



# Forge Online Training – Design Automation Revit

小笠原 龍司

Developer Advocacy & Support、Autodesk Developer Network

# アジェンダ – ご紹介する内容

- はじめに
- Design Automation を利用するソリューション
- Learn Forge : モデルを修正する
- 付録 : .NET Core サンプルに Viewer 機能を追加する
- 付録 : コストについて



はじめに

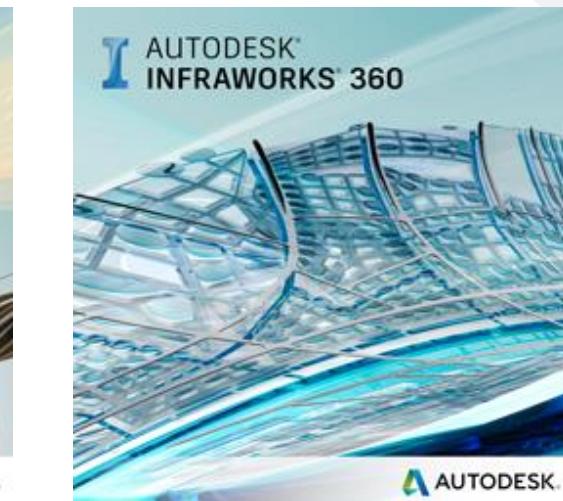
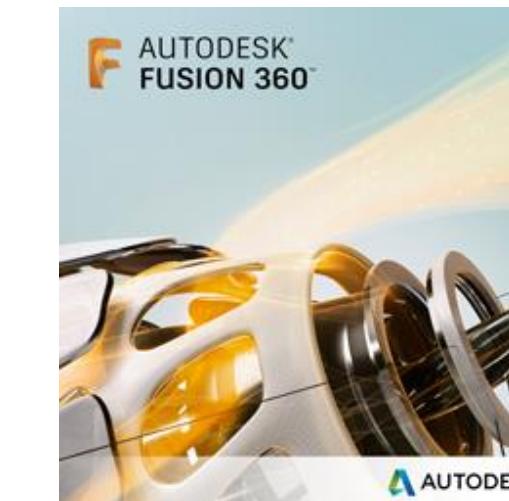
# インターネットで各社提供 API への接続が可能な時代

- 業界標準＆オープンソース API テクノロジ



RESTful API  
GET PUT POST DELETE

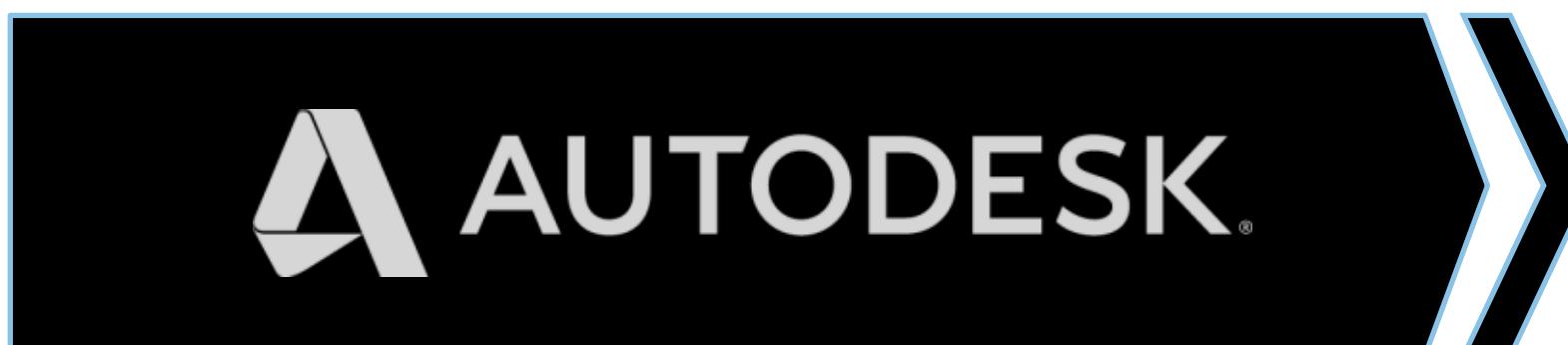




AUTODESK  
Forge

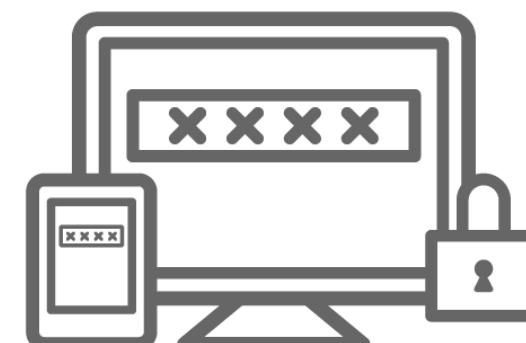
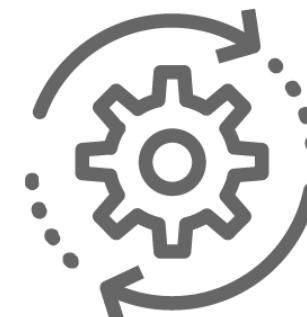
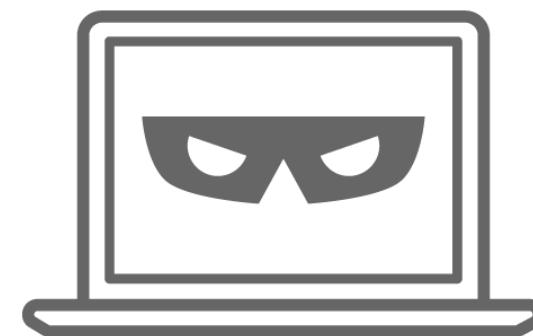
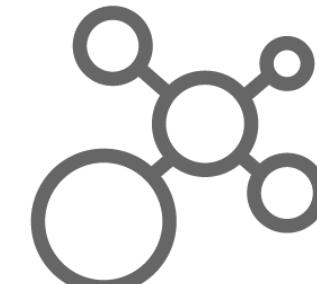


# 要素技術



# デスクトップ製品のアドイン開発者の方へ

- デスクトップ開発と Web 開発の違いをご理解ください
  - Web アプリの特性 – セキュリティ視点
  - 通信経路
  - 呼び出し数制限
  - 非同期処理
  - 仮想環境
  - オンプレミスとの違い



関連ブログ記事



# 公式ドキュメント

- Forge ポータル (<https://forge.autodesk.com/>) に記載

The screenshot shows the Autodesk Documentation page for Core Components. The top navigation bar includes links for Platform Vision, Solutions, Getting Started, Documentation (which is highlighted with a blue arrow), Community, Support, and Pricing. A search icon and a 'SIGN IN' button are also present. The main content area features a section titled 'Documentation' with a sub-section 'Core Components'. It describes a set of foundational building blocks for tying data and workflows together. Three cards are displayed: 'Authentication (V1)', 'Data Exchange', and 'Data Management'. Each card has a corresponding 'API Reference' link at the bottom.

**AUTODESK** Platform Vision Solutions ▾ Getting Started Documentation Community ▾ Support ▾ Pricing SIGN IN

Core Components

- Authentication (V1)
- Data Exchange
- Data Management
- Data Visualization
- Design Automation
- Model Derivative
- Premium Reporting
- Reality Capture
- Token Flex
- Viewer SDK
- Webhooks

**ACC**

- ACC APIs
- BIM 360 APIs

**Code Samples**

## Documentation

Here you'll find references, guides, resources and tutorials to build your solutions with Forge.

### Core Components

A set of foundational building blocks that enable you to tie data and workflows together.

**Authentication (V1)**  
Generate tokens based on the OAuth 2.0 standard to authenticate requests made to Forge APIs and SDKs.

**Data Exchange**  
Connect and access subsets of design data in your app of choice. Data exchanges enable you to share design data with collaborators. Consumers of that data can selectively use the data they care about. Currently available for Revit data.

**Data Management**  
Access data across BIM 360 team, Fusion Team, BIM 360 Docs, and the Object Storage Service to build apps to display and extend your data in ways that add value to your users.

API Reference API Reference API Reference

# Forge サンプル

- Forge ポータル下部 Code Samples からのアクセス

The screenshot shows the bottom section of the Autodesk Forge website. At the top left is an orange 'SUBMIT' button with a right-pointing arrow. Below it is a message: 'This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.' The footer is divided into five columns: 'FOLLOW FORGE' (Twitter, Facebook), 'SOLUTIONS' (Autodesk Construction Cloud, BIM 360, Data Management, Data Visualizations, Design Automation, Model Derivative, Reality Capture, Token Flex, Viewer, Webhooks), 'DOCUMENTATION' (Authentication, BIM 360, Data Management, Data Visualizations, Design Automation, Model Derivative, Reality Capture, Token Flex, Viewer, Webhooks), 'RESOURCES' (Get Help, API Status, Blog, FAQ, Code Samples, Videos & Slides, Service Issue? Contact Us, Get in Touch!), and 'ABOUT' (About Forge, Pricing, Success Stories, Partners, DevCon 2019, DevCon 2018, DevCon 2017, DevCon 2016). A blue arrow points to the 'Code Samples' link in the 'RESOURCES' column.

FOLLOW FORGE		SOLUTIONS	DOCUMENTATION	RESOURCES	ABOUT
<a href="#">Twitter</a>	<a href="#">Facebook</a>	Autodesk Construction Cloud	Authentication	Get Help	About Forge
		BIM 360	BIM 360	API Status	Pricing
		Data Management	Data Management	Blog	Success Stories
		Data Visualizations	Data Visualizations	FAQ	Partners
		Design Automation	Design Automation	Code Samples	DevCon 2019
		Model Derivative	Model Derivative	Videos & Slides	DevCon 2018
		Reality Capture	Reality Capture	Service Issue? Contact Us	DevCon 2017
		Token Flex	Token Flex	Get in Touch!	DevCon 2016
		Viewer	Viewer		
		Webhooks	Webhooks		

Privacy/Cookies | Privacy Settings | Do not sell my personal information | Terms of Service | Legal Notices & Trademarks | Report Noncompliance | © 2022 Autodesk Inc. All rights reserved.

# Forge SDK

- RESTful API をラップするサーバー実装用ユーティリティ

The collage includes:

- A screenshot of the Autodesk Forge Responsive Connected Database application, showing a 3D model of a building and a database interface.
- A screenshot of the Forge Node.js SDK documentation or landing page.
- A screenshot of a command-line application titled "Authorisation and translation via console application".
- Logos for Autodesk Forge, Node.js, and various programming frameworks (React, Angular, etc.) at the bottom.

# 学習リソース：Learn Forge

- <https://learnforge.autodesk.io/>

The screenshot shows the left sidebar with various navigation links and the main content area for the 'Autodesk Forgeについて' page. A blue arrow points to the 'JA' link in the top right corner of the header, which is underlined to indicate it's the selected language.

Type to search

EN JA ZH-CN ZH-TW

Autodesk Forge

ホーム

コーディングを開始する前に

ツール

OAuth

モデルを表示する

サーバを作成する

認証する

ファイルを OSS にアップロードする

ファイルを変換する

ビューアに表示する

BIM 360 と Fusion のモデルを表示する

サーバを作成する

## Autodesk Forge について

クイックスタートガイドを使用して、認証、データ管理、ファイル変換、およびモデルレンダリングの基本について説明します。

## 概要

Forge を使用すると、設計およびエンジニアリングのデータを活用して、製造、メディア/エンターテインメント、アーキテクチャ、エンジニアリング、建設のカスタム ソフトウェア アプリケーションおよび接続済みのワークフローを開発できます。

- **3D モデルをブラウザで直接表示する:** ビューアを使用すると、追加のソフトウェアをインストールしなくても、設計ファイルに関する 50 以上の形式のメタデータを埋め込み、操作して、ブラウザに取得することができます。
- **データを一元管理する:** Data Management API を使用すると、A360、Fusion、およびオブジェクトストレージサービス全体のデータにアクセスできます。

# Forge を始めるには?

- まずは <https://forge.autodesk.com/> へ



Autodesk ID で  
サインイン



Forge アプリ作成



開発

- 目的はデベロッパー キーの取得
  - Client ID** (別名 : Consumer Key)
  - Client Secret** (別名 : Consumer Secret)

# My Apps からのアプリ登録

どのAPIを使うか指定

CREATE APP >

The screenshot shows the Autodesk My Apps API registration interface. A large yellow rectangular border highlights the central area where APIs are selected. Inside this area, there are two rows of four boxes each, representing different APIs. Each box contains an icon and the API name, with a blue circular checkbox in the top right corner. All ten checkboxes are checked. The APIs listed are:

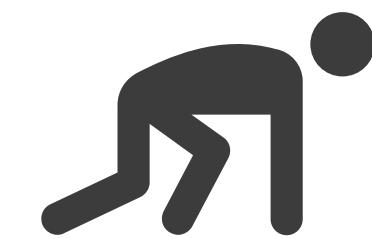
- Autodesk Construction Cloud API
- BIM 360 API
- Data Exchange API
- Data Management API
- Design Automation API
- Model Derivative API
- Premium Reporting API
- Reality Capture API
- Token Flex Usage Data API
- Webhooks API

At the bottom right of the highlighted area is a large orange button labeled "CREATE APP >". To the right of the button is a large yellow arrow pointing left.



# Design Automation を利用する ソリューション

# Forge の主なソリューション



Viewer ソリューション

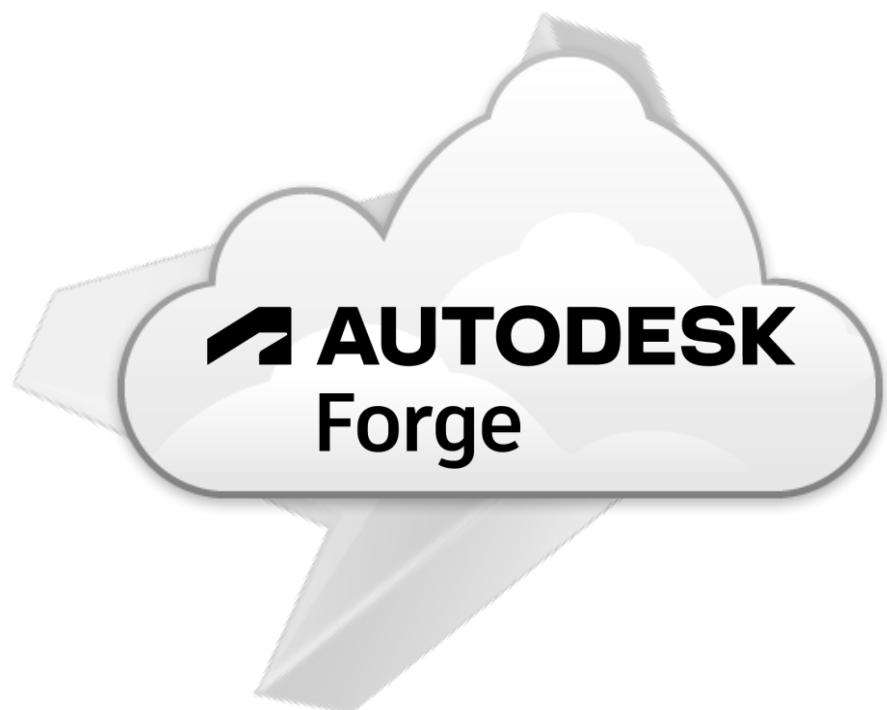
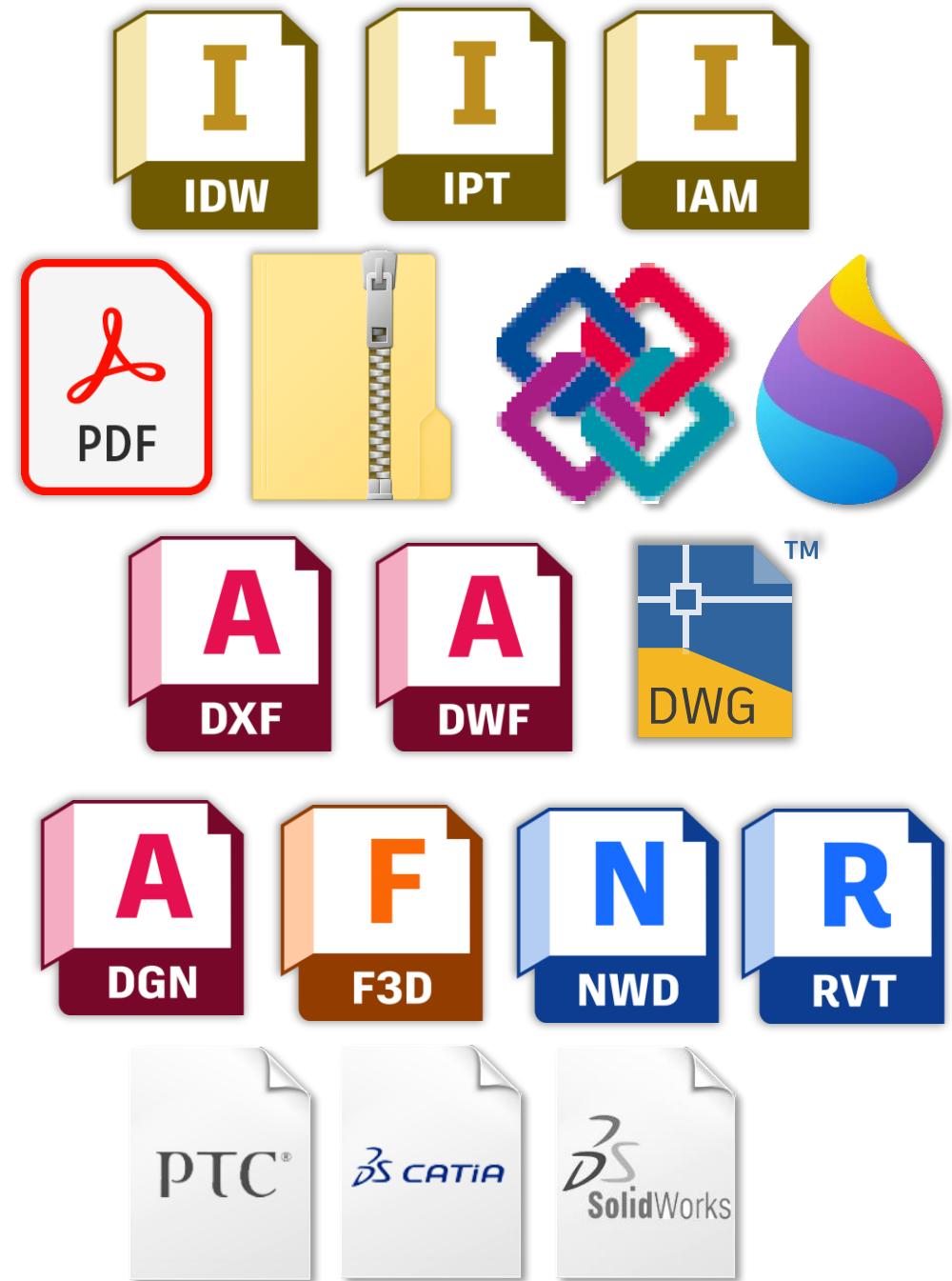


