色、線種パターンを設定します。



<https://thebuildingcoder.typepad.com/blog/2016/10/how-to-create-a-new-line-style.html>

サンプルコード

```
FilteredElementCollector collector = new FilteredElementCollector(doc).OfClass(typeof(LinePatternElement));

LinePatternElement linePatternElem = collector
    .Cast<LinePatternElement>()
    .First<LinePatternElement>(linePattern
        => linePattern.Name == "Long dash");

Categories categories = doc.Settings.Categories;
Category lineCat = categories.get_Item(BuiltInCategory.OST_Lines);

using (Transaction t = new Transaction(doc))
{
    t.Start("Create LineStyle");

    Category newLineStyleCat = categories.NewSubcategory(lineCat, "New LineStyle");

    doc.Regenerate();

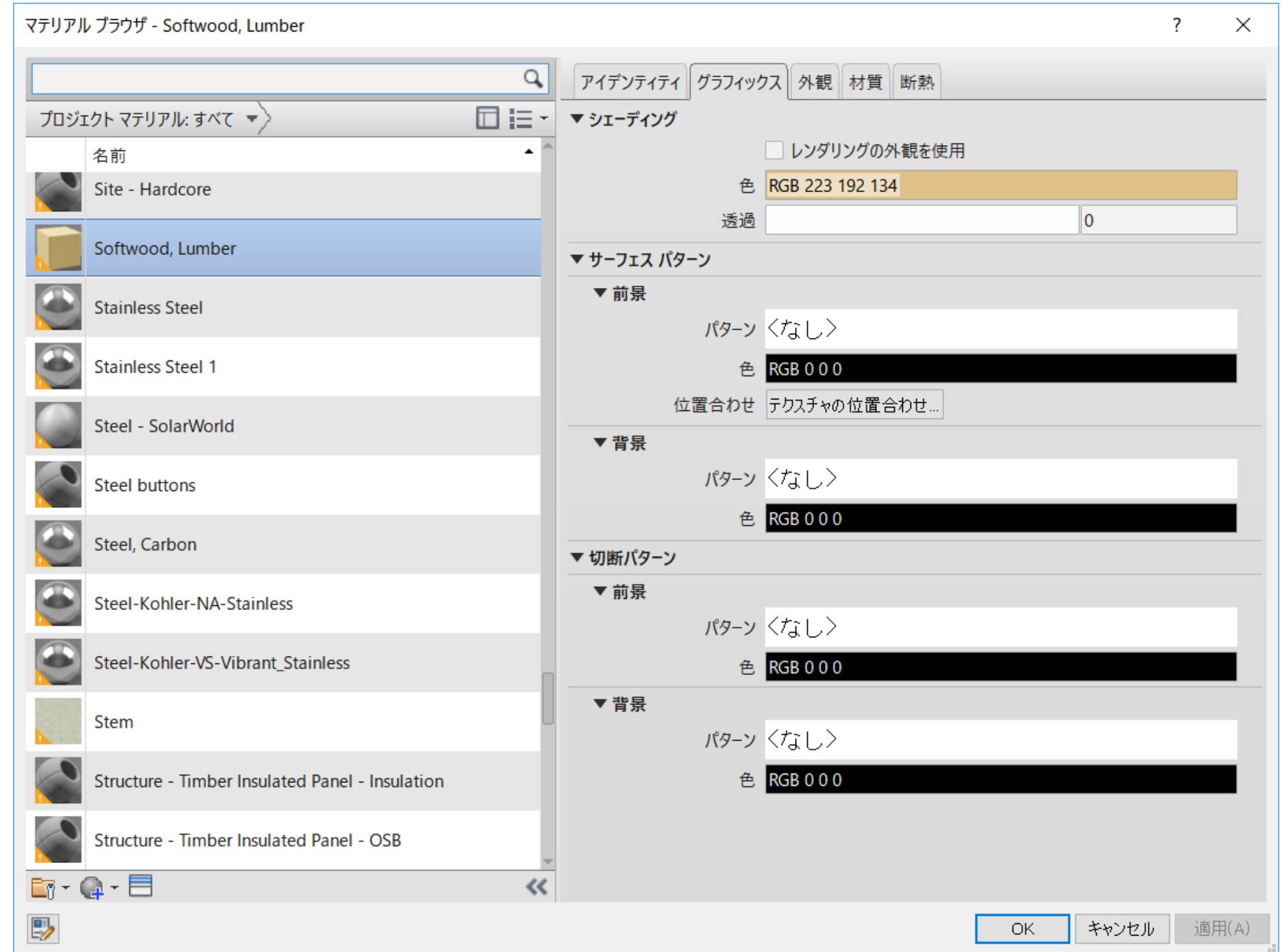
    newLineStyleCat.SetLineWeight(8, GraphicsStyleType.Projection);
    newLineStyleCat.LineColor = new Color(0xFF, 0x00, 0x00);
    newLineStyleCat.SetLinePatternId(linePatternElem.Id, GraphicsStyleType.Projection);

    t.Commit();
}
```

