AUTODESK

Dynamo の 「イロハ」 ~ 番外編：

Dynamo！Python！API！実は簡単？！
経験ゼロの現役大学生が徹底解説 ~

末廣 隆介, 日下部 達哉

本日の主な講師は、弊社 Intern（末廣さん）です

- 本日の流れ
  - セミナー本編 by 末廣　　（Dynamo for Revit 上での、Python の使い方 を解説）
  - 自由討論 by 末廣＆日下部　（Dynamo 上で Python を「初めて触ってみた」感想）
  - 補足説明 by 日下部　　（Dynamo for Civil 3D 上での、Python の使い方 に関して補足）
- このセミナーを通したメッセージ：**Dynamo を使用している方**
  - このセミナーの手順に従うと、 Dynamo 上で Python が書けます。
- このセミナーを通したメッセージ：**Dynamo を使用したことがない方（調査中を含む）**
  - Dynamo 未経験の学生さんでも、一か月でここまで出来ます。あなたにも、きっと出来ます。
  - Dynamo 上で Python が書けると、Dynamo をもっと便利に使えます。
  - Dynamo の基礎を学んで、Python の世界に踏み出したくなったら、このセミナーがあります。

セミナー 本編

Dynamo の 「イロハ」 ~ 番外編 ~

今日のコンテンツ

自己紹介

API とは？

API 実装後の最終形態

API リファレンス (API Docs) を読み解く

Python を書き始める下準備

Revit タイプと Dynamo タイプの変換方法

デバッグのテクニック

Python を使って Dynamo を実装するメリット

自己紹介：末廣隆介

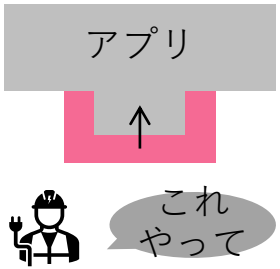
技術営業 (テクニカルセールス) のインターンです！

基本的なバックグラウンド

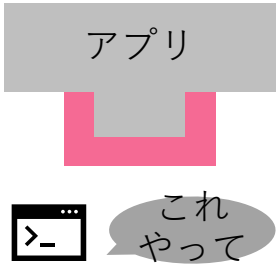
- **在籍校:** カリフォルニア大学バークレー校 (UC Berkeley)
- **学年:** 二年生
- **専攻:** 数学、コンピューターサイエンス、ビジネス
- **今まで住んだ事がある場所:** ボストン（生まれ）、ロンドン（６年間）、ニューヨーク（５年）、東京（５年）、バークレー（1年）
- **経験**：コンピュータービジョンを用いてマスク着用の有無、また正しくつけているか検知するプログラム、ゲーム会社のユーザー分析
- **趣味**：テニス、ゴルフ、旅行、食べること

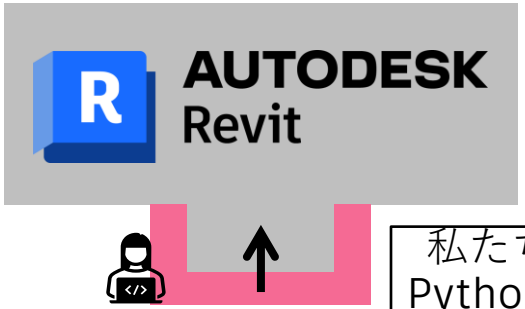
API (Application Programming Interface) とは