ストレージ統合ソリューション

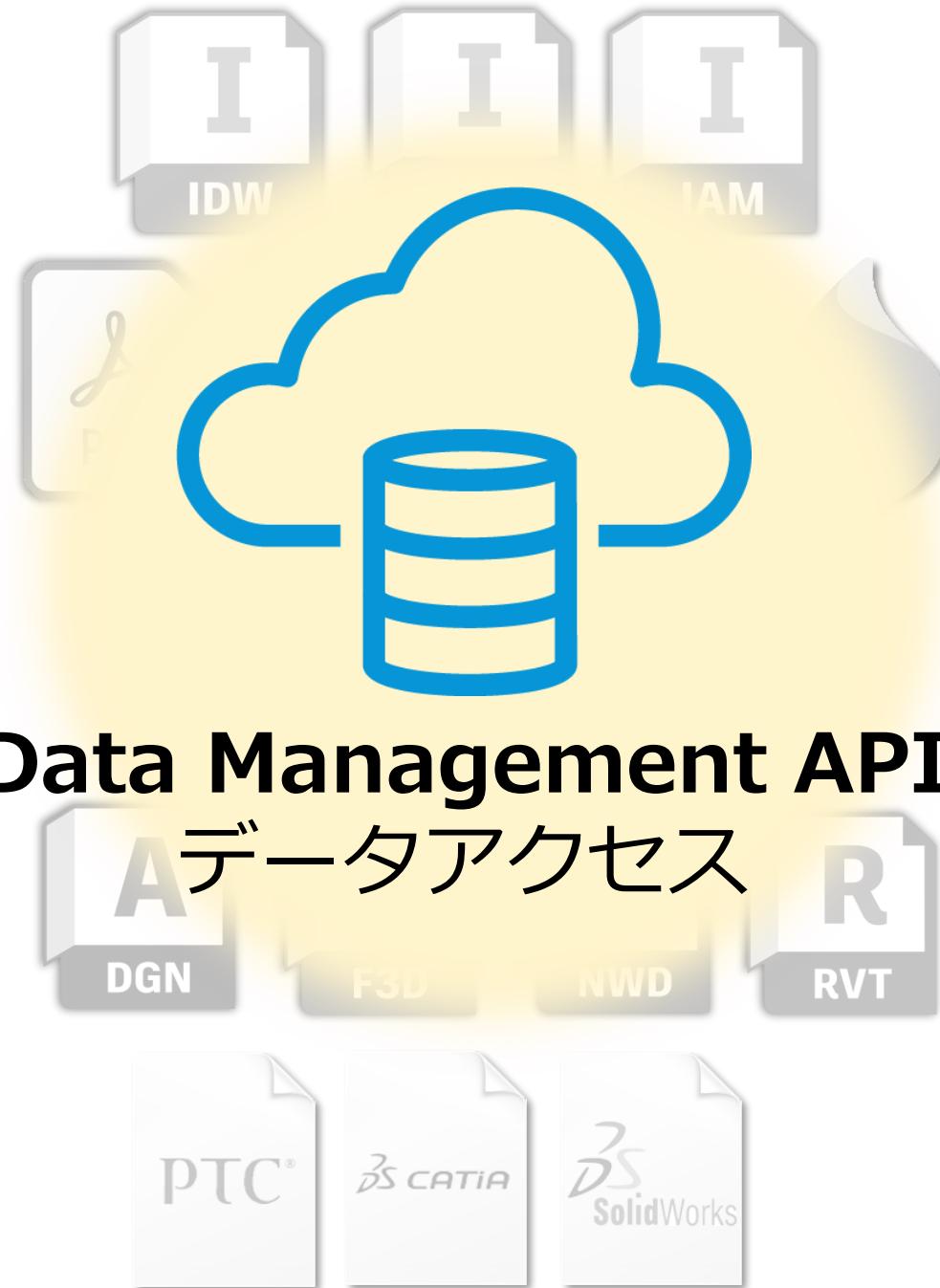


自動化ソリューション

# Viewer ソリューション

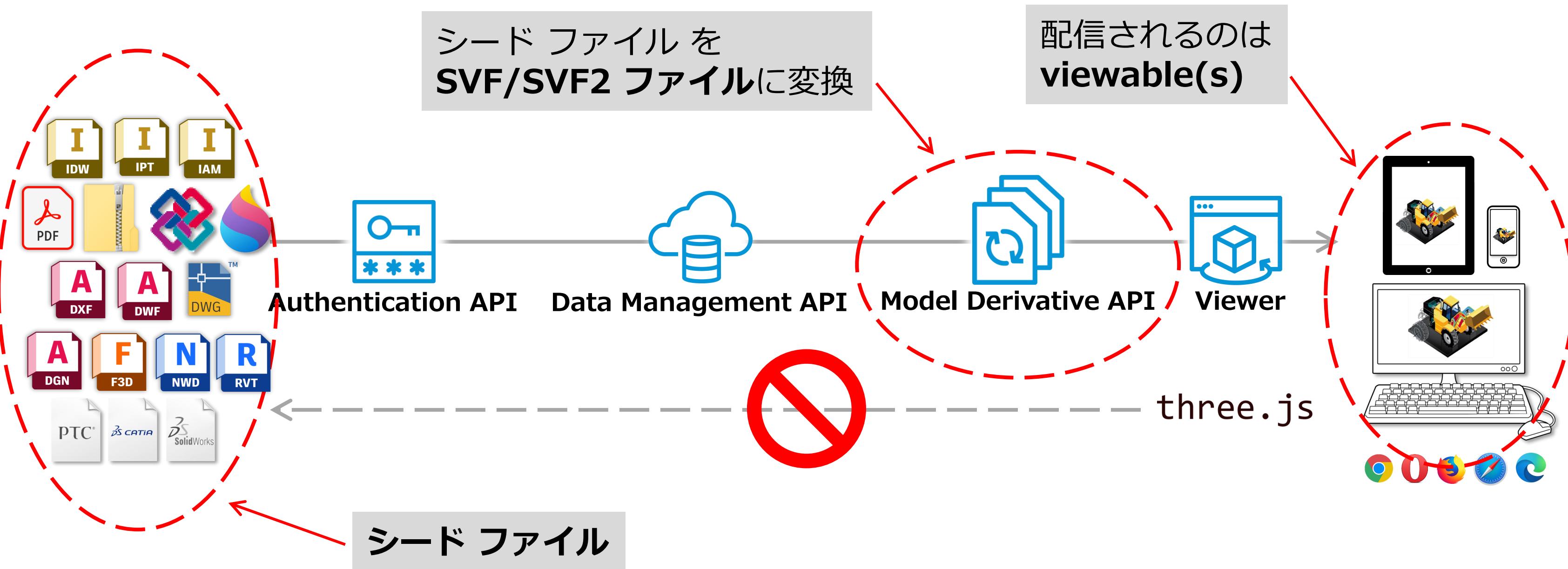


# Viewer ソリューション



# Forge Viewer ソリューションのながれ (OSS)

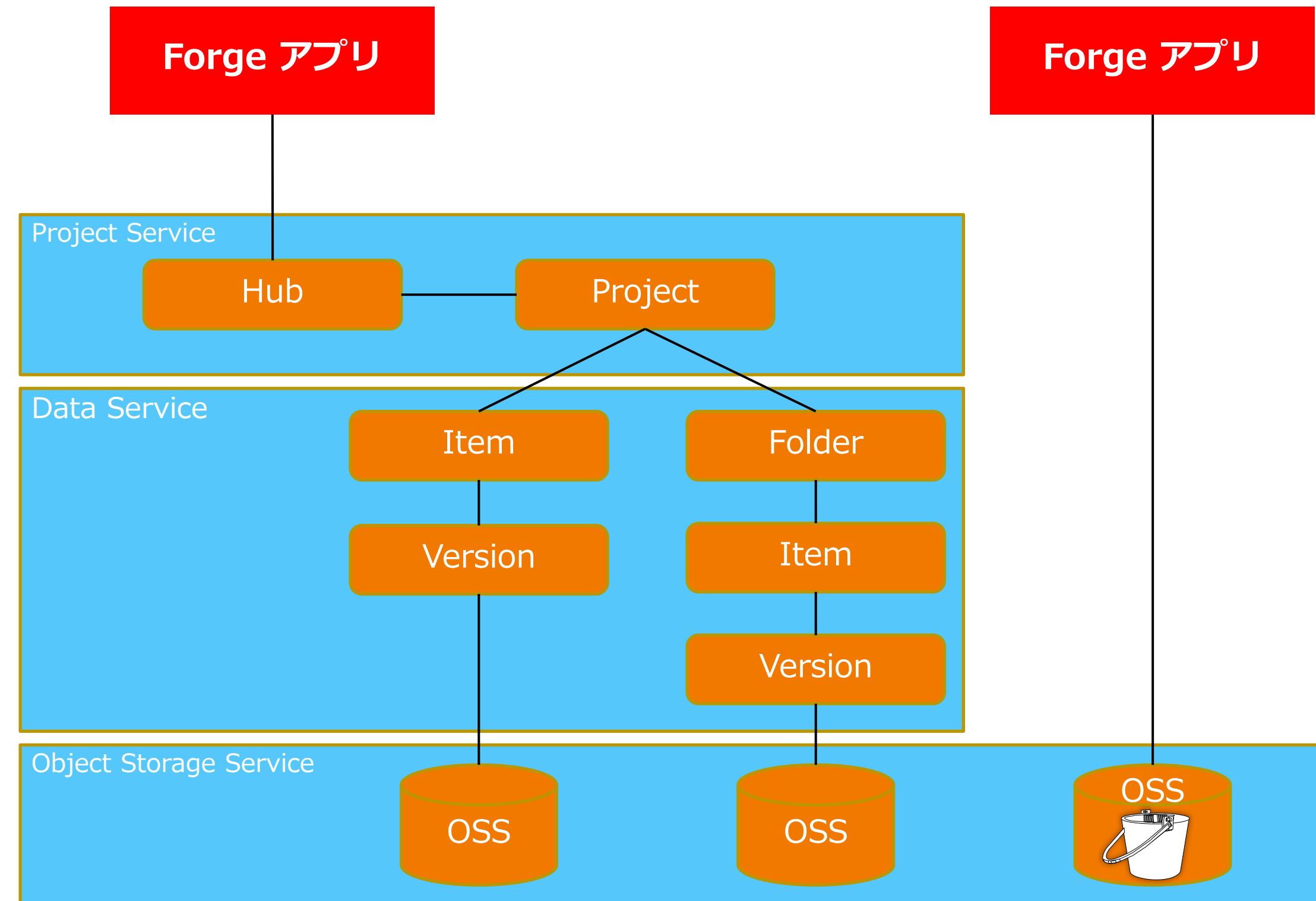
- Forge Viewer からのオリジナルデータへの反映は不可



# オートデスク ストレージサービス

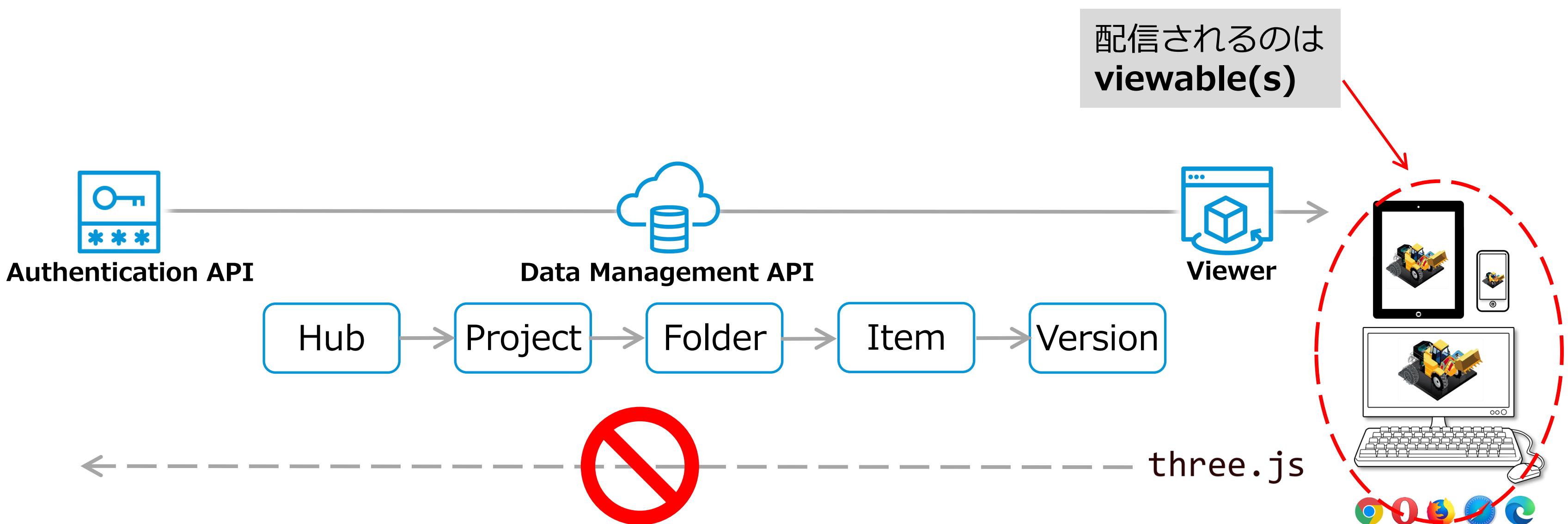
- プロジェクトベース
  - A360 Personal、Fusion Team、Autodesk Docs、BIM 360 Docs
  - **Forge Data Management API** でアクセス可能
- フォルダベース
  - A360 Drive、Autodesk Drive
  - プロジェクトベースのストレージ領域とは独立
  - **API は未公開のため未サポート**

# オートデスクストレージサービスの構造

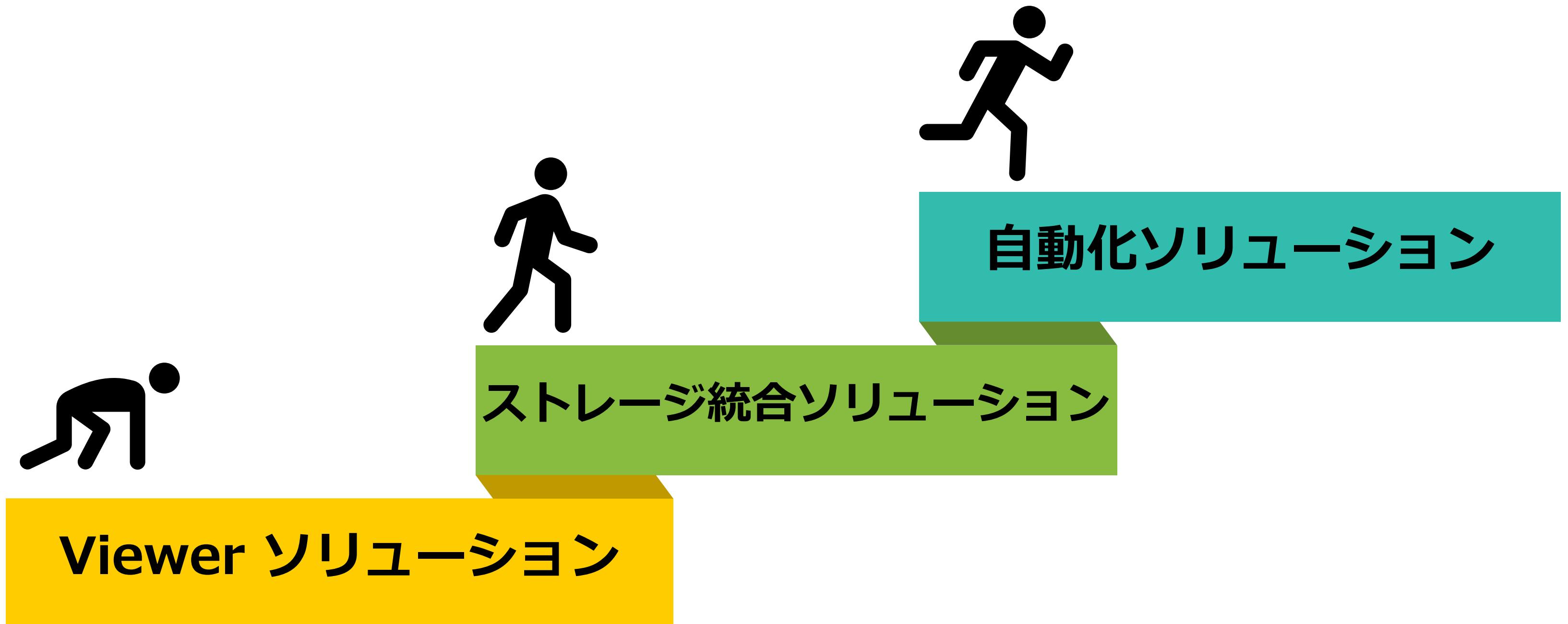


# Forge Viewer ソリューションのながれ (Hub)

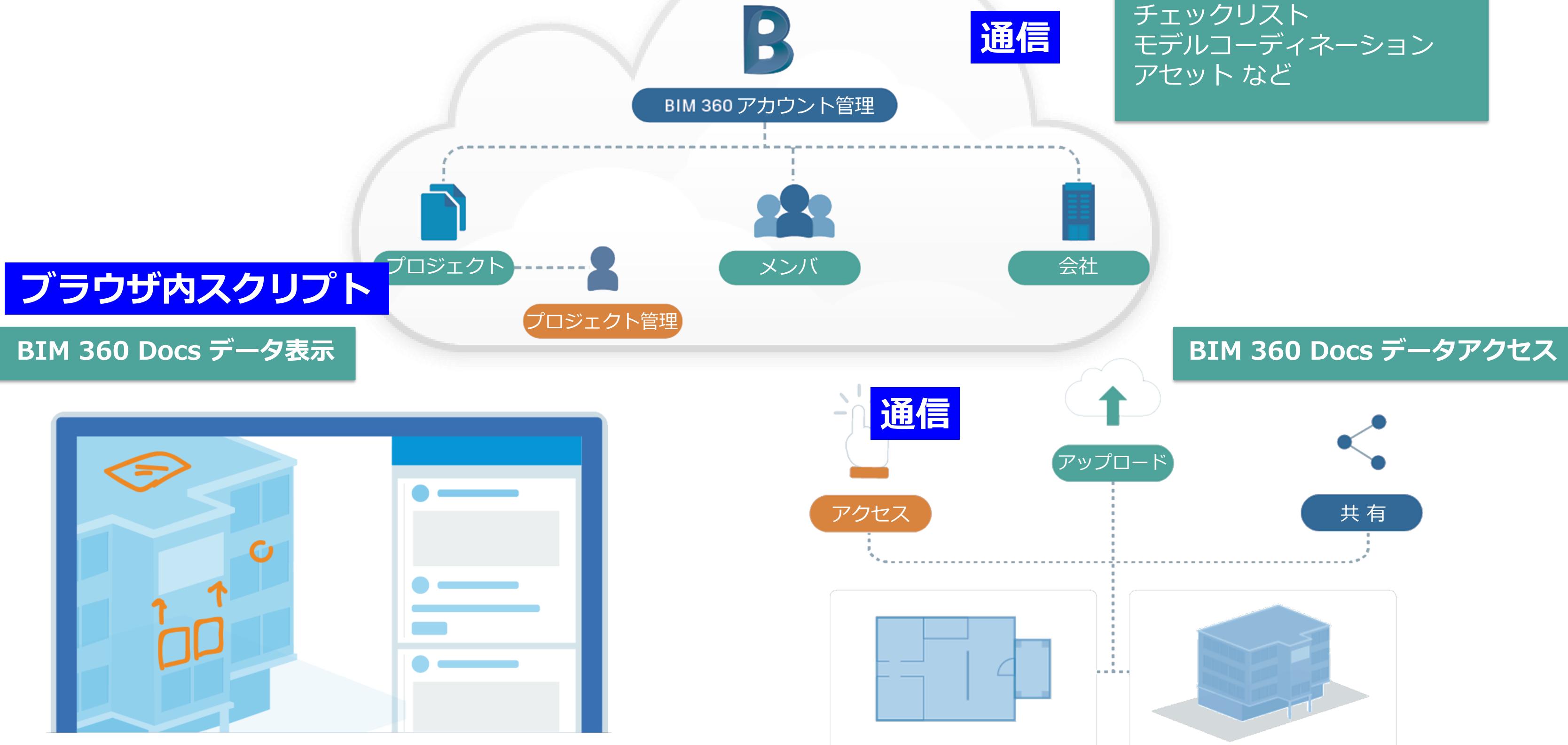
- ストレージサービスが実行した変換結果を利用可



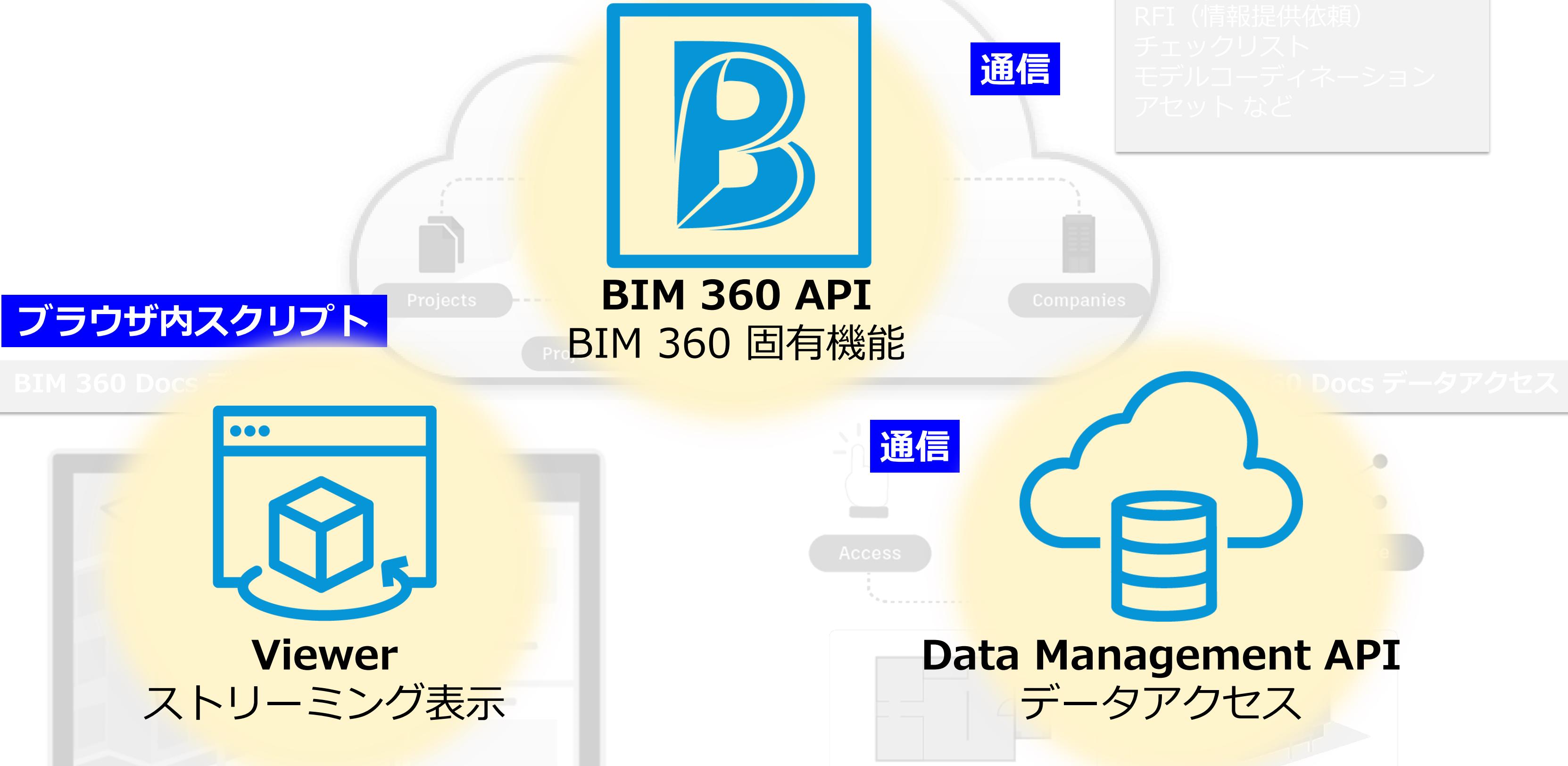
# Forge の主なソリューション



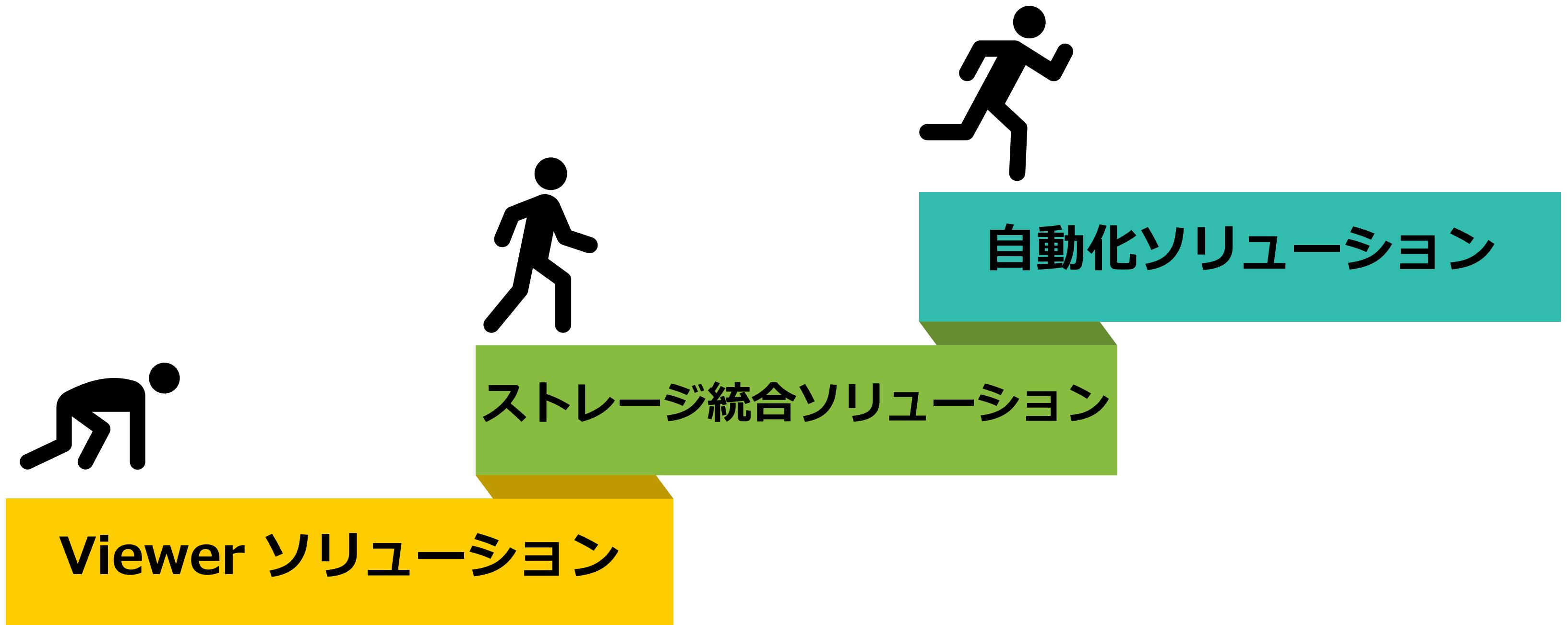
# BIM 360 統合ソリューション



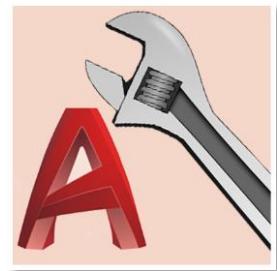
# BIM 360 統合ソリューション



# Forge の主なソリューション



# オートデスク デスクトップ製品と カスタマイズテクノロジ (API)



## ■ AutoCAD

- AutoLISP : AutoCAD 機能として独自実装
- ObjectARX : ネイティブ C++
- ActiveXオートメーション : Component Object Model
- AutoCAD .NET API : .NET Framework