マテリアル

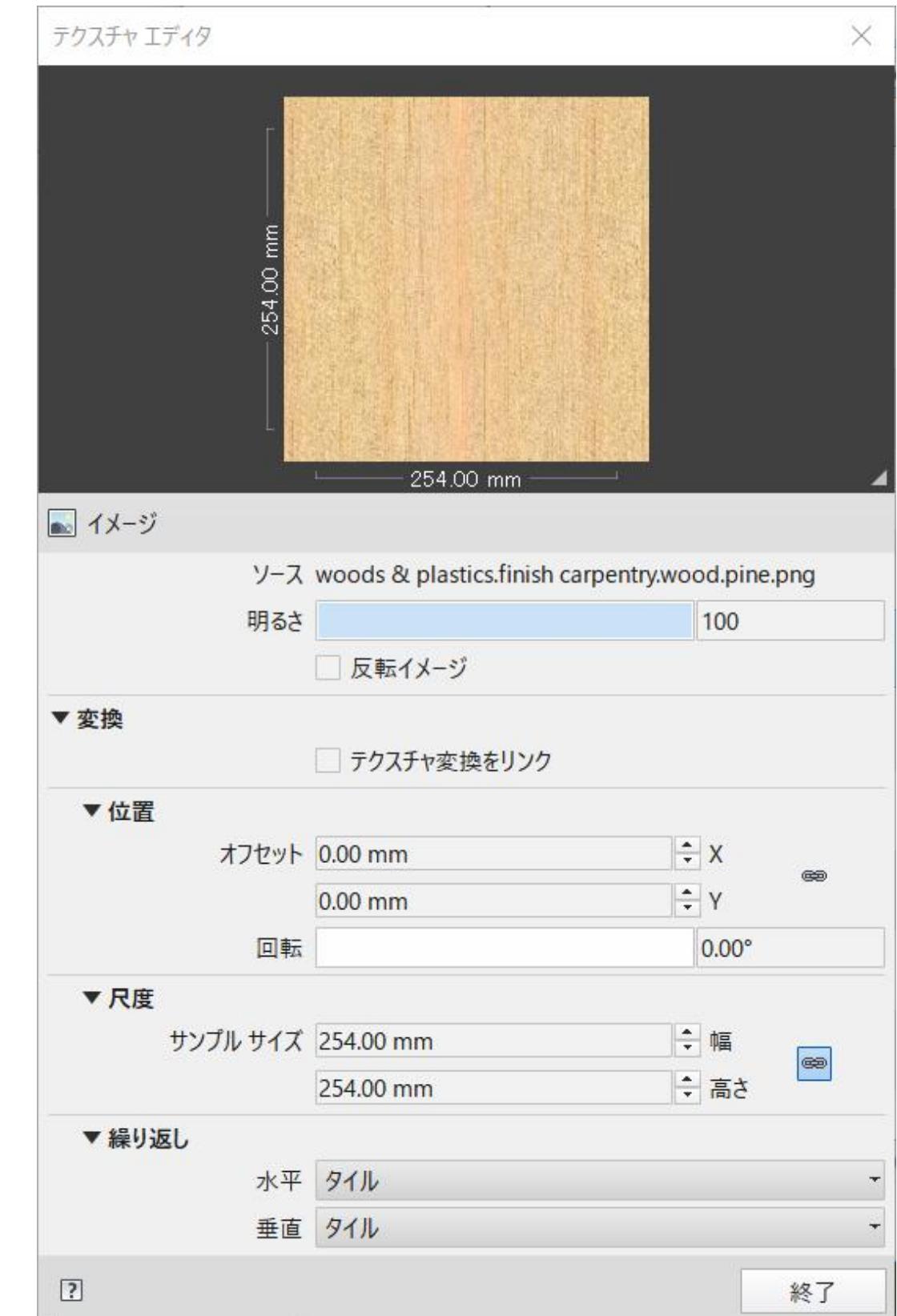
- Material クラス
- 各プロパティの情報にアクセス
- 各アセット要素を取得

外観アセット	AppearanceAssetId
材質アセット	StructuralAssetId
断熱アセット	ThermalAssetId



マテリアル テクスチャのファイルパスを取得したい

- 外観アセットのプロパティにアクセスするために必要な名前空間
 - Autodesk.Revit.DB.Visual 名前空間
- UnifiedBitmap クラスでサポート
 - Asset[UnifiedBitmap.UnifiedbitmapBitmap]
- デフォルトのマテリアルライブラリの場所が設定されている場合は、相対パスが返却されます。
 - C:\Program Files (x86)\Common Files\Autodesk Shared\Materials\Textures\
- もし相対パスの場合は、デフォルトのマテリアルライブラリの場所にそのファイルが存在するか確認し、存在しない場合は、Revit.ini ファイルからルートパスを取得するようなロジックを追加する必要があります。



サンプルコード

```
Material material = elem as Material;
ElementId appearanceAssetId = material.AppearanceAssetId;
AppearanceAssetElement assetElem = material.Document.GetElement(appearanceAssetId) as AppearanceAssetElement;
Asset asset = assetElem.GetRenderingAsset();

int size = asset.Size;
for (int assetIdx = 0; assetIdx < size; assetIdx++)
{
    AssetProperty aProperty = asset.Get(assetIdx);

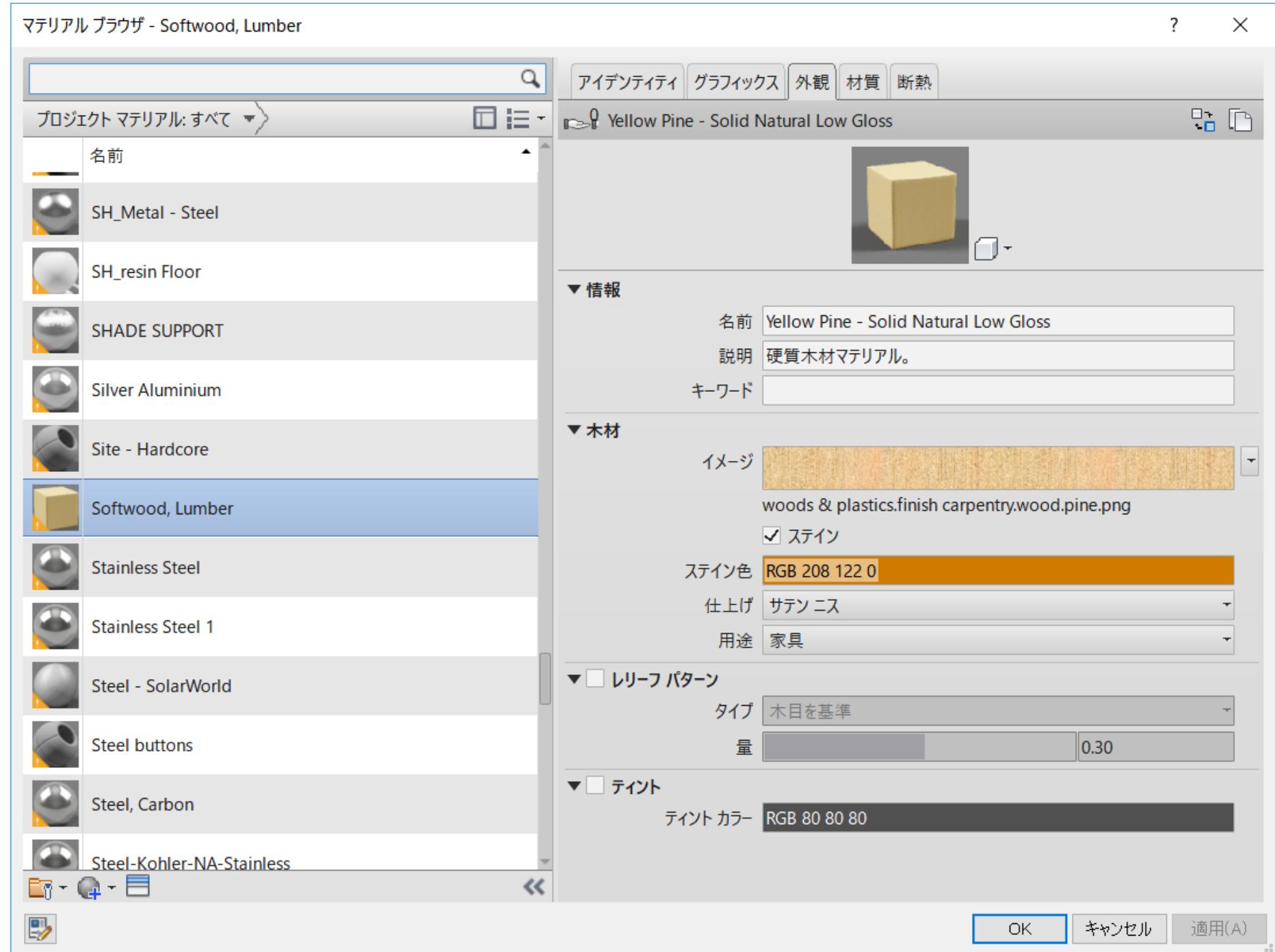
    if (aProperty.NumberOfConnectedProperties < 1)
        continue;

    Asset connectedAsset = aProperty.GetConnectedProperty(0) as Asset;

    if (connectedAsset.Name == "UnifiedBitmapSchema")
    {
        AssetPropertyString path = connectedAsset[UnifiedBitmap.UnifiedbitmapBitmap] as AssetPropertyString;
    }
}
```