GUI：人間がアプリに指示



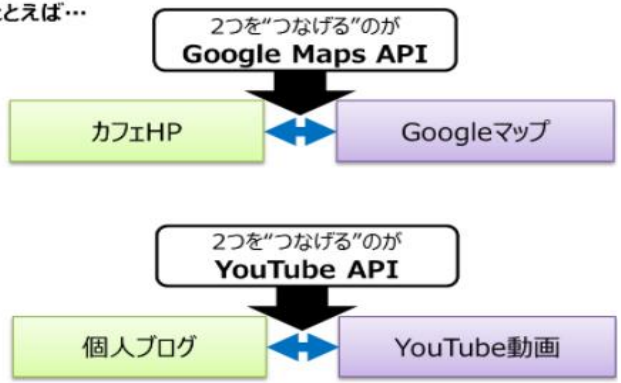
API：プログラムがアプリに指示





私たちが Dynamo 上で Python を書き、Revit のサービスを使うよう指示する必要がある

たとえば…

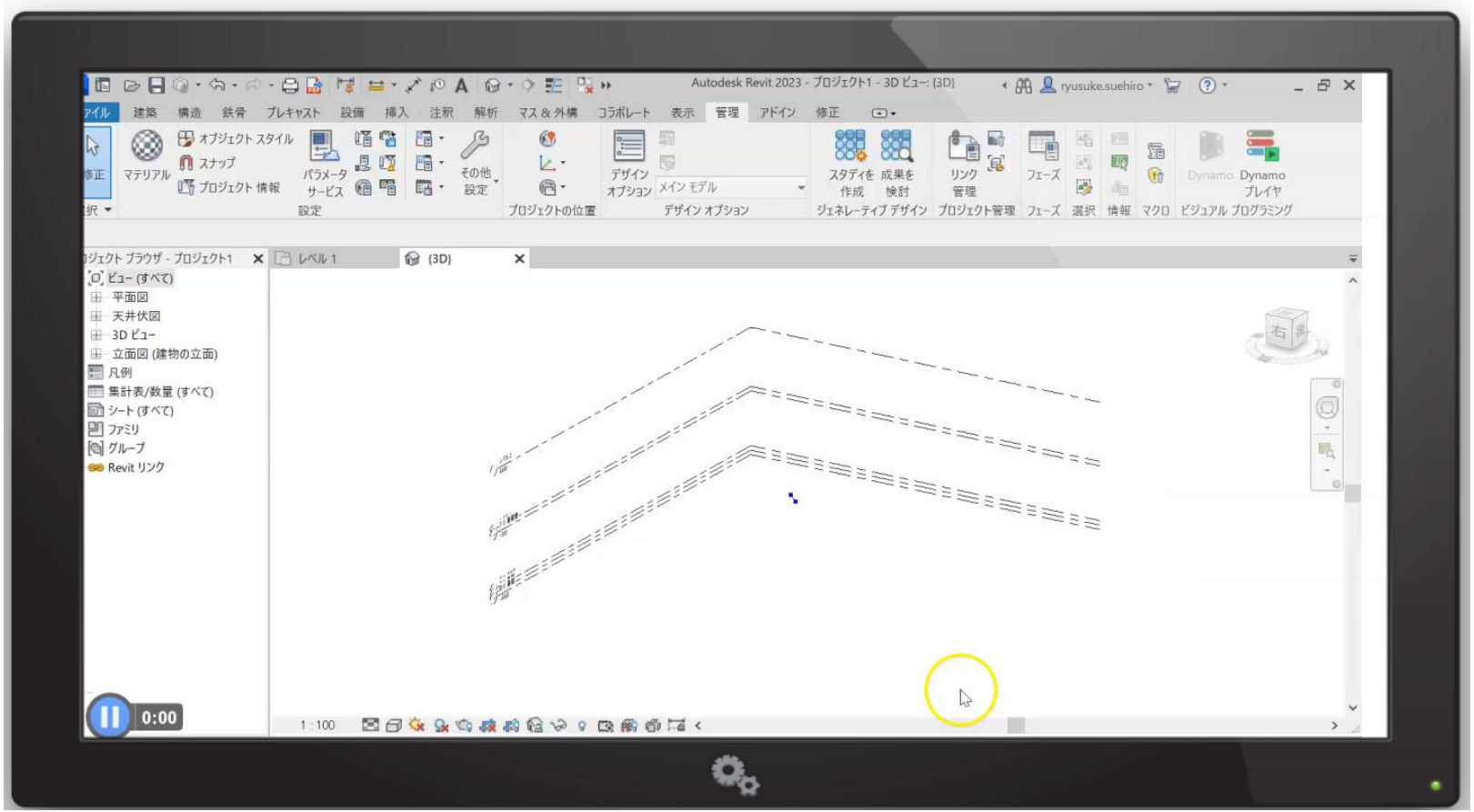


API があると、複数のサービスを自動で接続できるようになる

Revit にある膨大なサービスを自動的に使えるようになれば**無限の可能性がある！**

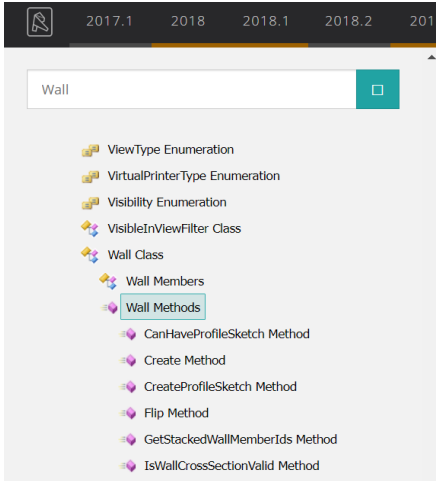
結果動画

Dynamo と Python Script を使うと、一瞬で壁が作れる！

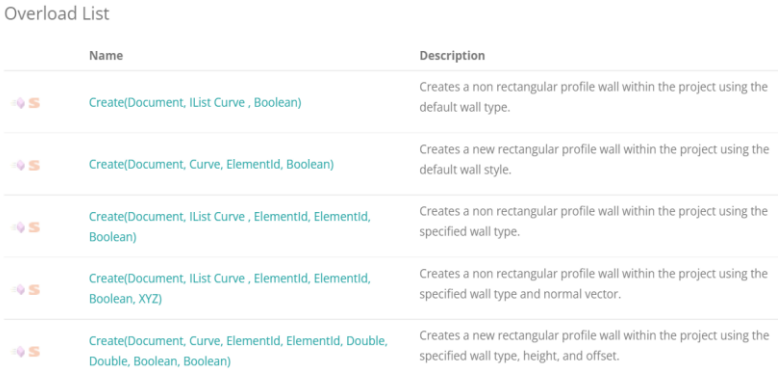


Revit API Docs の検索、読み方

Revit API Docs で使いたい Method (関数) を検索 → 複数同じメソッドがある場合は、必要なパラメータ情報と生成結果の説明 (Description) をもとに選択 → パラメータの詳しい情報を確認



注意：壁などを作りたい場合は作りたいものが所属する “Wall” というクラスから検索し、その後 “Create” を検索する



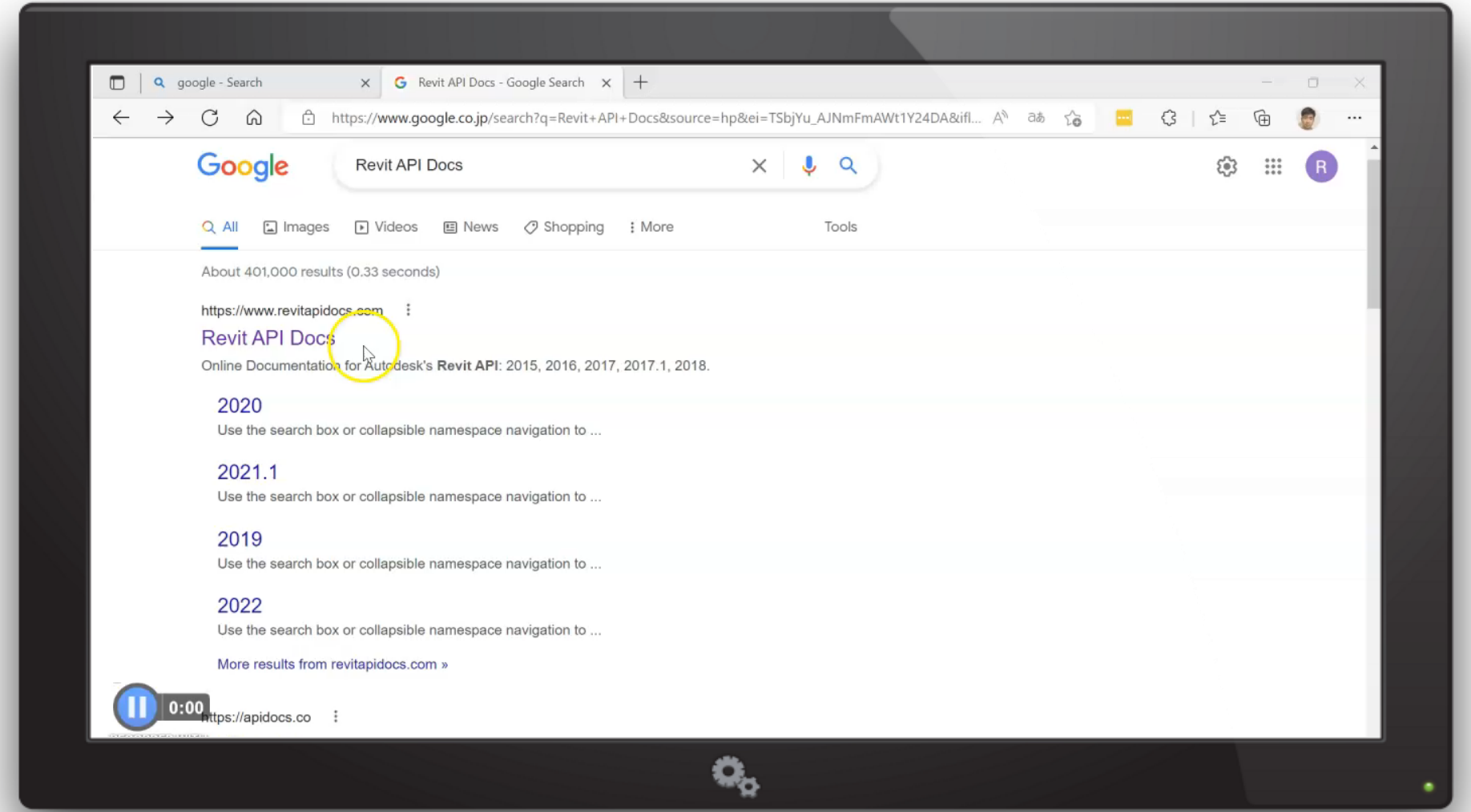
注意：同じメソッドでもパラメータの種類によって作る壁の種類が変わる

```
public static Wall Create(
        Document document,
        Curve curve,
        ElementId wallTypeId,
        ElementId levelId,
        double height,
        double offset,
        bool flip,
        bool structural
)
```