## ■ Revit

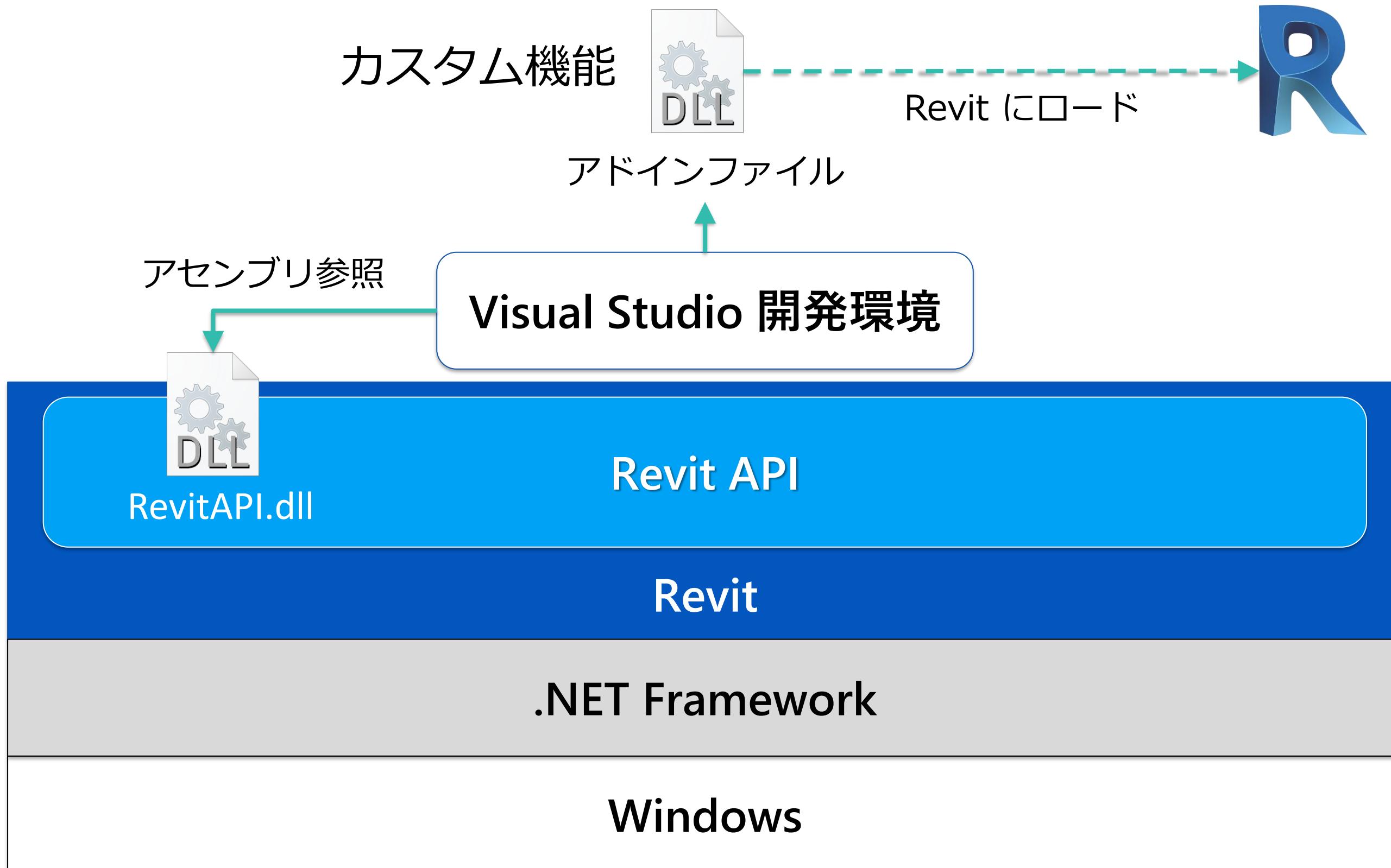
- Revit API : .NET Framework



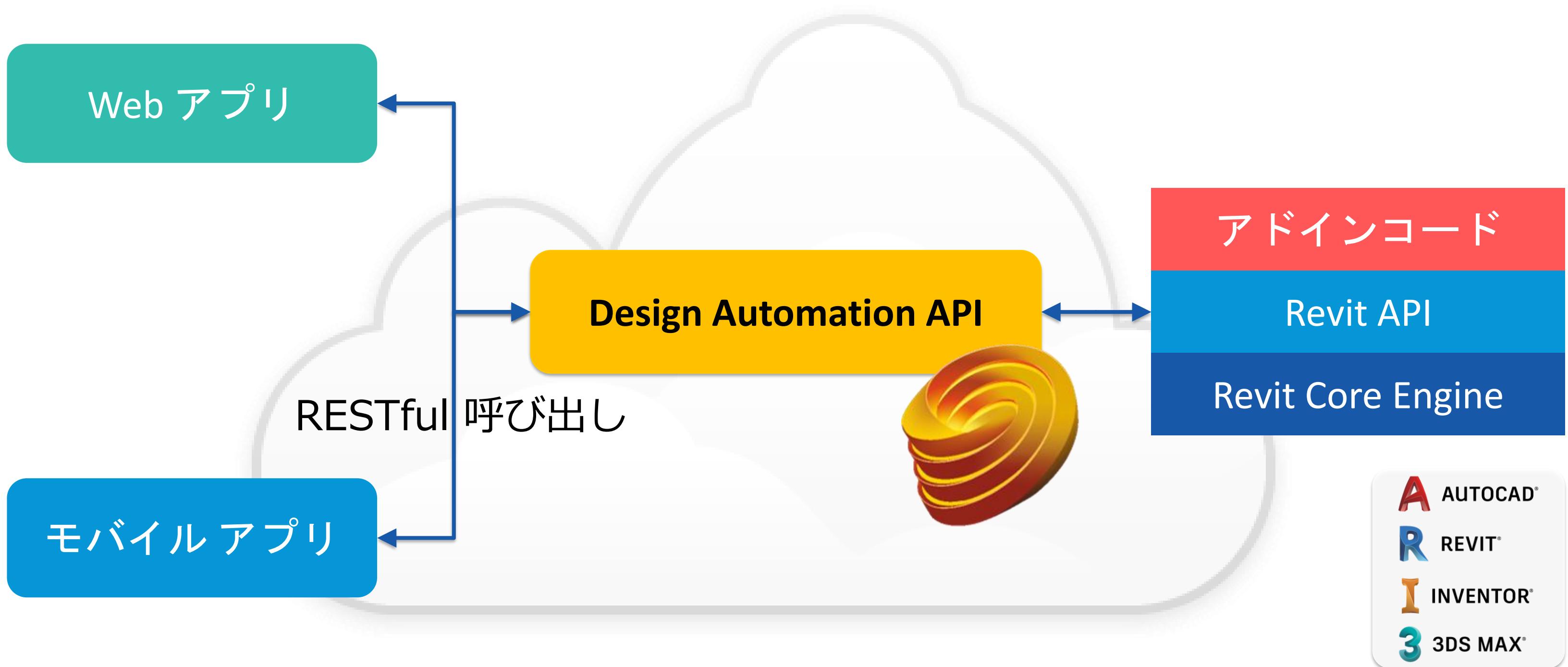
## ■ Inventor

- Inventor API : Component Object Model  
VBA, iLogic

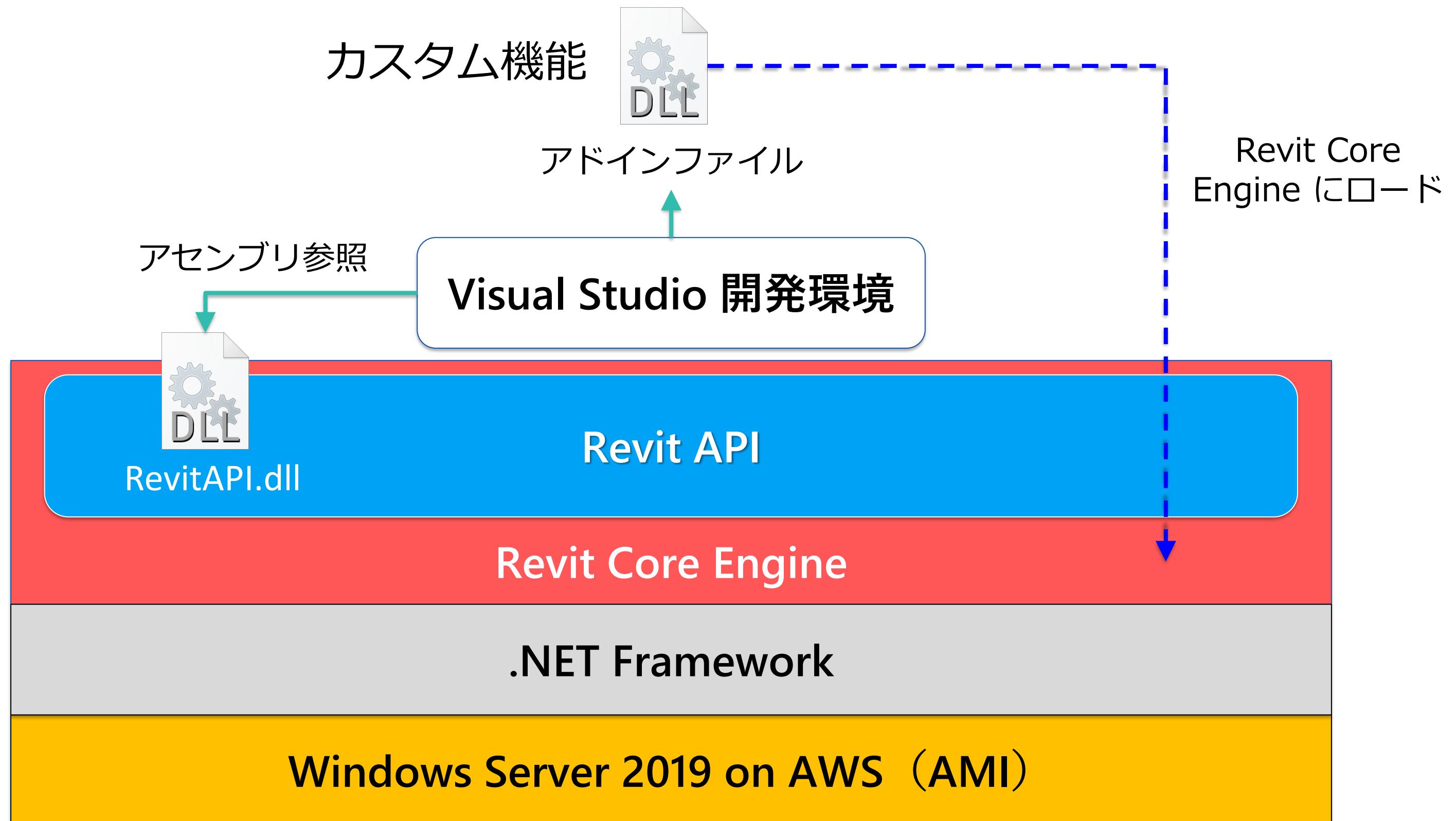
# Revit アドイン開発の仕組み



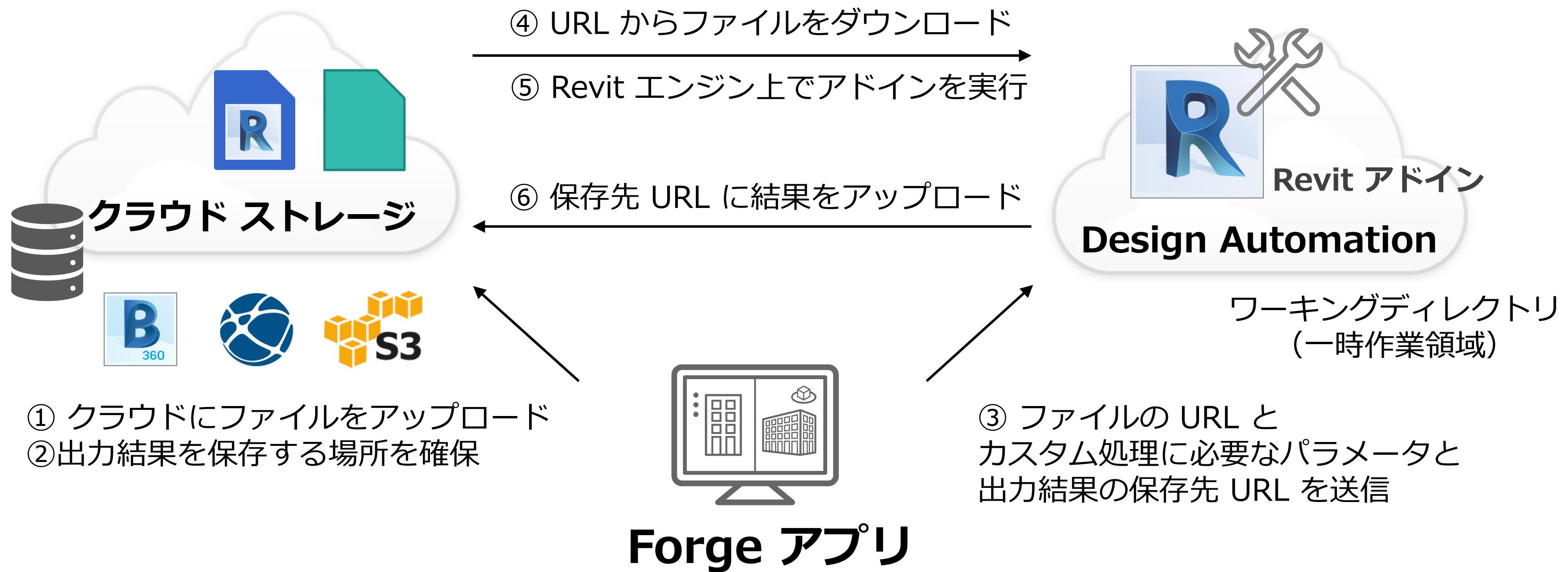
# Revit アドインをクラウドで実行するための API



# Revit アドイン開発の仕組み

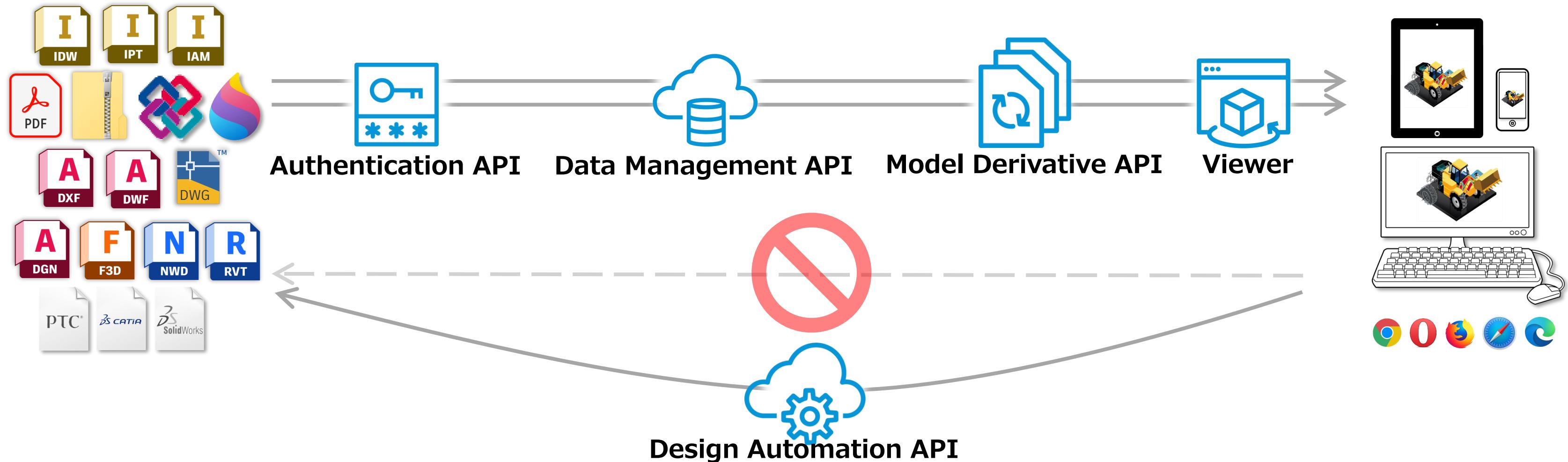


# Design Automation for Revit のワークフロー



# Viewer と Design Automation API の連携

- Design Automation API の連携は可能
- AutoCAD、Revit、Inventor、3ds Max に対応



# Design Automation API を正しく理解しましょう

## 対話的な表示/編集機能は ありません

- Revit Web 版 (Revit UI 画面) のようなものではありません

## ビューア機能は ありません

- 必要に応じて Forge Viewer の利用を検討出来ます

## エンドユーザ向けのサービスでは ありません

- 開発者向けのサービスです

## サーバー モジュールでは ありません

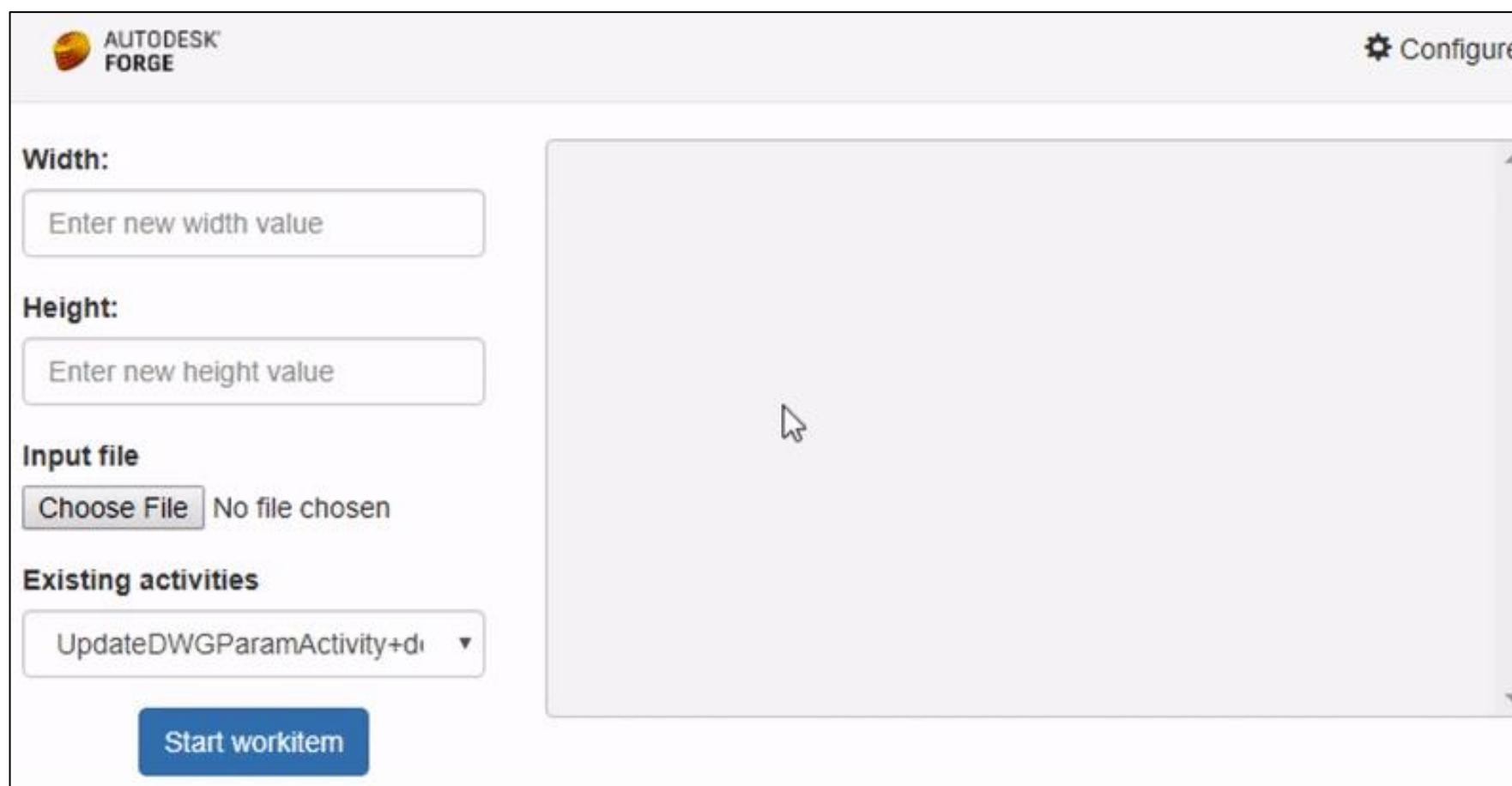
- オンプレミス(プライベート)サーバー版はありません



# Learn Autodesk Forge

# Forge APIs を学習するためのチュートリアル

- Learn Autodesk Forge (日本語)
  - 「モデルを修正する」セクション
    - Forge アプリ専用のストレージ (OSS Bucket) を作成して、デザインファイルをアップロードし、Revit, Inventor, AutoCAD, 3ds Max のモデルのパラメータを変更後、ファイルに保存してダウンロード

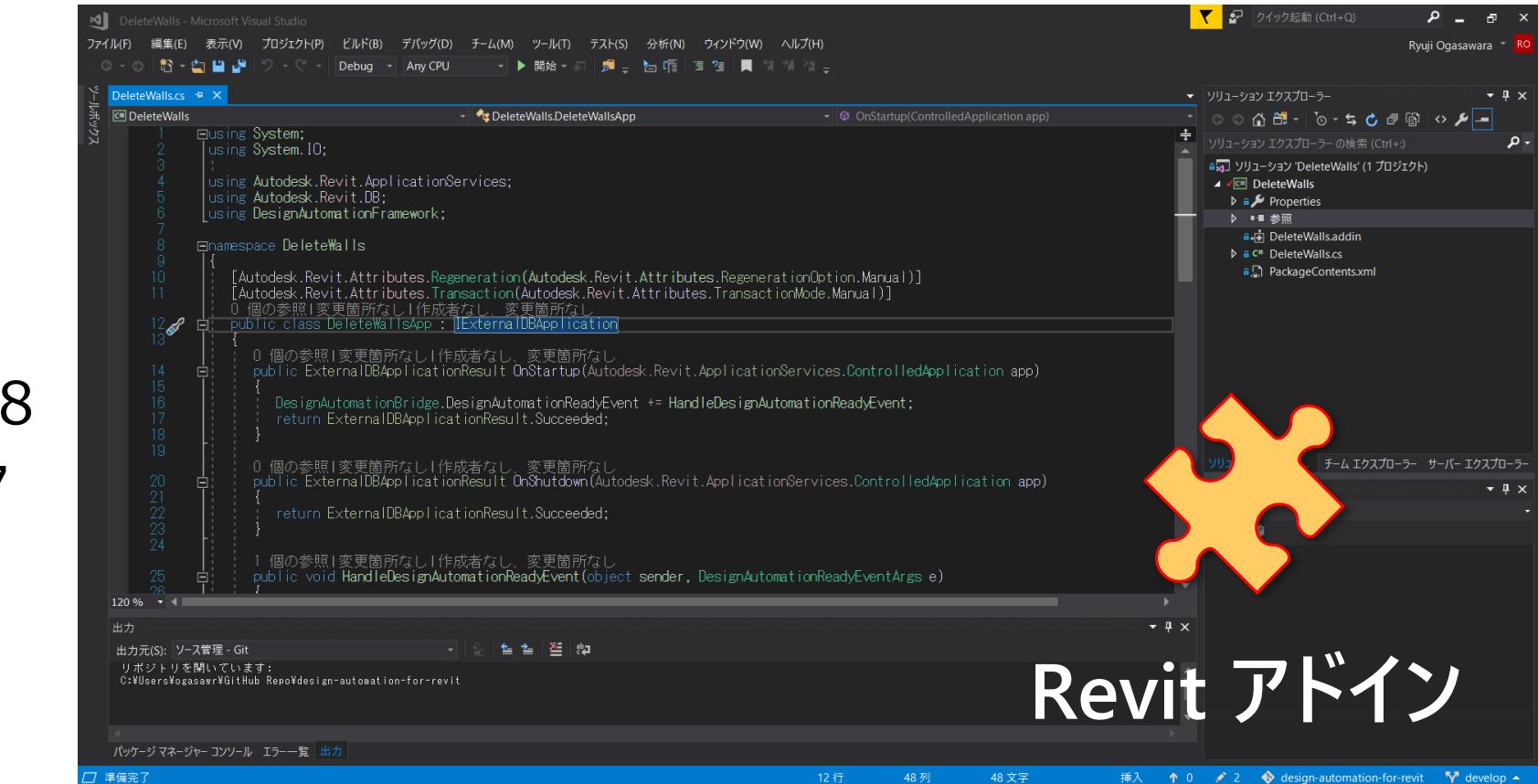


## 言語の選択

- Node.js
- .NET Core

# Design Automation for Revit アドインの開発環境

- Revit API
  - C#、VB.NET、マネージ C++、その他
- .NET Framework
  - Revit 2021 - 2023 -> .NET Framework 4.8
  - Revit 2019, 2020 -> .NET Framework 4.7
- Microsoft Visual Studio
  - 各バージョンに一致するアセンブリ出力が可能なものの
- 参照ライブラリ
  - RevitAPI.dll
  - Nuget パッケージ
    - Autodesk.Forge.DesignAutomation.Revit
    - Newtonsoft.Json



```

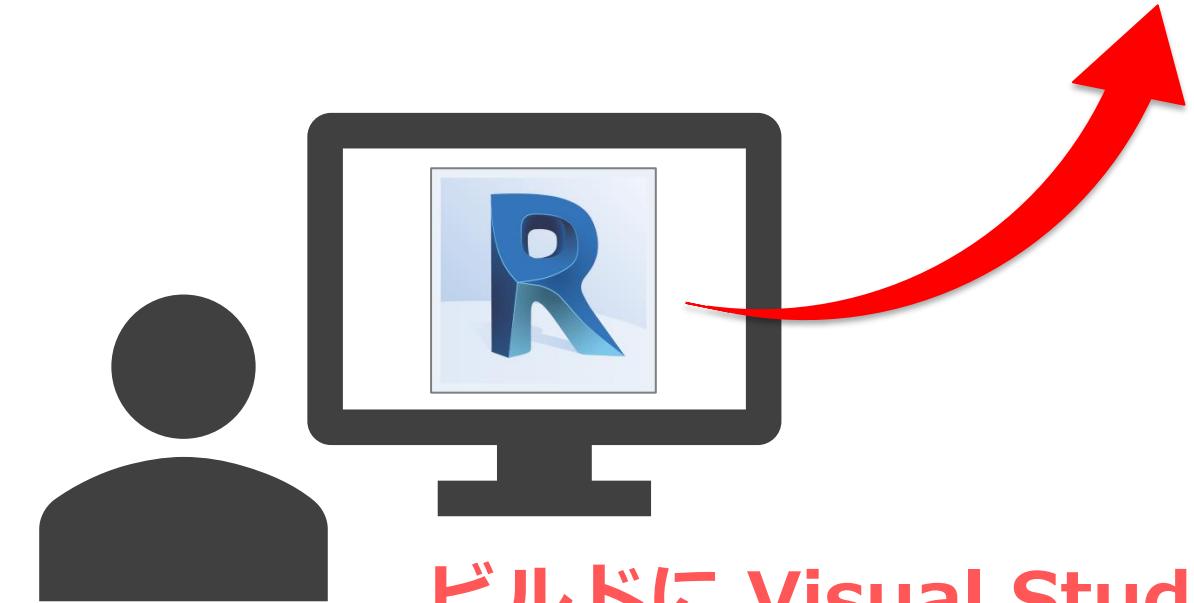
using System;
using System.IO;
using Autodesk.Revit.ApplicationServices;
using Autodesk.Revit.DB;
using DesignAutomationFramework;

namespace DeleteWalls
{
    [Autodesk.Revit.Attributes.Regeneration(Autodesk.Revit.Attributes.RegenerationOption.Manual)]
    [Autodesk.Revit.Attributes.Transaction(Autodesk.Revit.Attributes.TransactionMode.Manual)]
    public class DeleteWallsApp : IExternalDBApplication
    {
        public ExternalDBApplicationResult OnStartup(Autodesk.Revit.ApplicationServices.ControlledApplication app)
        {
            DesignAutomationBridge.DesignAutomationReadyEvent += HandleDesignAutomationReadyEvent;
            return ExternalDBApplicationResult.Succeeded;
        }

        public ExternalDBApplicationResult OnShutdown(Autodesk.Revit.ApplicationServices.ControlledApplication app)
        {
            return ExternalDBApplicationResult.Succeeded;
        }

        public void HandleDesignAutomationReadyEvent(object sender, DesignAutomationEventArgs e)
        {
        }
    }
}

```



# 今回のチュートリアルで使用する開発環境

- Visual Studio 2022 (2019でも可)
  - Community (無償)
  - Professional (有償)
- Revit 2019 ~ 2023
- 7zip
- ngrok ツール
- Google Chrome ブラウザ
- Git for Windows (必須ではありません)

[Design Automation API for Revit - Learn Autodesk Forge チュートリアル .NET Core Sample のセットアップ](#)



サーバーを作成する

# ngrok ツール Agent v3 の注意点

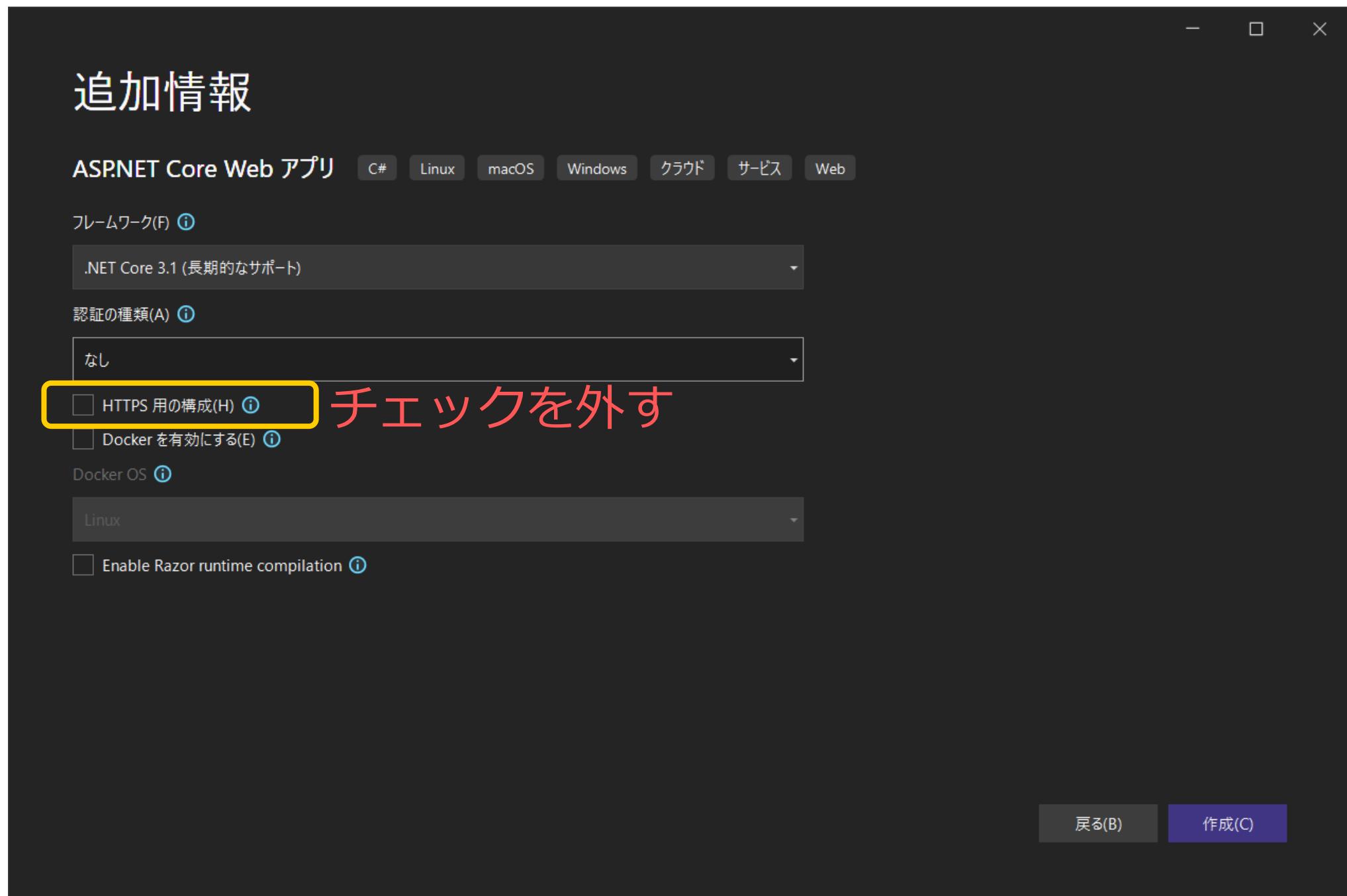
- デフォルトでは HTTPS のみ有効のため、下記のコマンドを実行してください。

```
ngrok http 3000 --host-header=localhost:3000 --scheme http
```