マテリアルの編集

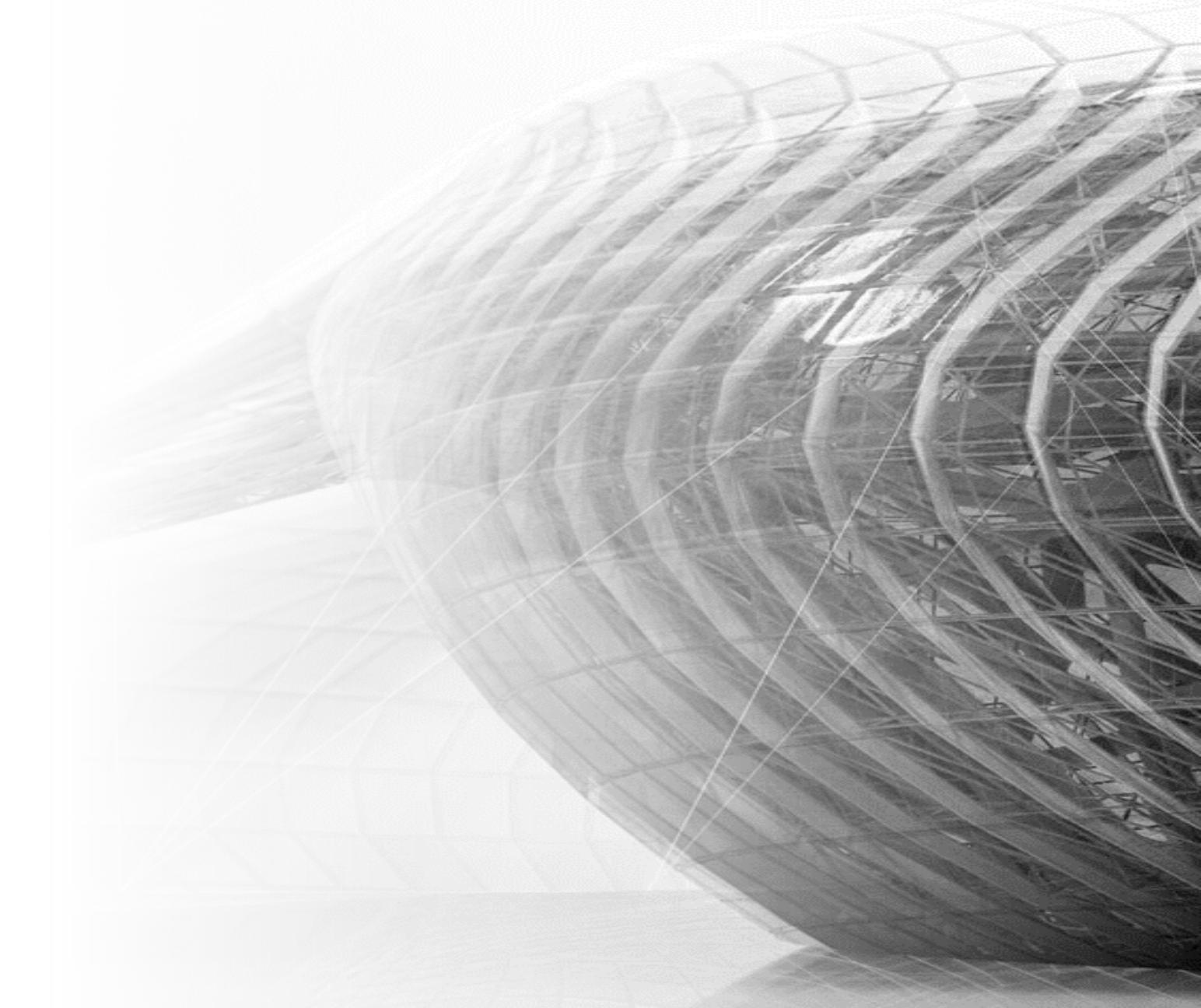
- Revit 2018.1 から、マテリアルの外観アセットに含まれているプロパティ（テクスチャも含む）の編集をサポート。
 - AppearanceAssetEditScope クラス
- [マテリアル ブラウザ]ダイアログの[外観]タブに表示されるプロパティが対象。
- 2018.1 以前のバージョンでは、プロパティの読み取りのみ。



Material API 解説コンテンツ

- Revit API 開発者用ガイド
 - http://help.autodesk.com/view/RVT/2019/JPN/?guid=Revit_API_Revit_API_Developers_Guide_Revit_Geometric_Elements_Material_html
- ブログ記事解説
 - <https://thebuildingcoder.typepad.com/blog/2017/11/modifying-material-visual-appearance.html>
- AU 2017 (New API to Modify Visual Appearance of Materials in Revit)
 - <https://www.autodesk.com/autodesk-university/class/New-API-Modify-Visual-Appearance-Materials-Revit-2017>
 - セッション動画
 - ハンドアウト
 - スライド