注意：太字のものは必須。その他は指定しないとデフォルトの値が使われる

後ほどこのように Python を実装するためにはどんな情報を用意するべきか分かる

```
wall = Wall.Create(doc, curve, wallTypeId, levelId, 250.0,0.0,False,True)
```

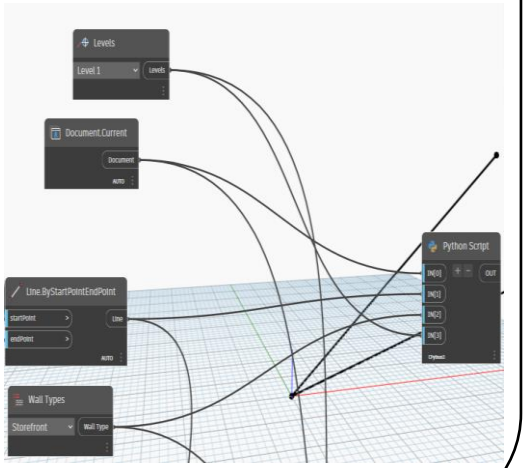

google - Search
Revit API Docs - Google Search
https://www.google.co.jp/search?q=Revit+API+Docs&source=hp&ei=TSbjYu_AJNmFmAWt1Y24DA&ifl...
Google
Revit API Docs
All
Images
Videos
News
Shopping
More
Tools
About 401,000 results (0.33 seconds)
https://www.revitapidocs.com
Revit API Docs
Online Documentation for Autodesk's Revit API: 2015, 2016, 2017, 2017.1, 2018.
2020
Use the search box or collapsible namespace navigation to ...
2021.1
Use the search box or collapsible namespace navigation to ...
2019
Use the search box or collapsible namespace navigation to ...
2022
Use the search box or collapsible namespace navigation to ...
More results from revitapidocs.com »
0:00
https://apidocs.co