- なお、事前にユーザー登録をしてトークンをセットしておく必要もございます。

```
ngrok config add-authtoken Your Authtoken
```

# プロジェクト新規作成時の注意点



# プロジェクトの設定を確認する

Visual Studio 上で forgeSample プロジェクトをダブルクリック、または forgeSample.csproj をテキストエディタで開く

```
<Project Sdk="Microsoft.NET.Sdk.Web">

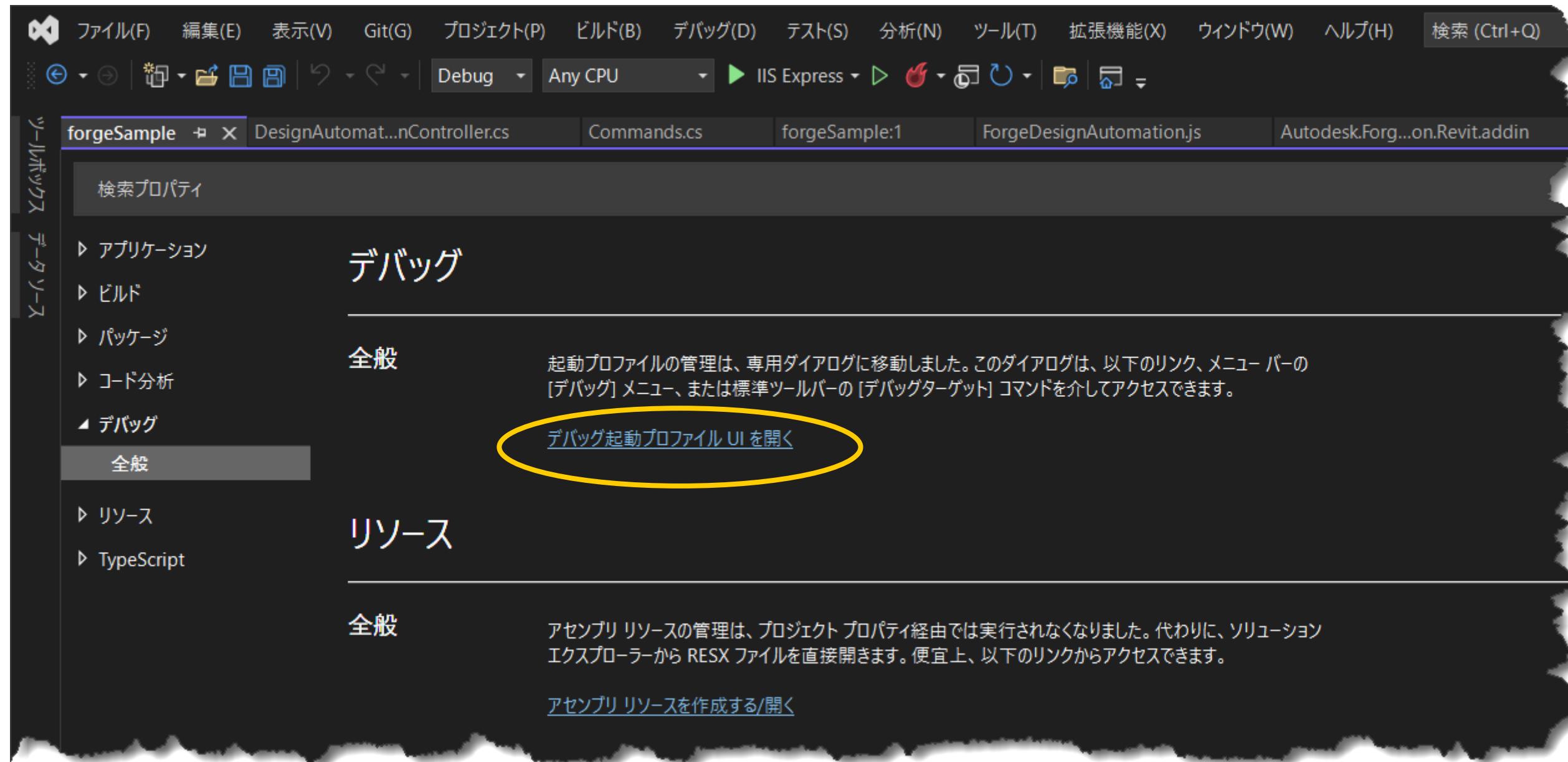
<PropertyGroup>
  <TargetFramework>netcoreapp3.1</TargetFramework>
</PropertyGroup>

<ItemGroup>
  <PackageReference Include="Autodesk.Forge" Version="1.9.0" />
  <PackageReference Include="Autodesk.Forge.DesignAutomation" Version="3.0.6" />
  <PackageReference Include="Microsoft.AspNetCore.Mvc.NewtonsoftJson" Version="3.1.26" />
  <PackageReference Include="Microsoft.AspNetCore.SignalR.Protocols.NewtonsoftJson" Version="3.1.26" />
</ItemGroup>

<ItemGroup>
  <Folder Include="wwwroot\$bundles\$" />
</ItemGroup>

</Project>
```

# Visual Studio 2022 での環境変数の設定の注意点



```
ASPNETCORE_ENVIRONMENT=Development,FORGE_CLIENT_ID=XXXXXXXXXXXX,FORGE_CLIENT_SE  
CRET=XXXXXXXXXXXX,FORGE_WEBHOOK_URL=http://XXX.XXXXXXX.jp.ngrok.io
```

# プロファイルの設定を確認する

Visual Studio 上で Properties/launchSettings.json をダブルクリック、またはテキストエディタで開く

```
{  
  "iisSettings": {  
    "windowsAuthentication": false,  
    "anonymousAuthentication": true,  
    "iisExpress": {  
      "applicationUrl": "http://localhost:3000",  
      "sslPort": 0  
    }  
  },  
  "profiles": {  
    "IIS Express": {  
      "commandName": "IISExpress",  
      "launchBrowser": true,  
      "environmentVariables": {  
        "ASPNETCORE_ENVIRONMENT": "Development",  
        "FORGE_WEBHOOK_URL": "http://XXXXXXXXXX.ngrok.io",  
        "FORGE_CLIENT_ID": "XXXXXXXXXX",  
        "FORGE_CLIENT_SECRET": "XXXXXXXXXX"  
      },  
      "applicationUrl": "http://localhost:3000"  
    },  
    "forgeSample": {  
      // 省略  
    }  
  }  
}
```

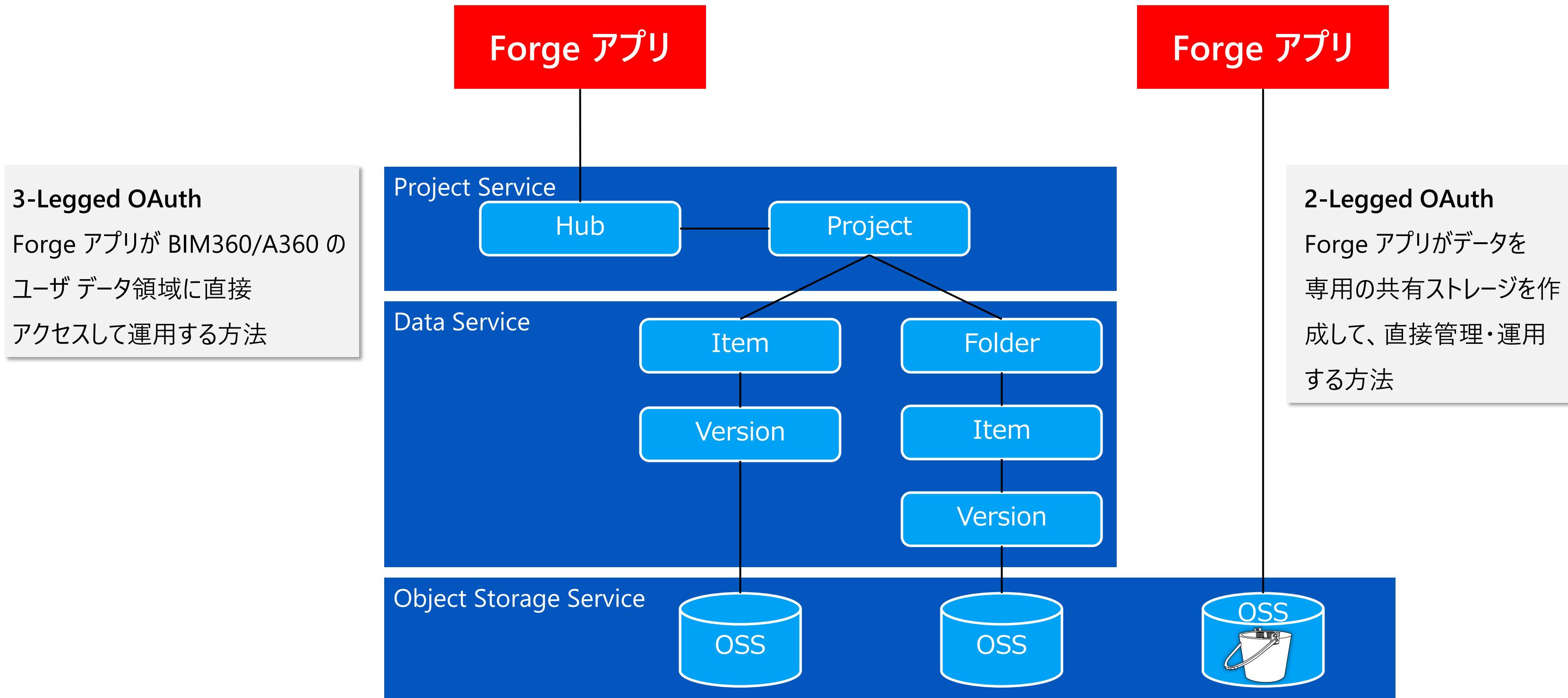
# Access Token (アクセストークン) とは

- クラウド リソースへのアクセス権限をチェックする仕組み
  - 有効期限が設定される
  - クライアントからの取得は CORS エラーになる（仕様）
- 生成に必要なもの
  - デベロッパキー
    - Client ID と Client Secret のペア
    - Forge ポータルで App (アプリ) 作成時に取得可能
  - Scope
    - リソースへのアクセス権限範囲を指定
    - 使用するリソースによって適切に使い分けが必要
    - 複数の Scope 文字列を半角スペースで結合して指定

# Authentication API によるアクセストークンの取得

- **2-legged Authentication**
  - エンドユーザーの許可（認可）を必要とするリソースにアクセスする必要がない
  - アプリは Forge からの認証を得ればアクセス可能（2者間通信）
  - 専用の共有ストレージ Object Storage Service (OSS) にアクセス
- **3-legged Authentication**
  - アプリが Autodesk ログイン ページにリダイレクトして、エンドユーザーのどのリソースにアクセスしたいかという情報を提供
  - ユーザーから同意が得られてアクセス許可（認可）を得た後にアプリにリダイレクト
  - オートデスク クラウドサービスのユーザストレージ領域にアクセス
- 認証・認可後に **Access Token**（アクセストークン）を発行 

# シナリオ/目的に応じた認証・認可

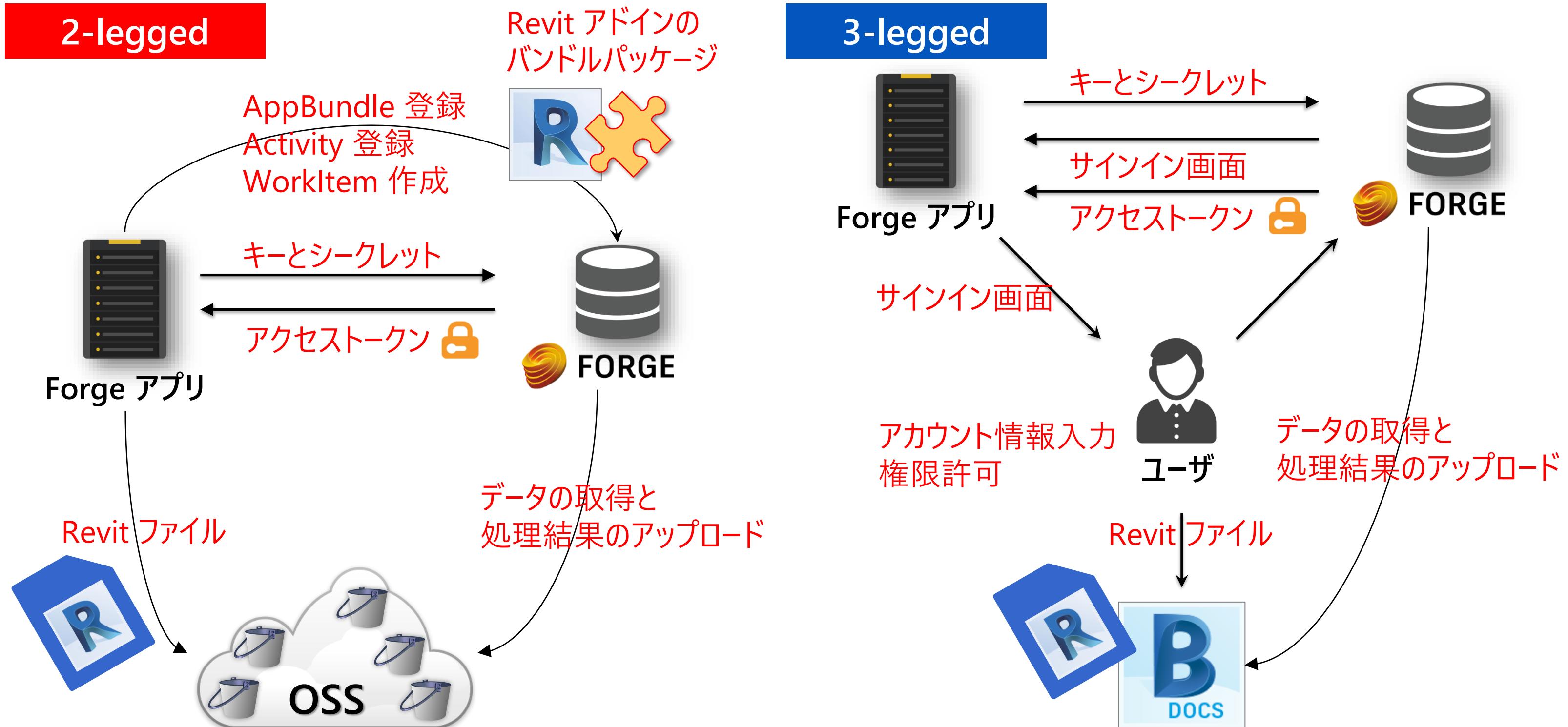


# 2-legged or 3-legged

- エンドポイント毎に必要とされるコンテキストは異なります。

<p><b>GET</b> <a href="#">projects/:project_id/items/:item_id</a></p> <p>Retrieves metadata for a specified item. Items represent word documents.</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>The tip version for the item is included in the <code>included</code> array.</li> <li>To retrieve metadata for multiple items, see the <a href="#">ListItems</a> command.</li> </ul> <p>New! Autodesk Construction Cloud platform (ACC). Note that this endpoint is part of the Autodesk Construction Cloud APIs, see the <a href="#">Autodesk Construction Cloud documentation</a>.</p> <p><b>Resource Information</b></p> <table border="1"> <tbody> <tr> <td>Method and URI</td> <td><b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/projects/:project_id/items/:item_id">https://developer.api.autodesk.com/oss/v2/projects/:project_id/items/:item_id</a></td> </tr> <tr> <td>Authentication Context</td> <td>user context optional</td> </tr> <tr> <td>Required OAuth Scopes</td> <td>data:read</td> </tr> </tbody> </table>	Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/projects/:project_id/items/:item_id">https://developer.api.autodesk.com/oss/v2/projects/:project_id/items/:item_id</a>	Authentication Context	user context optional	Required OAuth Scopes	data:read	<p><b>GET</b> <a href="#">users/me</a></p> <p>Retrieves the profile information about an end user, including the user's name, email address, and profile picture. This endpoint does not require full control permissions to create issues. However it does not currently provide full information about a user's account, such as their role or organization.</p> <p>New! Autodesk Construction Cloud platform (ACC). Note that this endpoint is part of the Autodesk Construction Cloud APIs, see the <a href="#">Autodesk Construction Cloud documentation</a>.</p> <p><b>Resource Information</b></p> <table border="1"> <tbody> <tr> <td>Method and URI</td> <td><b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/users/me">https://developer.api.autodesk.com/oss/v2/users/me</a></td> </tr> <tr> <td>Authentication Context</td> <td>user context required</td> </tr> <tr> <td>Required OAuth Scopes</td> <td>account:read</td> </tr> </tbody> </table>	Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/users/me">https://developer.api.autodesk.com/oss/v2/users/me</a>	Authentication Context	user context required	Required OAuth Scopes	account:read	<p><b>GET</b> <a href="#">buckets/:bucketKey/objects</a></p> <p>List objects in a bucket. It is only available to the bucket creator.</p> <p><b>Resource Information</b></p> <table border="1"> <tbody> <tr> <td>Method and URI</td> <td><b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects">https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects</a></td> </tr> <tr> <td>Authentication Context</td> <td>app-only</td> </tr> <tr> <td>Required OAuth Scopes</td> <td>data:read</td> </tr> </tbody> </table>	Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects">https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects</a>	Authentication Context	app-only	Required OAuth Scopes	data:read
Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/projects/:project_id/items/:item_id">https://developer.api.autodesk.com/oss/v2/projects/:project_id/items/:item_id</a>																			
Authentication Context	user context optional																			
Required OAuth Scopes	data:read																			
Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/users/me">https://developer.api.autodesk.com/oss/v2/users/me</a>																			
Authentication Context	user context required																			
Required OAuth Scopes	account:read																			
Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects">https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects</a>																			
Authentication Context	app-only																			
Required OAuth Scopes	data:read																			

# Design Automation API に必要なアクセストークン



# OSS 利用時に必要なスコープ\*

- バケットの作成やファイルの読み書きの権限が必要です。

<b>user-profile:read</b>	プロファイル (Autodesk ID) の表示
<b>user:read</b>	プロファイル (Autodesk ID) の読み取り
<b>user:write</b>	プロファイル (Autodesk ID) の書き込み
<b>viewables:read</b>	変換後のデザインデータ (SVF) の読み取り (表示)
<b>data:read</b>	ストレージ データの読み取り
<b>data:write</b>	ストレージ データの書き込み (編集)
<b>data:create</b>	ストレージ データの作成
<b>data:search</b>	ストレージ データの検索
<b>bucket:create</b>	新しい Bucket の作成
<b>bucket:read</b>	Bucket の読み取り
<b>bucket:update</b>	Bucket の更新
<b>bucket:delete</b>	Bucket の削除
<b>code:all</b>	コードの生成または実行 (Design Automation API)
<b>account:read</b>	アプリやサービス アカウントの読み取り
<b>account:write</b>	アプリやサービス アカウントの書き込み



# Design Automation API 利用時に必要なスコープ<sup>®</sup>

- コードの生成と実行の権限に相当する、code:all を指定する必要があります。

<b>user-profile:read</b>	プロファイル (Autodesk ID) の表示
<b>user:read</b>	プロファイル (Autodesk ID) の読み取り
<b>user:write</b>	プロファイル (Autodesk ID) の書き込み
<b>viewables:read</b>	変換後のデザインデータ (SVF) の読み取り (表示)
<b>data:read</b>	ストレージ データの読み取り
<b>data:write</b>	ストレージ データの書き込み (編集)
<b>data:create</b>	ストレージ データの作成
<b>data:search</b>	ストレージ データの検索
<b>bucket:create</b>	新しい Bucket の作成
<b>bucket:read</b>	Bucket の読み取り
<b>bucket:update</b>	Bucket の更新
<b>bucket:delete</b>	Bucket の削除
<b>code:all</b>	コードの生成または実行 (Design Automation API)
<b>account:read</b>	アプリやサービス アカウントの読み取り
<b>account:write</b>	アプリやサービス アカウントの書き込み



# 必要なスコープの見分け方

- API リファレンスで Endpoint 毎に必要なスコープをチェック！

**GET buckets**

This endpoint will return the buckets owned by the application. This endpoint supports pagination.

**Resource Information**

---

Method and URI	<b>GET</b> <a href="https://developer.api.autodesk.com/oss/v2/buckets">https://developer.api.autodesk.com/oss/v2/buckets</a>
Authentication Context	app-only
Required OAuth Scopes	<b>bucket :read</b>
Data Format	JSON

bucket:read

**POST buckets**

Creates a bucket. Buckets are arbitrary spaces that are created by applications and are used to store objects for later retrieval. A bucket is owned by the application that creates it.

**Resource Information**

---

Method and URI	<b>POST</b> <a href="https://developer.api.autodesk.com/oss/v2/buckets">https://developer.api.autodesk.com/oss/v2/buckets</a>
Authentication Context	app-only
Required OAuth Scopes	<b>bucket :create</b>
Data Format	JSON

bucket:create

# OAuthController.cs

```
private static dynamic InternalToken { get; set; }

public static async Task<dynamic> GetInternalAsync()
{
    if (InternalToken == null || InternalToken.ExpiresAt < DateTime.UtcNow)
    {
        InternalToken = await Get2LeggedTokenAsync(new Scope[] { Scope.BucketCreate, Scope.BucketRead, Scope.BucketDelete,
Scope.DataRead, Scope.DataWrite, Scope.DataCreate, Scope.CodeAll });
        InternalToken.ExpiresAt = DateTime.UtcNow.AddSeconds(InternalToken.expires_in);
    }

    return InternalToken;
}

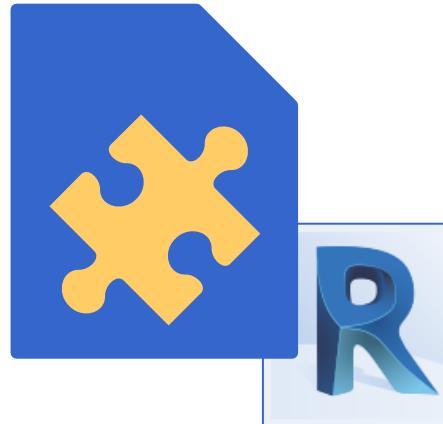
private static async Task<dynamic> Get2LeggedTokenAsync(Scope[] scopes)
{
    TwoLeggedApi oauth = new TwoLeggedApi();
    string grantType = "client_credentials";
    dynamic bearer = await oauth.AuthenticateAsync(
        GetAppSetting("FORGE_CLIENT_ID"),
        GetAppSetting("FORGE_CLIENT_SECRET"),
        grantType,
        scopes);
    return bearer;
}
```



# Design Automation の仕組み

# 3つのキーワード

## AppBundle

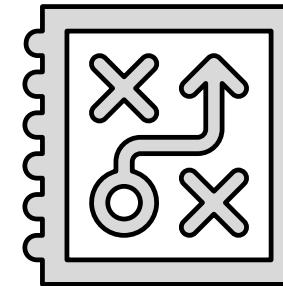


Revit アドインのパッケージ

```
Id : CustomApp  
Engine : 2023  
Description: Test custom app  
Package: Storage URL
```

Revit の .NET API で作成したアセンブリや関連ファイルを ZIP 圧縮してアップロードし、AppBundle として登録する。

## Activity

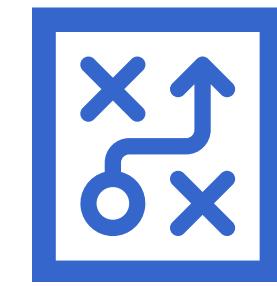


実行されるアクションの定義

```
Id: UpdateParamActivity  
Input Parameter A : RVT, TXT  
Output Parameter B: RVT  
AppBundle: CustomApp
```

カスタム処理の雛型を定義する。  
.NET アセンブリ内でどんなデータを入力して、どんなデータを出力するか定義する。

## WorkItem



指定のアクションを呼び出すジョブ

```
Id: 返却される文字列  
Activity : UpdateParamActivity  
Input Parameter A : File URL  
Output Parameter B: Storage URL
```