ジオメトリ



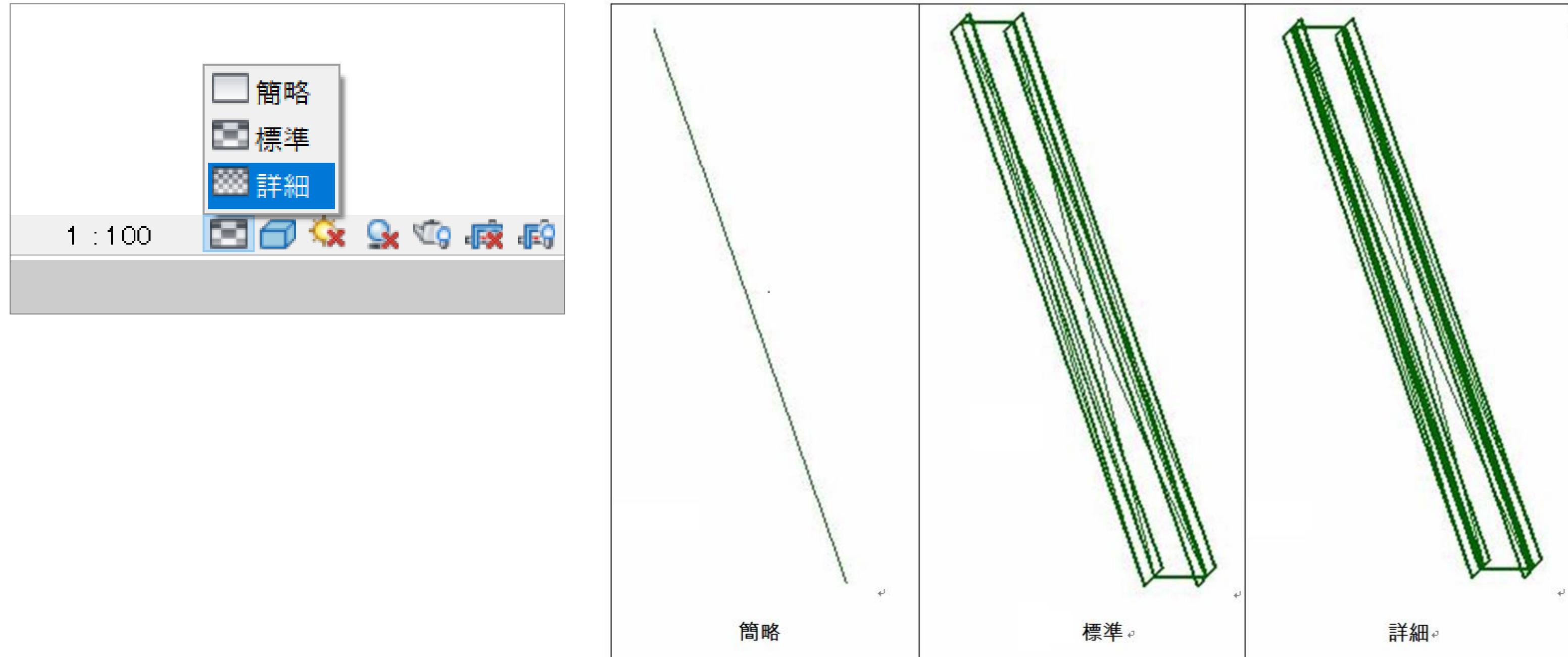
オプションの設定

- Options クラスのプロパティを設定することで、ジオメトリの取得結果が変わります。

ComputeReferences	ジオメトリ情報を取得する際に、ジオメトリ参照を計算するかどうかを示します。 ※既定は <code>False</code> なので、このプロパティが <code>True</code> に設定されていない場合は、参照にはアクセスできません。
IncludeNonVisibleObjects	既定のビューに表示されないジオメトリ オブジェクトも返すかどうかを示します。
View	特定のビューからジオメトリ情報を取得します。
DetailLevel	優先される詳細レベルを示します。既定は標準です。

梁からジオメトリを取得した場合

- DetailLevel プロパティの設定に応じて、異なるジオメトリが返されます。



インスタンス ジオメトリとシンボル ジオメトリ

- ファミリ インスタンスからジオメトリを取得すると、多くの場合、GeometryInstance オブジェクトが返されます。

GeometryInstance.GetSymbolGeometry()

ファミリ インスタンスの元となっている[ファミリ タイプのジオメトリ](#)を取得。
ファミリのローカル座標系で表されます。
方向や配置場所を気にしない場合に使用します。

GeometryInstance.GetInstanceGeometry()

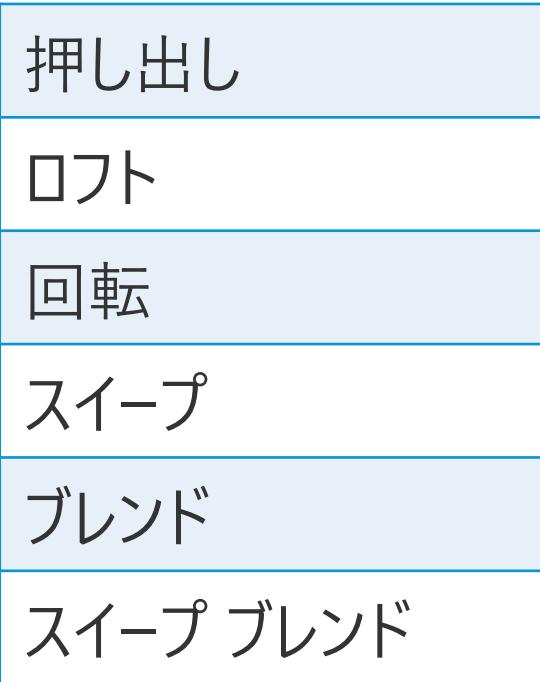
プロジェクト上に配置されている[ファミリ インスタンスのジオメトリ](#)を取得。
プロジェクト座標系で表されます。

FamilyInstance.GetOriginalGeometry()