Python Script で API を実装する下準備

API Docs にあるパラメータを Dynamo で作り、Python Script が使えるような準備が必要

Python Script の Input の設定

API Docs で見たパラメータを Dynamo で作り、Python Script の IN につなげる



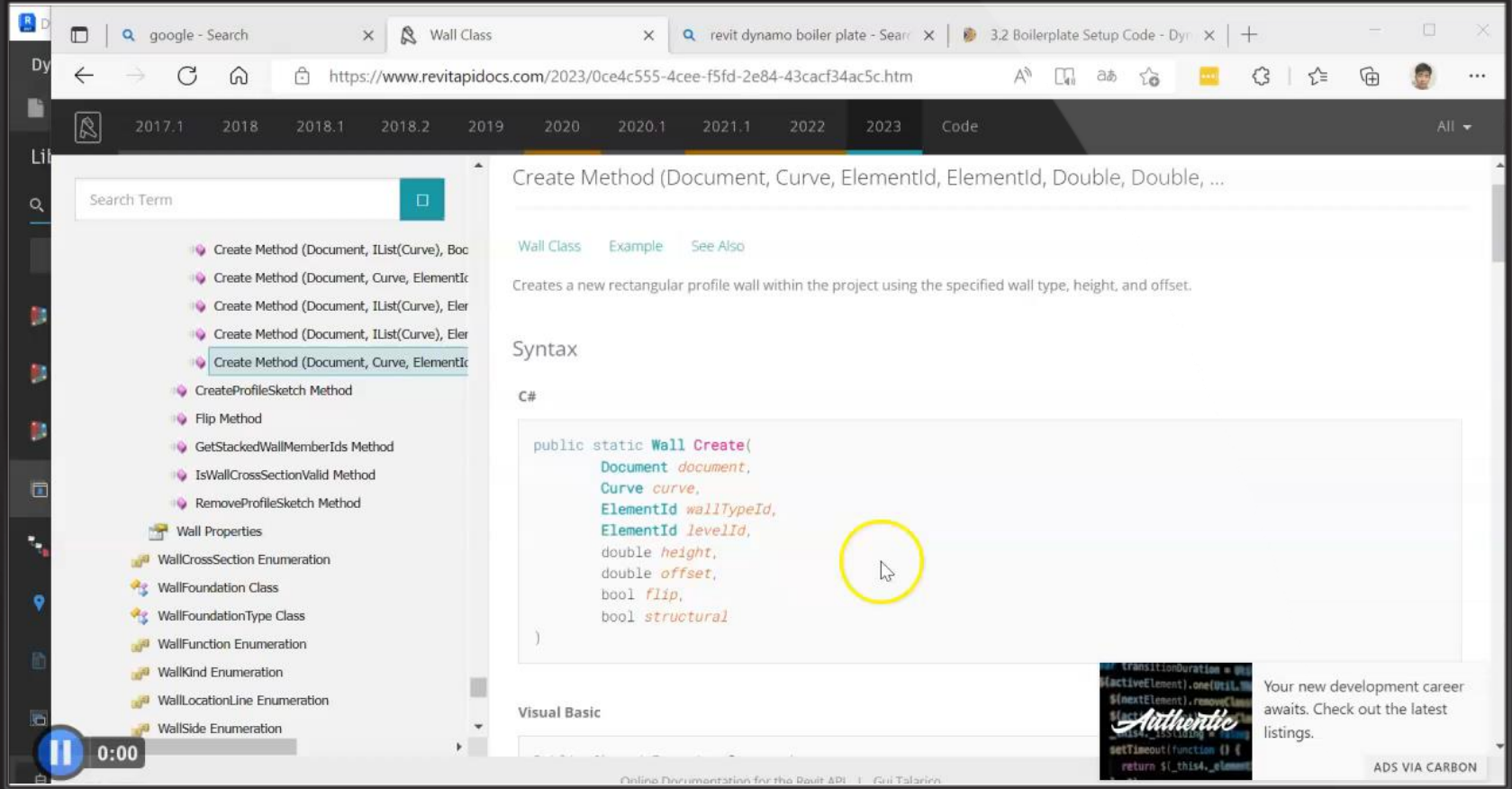
Python Script ライブラリーの読み込み

RevitBoilerplate.py

```python
import clr
import sys
sys.path.append('C:\Program Files (x86)\IronPython 2.7\Lib')
import System
from System import Array
from System.Collections.Generic import *
clr.AddReference('ProtoGeometry')
from Autodesk.DesignScript.Geometry import *
clr.AddReference("RevitNodes")
import Revit
clr.ImportExtensions(Revit.Elements)
clr.ImportExtensions(Revit.GeometryConversion)
clr.AddReference("RevitServices")
import RevitServices
from RevitServices.Persistence import DocumentManager
from RevitServices.Transactions import TransactionManager

clr.AddReference("RevitAPI")
clr.AddReference("RevitAPIUI")

import Autodesk
from Autodesk.Revit.DB import *
from Autodesk.Revit.UI import *

doc = DocumentManager.Instance.CurrentDBDocument
uiapp = DocumentManager.Instance.CurrentUIApplication
app = uiapp.Application
uidoc = uiapp.ActiveUIDocument

#######OK NOW YOU CAN CODE########
```

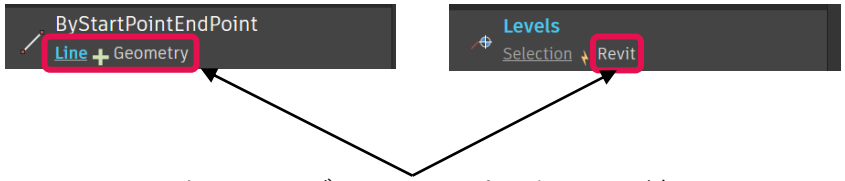
ボイラープレート からコピーして Python が Revit の API を認識できるようセットアップ。このステップを忘れると必ずエラーが出る

* IronPython2, CPython3 どちらでも使用可能


google - Search
Wall Class
revit dynamo boiler plate - Sear
3.2 Boilerplate Setup Code - Dyn
https://www.revitapidocs.com/2023/0ce4c555-4cee-f5fd-2e84-43cacf34ac5c.htm
2017.1 2018 2018.1 2018.2 2019 2020 2020.1 2021.1 2022 2023 Code All
Search Term
Create Method (Document, IList(Curve), Boc
Create Method (Document, Curve, ElementIc
Create Method (Document, IList(Curve), Eler
Create Method (Document, IList(Curve), Eler
Create Method (Document, Curve, ElementIc
CreateProfileSketch Method
Flip Method
GetStackedWallMemberIds Method
IsWallCrossSectionValid Method
RemoveProfileSketch Method
Wall Properties
WallCrossSection Enumeration
WallFoundation Class
WallFoundationType Class
WallFunction Enumeration
WallKind Enumeration
WallLocationLine Enumeration
WallSide Enumeration
Create Method (Document, Curve, ElementId, ElementId, Double, Double, ...
Wall Class Example See Also
Creates a new rectangular profile wall within the project using the specified wall type, height, and offset.
Syntax
C#
public static Wall Create(
Document document,
Curve curve,
ElementId wallTypeId,
ElementId levelId,
double height,
double offset,
bool flip,
bool structural
)
Visual Basic
0:00
Your new development career awaits. Check out the latest listings.
ADS VIA CARBON

Dynamo の様々なタイプから Revit への変換方法

現在あるエレメントが何タイプか確認



Dynamo 上で一つずつのコンポーネントが何のタイプなのかが見られる。
Revit と書いてない場合は Revit への変換が必要
Revit と書いてある場合は Unwrap が必要（後述）

API 実装に必要な Revit Type (オレンジ) と照合

```
public static Wall Create(
        Document document,
        Curve curve,
        ElementId wallTypeId,
        ElementId levelId,
        double height,
        double offset,
        bool flip,
        bool structural
)
```