REST API でリクエストするジョブ。  
対象のモデルやテキストデータ、出力先の URL と、実行する Activity を指定する。

# AppBundle と Activity の登録手順

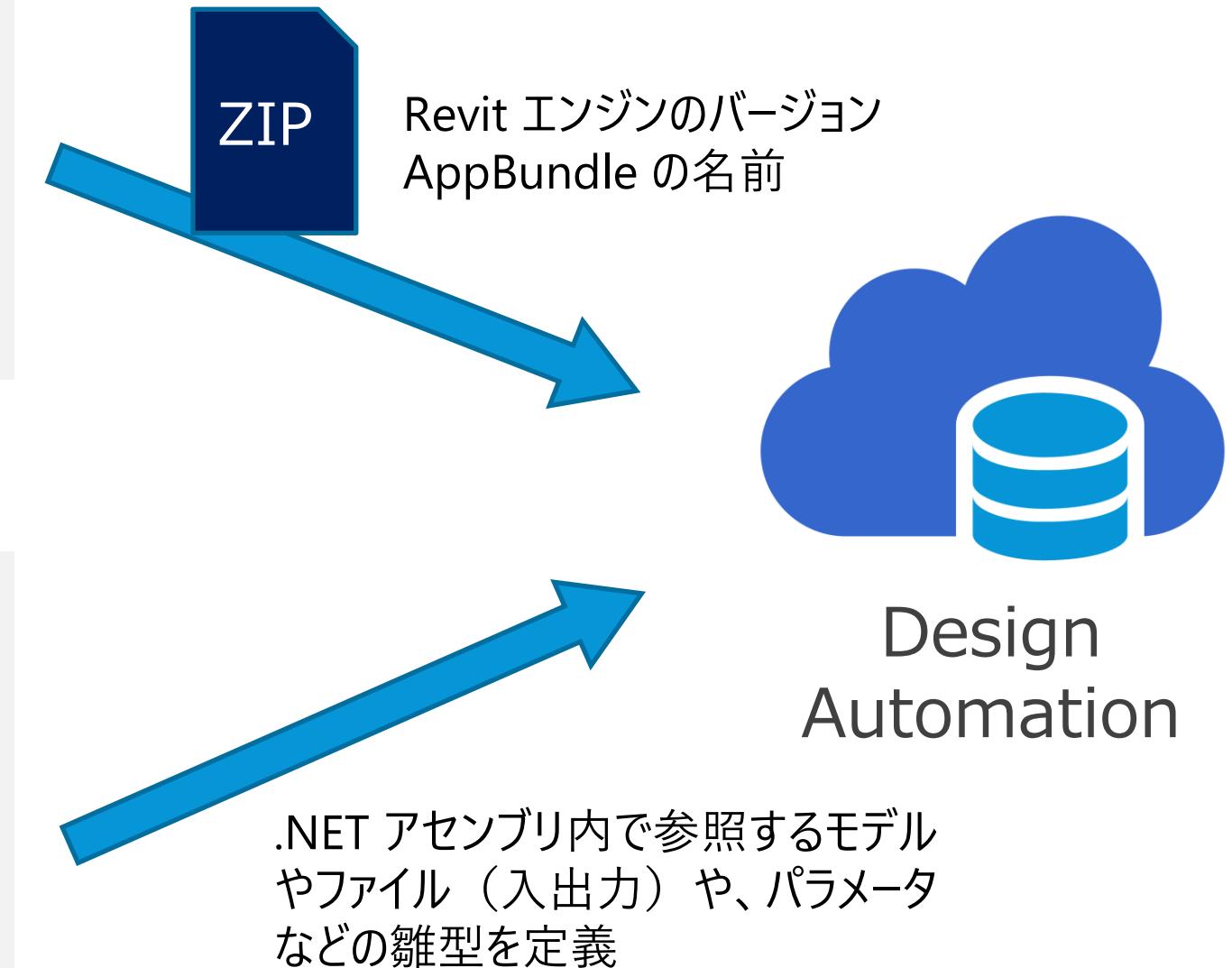
## AppBundle を登録

1. 新しい AppBundle を作成
2. バンドルパッケージをアップロード
3. エイリアスとバージョンを設定



## Activity を登録

1. 新しい Activity を作成
2. エイリアスとバージョンを設定
3. 必要があればロケールを設定



# Alias と Version

- AppBundle と Activity は、それぞれバージョンを追加していくことができます。
- 特定のバージョンに任意のラベルで名前をつけてエイリアスを作成できます。

	Version	Alias
AppBundle	1	
	2	Production
	3	
	4	Test
	5	Development

	Version	Alias
Activity	1	
	2	
	3	Production
	4	Beta
	5	Test
	6	Development

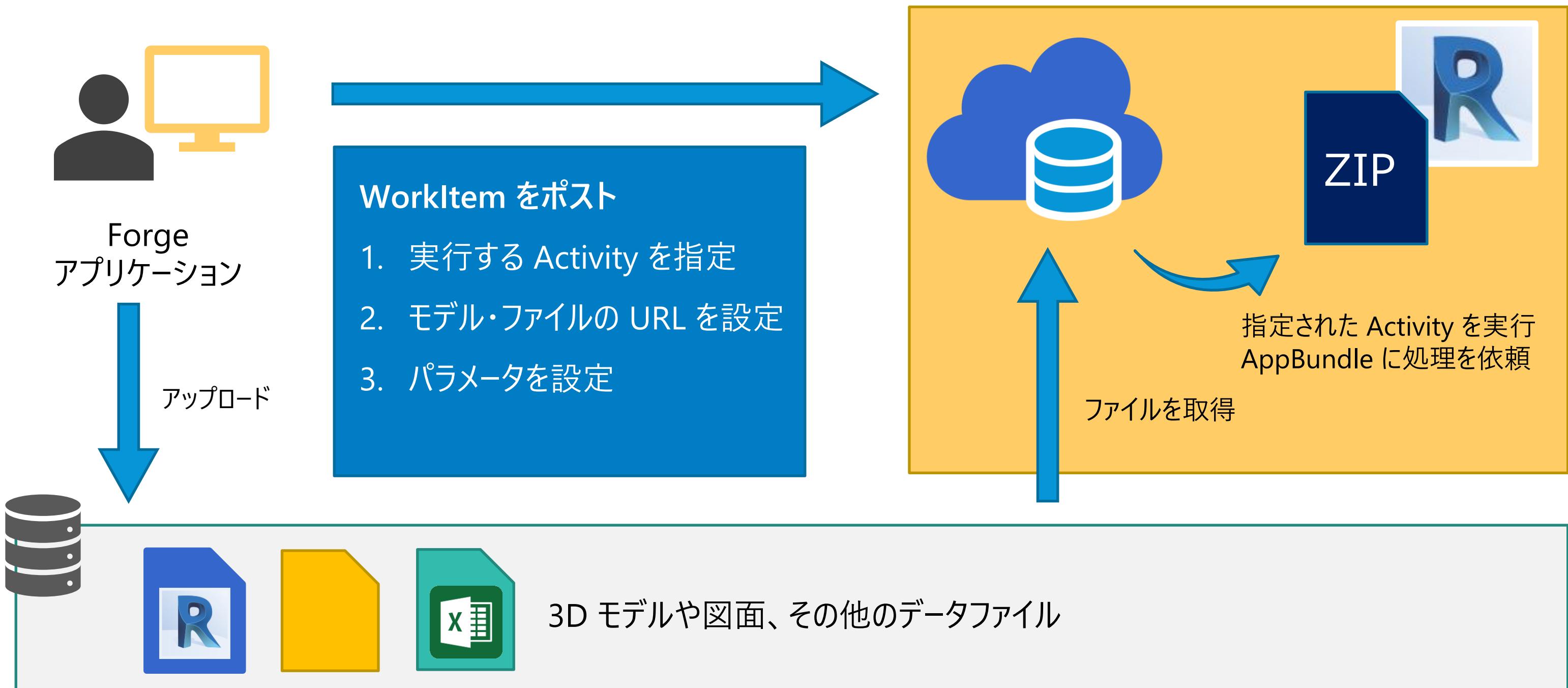
- 形式: YourNickname.SomeAppBundleId+SomeAliasName
- 例: MyFirstForgeAppNickname.DeleteWallApp+test

例えば、Activity のテストを行う際に、Test エイリアスを作成してテストを実行する。ただし、AppBundle は、稼働中の Production のエイリアスを呼びだす、といった開発・テスト用の使い方ができます。

[https://forge.autodesk.com/en/docs/design-automation/v3/developers\\_guide/aliases-and-ids/](https://forge.autodesk.com/en/docs/design-automation/v3/developers_guide/aliases-and-ids/)

[https://adndevblog.typepad.com/technology\\_perspective/2019/02/understanding-steps-to-use-design-automation-api-for-revit.html](https://adndevblog.typepad.com/technology_perspective/2019/02/understanding-steps-to-use-design-automation-api-for-revit.html)

# WorkItem を登録する



# WorkItem のコールバック通知処理

- WorkItem を POST する際に、"onComplete" という引数にコールバック URL を設定することができます。
- この引数を事前に設定しておけば、WorkItem の完了時に、**指定の URL** を自動的に呼び出してくれます。
- コールバック URL には、**クエリパラメータ**を組み合わせることができます。
- 進捗状況を30秒毎に "onProgress" コールバック URL で受け取ることもできます。



# オプショナル：Nickname の作成

- Design Automation は、Forge アプリを識別するために **Client ID** を使用しますが、これは人間には覚えづらい文字列（例えば、YnhayiOjhgnsd&jsafh890ryehQW）となっています。
- Forge アプリから Design Automation に処理を依頼するときにも、この Client ID が必要となります。より覚えやすい**ニックネーム**（エイリアスのようなもの）で名前を付けて**マッピング**を行うことができます。
- 20 文字以内で、a-z, A-Z, 0-9, \_ から作成
- 今回のチュートリアルでは、Nickname は Client Id を使用します。



基本アプリの UI

# ドキュメントロード時のJavaScript コード

- 登録済み AppBundle と Activity の取得
- 各 UI ボタンのクリックイベントの登録
- サーバーとのリアルタイム通信開始

```

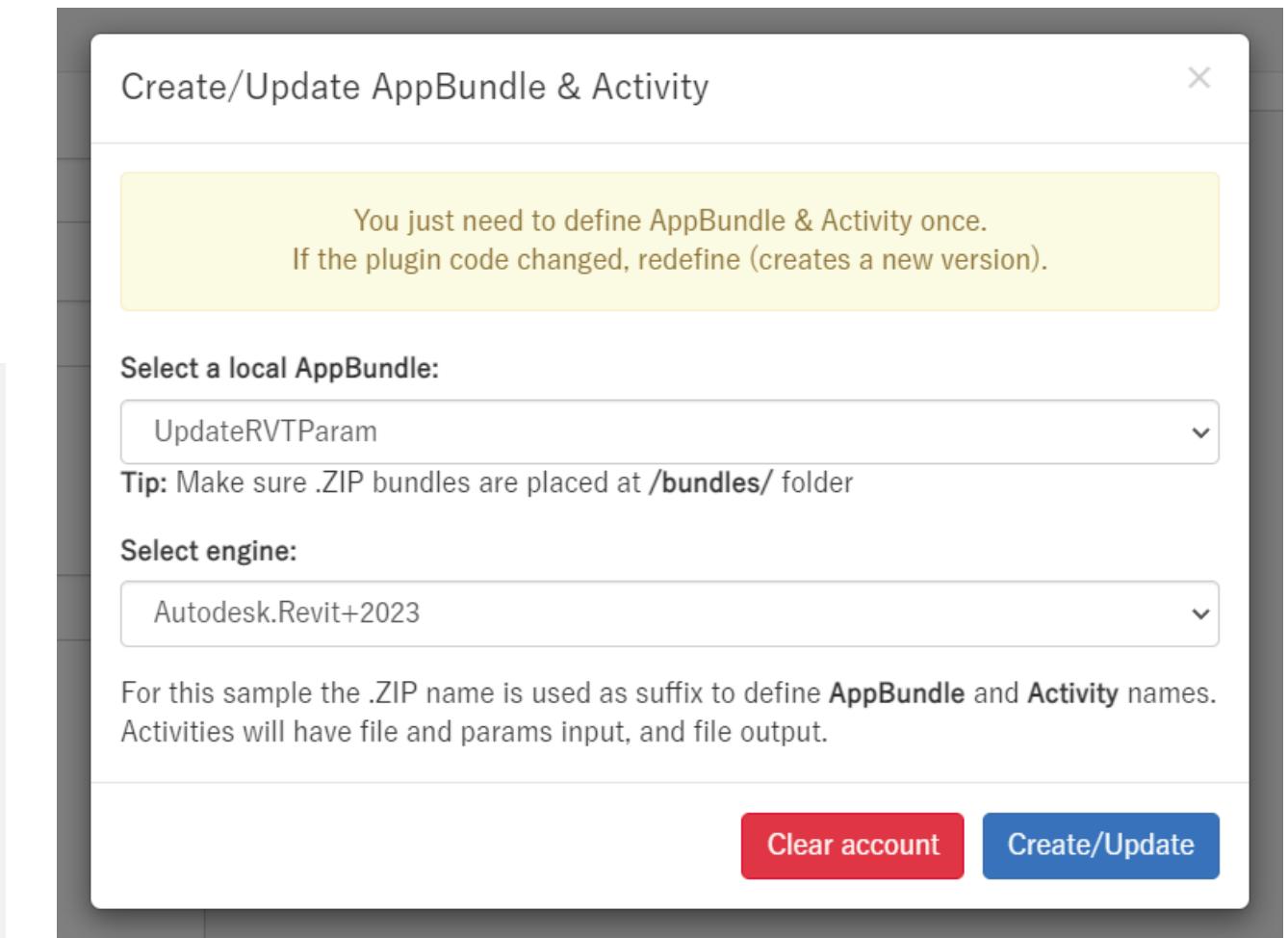
$(document).ready(function () {
    prepareLists();

    $('#clearAccount').click(clearAccount);
    $('#defineActivityShow').click(defineActivityModal);
    $('#createAppBundleActivity').click(createAppBundleActivity);
    $('#startWorkitem').click(startWorkitem);

    startConnection();
});

function prepareLists() {
    list('activity', '/api/forge/designautomation/activities');
    list('engines', '/api/forge/designautomation/engines');
    list('localBundles', '/api/appbundles');
}

```



# SignalR の version 3 系の最新バージョンに対応

- index.html で読み込むスクリプトを最新バージョン 3.1.26 に変更

```
<head>
  <title>Autodesk Forge – Design Automation</title>
  <meta charset="utf-8" />
  <link rel="shortcut icon" href="https://github.com/Autodesk-Forge/learn.forge.viewhubmodels/raw/master/img/favicon.ico">
  <!-- Common packages: jQuery, Bootstrap -->
  <script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="//cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/3.4.1/js/bootstrap.min.js"></script>
  <link rel="stylesheet" href="//cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/3.4.1/css/bootstrap.min.css">
  <!-- .NET SignalR -->
  <script src="https://cdnjs.cloudflare.com/ajax/libs/microsoft-signalr/3.1.26/signalr.min.js"></script> ←
  <!-- Files for this project -->
  <script src="/js/ForgeDesignAutomation.js"></script>
</head>
```



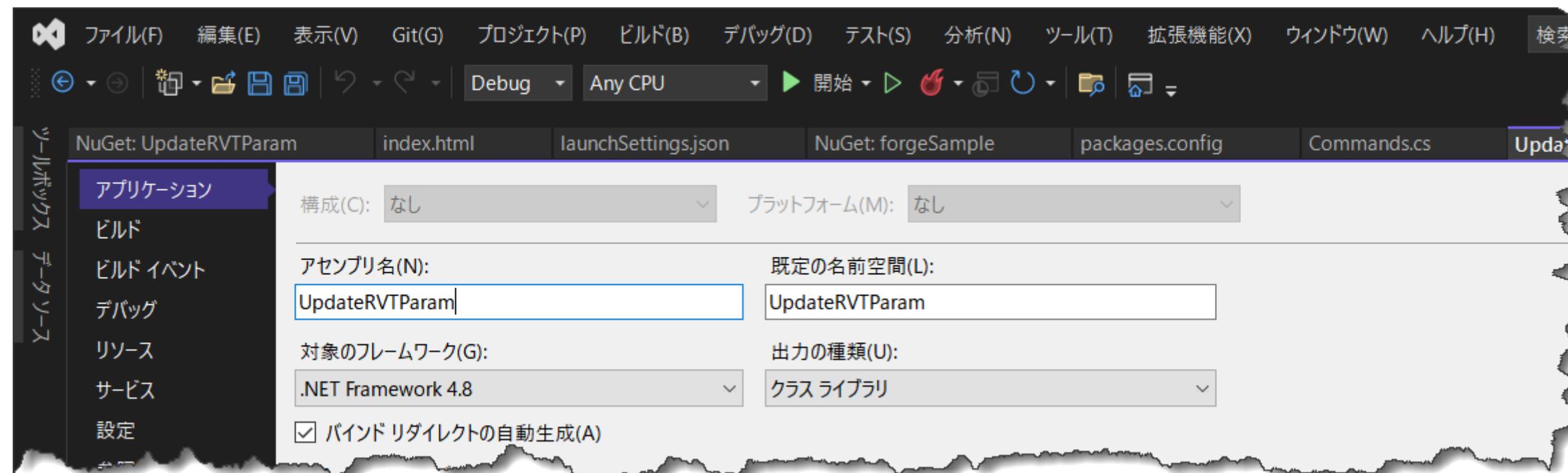
プラグインを準備する

# 新規プロジェクトの作成

## 新しいプロジェクトを作成する

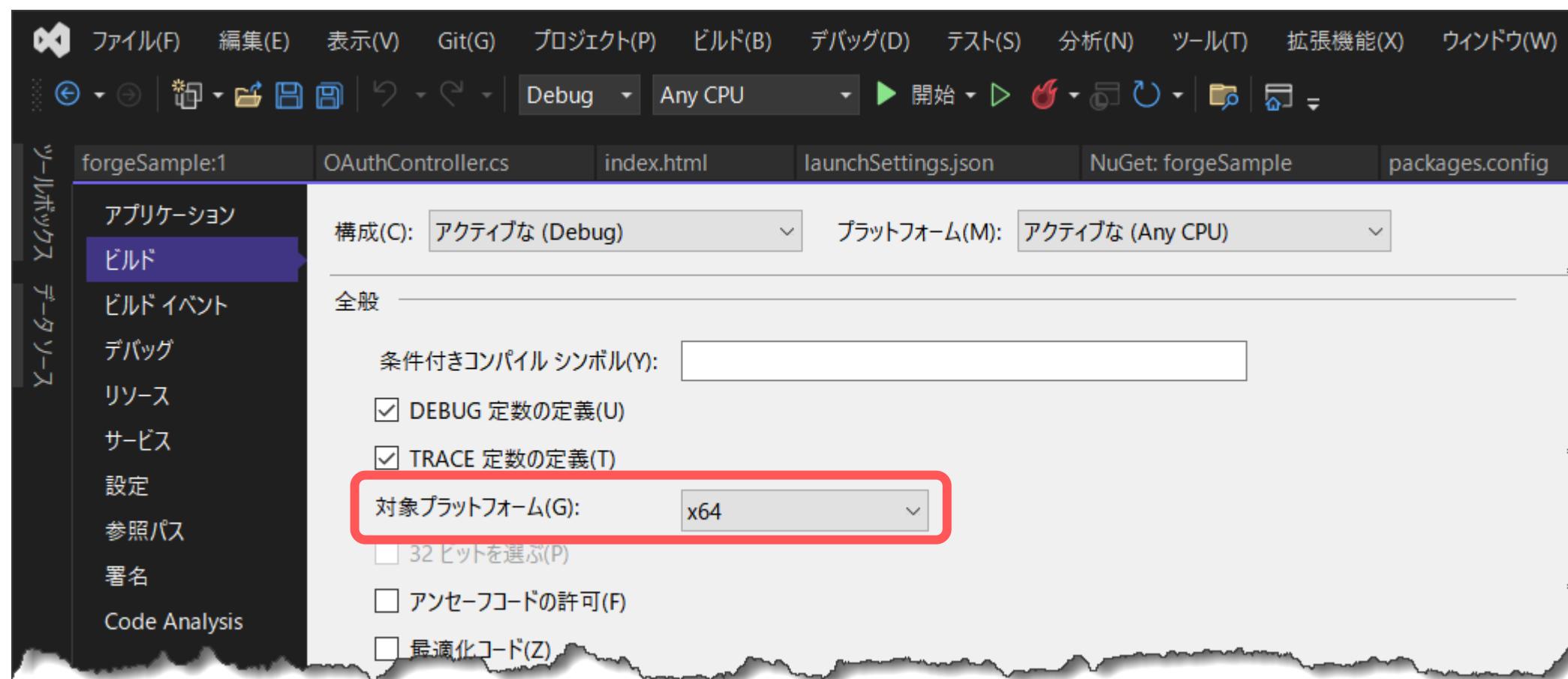
ソリューションを右クリックし、Add >> New Project を選択します。Windows Desktop、次に Class Library を選択し、最後に **UpdateRVTParam** という名前を付けます。

- Visual Studio 2022 でクラスライブラリを選択できない場合
  1. プロジェクトテンプレートで空のプロジェクト (.NET Framework 4.8) を指定して作成
  2. プロジェクトのプロパティを開いて、出力の種類をコンソールアプリケーションからクラスライブラリに変更



# NuGet パッケージ追加時の警告

- Autodesk.Forge.DesignAutomation.Revit を参照に追加した際に警告が表示される場合があります。
  - UpdateRVTParam プロジェクトのプロパティを開きます。
  - ビルドタブで対象プラットフォームを x64 に変更してください。



# プロジェクトの設定を確認する

- packages.config をダブルクリックで開く

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="Autodesk.Forge.DesignAutomation.Revit" version="2023.0.1" targetFramework="net48" />
  <package id="Microsoft.CSharp" version="4.5.0" targetFramework="net48" />
  <package id="Newtonsoft.Json" version="13.0.2-beta1" targetFramework="net48" />
</packages>
```

- 使用する Revit のバージョンに対応する .NET Framework と Autodesk.Forge.DesignAutomation.Revit パッケージをインストール
  - .NET Framework
    - Revit 2021 - 2023 -> .NET Framework 4.8
    - Revit 2019, 2020 -> .NET Framework 4.7

# Revit プラグインの作成に必要なアセンブリ参照



# IExternalDBApplication の実装

```

1  using Autodesk.Revit.ApplicationServices;
2  using Autodesk.Revit.Attributes;
3  using Autodesk.Revit.DB;
4  using DesignAutomationFramework;
5  using Newtonsoft.Json;
6  using System.Collections.Generic;
7  using System.IO;
8
9  namespace Autodesk.Forge.Sample.DesignAutomation.Revit
10 {
11     [Transaction(TransactionMode.Manual)]
12     [Regeneration(RegenerationOption.Manual)]
13     0 個の参照
14     public class Commands : IExternalDBApplication 実装するインターフェース
15     {
16         //Path of the project(i.e)project where your Window family files are present
17         string OUTPUT_FILE = "OutputFile.rvt";
18
19         0 個の参照
20         public ExternalDBApplicationResult OnStartup(ControlledApplication application) イベントハンドラの登録
21         {
22             DesignAutomationBridge.DesignAutomationReadyEvent += HandleDesignAutomationReadyEvent;
23             return ExternalDBApplicationResult.Succeeded;
24         }
25
26         1 個の参照
27         private void HandleDesignAutomationReadyEvent(object sender, DesignAutomationEventArgs e)
28         {
29             LogTrace("Design Automation Ready event triggered...");
30             e.Succeeded = true;
31             EditWindowParametersMethod(e.DesignAutomationData.RevitDoc); ← カスタム処理を呼び出し
32         }
33     }
34 }
```

# UI 画面



# サーバーからのデータを Revit アドインで受け取る方法

```
31     private void EditWindowParametersMethod(Document doc)
32     {
33         InputParams inputParameters = JsonConvert.DeserializeObject<InputParams>(File.ReadAllText("params.json"));
34
35         using (Transaction trans = new Transaction(doc))
36         {
37             trans.Start("Update window parameters");
38
39             FilteredElementCollector WindowCollector = new FilteredElementCollector(doc)
40                 .OfCategory(BuiltInCategory.OST_Windows).WhereElementIsNotElementType();
41             IList<ElementId> windowIds = WindowCollector.ToElementIds() as IList<ElementId>;
42
43             foreach (ElementId windowId in windowIds)
44             {
45                 Element Window = doc.GetElement(windowId);
46                 FamilyInstance FamInst = Window as FamilyInstance;
47                 FamilySymbol FamSym = FamInst.Symbol;
48                 SetElementParameter(FamSym, BuiltInParameter.WINDOW_HEIGHT, inputParameters.Height);
49                 SetElementParameter(FamSym, BuiltInParameter.WINDOW_WIDTH, inputParameters.Width);
50             }
51
52             trans.Commit();
53         }
54
55         ModelPath ProjectModelPath = ModelPathUtils.ConvertUserVisiblePathToModelPath(OUTPUT_FILE);
56         SaveAsOptions SAO = new SaveAsOptions();
57         SAO.OverwriteExistingFile = true;
58
59         LogTrace("Saving file... ");
60         doc.SaveAs(ProjectModelPath, SAO);
61     }
```

JSON ファイルでデータを取得可能

# 任意のタイミングでログにテキスト出力

- Revit アドインのプログラム内で、**Console.WriteLine()** メソッドを呼び出せば、任意のタイミングで、report.txt にカスタムのログを出力することができます。

```
private static void LogTrace(string format, params object[] args) { System.Console.WriteLine(format, args); }
```

# Revit アドインのエラーハンドリング

- Revit アドインで発生するエラーは、トランザクションのコミット時、あるいはロールバック時に呼び出されます。
- DesignAutomationBridge は、デフォルトのエラーハンドラを搭載しており、すべての警告を抑止し、エラーの場合は、それを解決しようと試みます。解決したらコミットします。
- デフォルトの解決方法が「要素の削除」となる場合は、エラーハンドラは要素を削除せずに、ロールバックします。
- 開発者は、カスタムのエラーハンドラを実装して、デフォルトのエラーハンドラを上書きすることができます。

```
public void HandleDesignAutomationReadyEvent(object sender, DesignAutomationReadyEventArgs e)
{
    // Hook up the CustomFailureHandling failure processor.
    Application.RegisterFailuresProcessor(new CustomFailureHandlingProcessor());

    // Run the application logic.
    SketchItFunc(e.DesignAutomationData);

    e.Succeeded = true;
}
```

# バンドルパッケージの作成

- AppBundle に登録するファイルは、アセンブリファイルとアドインマニフェストファイルを指定のフォルダ構成で格納して ZIP 圧縮したパッケージファイルです。

DeleteWallsApp.zip