結合、切り取り、切欠きなど、Revit で行われるその後処理によって変更される前の[インスタンスの元のジオメトリ](#)を取得。

ソリッドと面の作成

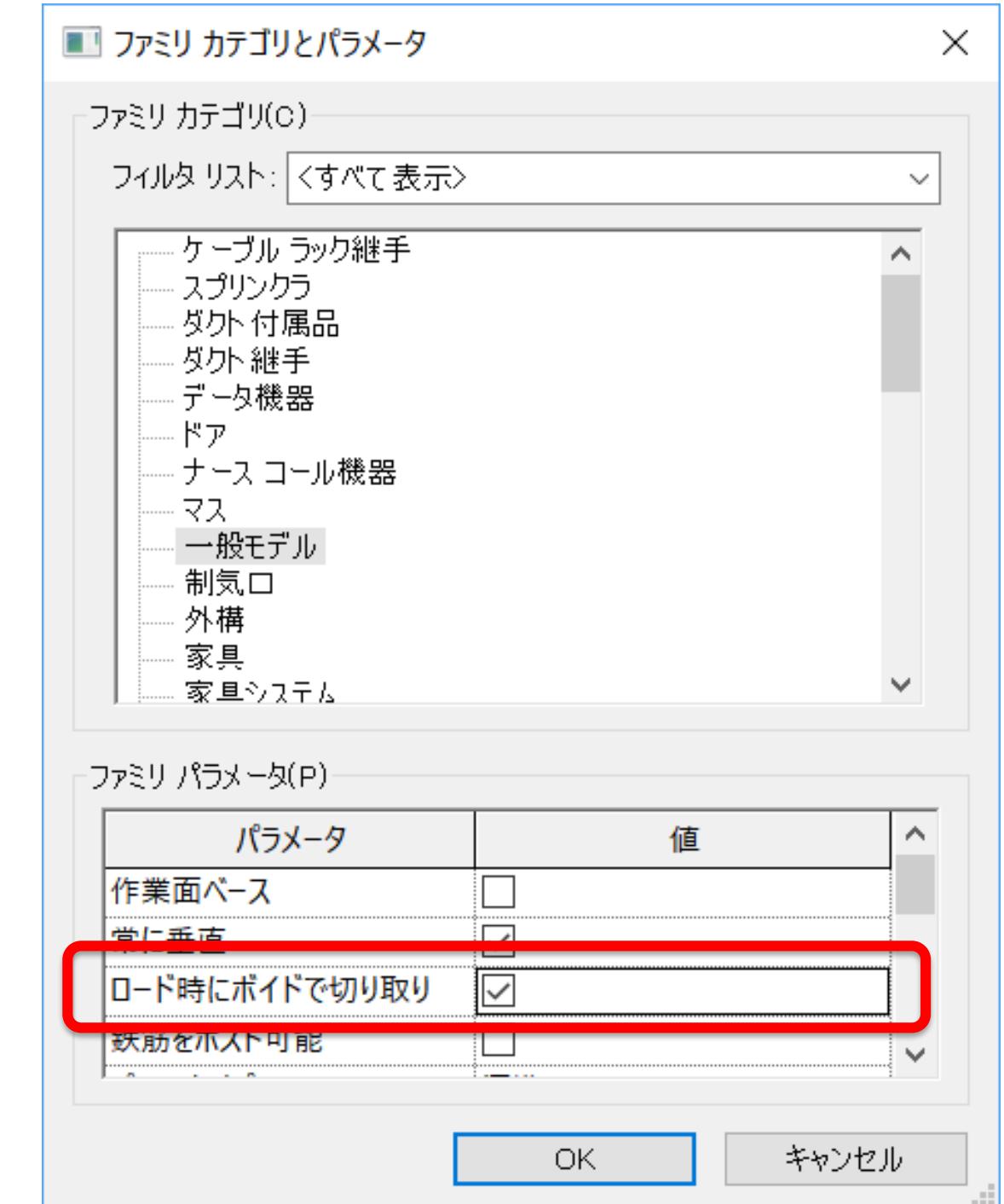
- GeometryCreationUtilities クラスは、基本的なソリッド形状が構築できるユーティリティ クラスです。



- 結果として得られるジオメトリは、任意の要素の一部としてプロジェクト ドキュメントには追加されません。
- 作成されたソリッドは、API 関数への入力として使用することができます。
 - 交点で 3D 要素を検索するための入力ソリッドとして
 - ブール演算への 1 つまたは複数の入力として
 - ジオメトリ計算の一部として

ブール演算ユーティリティクラス

- BooleanOperationsUtils クラス
 - 非要素のジオメトリを作成します。
 - 任意の要素の一部としてプロジェクト ドキュメントには追加されません。
- InstanceVoidCutUtils クラス
 - ファミリインスタンスをアタッチされていないボイドを使用して要素を切断します。
 - 最初にファミリ定義の一部として穴を含めることができない場合、メンバー内の個々の穴をドリルするような微調整作業には、より便利です。
- SolidSolidCutUtils クラス
 - ファミリエディタで使用することを想定。
 - 制約があるため、詳細は API リファレンスを確認。



ジオメトリ ユーティリティ クラス

HostObjectUtils

ホスト要素の特定の面を取得するためのショートカットメソッド。

- HostObjectUtils.GetSideFaces()
- HostObjectUtils.GetTopFaces(), HostObjectUtils.GetBottomFaces()

SolidUtils

ソリッドを複製、分割、変換するメソッド。

- SolidUtils.Clone()
- SolidUtils.SplitVolumes()
- SolidUtils.CreateTransformed()