変換方法が分からない場合：

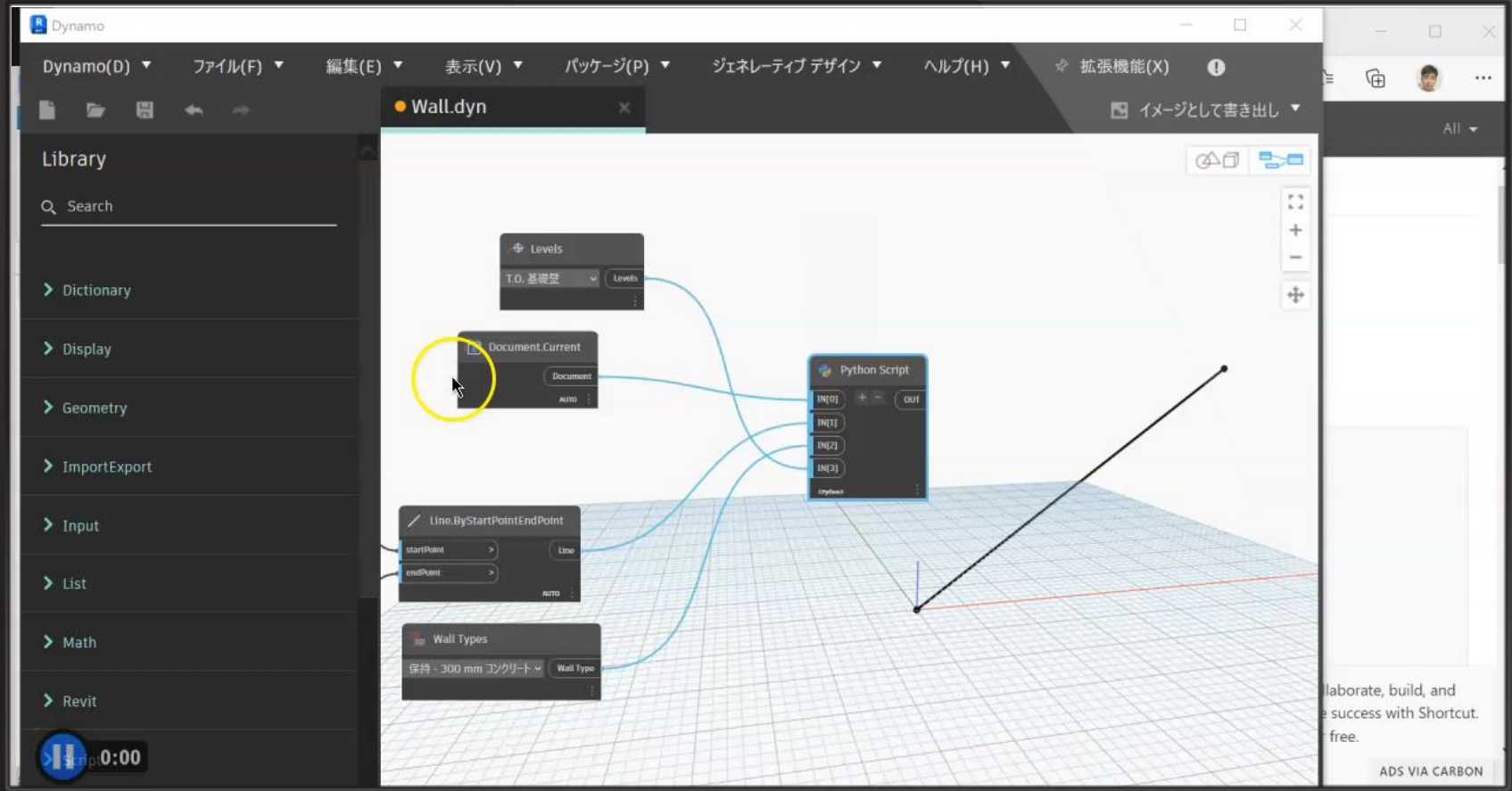
変換方法の検索

ネットや Dynamo フォーラムで変換する前と後のタイプをキーワードにして検索！もしくは API Docs でエレメントの継承ストラクチャーを見る

変換方法がすぐわかる場合：

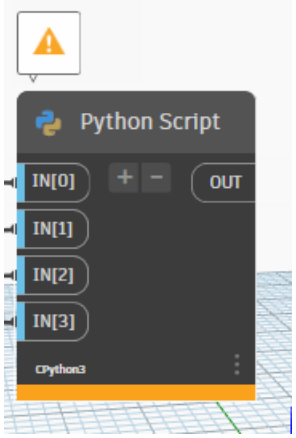
先程 IN に繋げた Dynamo エレメントを一つずつ読み込み、必須パラメータの数と一致するか確認し、それぞれ適正な Revit へタイプ変換を行う

```
35 document = IN[0]
36 curve = IN[1].ToRevitType()
37 wallTypeId = UnwrapElement(IN[2]).Id
38 levelId = UnwrapElement(IN[3]).Id
```


Dynamo
Dynamo(D)
ファイル(F)
編集(E)
表示(V)
パッケージ(P)
ジェネレーティブ デザイン
ヘルプ(H)
拡張機能(X)
Wall.dyn
イメージとして書き出し
Library
Search
Dictionary
Display
Geometry
ImportExport
Input
List
Math
Revit
Levels
T.O. 基礎笠
Levels
Document.Current
Document
AUTO
Python Script
IN[0]
IN[1]
IN[2]
IN[3]
OUT
Line.ByStartPointEndPoint
startPoint
endPoint
Line
AUTO
Wall Types
保持 - 300 mm コンクリート
Wall Type
0:00
All
laborate, build, and
success with Shortcut.
free.
ADS VIA CARBON