```

|-- DeleteWalls.bundle
|   |-- PackageContents.xml
|   |-- Contents
|       |-- DeleteWalls.dll
|       |-- DeleteWalls.addin

```

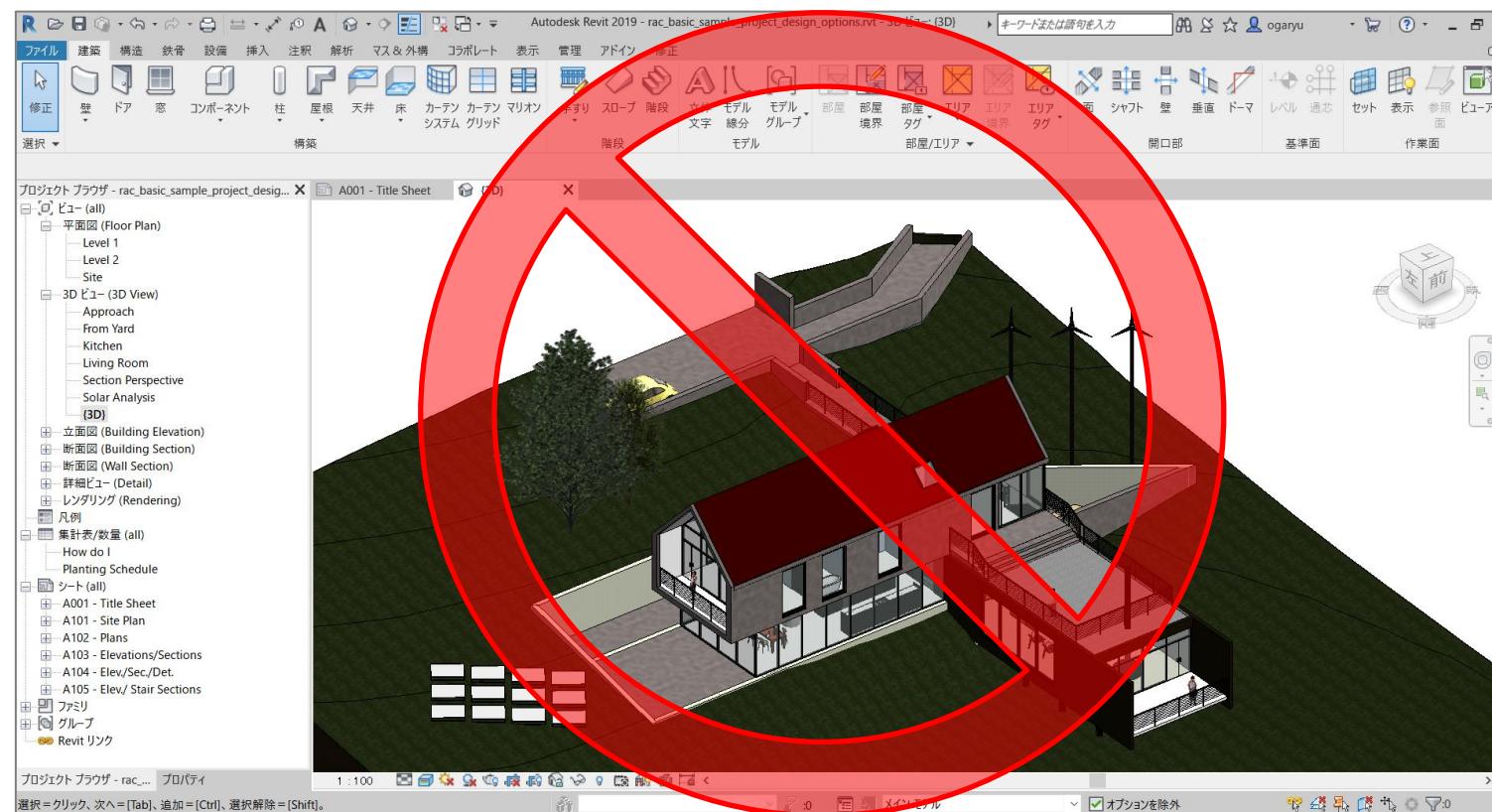
```

1  <?xml version="1.0" encoding="utf-8" ?>
2  <ApplicationPackage>
3      <Components Description="SketchIt">
4          <RuntimeRequirements OS="Win64"
5              Platform="Revit"
6                  SeriesMin="R2018"
7                  SeriesMax="R2018" />
8          <ComponentEntry AppName="SketchIt"
9              Version="1.0.0"
10             ModuleName="../Contents/SketchIt.addin"
11             AppDescription="Sketches some walls and a floor."
12             LoadOnCommandInvocation="False"
13             LoadOnRevitStartup="True" />
14     </Components>
15 </ApplicationPackage>

```

# Revit アドイン開発時の制約事項

- [x] Revit UI 名前空間へのアクセス、UI 画面へのアクセスはできません。
- [x] ユーザーとのインタラクションが発生する処理はサポートされておりません。
- [x] ActiveView と ActiveDocument プロパティにはアクセスできません。
- [x] 複雑なセッション管理は想定されていません。 (バッチ処理を想定)
- [x] Navisworks への書き出し、Desktop Connector は未サポートです。
- [x] Revit Could Worksharing の中央モデルとの同期は、現在プライベートベータ版。
- [x] Could Model for Revit non-workshared も一部制約があります。



# 追加のフォントサポート

- Revit を起動する OS は Windows Server 2019 英語版になります。
- Design Automation for Revit を使用してモデルを処理する場合、Worker インスタンスで使用できないフォントはすべて置換されます。エクスポートなどの操作では、フォントが置換されると、結果の外観が異なる場合があります。
- Design Automation for Revit では、リストに掲載されているフォントが Worker インスタンスに追加でプレインストールされるため、多くの場合、フォントの置換が不要になり、結果の外観が向上します。



AppBundle をアップロードする

# クライアントサイド (ForgeDesignAutomation.js)

```
49
50  function createAppBundleActivity() {
51    startConnection(function () {
52      writeLog("Defining appbundle and activity for " + $('#engines').val());
53      $('#defineActivityModal').modal('toggle');
54      createAppBundle(function () {
55        createActivity(function () {
56          prepareLists();
57        })
58      });
59    });
60  }
61
```

62 → `function createAppBundle(cb) {`

```
63   jQuery.ajax({
64     url: 'api/forge/designautomation/appbundles',
65     method: 'POST',
66     contentType: 'application/json',
67     data: JSON.stringify({
68       zipFileName: $('#localBundles').val(),
69       engine: $('#engines').val()
70     }),
71     success: function (res) {
72       writeLog('AppBundle: ' + res.appBundle + ', v' + res.version);
73       if (cb) cb();
74     }
75   });
76 }
77
```

サーバーの URL に POST リクエスト送信

78 `function createActivity(cb) {`

```
79   jQuery.ajax({
80     url: 'api/forge/designautomation/activities',
81     method: 'POST',
82     contentType: 'application/json',
83     data: JSON.stringify({
84       zipFileName: $('#localBundles').val(),
85       engine: $('#engines').val()
86     }),
87     success: function (res) {
88       writeLog('Activity: ' + res.activity);
89       if (cb) cb();
90     }
91   });
92 }
```

# サーバーサイド (DesignAutomationController.cs)

```
91  ///<summary>
92  /// Define a new AppBundle
93  ///</summary>
94  [HttpPost]
95  [Route("api/forge/designautomation/appbundles")]
0 個の参照
96  public async Task<IActionResult> CreateAppBundle([FromBody] JObject appBundleSpecs)
97  {
98      // basic input validation
99      string zipFileName = appBundleSpecs["zipFileName"].Value<string>();
100     string engineName = appBundleSpecs["engine"].Value<string>();
101
102     // standard name for this sample
103     string appName = zipFileName + "AppBundle";
```

クライアントからの POSTリクエスト  
で呼び出されるメソッド

```
176  ///<summary>
177  /// Define a new activity
178  ///</summary>
179  [HttpPost]
180  [Route("api/forge/designautomation/activities")]
0 個の参照
181  public async Task<IActionResult> CreateActivity([FromBody] JObject activitySpecs)
182  {
183      // basic input validation
184      string zipFileName = activitySpecs["zipFileName"].Value<string>();
185      string engineName = activitySpecs["engine"].Value<string>();
186
187      // standard name for this sample
188      string appName = zipFileName + "AppBundle";
```

# API エンドポイント : POST appbundles

Method and URI	<b>POST</b> <a href="https://developer.api.autodesk.com/da/us-east/v3/appbundles">https://developer.api.autodesk.com/da/us-east/v3/appbundles</a>	
Authentication Context	app only	
Required OAuth Scopes	code:all	
Data Format	JSON	
	<b>package</b> <i>string</i>	The URL that points to the zip package for the AppBundle or Engine.
	<b>id</b> <i>string</i>	Name of AppBundle, see the example section.
	<b>engine *</b> <i>string</i>	The actual processing engine that runs the WorkItem job and processes the Activity.
	<b>appbundles</b> <i>array: string</i>	A module referenced by an Activity in order to perform specific functions. Typically this is a DLL or some other form of custom code.
	<b>▼ settings</b> <i>object</i>	The url/string Settings for a given set of AppBundle.
	<b>&gt; *</b> <i>any of</i>	Type: dictionary<string, *>
	<b>description</b> <i>string</i>	Human readable description of the object.

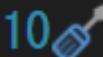
\* Required

# API エンドポイント : POST appbundles/:id/aliases

Method and URI	<b>POST</b> <a href="https://developer.api.autodesk.com/da/us-east/v3/appbundles/:id/aliases">https://developer.api.autodesk.com/da/us-east/v3/appbundles/:id/aliases</a>
Authentication Context	app only
Required OAuth Scopes	code:all
Data Format	JSON
	<b>Request</b>
	<b>URI Parameters</b>
	<b>id</b> <i>string</i>
	Name of AppBundle (unqualified).
	<b>Request</b>
	<b>Body Structure</b>
	<b>version</b> <i>int</i>
	The version that this alias refers to.
	<b>receiver</b> <i>string</i>
	The user to share the alias with.
	<b>id</b> <i>string</i>

# AppBundle と Alias の作成

```

110  Page<string> appBundles = await _designAutomation.GetAppBundlesAsync();
```

112 // check if app bundle is already define 既に登録されている AppBundle の一覧を取得

113 dynamic newAppVersion;

114 string qualifiedAppBundleId = string.Format("{0}.{1}{2}", NickName, AppBundleName, Alias);

115 if (!appBundles.Data.Contains(qualifiedAppBundleId))

116 {

117 // create an AppBundle (version 1)

118 AppBundle AppBundleSpec = new AppBundle()

119 {

120 Package = AppBundleName,

121 Engine = engineName,

122 Id = AppBundleName,

123 Description = string.Format("Description for {0}", AppBundleName),

124 };

125 newAppVersion = await \_designAutomation.CreateAppBundleAsync(appBundleSpec); 新規 AppBundle を作成

126 if (newAppVersion == null) throw new Exception("Cannot create new app");

127

128 // create alias pointing to v1 新規 Alias を作成

129 Alias aliasSpec = new Alias() { Id = Alias, Version = 1 };

130 Alias newAlias = await \_designAutomation.CreateAppBundleAliasAsync(appBundleName, aliasSpec);

131 }

132 }

133 else

134 {

新規作成時は、Version は自動的に 1 が設定されます。

Alias 登録後、バンドルレパッケージを AppBundle にアップロードします。



Activity を定義する

# API エンドポイント : POST activities

Method and URI

**POST** <https://developer.api.autodesk.com/da/us-east/v3/activities>

Authentication Context

app only

Required OAuth Scopes

code:all

Data Format

JSON

Request

Body Structure

**commandLine** \* Path to Engine executable with arguments. [Activity command line](#).

*array: string*

> **parameters** Each parameter represents an input or output file. Named parameters of an Activity have corresponding named arguments of a WorkItem.

**id**

Name of Activity, see the example section.

*string*

**engine** \*

The actual processing engine that runs the WorkItem job and processes the Activity.

*string*

**appbundles**

A module referenced by an Activity in order to perform specific functions. Typically this is a DLL or some other form of custom code.

> **settings**

The url/string Settings for a given set of AppBundles.

*object*

**description**

Human readable description of the object.

*string*

[Expand](#)



# Activity と Alias の作成

```

190  Page<string> activities = await _designAutomation.GetActivitiesAsync();
191  string qualifiedActivityId = string.Format("{0}. {1}+{2}", NickName, activityName, Alias);
192  if (!activities.Data.Contains(qualifiedActivityId)) 既に登録されている Activity の一覧を取得
193  {
194      // define the activity
195      // ToDo: parametrize for different engines...
196      dynamic engineAttributes = EngineAttributes(engineName);
197      string commandLine = string.Format(engineAttributes.commandLine, appBundleName);
198      Activity activitySpec = new Activity()
199      {
200          Id = activityName,
201          Appbundles = new List<string>() { string.Format("{0}. {1}+{2}", NickName, appBundleName, Alias) },
202          CommandLine = new List<string>() { commandLine },
203          Engine = engineName,
204          Parameters = new Dictionary<string, Parameter>()
205          {
206              {
207                  "inputFile", new Parameter() { Description = "input file", LocalName = "$(inputFile)", Ondemand = false, Req },
208                  "inputJson", new Parameter() { Description = "input json", LocalName = "params.json", Ondemand = false, Req },
209                  "outputFile", new Parameter() { Description = "output file", LocalName = "outputFile." + engineAttributes.e
210              },
211              Settings = new Dictionary<string, ISetting>()
212              {
213                  "script", new StringSetting() { Value = engineAttributes.script } }
214              };
215          };
216          Activity newActivity = await _designAutomation.CreateActivityAsync(activitySpec);
217

```

既に登録されている Activity の一覧を取得

新規 Activity を作成

Alias と Version の作成が続きます。

# Activity の定義

```
1  {
2      "id": "UpdateRVTParamActivity", Activity ID
3      "commandLine": [ "$(engine.path)\\\\revitcoreconsole.exe /i \"$(args[inputRvtFile].path)\" /al $(appbundles[UpdateRVTParam].path)" ],
4      "parameters": {
5          "inputFile": {
6              "zip": false,
7              "ondemand": false,
8              "verb": "get",
9              "description": "input file",
10             "required": true
11         },
12         "inputJson": {
13             "zip": false,
14             "ondemand": false,
15             "verb": "get",
16             "description": "input json",
17             "required": false,
18             "localName": "params.json"
19         },
20         "outputFile": {
21             "zip": false,
22             "ondemand": false,
23             "verb": "put",
24             "description": "output file",
25             "required": true,
26             "localName": "OutputFile.rvt"
27     }
28 },
29 "engine": "Autodesk.Revit+2021", Engine
30 "appbundles": [ "MyFirstForgeAppNickname.UpdateRVTParam+dev" ], AppBundle ID
31 "description": "."
32 }
```

Activity ID

入力 Revit ファイル

入力 JSON ファイル

出力 Revit ファイル

**verb: get** Revit アドインに渡すファイル

**verb: put** Revit アドインから出力するファイル

**localName** ワーキングディレクトリ上でのファイル名

**required** 必須かそうでないかを指定

**ondemand** WorkItem の処理中に任意のタイミングで  
外部データファイルを取得する方法



WorkItem を実行する

# クライアントサイド (ForgeDesignAutomation.js)

```
94  function startWorkitem() {
95      var inputFileField = document.getElementById('inputFile');
96      if (inputFileField.files.length === 0) { alert('Please select an input file'); return; }
97      if ($('#activity').val() === null) { alert('Please select an activity'); return; }
98      var file = inputFileField.files[0];
99      startConnection(function () {
100          var formData = new FormData();
101          formData.append('inputFile', file);
102          formData.append('data', JSON.stringify({
103              width: $('#width').val(),
104              height: $('#height').val(),
105              activityName: $('#activity').val(),
106              browserConnectionId: connectionId
107          }));
108          writeLog('Uploading input file...');
109          $.ajax({
110              url: 'api/forge/designautomation/workitems',
111              data: formData,
112              processData: false,
113              contentType: false,
114              type: 'POST',
115              success: function (res) {
116                  writeLog('Workitem started: ' + res.workItemId);
117              }
118          });
119      });
120  }
```

フォームでファイルと  
パラメータ値を送信

# サーバーサイド (DesignAutomationController.cs)

```
247     /// <summary>
248     /// Start a new workitem
249     /// </summary>
250     [HttpPost]
251     [Route("api/forge/designautomation/workitems")]
252     // クライアントからのフォームデータを取得
// 0 個の参照
253     public async Task<IActionResult> StartWorkitem([FromForm] StartWorkitemInput input)
254     {
255         // basic input validation
256         JObject workItemData = JObject.Parse(input.data);
257         string widthParam = workItemData["width"].Value<string>();
258         string heightParam = workItemData["height"].Value<string>();
259         string activityName = string.Format("{0}. {1}", NickName, workItemData["activityName"].Value<string>());
260         string browerConnectionId = workItemData["browerConnectionId"].Value<string>();
261
262         // save the file on the server
263         var fileSavePath = Path.Combine(_env.ContentRootPath, Path.GetFileName(input.inputFile.FileName));
using (var stream = new FileStream(fileSavePath, FileMode.Create)) await input.inputFile.CopyToAsync(stream);
```

# API エンドポイント : POST workitems

Method and URI	<b>POST</b> <a href="https://developer.api.autodesk.com/da/us-east/v3/workitems">https://developer.api.autodesk.com/da/us-east/v3/workitems</a>
Authentication Context	user context optional
Required OAuth Scopes	code:all
Data Format	JSON
	<b>Request</b>
	<b>Body Structure</b>
	<b>id</b> <i>string</i> Id.
	<b>activityId *</b> <i>string</i> Reference to the Activity that this WorkItem will invoke. Examples: <i>MyPlot+Prod</i> (an Activity created by the caller) or <i>Autodesk.PlotToPdf</i> (an Activity created by someone else and shared with this caller).
	<b>&gt; arguments</b> <i>object</i> Arguments of the WorkItem. Named parameters of an Activity have corresponding named arguments of a WorkItem.
	<b>&gt; signatures</b> <i>object</i> Signatures for various WorkItem attributes.
	<b>limitProcessingTimeSec</b> <i>int</i> Max duration of processing in seconds per workitem (includes download and upload time).

# WorkItem のパラメータ

```
1  {
2    "activityId": "{{dasNickName}}.{{activityName}}+{{activityAlias}}",
3    "arguments": {
4      "inputFile": {
5        "url": "{{base_domain}}/oss/v2/buckets/{{bucketKey}}/objects/input.rvt",
6        "headers": {
7          "Authorization": "Bearer {{access_token}}"
8        }
9      },
10     "inputJson": {
11       "url": "data:application/json,{{\"Width\":\"2\", \"Height\":\"4\"}}"
12     },
13     "outputFile": {
14       "verb": "put",
15       "url": "{{base_domain}}/oss/v2/buckets/{{bucketKey}}/objects/output.rvt",
16       "headers": {
17         "Authorization": "Bearer {{access_token}}"
18       }
19     }
20   }
21 }
```

The diagram illustrates the mapping of various parameters in the WorkItem JSON object to specific inputs and outputs:

- Input Revit File:** This group includes the `inputFile.url` (line 5) and `inputFile.headers` (lines 6-8). A blue bracket on the right side of the code spans from the start of the `inputFile` block to the end of the `headers` block, labeled "Input Revit File".
- Input JSON Data:** This group includes the `inputJson.url` (line 11). A blue bracket on the right side of the code spans from the start of the `inputJson` block to the end of the `url` value, labeled "Input JSON データ".
- Output Revit File:** This group includes the `outputFile.verb` (line 14), `outputFile.url` (line 15), and `outputFile.headers` (lines 16-18). A blue bracket on the right side of the code spans from the start of the `outputFile` block to the end of the `headers` block, labeled "Output Revit ファイル".

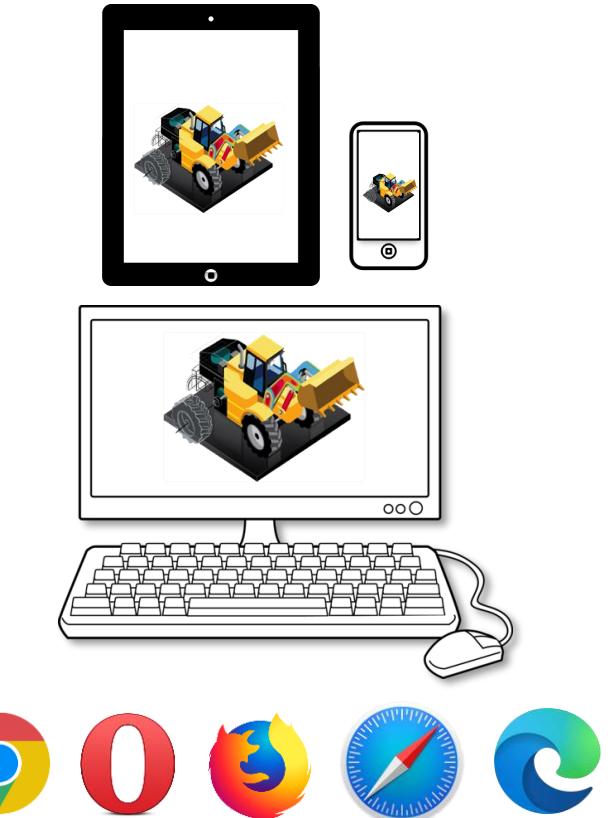
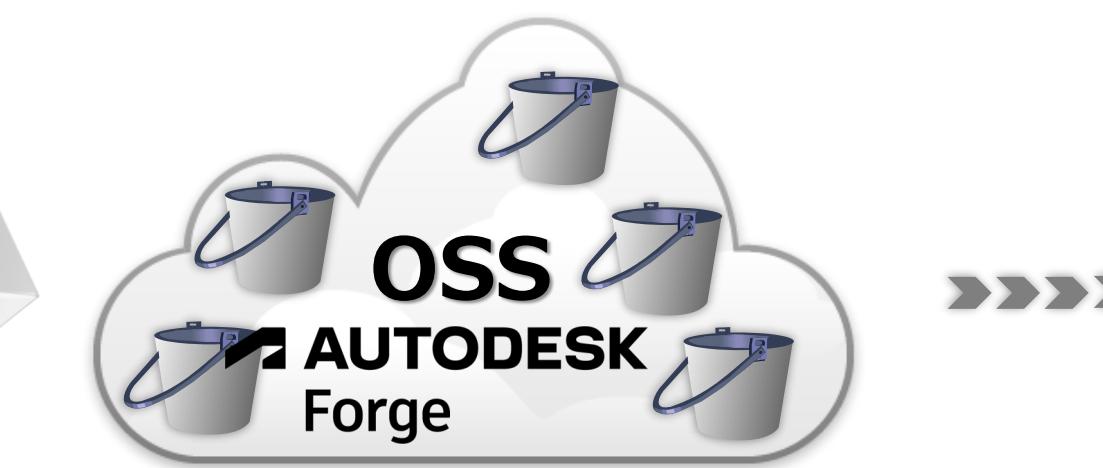
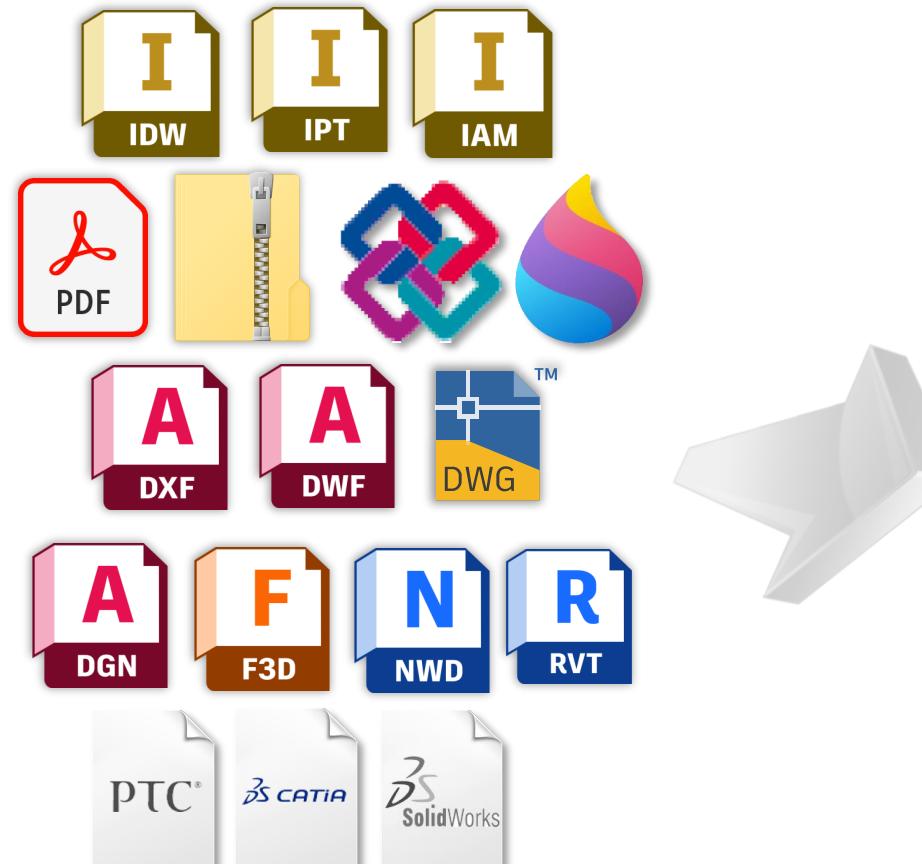
# 入力 Revit ファイルを OSS にアップロードする