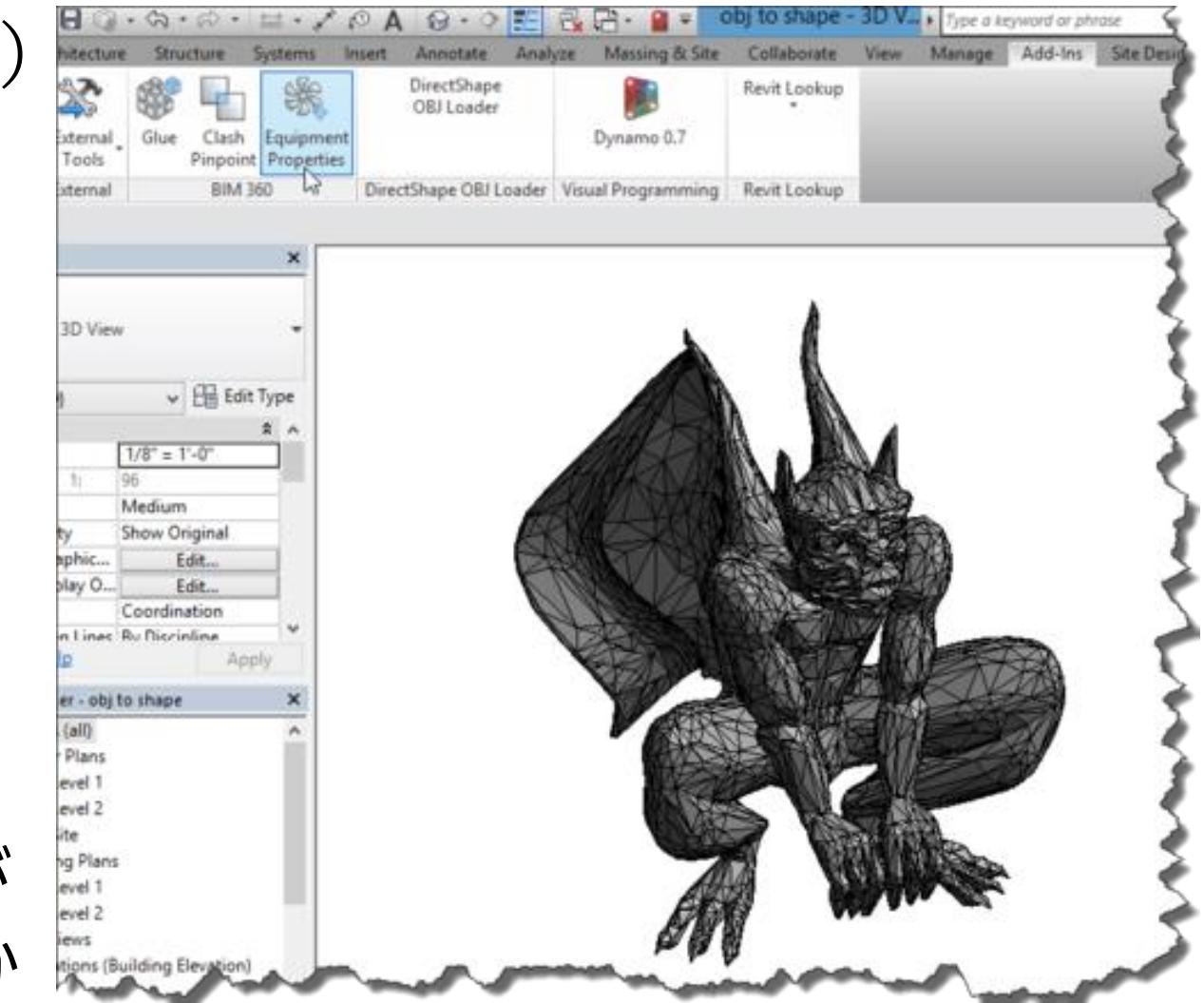
JoinGeometryUtils

要素の結合/結合解除のためのメソッドや、要素の結合順序を管理するためのメソッド。

これらのユーティリティはファミリドキュメントでは使用できません。

DirectShape

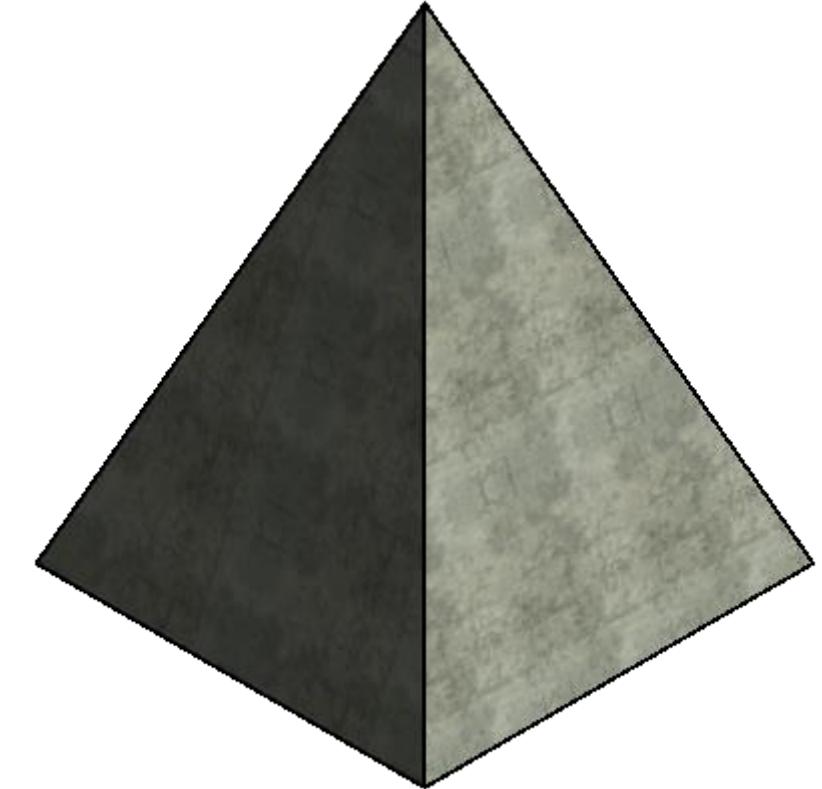
- DirectShape は、任意のジオメトリ要素（ソリッドやメッシュなど）を格納するための入れ物です。ビューに作図され、ドキュメントにも保存することができます。
 - IFCなどの他のデータフォーマットから形状をインポートする
 - プロジェクトのビュー上に、直接、ジオメトリデータを作図する
 - ファミリのジオメトリ要素を作成する
- Wall カテゴリなどの最上位の Model カテゴリを割り当てることができます。このカテゴリに応じて、利用可能なパラメータと、いくつかの制限された動作がオブジェクトに付与されます。
- DirectShape 要素にサブカテゴリを割り当てるることはできません。
- ファミリと同じように詳細に振る舞いやパラメータを定義することはできません。