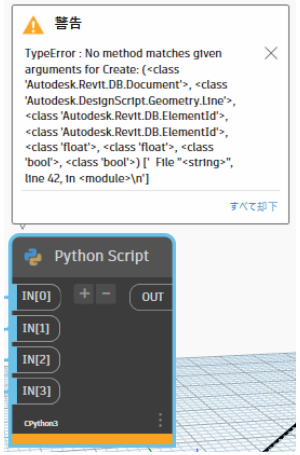
Python Script のデバッグの仕方

Dynamo の Python Script がオレンジ色で警告サインが見える

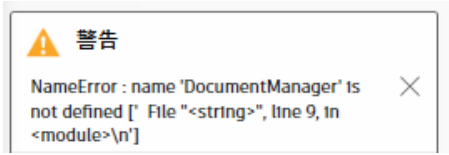


警告サインを押す

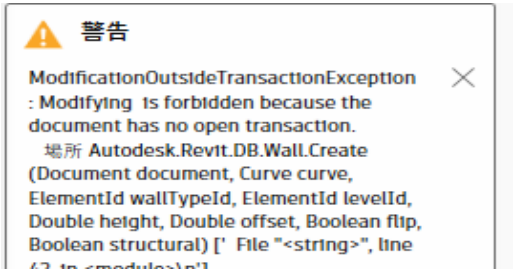
エラーの種類、エラー箇所の行数が見える

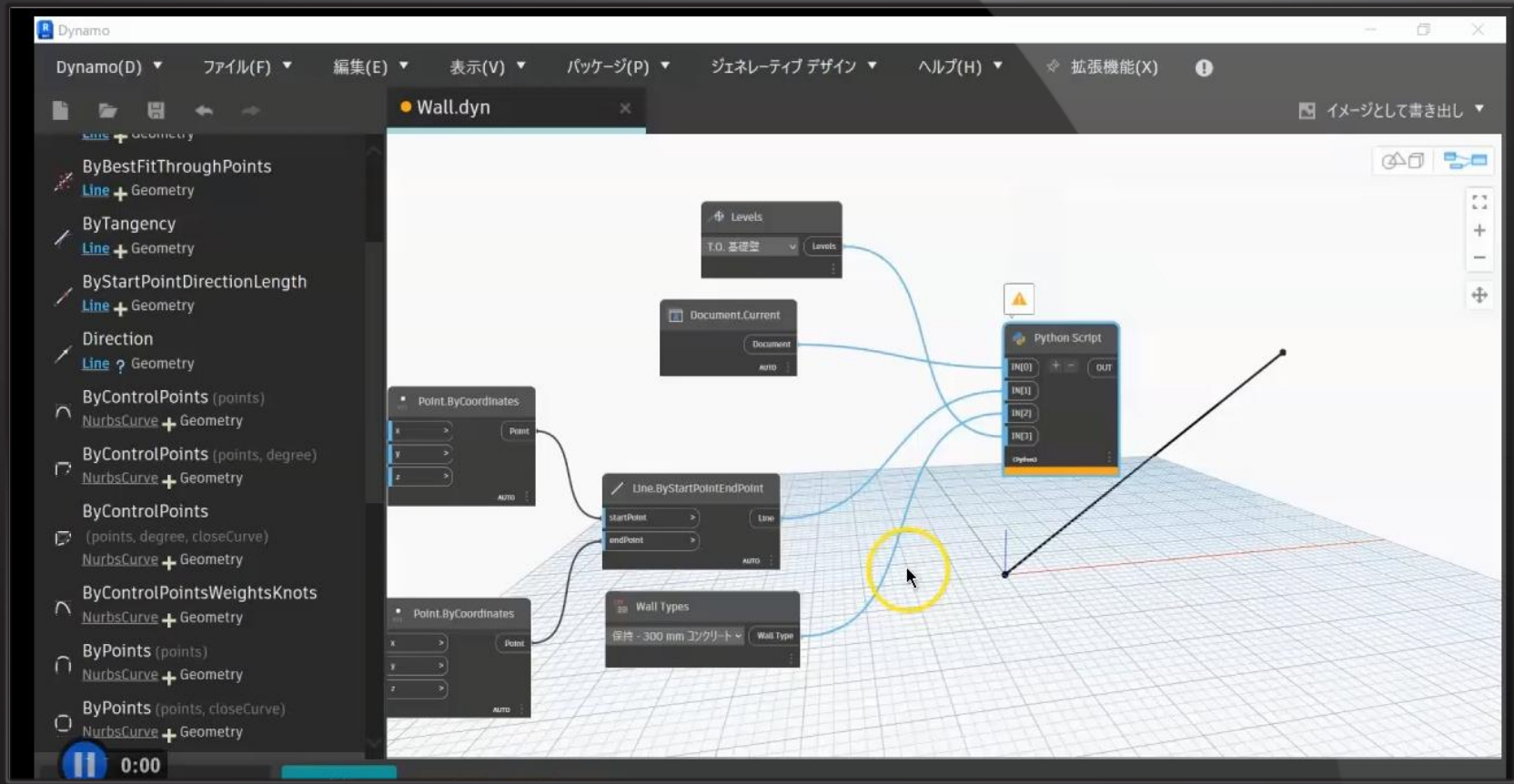


Not Defined など明確なエラーの場合は指定された行に行き、対処



複雑なエラーの場合はすぐ検索！




Dynamo
Dynamo(D)
ファイル(F)
編集(E)
表示(V)
パッケージ(P)
ジェネレーティブ デザイン
ヘルプ(H)
拡張機能(X)
Wall.dyn
イメージとして書き出し
ByBestFitThroughPoints
Line + Geometry
ByTangency
Line + Geometry
ByStartPointDirectionLength
Line + Geometry
Direction
Line ? Geometry
ByControlPoints (points)
NurbsCurve + Geometry
ByControlPoints (points, degree)
NurbsCurve + Geometry
ByControlPoints
(points, degree, closeCurve)
NurbsCurve + Geometry
ByControlPointsWeightsKnots
NurbsCurve + Geometry
ByPoints (points)
NurbsCurve + Geometry
ByPoints (points, closeCurve)
NurbsCurve + Geometry
0:00
Levels
Document.Current
Python Script
Point.ByCoordinates
Line.ByStartPointEndPoint
Point.ByCoordinates
Wall Types

なぜ Python と Dynamo を試してみるべきか

反復作業を効率化できる

例：モデル作成に要する時間を
2 週間から 2 日に短縮（こちら）

Dynamo ノードだけでは不可能な Revit カスタマイズもできる

・Dynamo で未対応の要素の処理
(CAD リンク, MEP 製造用パーツ, …)
・複雑なリスト処理 (for, while, …)
・Python ライブラリを活用した計算
(NumPy, Pandas, …)

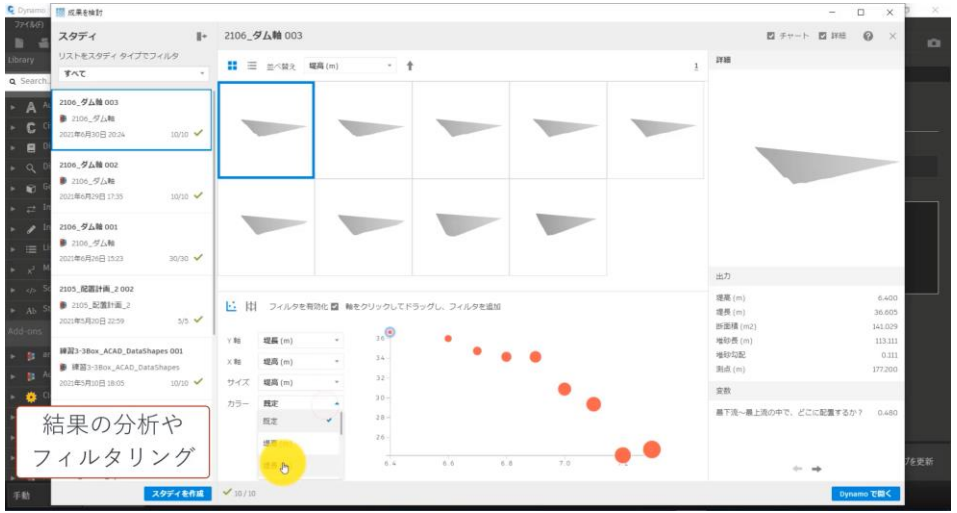
簡単に壁の間隔や高さもカスタマイズできる

なぜ Python と Dynamo を試してみるべきか 2

オートデスクの最先端技術を
試せるので、楽しい

例：Dynamo for Civil 3D, Python に加えて
Generative Design の技術も活用

→ 砂防施設の最適配置（詳細は こちら）



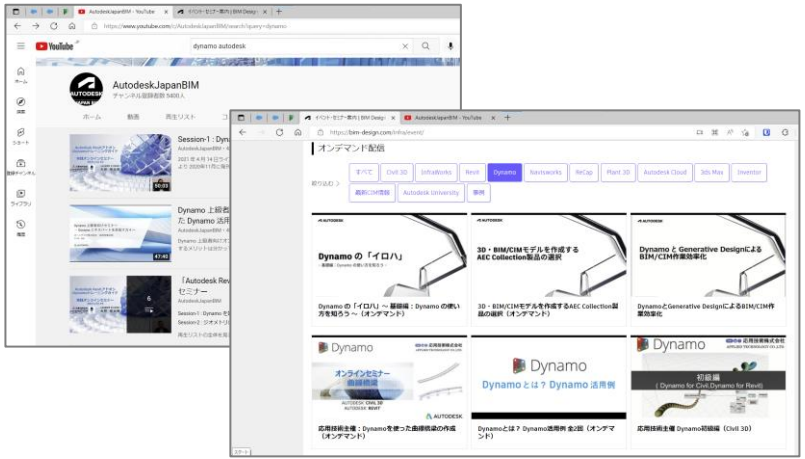
Free Talk Session

Dynamo 学習、最初はどこから？

様々な動画コンテンツ

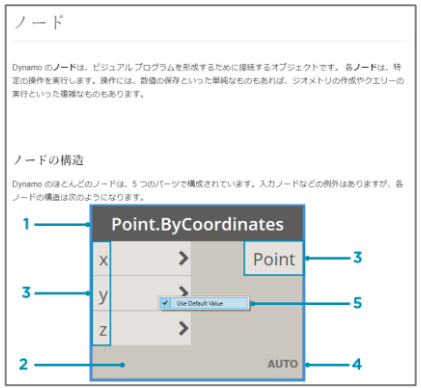
Dynamo の操作方法を確認する、「手順書」
- YouTube でいろいろ検索（こちら）
- Autodesk Japan のセミナー
  （建築編は こちら、土木編は こちら）