```
268 // upload file to OSS Bucket
269 // 1. ensure bucket exists
270 string bucketKey = NickName.ToLower() + "-designautomation";
271 BucketsApi buckets = new BucketsApi();
272 buckets.Configuration.AccessToken = oauth.access_token;
273 try Bucket を作成。
274 {
275     PostBucketsPayload bucketPayload = new PostBucketsPayload(bucketKey, null, PostBucketsPayload.PolicyKeyEnum.Transient);
276     await buckets.CreateBucketAsync(bucketPayload, "US");
277 }
278 catch { }; // in case bucket already exists
279 // 2. upload inputFile
280 string inputFileNameOSS = string.Format("{0}_input_{1}", DateTime.Now.ToString("yyyyMMddhhmmss"), Path.GetFileName(input.input));
281 ObjectsApi objects = new ObjectsApi();
282 objects.Configuration.AccessToken = oauth.access_token;
283 using (StreamReader streamReader = new StreamReader(fileSavePath)) ファイルをアップロード
284     await objects.UploadObjectAsync(bucketKey, inputFileNameOSS, (int)streamReader.BaseStream.Length, streamReader.BaseStream,
285 System.IO.File.Delete(fileSavePath); // delete server copy
286
287 // prepare workitem arguments
288 // 1. input file
289 XrefTreeArgument inputFileArgument = new XrefTreeArgument()
290 {
291     Url = string.Format("https://developer.api.autodesk.com/oss/v2/buckets/{0}/objects/{1}", bucketKey, inputFileNameOSS),
292     Headers = new Dictionary<string, string>()
293     {
294         { "Authorization", "Bearer " + oauth.access_token }
295     }
296 };
```

# Bucket とは

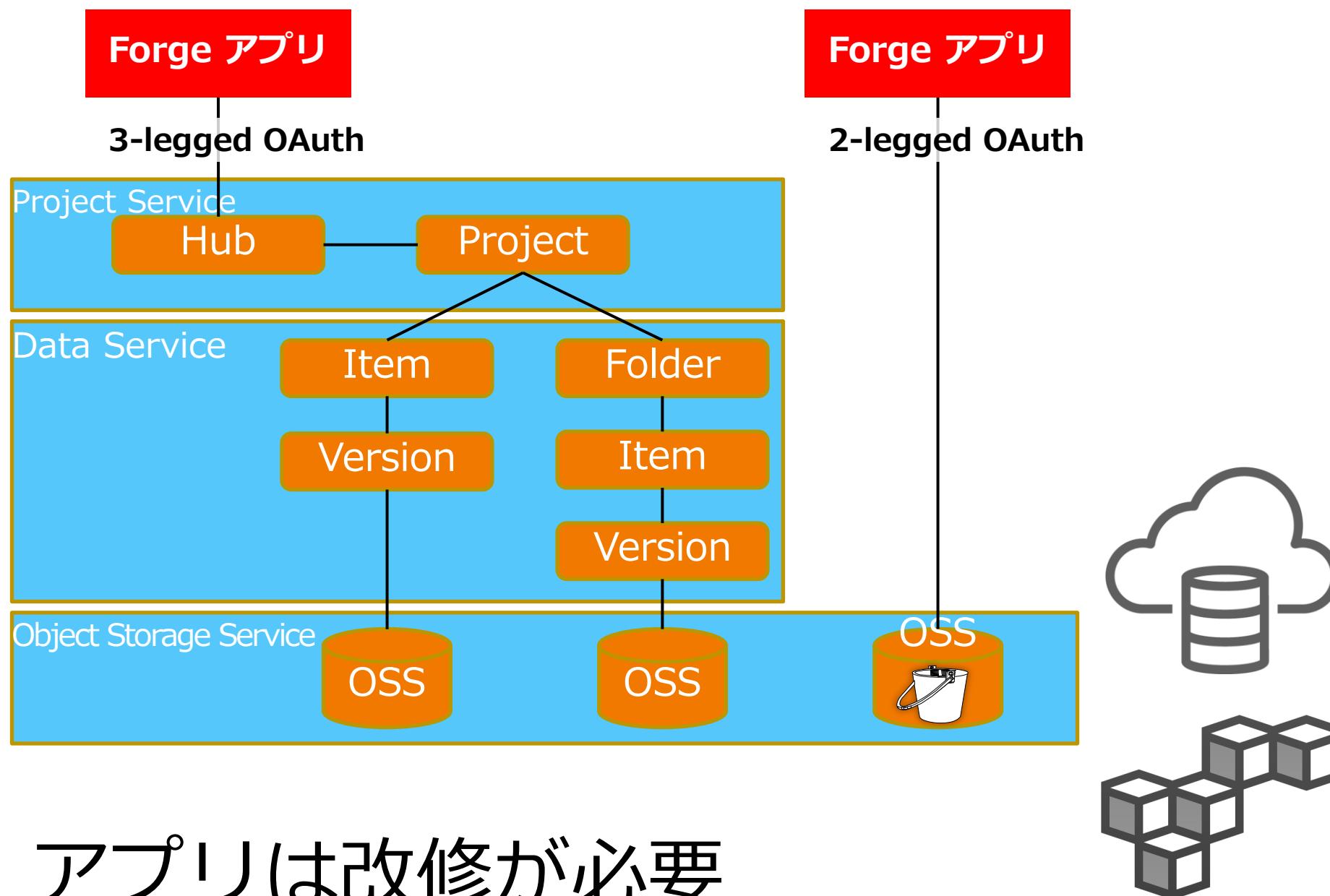
- Bucket は Forge 共有ストレージ内の‘フォルダ’
  - Bucket はデザイン ファイルをアップロード/変換する場所
  - 他の Bucket と重複しない一意な名前が必要
  - 半角英字(小文字)、数字、記号(- \_ .)

Client ID 利用を推奨

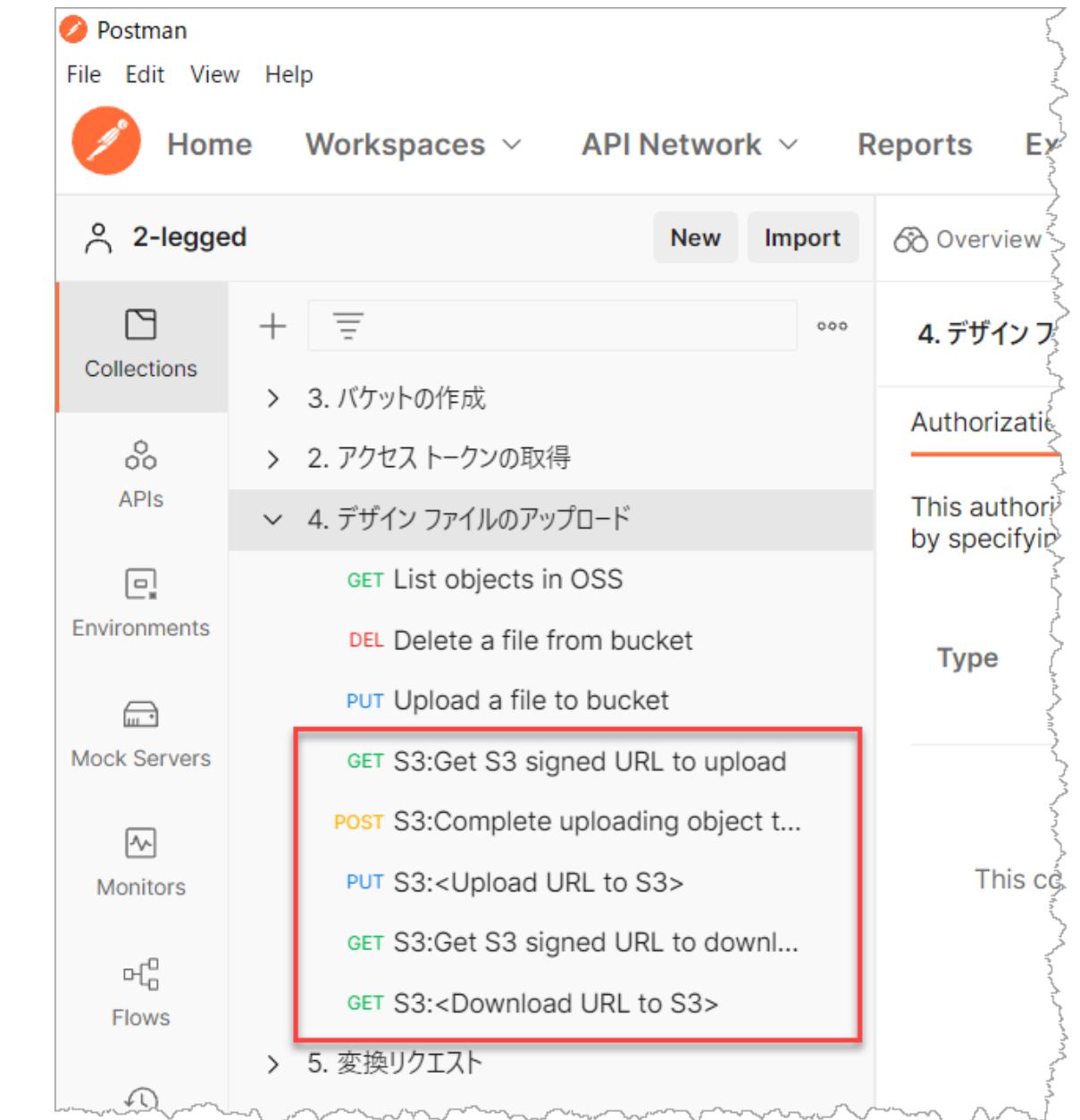


# OSS の Direct-to-S3 アプローチへの移行

- アップロードとダウンロードで AWS S3 に直接アクセス



- アプリは改修が必要



# 入力 JSON データの作成

```
"parameters": {  
    "inputFile": {  
        "zip": false,  
        "ondemand": false,  
        "verb": "get",  
        "description": "input file",  
        "required": true,  
    },  
  
    "inputJson": {  
        "zip": false,  
        "ondemand": false,  
        "verb": "get",  
        "description": "input json",  
        "required": false,  
        "localName": "params.json"  
    },  
  
    "outputFile": {
```

Activity

```
"arguments": {  
    "inputFile": {  
        "url": "https://myWebsite/revit sample file.rvt"  
    },  
  
    "inputJson": {  
        "url": "data:application/json, {  
            'width': 3,  
            'height': 7  
        }"  
    },  
  
    "outputFile": {  
        "verb": "put",  
        "url": "https://myWebsite/signed/url/to/revit sample file.rvt"  
    }  
}
```

WorkItem

JSONデータをリクエストのパラメータに埋め込むと、  
Revit アドインから JSON ファイルとして取得可能。

```
// 2. input json  
dynamic inputJson = new JObject();  
inputJson.Width = widthParam;  
inputJson.Height = heightParam;  
XrefTreeArgument inputJsonArgument = new XrefTreeArgument()  
{  
    Url = "data:application/json, " + ((JObject)inputJson).ToString(Formatting.None).Replace("¥", "")  
};
```

# 出力 Revit ファイルの保存場所を設定

- バケットとファイル名を設定したエンドポイントを予め用意しておき、PUT メソッドでアップロードするように設定します。

Method and URI

**PUT https://developer.api.autodesk.com/oss/v2/buckets/:bucketKey/objects/:objectKey**

Authentication Context

user context optional

Required OAuth Scopes

data:write or data:create ( data:write allows overwriting existing objects)

Data Format

JSON

```

307
308 // 3. output file
309 string outputFileNameOSS = string.Format("{0}_output_{1}", DateTime.Now.ToString("yyyyMMddhhmmss"),
310 Path.GetFileName(input.inputFile.FileName));
311
312 XrefTreeArgument outputFileArgument = new XrefTreeArgument()
313 {
314     Url = string.Format("https://developer.api.autodesk.com/oss/v2/buckets/{0}/objects/{1}", bucketKey, outputFileNameOSS),
315     Verb = Verb.Put,
316     Headers = new Dictionary<string, string>()
317     {
318         {"Authorization", "Bearer " + oauth.access_token }
319     }
320 };

```

# WorkItem の作成

onComplete コールバックにサーバーで受け取る URL を設定します。

WorkItem が完了すると、Design Automation はアプリケーションにコールバックします。

- Forge\_WEBHOOK\_URL : ngrok の Forwarding URL

```
322     // prepare & submit workitem
323     string callbackUrl = string.Format("{0}/api/forge/callback/designautomation?id={1}&outputFileName={2}",
324         OAuthController.GetAppSetting("FORGE_WEBHOOK_URL"), browserConnectionId, outputFileNameOSS);
325
326     WorkItem workItemSpec = new WorkItem()
327     {
328         ActivityId = activityName,
329         Arguments = new Dictionary<string, IArgument>()
330         {
331             { "inputFile", inputFileArgument },
332             { "inputJson", inputJsonArgument },
333             { "outputFile", outputFileArgument },
334             { "onComplete", new XrefTreeArgument { Verb = Verb.Post, Url = callbackUrl } }
335         };
336     };
337     WorkItemStatus workItemStatus = await _designAutomation.CreateWorkItemAsync(workItemSpec);
338
339     return Ok(new { WorkItemId = workItemStatus.Id });
340 }
```

# onComplete コールバックで呼び出されるメソッド

```
351     /// <summary>
352     /// Callback from Design Automation Workitem (onProgress or onComplete)
353     /// </summary>
354     [HttpPost]
355     [Route("/api/forge/callback/designautomation")]
356     public async Task<IActionResult> OnCallback(string id, string outputFileName, [FromBody] dynamic body)
357     {
358         try
359         {
360             // your webhook should return immediately! we can use Hangfire to schedule a job
361             JObject bodyJson = JObject.Parse((string)body.ToString());
362             await _hubContext.Clients.Client(id).SendAsync("onComplete", bodyJson.ToString()); onComplete で指定した URL
363
364             var client = new RestClient(bodyJson["reportUrl"].Value<string>()); SignalR でクライアントに通知
365             var request = new RestRequest(string.Empty); レポートログの取得
366             byte[] bs = client.DownloadData(request);
367             string report = System.Text.Encoding.Default.GetString(bs);
368
369             await _hubContext.Clients.Client(id).SendAsync("onComplete", report); SignalR でクライアントに通知
370
371             ObjectsApi objectsApi = new ObjectsApi();
372             dynamic signedUrl = await objectsApi.CreateSignedResourceAsyncWithHttpInfo(NickName.ToLower() + "-design");
373             await _hubContext.Clients.Client(id).SendAsync("downloadResult", (string)(signedUrl.Data.signedUrl)); Revit ファイルのダウンロード URL の作成
374         }
375         catch (Exception e) { }
376
377         // ALWAYS return ok (200)
378         return Ok();
379     }
```

# クライアントサイドのスクリプト

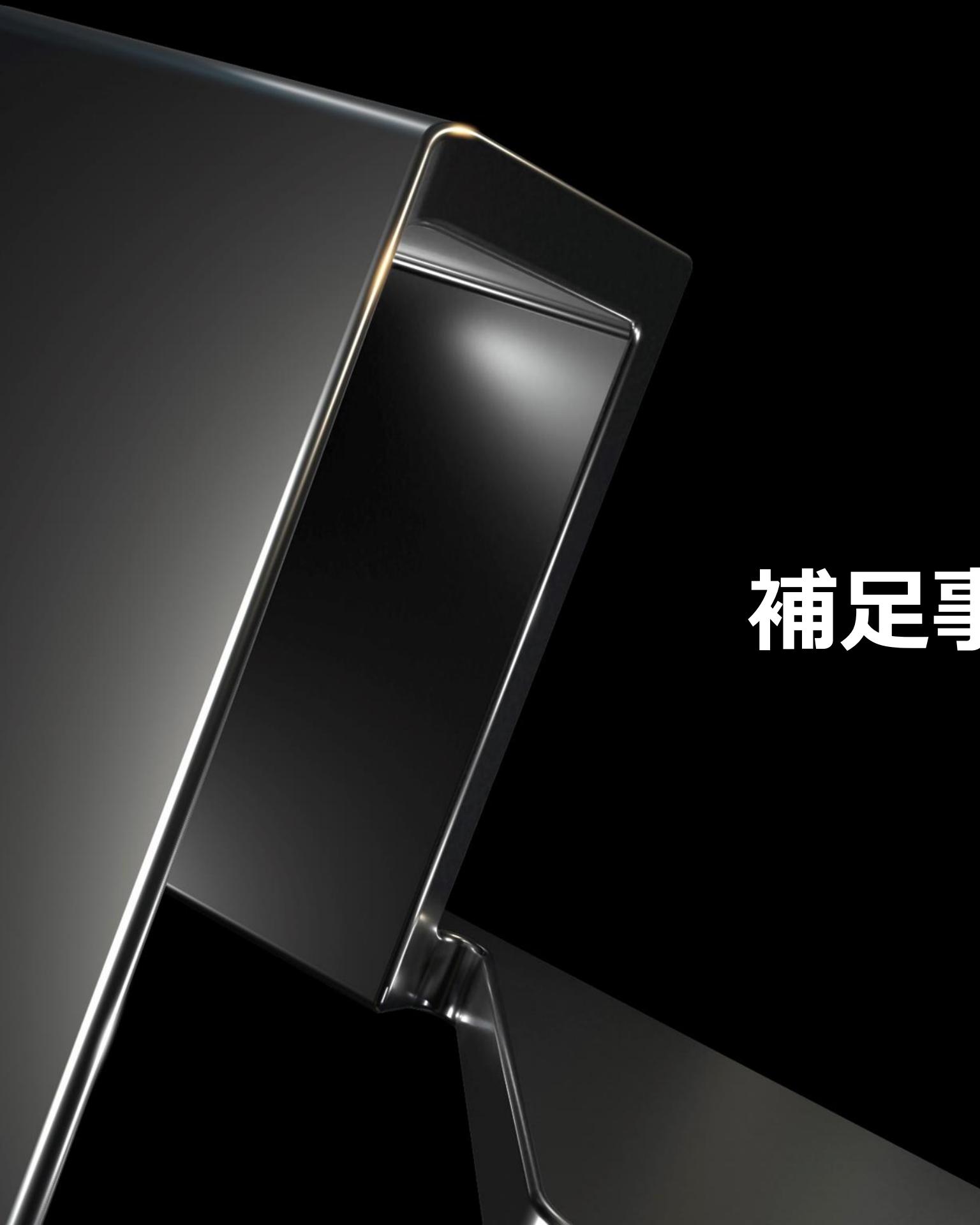
```
var connection;
var connectionId;

function startConnection(onReady) {
    if (connection && connection.connectionState) { if (onReady) onReady(); return; }
    connection = new signalR.HubConnectionBuilder().withUrl("/api/signalr/designautomation").build();
    connection.start()
        .then(function () {
            connection.invoke('get ConnectionId')
                .then(function (id) {
                    connectionId = id; // we'll need this...
                    if (onReady) onReady();
                });
        });
}

connection.on("downloadResult", function (url) {
    writeLog('<a href=' + url + '>Download result file here</a>');
});

connection.on("onComplete", function (message) {
    writeLog(message);
});
}
```

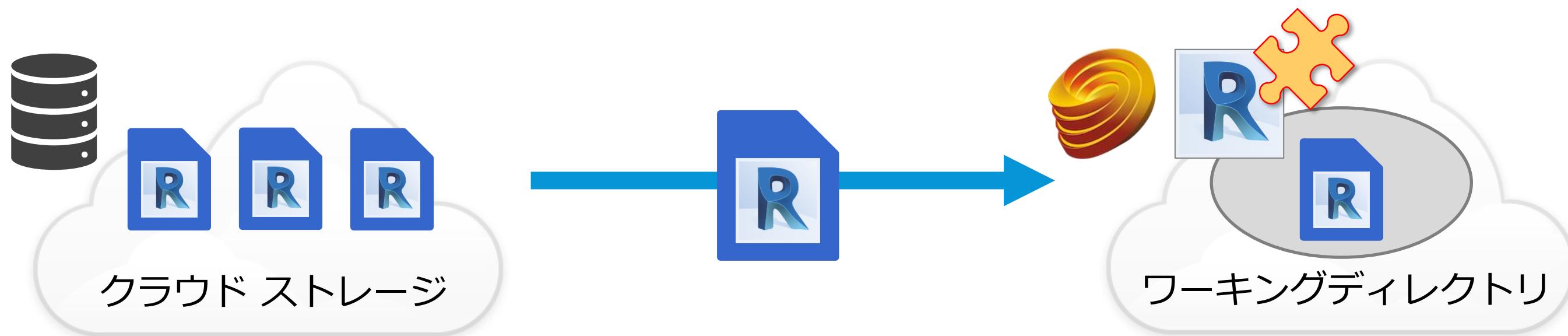
サーバーからのプッシュ通知で呼び出される



# 補足事項

# Design Automation 実行時のデータの取り扱い

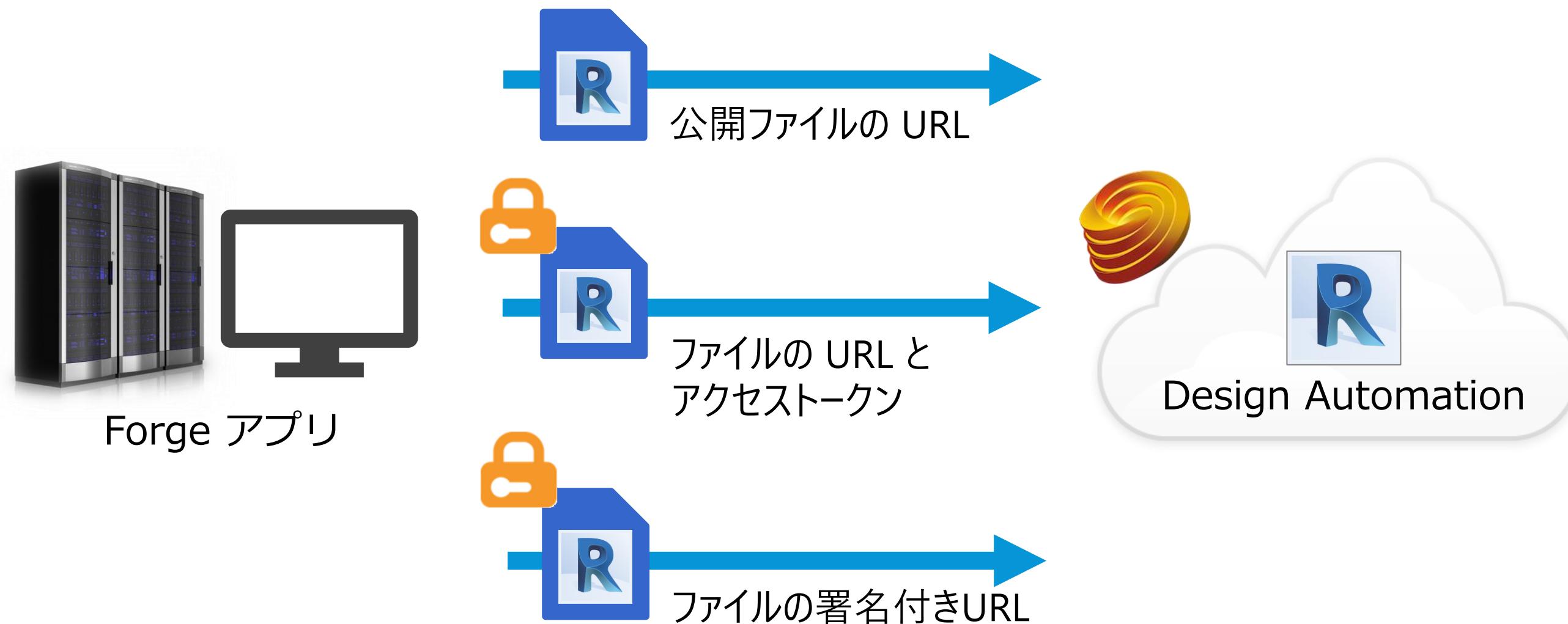
- Design Automation は、クライアント/Forge アプリが指定した任意のクラウドストレージのデータにアクセスして、Forge の一時的なデータ領域（ワーキングディレクトリ）にダウンロードします。



- Forge の一時的なデータ領域のデータは、Design Automation の実行後に破棄されます。

# クラウドストレージ上のファイル URL の取り扱い

- クラウドストレージ上のファイルを Design Automation に送信する方法
  - ファイルの URL のみ（インターネット上の公開ファイルなど）
  - ファイルの URL とアクセストークン → オートデスク クラウドストレージ/OSS
  - 署名付き URL （URLの有効期限、One Time URL、読み書きアクセス権） → OSS



# 署名付き URL の取得

- Data Management API には、OSS にファイルの保存場所を事前に確保して、署名付き URL を発行することもできます。
- この URL に対して、ファイルをアップロードする必要があるため、書き込みアクセス権のある署名付きURLを取得する必要があります。
  - 有効期限の設定
  - 最初に 1 度だけ使用された後に期限切れになる One Time URL の設定
  - アクセス権の設定
    - access=read
    - access=write
    - access=readwrite

# Revit ローカライズ版のエンジンを指定可能

- 起動する Revit のエンジンを、英語以外の言語も指定することができます。
  - /I JPN を指定。
- 日本語版で起動すれば、すべての Revit の出力も日本語版で実行されます。
- 日本語の名称を含んだビルトイインの要素やタイプをサポートする場合に利用します。
- ただし、Revit を起動する OS は Windows Server 2019 英語版になります。

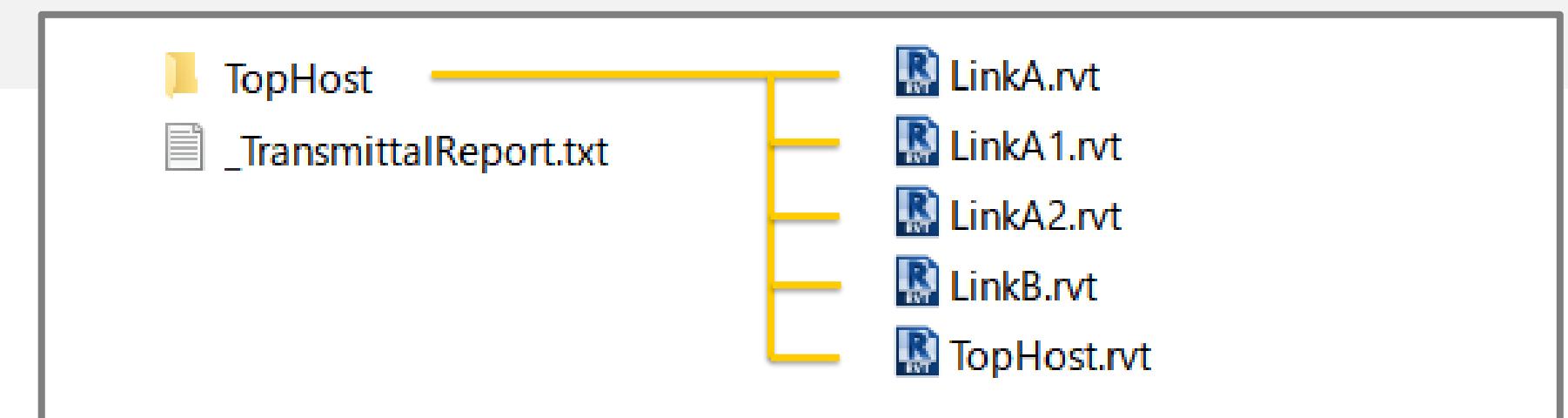


```
{  
  "commandLine": [  
    "$(engine.path)¥¥¥revitcoreconsole.exe /i $(args[rvtFile].path) /al $(apps[DeleteWallsApp].path) /I DEU"  
  ],  
  ...  
}
```

# サポートされているファイル: eTransmit ファイル

- eTransmit for Revit で出力した ZIP ファイルをサポート。