DirectShape のジオメトリを作成

- DirectShape では次のジオメトリ タイプを入力として使用できます。

- ソリッド(閉じたシェルまたは開いたシェル)
- メッシュ
- 曲線
- 点



- GeometryCreationUtilities や ShapeBuilder クラスのサブクラスでジオメトリを作成

GeometryCreationUtilities 基本的なソリッド形状が構築できるユーティリティクラス。

TessellatedShapeBuilder 接続された平面ファセットを境界としたソリッド、シェル、ポリメッシュを作成。

BRepBuilder サーフェス、エッジ、エッジの境界ループの入力の結果として、Revit の境界表現ジオメトリ(ソリッド、開いたシェルなど)を構築する。

ShapeImporter 外部の形式(SAT や Rhino など)で保存されたジオメトリを Revit 用に変換。

DirectShape Tips

- バリデーション メソッドを活用。

IsValidCategoryId()	カテゴリが最上位のカテゴリか検証。
IsValidGeometry()	ジオメトリの検証と、アサインされたカテゴリに適切な形状表現か確認。
IsValidShape()	IsValidGeometry() の拡張メソッド。
IsValidTypeId()	DirectShapeType に対応している要素 ID か確認。

- さらに応用として、
 - DirectShapeLibrary を利用すれば、事前に作成されたジオメトリの定義を参照して DirectShape を作成することができます。
 - ジオメトリ オブジェクトの配列として定義を追加
 - DirectShapeType としてシェイプ定義を追加

パフォーマンス



ファミリをロードするか、ファミリ タイプをロードするか

- ファミリのロード処理に時間がかかる場合にお試しください。
- Document クラスには、LoadFamily() メソッドと LoadFamilySymbol() メソッドがあります。

LoadFamily()	ファミリ全体とそのすべてのファミリ タイプをプロジェクトにロードします。
LoadFamilySymbol()	指定されたファミリ タイプのみをファミリ ファイルからプロジェクトにロードします。

- 注: アプリケーションのパフォーマンスを向上し、メモリ使用量を抑えるには、可能であれば、ファミリ オブジェクト全体ではなく特定の FamilySymbols をロードします。

ファミリ インスタンスの一括作成

- ファミリ インスタンスの作成処理に時間がかかる場合にお試しください。
- Document.NewFamilyInstances2() メソッドで、一度に複数のファミリ インスタンスを作成。
- メソッドの引数は、 FamilyInstanceCreationData オブジェクトのリストです。
- FamilyInstanceCreationData オブジェクトは、ファミリ インスタンスの作成に必要な引数をラップするオブジェクト。

API オブジェクトのメモリ解放

- Revitは、API アプリケーション（外部コマンドなど）から、コマンド制御が Revit に返されるたびに、API オブジェクトを自動的にページします。
- API アプリケーションが1つのコマンドで多くの API オブジェクトを使用している場合、削除された API オブジェクトをメモリから解放するために、ページ処理を明示的に呼び出すことで、パフォーマンスが改善する可能性があります。
- `Application.PurgeReleasedAPIObjects()` メソッド

未使用の項目を削除

- UI の[未使用の項目を削除]コマンドに対応する API は未サポート。
- パフォーマンス アドバイザ
 - ドキュメントを解析し、パフォーマンスの低下につながる可能性のある要素や設定についてユーザに警告を与えることができます。
 - 一連の規則を実行し、その結果を標準の[警告を一覧表示]ダイアログに表示します。
 - http://help.autodesk.com/view/RVT/2019/JPN/?guid=Revit_API_Revit_API_Developers_Guide_Advanced_Topics_Performance_Adviser_html
- PerformanceAdviserRuleId: 規則ID
 - プロジェクトが未使用のファミリとタイプを保持しているか確認する規則
 - GUID("e8c63650-70b7-435a-9010-ec97660c1bda")
 - <https://thebuildingcoder.typepad.com/blog/2018/08/purge-unused-using-performance-adviser.html>

API の調べ方



Step 1

- Revit API 開発者用ガイドで日本語で解説を読む。
 - http://help.autodesk.com/view/RVT/2019/JPN/?guid=Revit_API_Revit_API_Developers_Guide_html

The screenshot shows the navigation menu for the Revit API Developers Guide. The menu is organized into several sections, each with a plus sign icon indicating expandable content.

- Revit 開発者用ガイド
 - Revit API 開発者用ガイド
 - はじめに
 - + スタートアップ ガイド
 - + アドインの統合
 - + アプリケーションとドキュメント
 - + 要素の基本
 - Revit 要素との基本的なやりとり
 - + フィルタリング
 - + 選択
 - + パラメータ
 - + コレクション
 - + 要素を編集する
 - + ビュー
 - + トランザクション
 - Revit のジオメトリ要素
 - + 壁、床、天井、屋根、開口部
 - + ファミリ インスタンス
 - + ファミリ ドキュメント
 - + コンセプト デザイン
 - + 基準面および情報要素
 - + 注釈要素
 - + ジオメトリ
 - + スケッチ
 - + マテリアル
 - + 階段と手すり
 - サーフェス
 - DirectShape
 - サブ要素
 - 専門分野固有の機能
 - 建築設計
 - 部屋
 - 構造エンジニアリング
 - + 構造モデル要素
 - 解析モデル
 - 荷重
 - 解析リンク
 - 解析用リンク
 - 鉄骨製造
 - MEP エンジニアリング
 - + MEP 要素の作成
 - MEP システム
 - コネクタ
 - MEP 製造の詳細
 - ファミリの作成
 - 機器設定
 - 電気設定
 - 経路設定時の優先指定
 - 高度なトピック
 - + Revit モデルにデータを格納する
 - + イベント
 - 外部イベント
 - ドッキング可能なダイアログ ベイン
 - + ダイナミック モデル アップデータ
 - コマンド
 - + ポストおよび処理エラー
 - パフォーマンス アドバイザ
 - + 点群データの取り込み
 - + 解析
 - + 位置と場所
 - + ワークシェアリング
 - + 建設モデリング
 - + リンクされたファイル
 - + 書き出し
 - ウインドウ ハンドル
 - ブラウザ構成