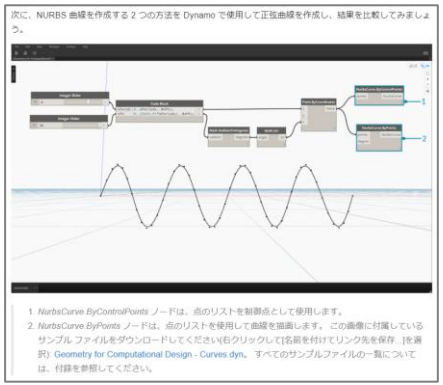
* 今回のセミナーのアーカイブは こちら に掲載予定



Dynamo Primer

Dynamo の機能を網羅的に解説した、「辞書」
- Dynamo の起動、UI
- ビジュアルプログラムの構造
- 形状を作成 (ジオメトリの操作)
- データを整理 (リスト、ディクショナリ)
- プログラムを記述 (Code Block, Python)
- …

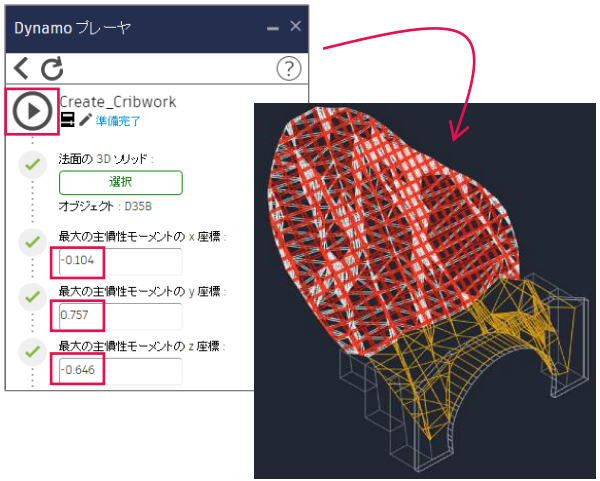


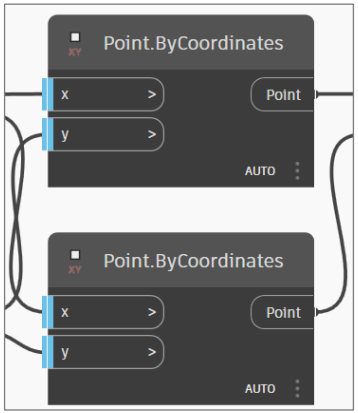


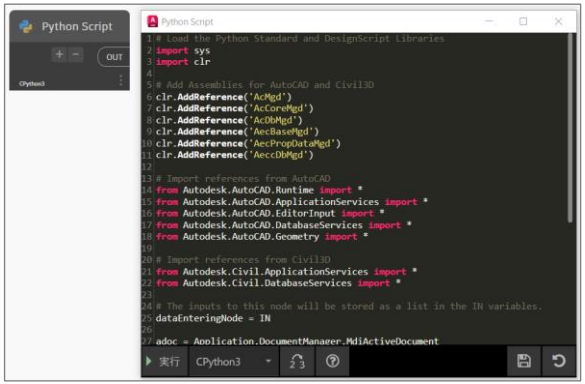
Dynamo の特色は何だと思うか？

- ビジュアルプログラミングの「分かりやすさ」と、Python の「拡張性」を両立
- 使い分けのカギは、ハンズオン動画を見たり自分で実際に触って試行錯誤をすること！

Dynamo Player	Dynamo	Dynamo + Python
既にある Dynamo のプログラムに、 入力値だけ指定して、処理を実行	ビジュアル プログラミングで プログラムを作成	Dynamo のノードだけでは 実現できない機能を、 Python を書いて実現

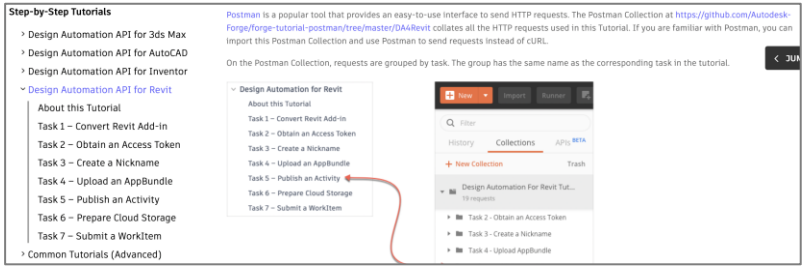


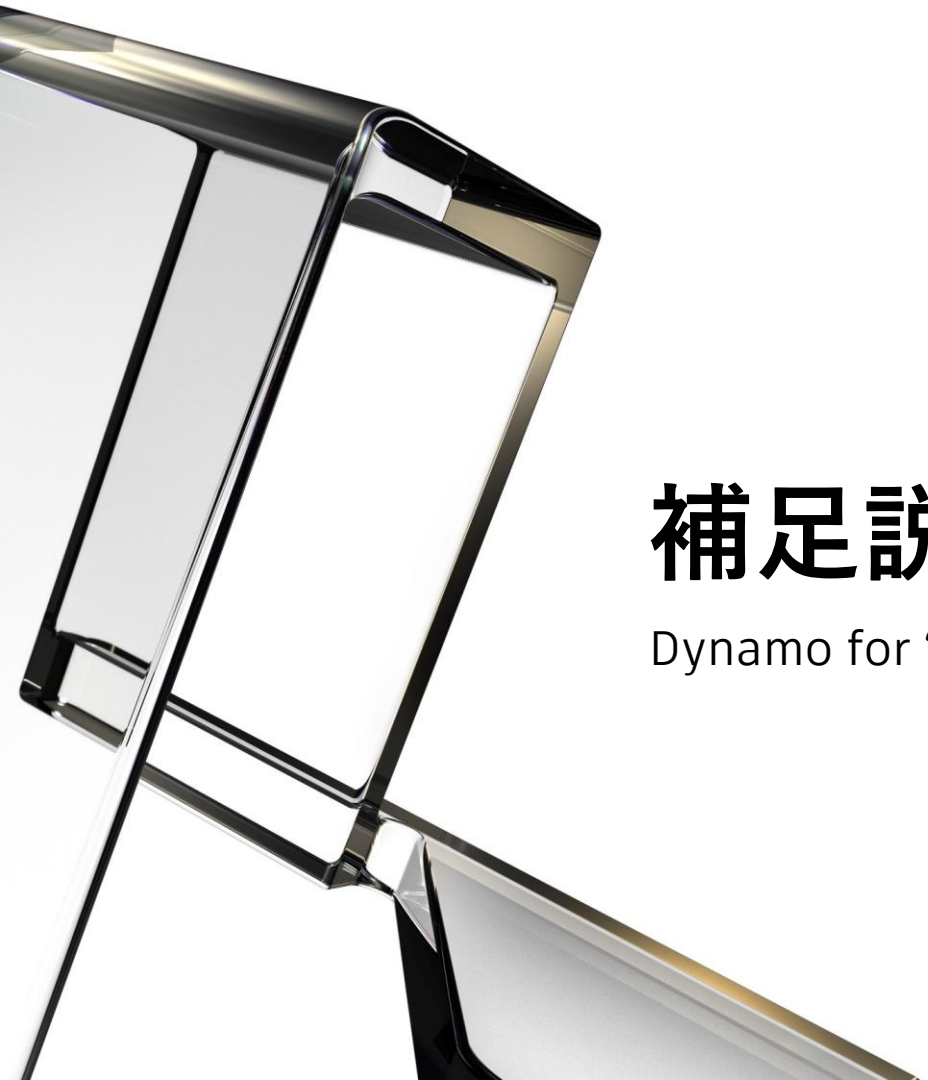




オートデスクの特色は何だと思うか？

- オープンなところ
  - API が公開されているので、自由にカスタマイズができる（Web API も！）
  - ユーザ同士のコミュニティがあるので、知見が Web 上に蓄積されている