```
"arguments": {  
    "rvtFile": {  
        "url": "https://s3-us-west-2.amazonaws.com/revitio-dev/test-data/TopHost.zip"  
    },  
    "countItParams": {  
        "url": "data:application/json,{\"walls\": true, \"floors\": true, \"doors\": true, \"windows\": true}"  
    },  
    "result": {  
        "verb": "put",  
        "url": "https://myWebsite/signed/url/to/result"  
    }  
}
```

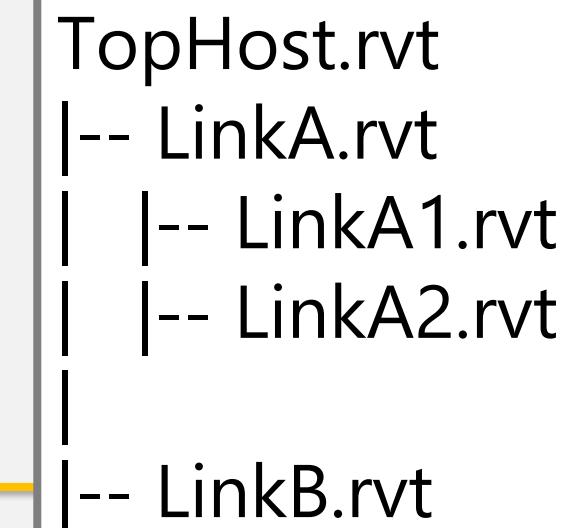


# サポートされているファイル: Revit リンクモデル

```

"rvtFile": {
    "url": "https://s3-us-west-2.amazonaws.com/revitio-dev/test-data/TopHost.rvt",
    "references": [
        {
            "url": "https://s3-us-west-2.amazonaws.com/revitio-dev/test-data/LinkA.rvt",
            "references": [
                {
                    "url": "https://s3-us-west-2.amazonaws.com/revitio-dev/test-data/LinkA1.rvt"
                },
                {
                    "url": "https://s3-us-west-2.amazonaws.com/revitio-dev/test-data/LinkA2.rvt"
                }
            ]
        },
        {
            "url": "https://s3-us-west-2.amazonaws.com/revitio-dev/test-data/LinkB.rvt"
        }
    ]
},
}

```



※ サブフォルダのリンクモデルもサポート

# サポートされているファイル: ZIP ファイル

- アップロード速度を向上させるために、Revit モデルを ZIP 圧縮して送信することができます。
- “pathInZip” オプションで、メインの Revit モデルのパスを指定します。

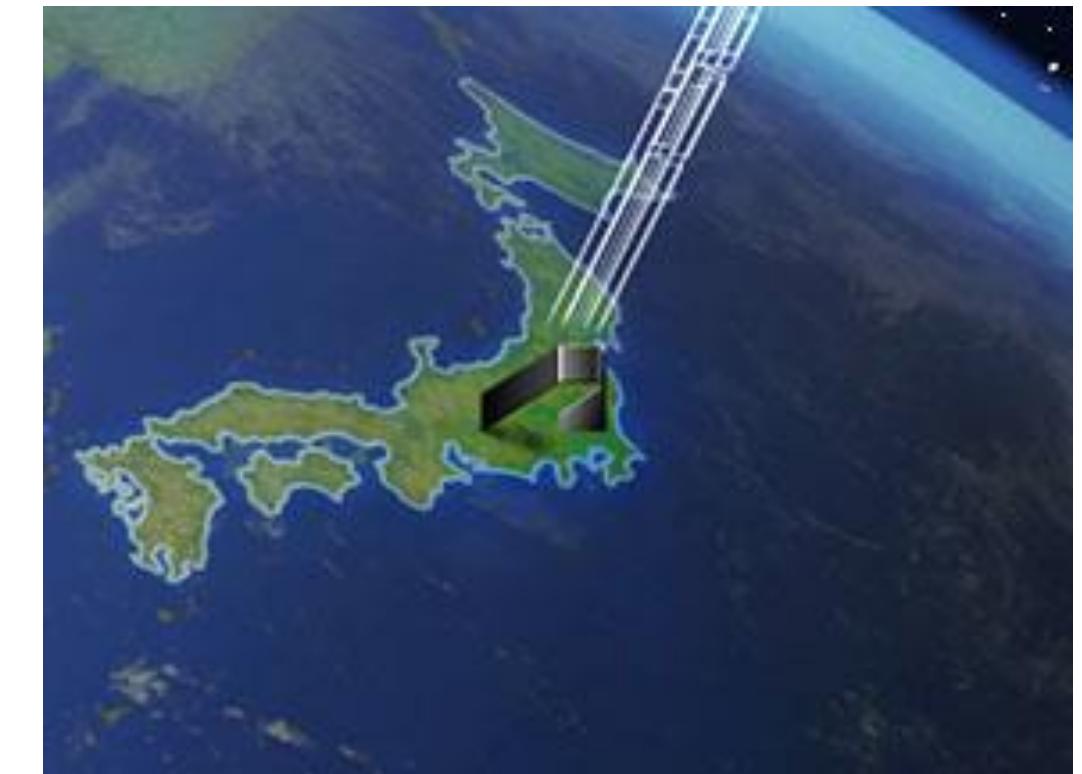
```
"arguments": {  
    "rvtFile": {  
        "url": "https://path/to/zip/file.zip",  
        "pathInZip": "RevitFile.rvt"  
    },  
}
```

# ご注意：進化し続ける Forge・開発作業に終わりはなし クラウドサービス

- Forge の機能向上によって動作が変わる可能性
  - API によっては特定バージョン指定も可能
  - 例) [POST CreateFolder Command の廃止](#)
- クラウド開発にはメインテナンスが必須
- デスクトップ製品のアドイン開発とは異なります！
- 開発ベンダーとのサブスクリプション？契約が必要！！

# Forge Data Days

- Forge Data を紹介するワールドツアー
  - 5月31日～、無償、対面イベント（同時オンライン配信なし）
  - [Forge Data Days | Autodesk Forge](#)
- 日本は東京オフィスで開催
  - 7月29日
  - [Forge Data Days 東京お申込み開始](#)

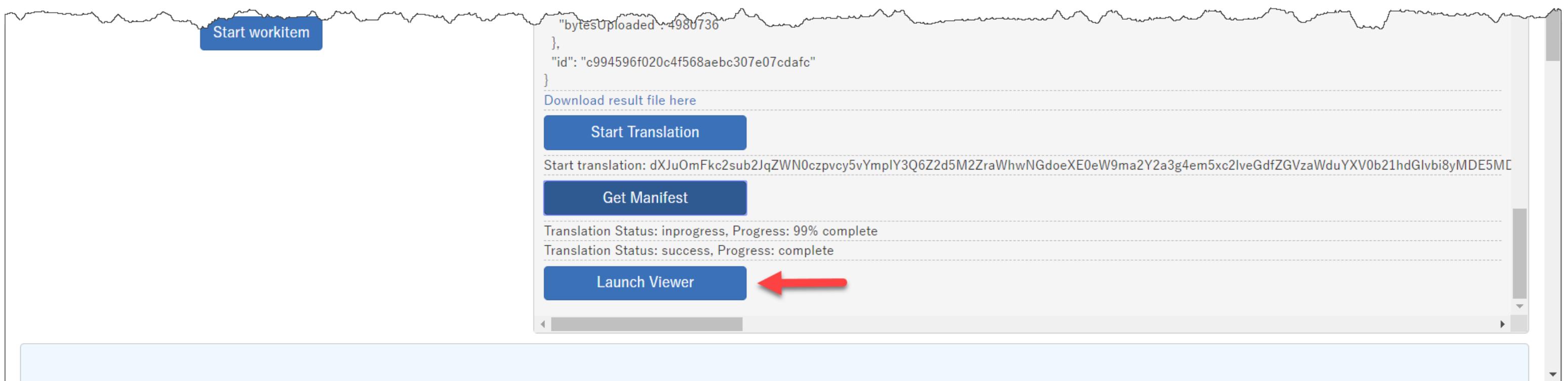




# 付録： .NET Core サンプルに Viewer 機能を 追加する

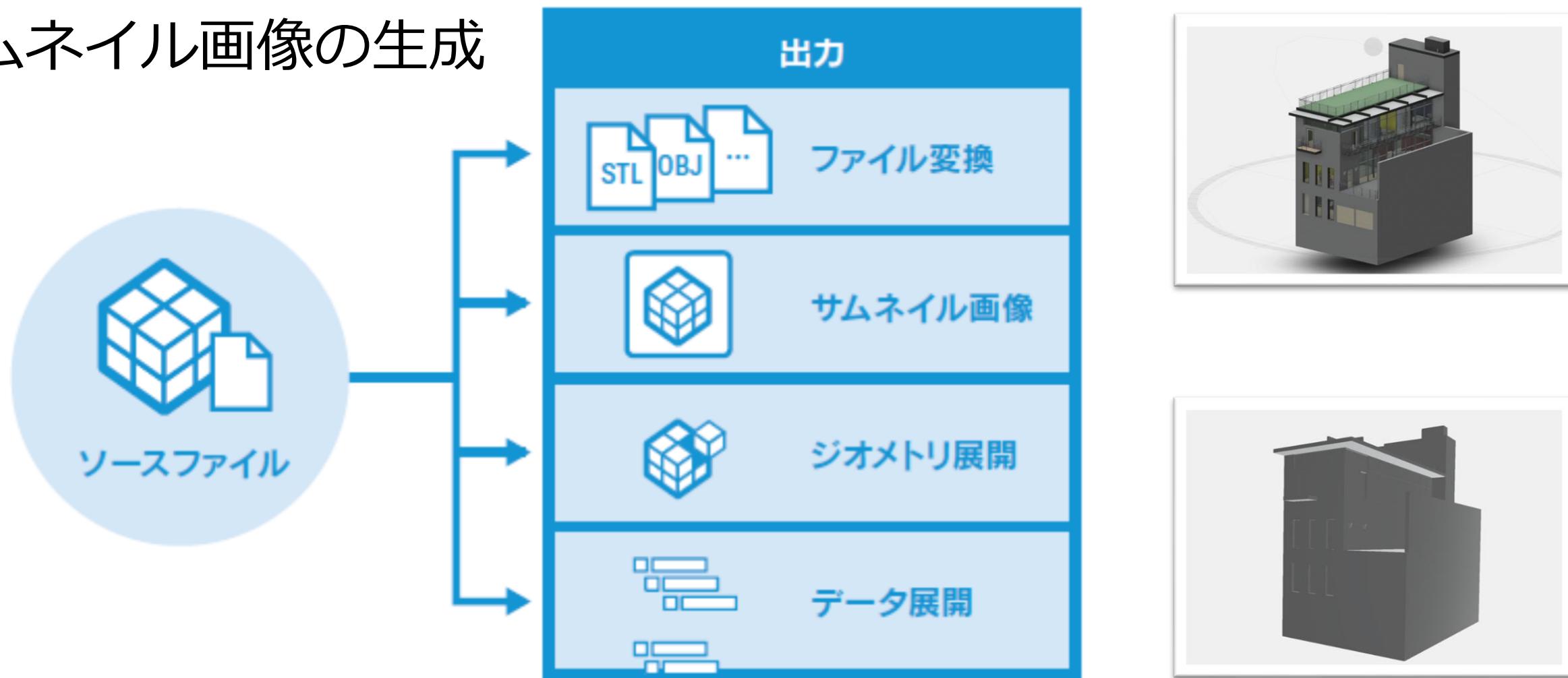
# .NET Core サンプルに Viewer 機能を追加

- 下記のブログ記事を参照しながら、Bucket に保存されている Revit プロジェクトファイルを Model Derivative API で SVF に変換して、Forge Viewer に表示する部分を追記していきましょう。
  - [https://adndevblog.typepad.com/technology\\_perspective/2019/03/design-automation-api-for-revit-learn-autodesk-forge-tutorial-sample-part2.html](https://adndevblog.typepad.com/technology_perspective/2019/03/design-automation-api-for-revit-learn-autodesk-forge-tutorial-sample-part2.html)



# Model Derivative API で SVF に変換

- デザイン ファイルを変換
  - Viewer 用に SVF 形式に変換してブラウザで表示
  - 他のデザイン ファイル形式に変換
  - ジオメトリ データやモデル階層の展開
  - サムネイル画像の生成



# マニフェストの取得

- ソースファイルの変換処理（ジョブ）の進捗状況とデリバティブ（派生ファイル）の情報（デリバティブの形式、URNなど）をJSON形式で表したもの

ソースファイルの URN を指定して SVF 形式（または他の形式）への変換処理（ジョブ）を依頼  
例えば、RVT から変換できる形式は、DWG, IFC, SVF, サムネイル画像のみ



マニフェストを取得して、デリバティブ（派生ファイル）の変換処理（ジョブ）の進捗状況をチェック



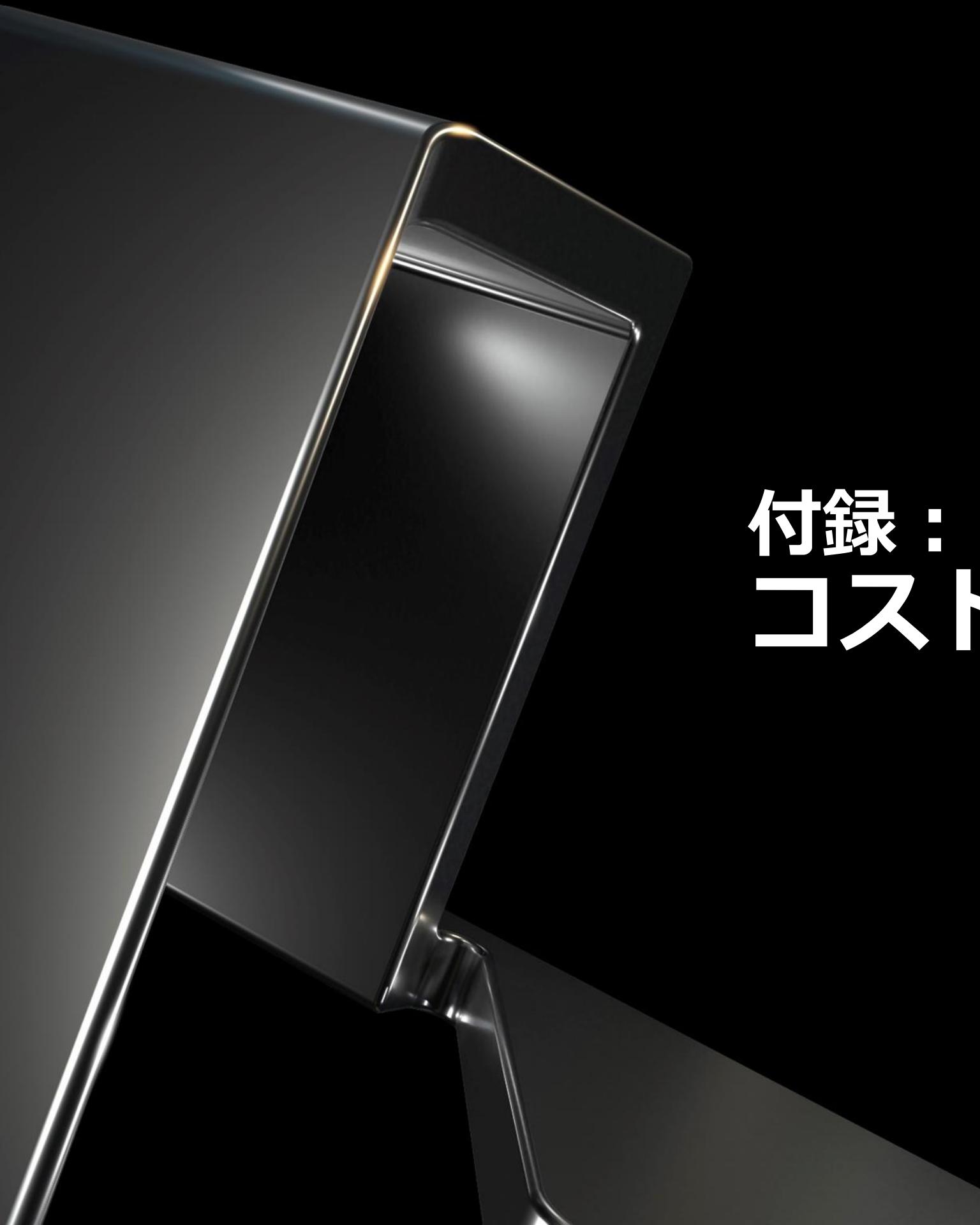
変換処理（ジョブ）が完了していれば、Viewer に表示可能な要素として viewableID が割り振られる



デリバティブ（派生ファイル）の URN を指定してダウンロードすることもできる

# Viewer 表示のためのアクセストークン

- Forge を利用するにあたっては、セキュリティの観点から、サーバーサイドで利用するアクセストークンと、クライアントサイドで利用するアクセストークンを、それぞれ分けて利用するよう推奨しております。
- インターナルトークン
  - Design Automation API, Data Management API, Model Derivative API
  - Scope.BucketCreate, Scope.BucketRead, Scope.BucketDelete, Scope.DataRead, Scope.DataWrite, Scope.DataCreate, Scope.CodeAll
- パブリックトークン
  - Forge Viewer のみ
  - Scope.ViewablesRead



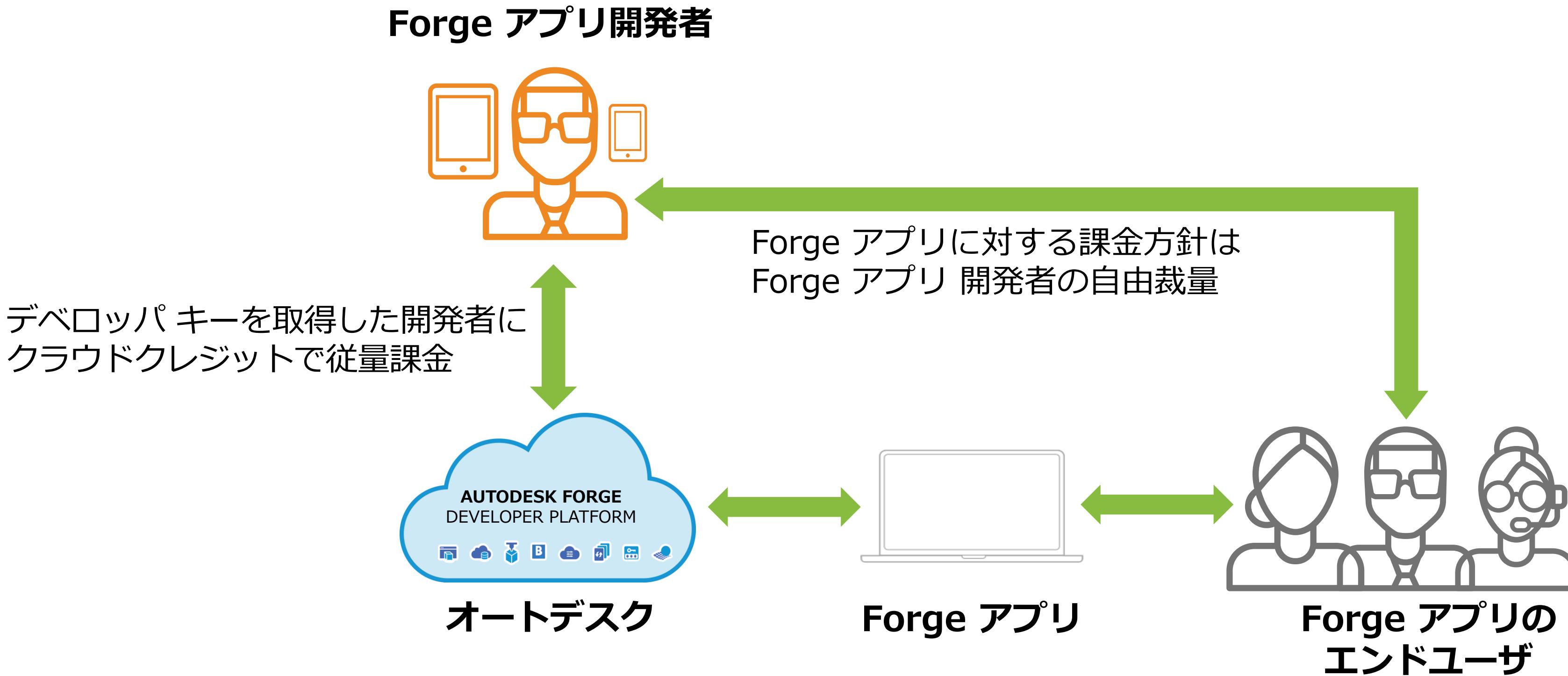
# 付録： コストについて

# Forge 利用に対する‘課金’とは？

- クラウド クレジットによる重量制‘課金’
  - オートデスク クラウド サービスのジョブ消費に対する**抽象単位**
  - <https://forge.autodesk.com/pricing>
- Design Automation API の課金とコスト算出について

API とサービス	コスト	補足
Design Automation API	2.0	クレジット クレジット / 1 時間の処理
Model Derivative API	0.5	クレジット クレジット / コンプレックス ジョブ
	0.1	クレジット クレジット / シンプル ジョブ
Reality Capture API	1.0	クレジット クレジットで 1 ギガピクセルの処理
その他すべての API とサービス	追加費用なし	Forge アカウントが必要

# Forge はデベロッパキー所有者へ‘課金’



# クラウドクレジットの Autodesk Flex への移行

- Forge クラウドクレジットの移行フェーズ
- 第1フェーズ（2022年3月29日PST）
  - クラウドクレジットは、1 クレジットあたり 1 US ドル（または現地通貨換算額）から 3 US ドルに価格改定
  - 日本では現在 1 クラウドクレジット税抜 160 円から 480 円へ
- 第2フェーズ（2022年Q3）
  - （Autodesk Flex の完全移行までForge で使用するクラウドクレジット追加購入分）は、Autodesk から直接購入のみとなり、Autodesk eStore や Autodesk パートナーを通じた購入不可

# API 別の消費クラウドクレジットとコスト（税抜き）

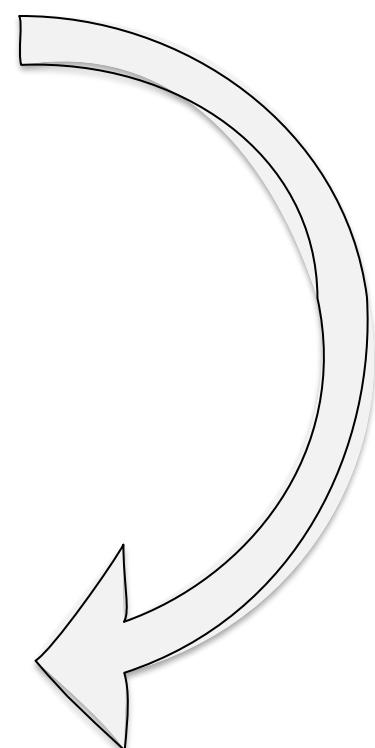
※ Design Automation API は 1 時間処理した 1 CPU 時間単位

API	消費クラウドクレジット	単位	実質コスト（税抜）
Model Derivative コンプレックス ジョブ (Revit、Navisworks)	1.5 クラウドクレジット	変換	1.5ドル - 240円
Model Derivative シンプル ジョブ (Revit、Navisworks 以外)	0.2 クラウドクレジット	変換	0.2ドル - 32円
Design Automation – AutoCAD	4 クラウドクレジット	処理時間	4.0ドル - 640円
Design Automation – Revit、Inventor、3ds Max	6 クラウドクレジット	処理時間	6.0ドル - 960円
Reality Capture	3.5 クラウドクレジット	1ギガピクセル毎	3.5ドル - 560円

API	消費クラウドクレジット	単位	実質コスト（税抜）
Model Derivative – コンプレックス ジョブ (Revit、Navisworks、IFC)	0.5 クラウドクレジット	変換	1.5ドル - 240円
Model Derivative – シンプル ジョブ (Revit、Navisworks、IFC 以外)	0.1 クラウドクレジット	変換	0.3ドル - 48円
Design Automation – すべて	2 クラウドクレジット	処理時間	6.0ドル - 960円
Reality Capture	1 クラウドクレジット	60枚の写真	3.0ドル - 480円

# 3月29日以降の購入方法

- Forge サブスクリプション新規購入
  - eStore
  - オフライン（見積書→署名→PO発行/請求書発行）
- サブスクリプション期間中のクラウドクレジットの購入
  - eStore
  - オフライン（見積書→署名→PO発行/請求書発行）
- Forge サブスクリプション新規購入/  
サブスクリプション期間中のクラウドクレジットの購入
  - オフライン（見積書→署名→PO発行/請求書発行）



# 購入アクションの方法

- [forge.orders@autodesk.com](mailto:forge.orders@autodesk.com) にメール  
または
- <https://forge.autodesk.com/ja/pricing> のフォーム



- Overuse の開発者にはオートデスクからコンタクト

# Pricing ページ

- <https://forge.autodesk.com/ja/pricing>

The screenshot shows the Autodesk Forge Pricing page. At the top, there's a navigation bar with the Autodesk logo, a search icon, and a 'SIGN IN' button. Below the navigation is a large banner with the text 'FORGE 価格' (Forge Pricing) and a '営業担当へのお問い合わせ' (Contact Sales Representative) button. The main content area features a dark background with a metallic 3D model of a chair. It includes a heading '無償でフレキシブル：3つのシンプルなステップで Forge での開発を始めよう' (Flexible and free: Start developing on Forge with 3 simple steps), a 'Forge 無償体験版' (Forge Free Trial) section with a cloud icon and a computer monitor icon, and a text block about the trial offer.

FORGE  
価格

営業担当へのお問い合わせ

無償でフレキシブル：3つのシンプルなステップで Forge での開発を始めよう

Forge 無償体験版

全機能へのアクセスと 100 クラウド クレジットで、まずは Forgeをお試しください

機能制限なし：すべての API とサービス、100 クラウド クレジット、5 GB のストレージをご利用いただけます。

無償体験版のチャージ：チケットトリアル ウォータースリーブサ

# クラウドクレジットは当面存続

- 未使用的クラウドクレジットは購入後 1 年間有効
  - 2022年3月28日に購入した CC は2023年3月27日まで使用可能
- Flex 各国で導入されるわけではない
  - Flex 未適用国では Flex 導入まで CC の購入と利用を継続
- Autodesk Flex トークン ≠ EBA Token Flex トークン

# 今後のリソース

- サンプル コード : GitHub
  - <https://github.com/Developer-Autodesk>
  - <https://github.com/Autodesk-Forge>
- サポート : StackOverflow
  - <https://forge.autodesk.com/en/support/get-help>
- ブログ : Forge Community Blog
  - <https://forge.autodesk.com/blog>



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.

© 2022 Autodesk. All rights reserved.