

 **AUTODESK Forge**

Forge Online Training – Design Automation Inventor

加藤 丈博

Developer Advocacy & Support、Autodesk Developer Network

アジェンダ – ご紹介する内容

- はじめに
- Design Automation APIの理解
- Learn Forge : モデルを修正する
- 付録 : Forge Design Automation でiLogicを実行する
- 付録 : コストについて

A sleek, dark, metallic-looking device, possibly a smartphone or tablet, is shown at an angle. The screen is illuminated with a soft, white light, creating a bright spot in the center. The device has a thin, polished frame that reflects light. The background is a solid black, which makes the device stand out.

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

<HTML>
HTML



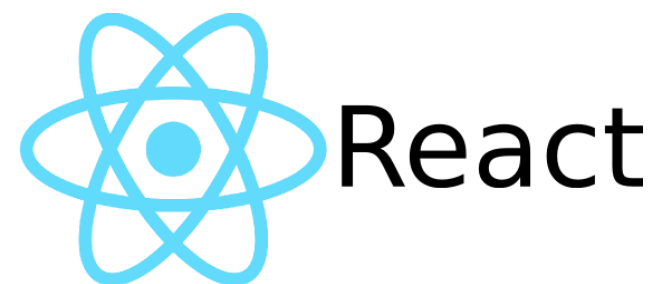
{CSS}
CSS



(JS)
JavaScript



RESTful API
GET PUT POST DELETE





要素技術

 **AUTODESK**
Forge

**Web
API**

アイデンティティ

**Web
API**

プロジェクト データ

**Web
API**

表示

**Web
API**

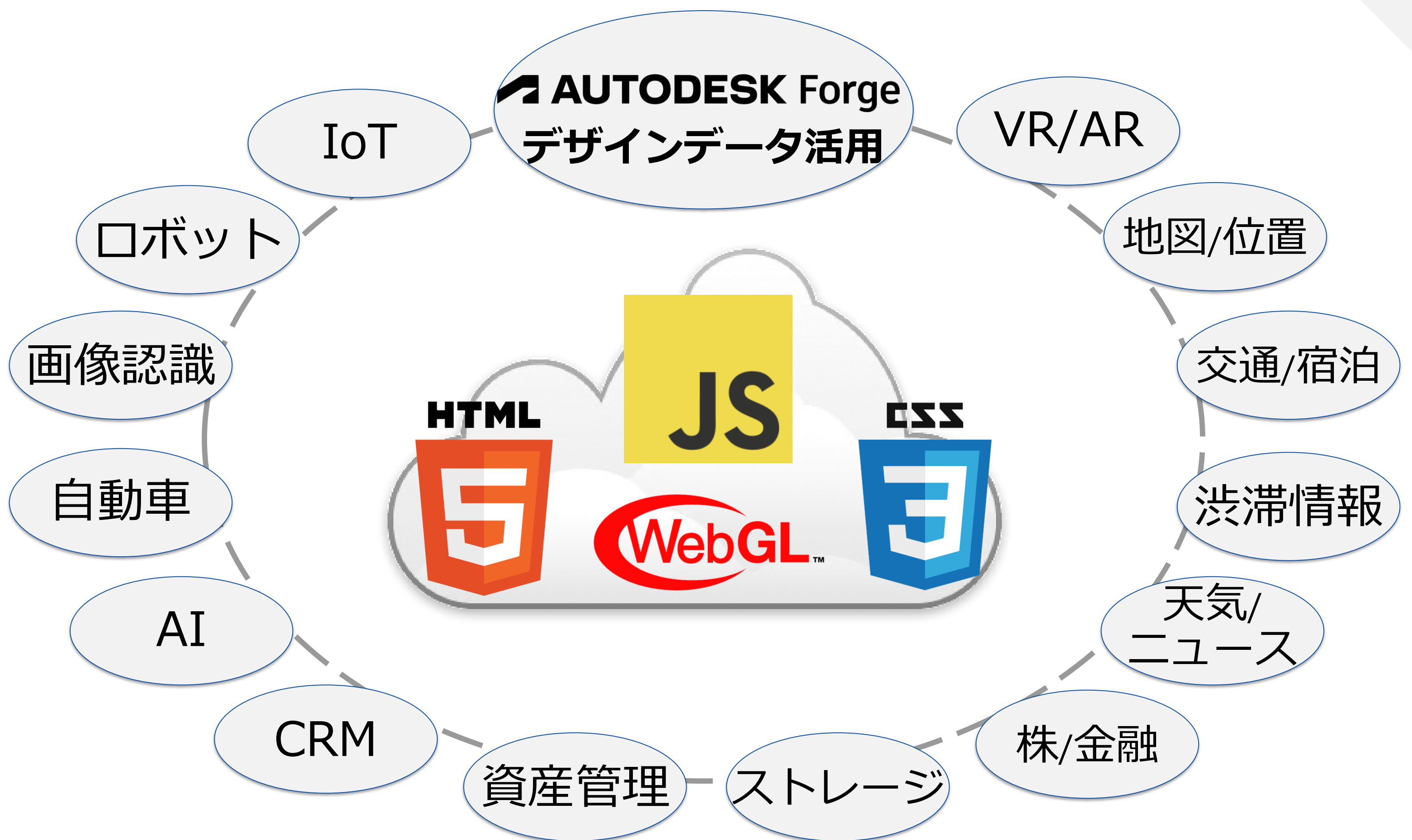
検索

**Web
API**

アクティビティ

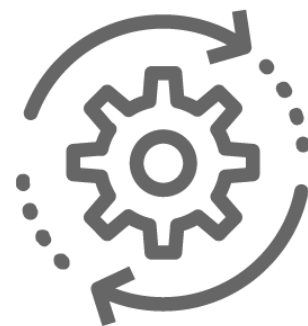
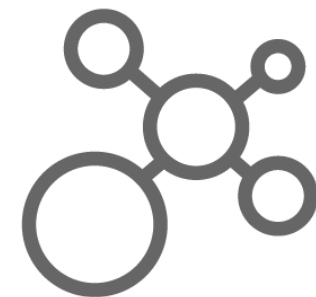
 **AUTODESK**

powered by
amazon
web services



デスクトップ製品のアドイン開発者の方へ

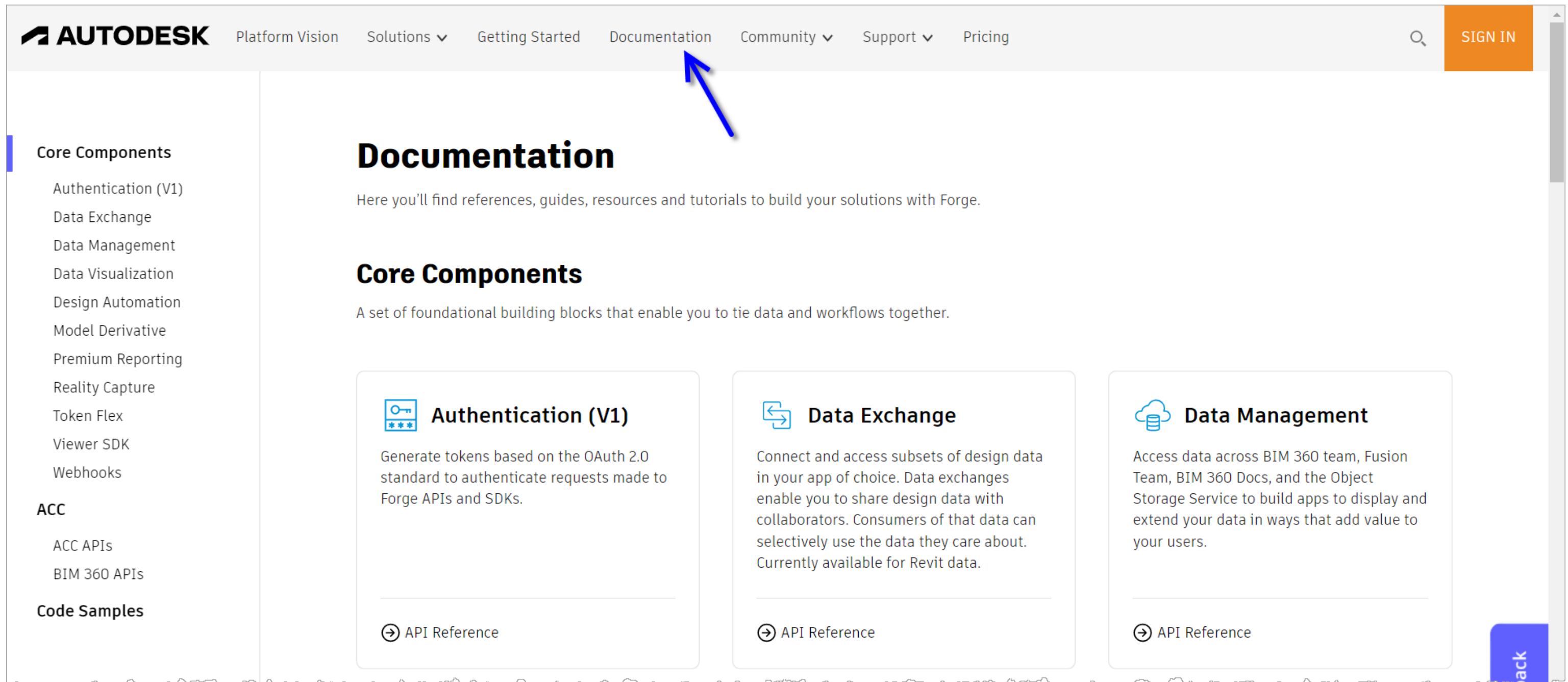
- デスクトップ開発と Web 開発の違いをご理解ください
 - Web アプリの特性 – セキュリティ視点
 - 通信経路
 - 呼び出し数制限
 - 非同期処理
 - 仮想環境
 - オンプレミスとの違い