Step 2

- Revit SDK に同梱されている API リファレンスで、クラスやメソッド、プロパティを検索し、引数の解説や注釈を確認する。

The screenshot shows the Revit 2019 API Reference application window. The left sidebar contains a navigation menu with options like '非表示', '同期', '戻る', '進む', '中止', '更新', 'ホーム', and 'オプション(O)'. Below this is a search bar with tabs for '目次(O)', 'キーワード(H)', and '検索(S)'. A list of namespaces is provided, starting with Autodesk.Revit.ApplicationServices Namespace, Autodesk.Revit.Attributes Namespace, Autodesk.Revit.Creation Namespace, Autodesk.Revit.DB Namespace, Autodesk.Revit.DB.Analysis Namespace, Autodesk.Revit.DB.Architecture Namespace, Autodesk.Revit.DB.DirectContext3D Namespace, Autodesk.Revit.DB.Electrical Namespace, Autodesk.Revit.DB.Events Namespace, Autodesk.Revit.DB.ExtensibleStorage Namespace, Autodesk.Revit.DB.ExternalService Namespace, Autodesk.Revit.DB.Fabrication Namespace, Autodesk.Revit.DB.IFC Namespace, Autodesk.Revit.DB.Lighting Namespace, Autodesk.Revit.DB.Macros Namespace, Autodesk.Revit.DB.Mechanical Namespace, Autodesk.Revit.DB.Plumbing Namespace, Autodesk.Revit.DB.PointClouds Namespace, Autodesk.Revit.DB.Steel Namespace, Autodesk.Revit.DB.Structure Namespace, Autodesk.Revit.DB.Structure.StructuralSe, Autodesk.Revit.DB.Visual Namespace, Autodesk.Revit.Exceptions Namespace, Autodesk.Revit.UI Namespace, Autodesk.Revit.UI.Events Namespace, Autodesk.Revit.UI.Macros Namespace, Autodesk.Revit.UI.Mechanical Namespace, Autodesk.Revit.UI.Plumbing Namespace, and Autodesk.Revit.UI.Selection Namespace. At the bottom of the sidebar is a 'What's New' link.

The main content area displays the 'Major changes and renovations to the Revit API' page. The title is 'Major changes and renovations to the Revit API'. Below it is a section titled 'API changes' with a sub-section for '.NET 4.7'. It states: '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.' Under 'View Filters support OR operators and nesting', it says: '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()'.

Below this, it says: '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 applicable only to filters with rules.

Step 3

- Revit API Forum で検索して、類似するスレッドをみつける。
 - <https://forums.autodesk.com/t5/revit-api-forum/bd-p/160>

The screenshot shows the Autodesk Knowledge Network Revit API Forum. At the top, there's a navigation bar with the Autodesk logo, account creation and sign-in links, and a language selector set to Japanese. Below the header is a search bar with a magnifying glass icon. The main title "REVIT" is prominently displayed with a large blue "R". Below it, there are two tabs: "ナレッジ" (Knowledge) and "フォーラム" (Forum), with "フォーラム" being the active tab. A secondary search bar is located below the main one. On the left side, there are links for "この掲示板" (This Board), "検索" (Search), and a "フォーラムに投稿" (Post to Forum) button. Below these are filters for "すべての投稿" (All Posts), "よくある質問" (FAQ), "解決策として承認済み" (Approved Solutions), and "未解決" (Unresolved). A pagination bar shows pages 1 through 246. On the right side, there's a sidebar with links to "フォーラムやアイデアに移動" (Move to Forum or Idea), "Revit Products のトップに戻る" (Return to Revit Products Top), "すべてのフォーラム" (All Forums), "すべてのアイデア" (All Ideas), and "ヘルプ" (Help). Below the sidebar, there's a section titled "話題のトピック" (Topics) with several posts listed:

- Surprising results from Face.Intersect(face) method
- How to create a Warning
- Ribbon Panel items are disabled when

At the bottom left, there's a comment from a user named jeremytammik: "Your input on Dynamo on the Web is needed %". The comment was posted on March 26, 2018, at 04:12 AM.

Step 4

- 弊社エンジニアの Jeremy Tammik が運営している「The Building Coder」で検索してみる。
 - <http://thebuildingcoder.typepad.com/>

The screenshot shows a blog post on the 'The Building Coder' website. The header features a large image of a building's interior structure. The title of the post is 'RvtSamples 2019'. The author's photo, a man with short brown hair, is displayed next to the text. Below the post, there is a search bar, a language selection dropdown ('G 言語を選択'), and a link to subscribe to the feed. The bottom navigation bar includes links for 'home', 'archives', 'about', 'topics', 'index', 'source', 'subscribe', 'code', 'forum', 'devguide', and '3dwc', along with social media icons for Facebook, Google+, LinkedIn, and Twitter.

The Building Coder
Programming Forge, BIM and the Revit API

* Forge RVT to IFC, ADN Xtra, TBC, AdnRme Updates | Main | What's New in the Revit 2019 API *

April 24, 2018

RvtSamples 2019

I already described how I installed Revit 2019, compiled the Revit 2019 SDK samples, migrated RevitLookup to the new version, The Building Coder samples, the AdnRme MEP HVAC and electrical samples and the AdnRevitApiLabsXtra training labs.

In the course of the last post, I forgot to mention setting up RvtSamples.

Just like the migration to previous versions, this is not a trivial undertaking.

Here are the discussions of the previous migrations to the Revit 2017 SDK and the Revit 2018.

To cut a long story short and simply share my current working RvtSamples source code for the Revit 2019 SDK, here is RvtSamples_2019.zip containing my modified files. It also includes some of the originals renamed by adding the suffix `_original`.

I hope you find this useful.

About – Topics – Index – Source

G 言語を選択 ▾

Search Submit

Subscribe to this blog's feed

Categories

home archives about topics index source subscribe code forum devguide 3dwc [f](#) [g+](#) [in](#) [t](#)

用語やプロパティの文言の英訳がわからないときは

- Revit ヘルプ（Revit API 開発者用ガイド）を英語版にして対訳を確認。

ヘルプのホームページ Ryuji Ogasawara 日本語 簡体中文 繁體中文 Čeština Deutsch English Español Français Italiano 한국어 Polski Português

AUTODESK® REVIT® 2019 キーワードを入力

ドアタイププロパティ

構築タイプ、機能、マテリアル、寸法などを変更するには、ドアのタイププロパティを修正します。タイププロパティを変更するには、要素を選択して、[修正]タブ ▶ [プロパティ]パネル ▶ (タイププロパティ)します。タイププロパティへの変更は、プロジェクト内のすべてのインスタンスに適用されます。

名前	説明
構築	ドアの周りの納まり。この設定は、ホストの設定より優先されます。
壁の納まり	ドアの周りの納まり。この設定は、ホストの設定より優先されます。
部分構成	ドアの周囲を構成する部品。
機能	ドアの開閉機能。
マテリアル	ドアのマテリアル。
ドアのマテリアル	ドアのフレームのマテリアル。



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.