補足説明

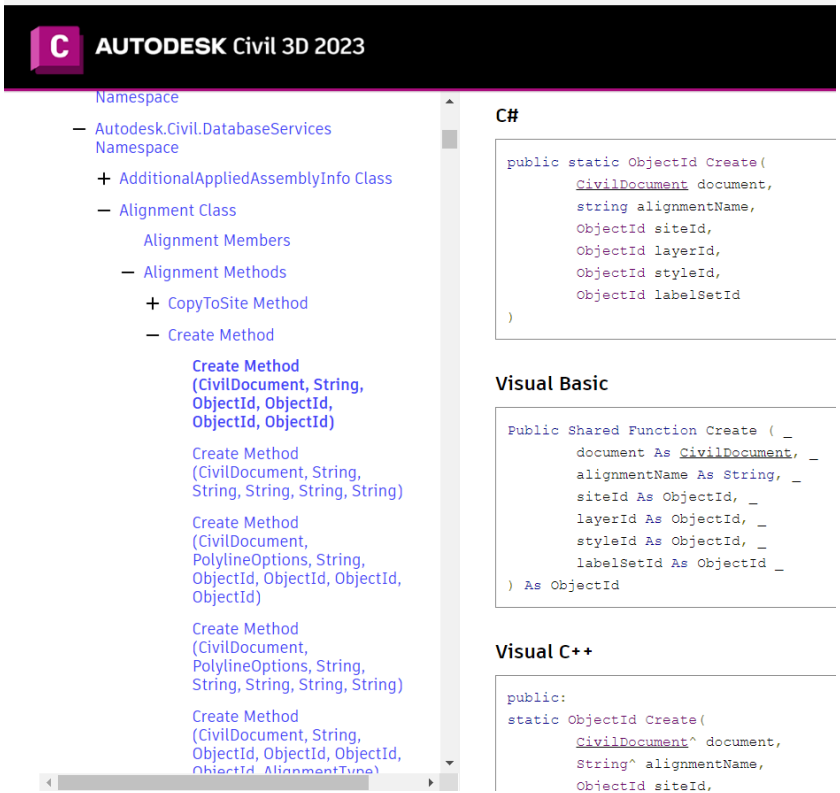
Dynamo for “Civil 3D” & Python の場合

結果動画

Dynamo と Python Script を使うと、一瞬でサブアセンブリが作れる！

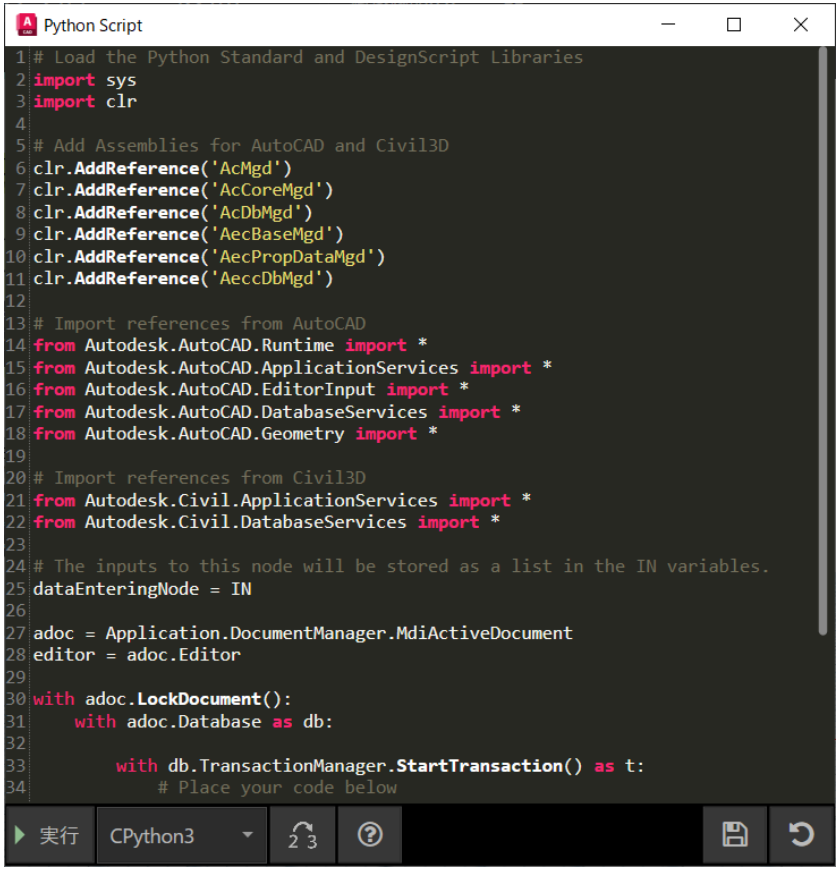
API Reference の読み方

- API Reference のありか
  - https://help.autodesk.com/view/CIV3D/2023/ENU/?guid=89ffd413-aada-d770-e322-89dfa7b99369
  - 年数 (2023) を変えれば、
    各 Ver の Reference を入手可能
- API Reference の読み方
  - 基本的には Revit API Docs と同じ
  - 使いたい Method (関数) を検索
  - 同じメソッドが複数ある場合は、
    説明書きを基に絞り込み
  - パラメータの詳しい情報を確認



Python Script で API を実装する下準備

- Python Script の Input の設定
  - Revit に同じ
- Python Script ライブラリーの読み込み
  - Python Script を作成した段階で、
    ボイラープレートを設定済



Dynamo タイプから Civil 3D タイプへの変換

- Revit と違い、変換不可
  - 下記の二つによる対応が基本
    - 最初から最後まで Dynamo ノードを使用
    - 最初から最後まで Python Script を使用
- Civil3DToolkit パッケージを使うと、Dynamo ノードだけで行える処理を飛躍的に増やすことが可能
  - パラメータの “参照” だけではなく、“作成” “更新” に対応できるように