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



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

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.
→ API Reference
- 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.
→ API Reference
- 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

Core Components

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

ACC

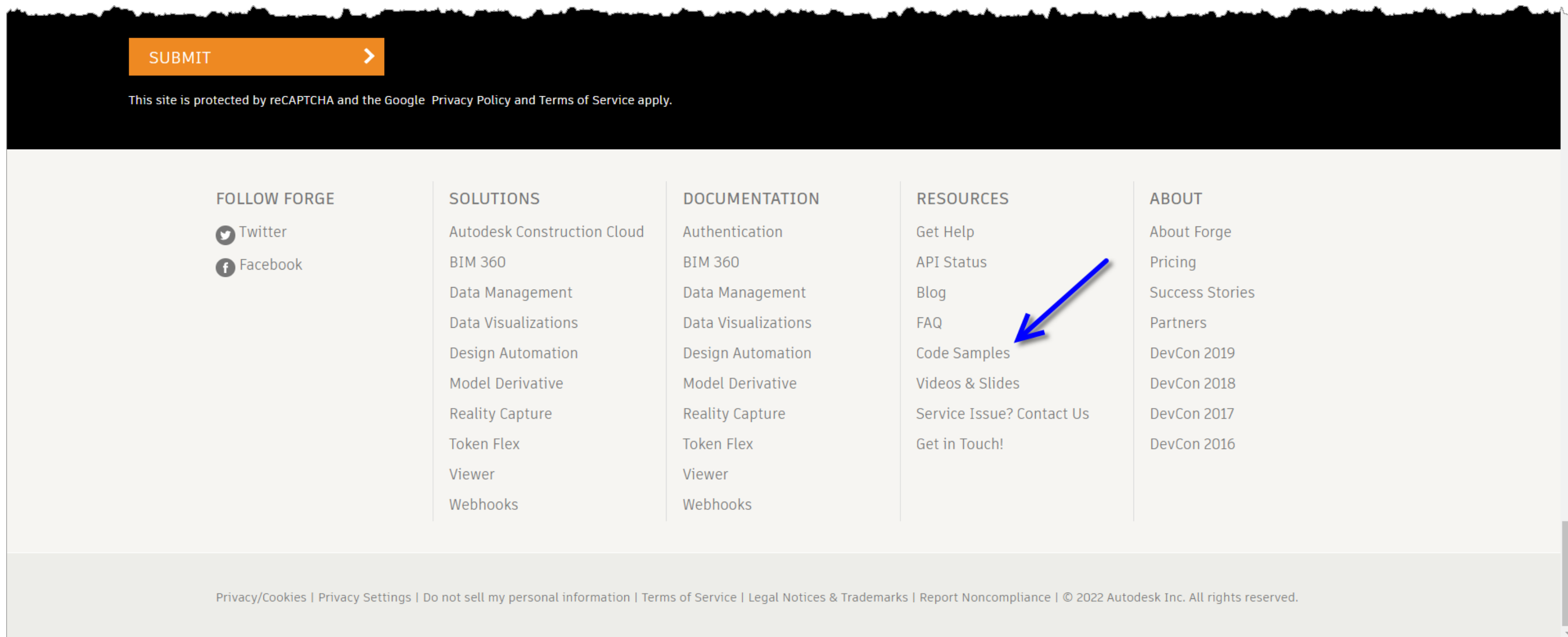
- ACC APIs
- BIM 360 APIs

Code Samples

back

Forge サンプル

- Forge ポータル下部 **Code Samples** からのアクセス



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 screenshot displays the Autodesk Forge SDK marketplace interface. It features three project cards, each with a header image, a title, a star rating, and a description.

- Forge Responsive Connected Database**: 238 stars. Description: "A responsive React-based web application that showcases the use of Autodesk Forge Viewer and Forge web services ..."
- Forge Node.js SDK**: 97 stars. Description: "Provides Node.js SDK to help you easily integrate Forge REST APIs into the application"
- Authorisation and translation via console application**: 14 stars. Description: "Command line tool demonstrating the authorisation and translation process using a Node.js console application"

At the bottom, there is a footer bar with the text "Developer Autodesk" and "Large Model Viewer - View & Data API", along with logos for various development tools and languages.

開発環境 – これに限定するものではありません

- Web ブラウザ : **Google Chrome**
- HTML/JavaScript エディタ : **VS Code**
- テスト ツール : **Postman**
- Web サーバー実装 : **Node.js**
- リポジトリ ユーティリティ : **git for Windows**



学習リソース : Learn Forge

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

The screenshot shows the 'Learn Forge' website interface. On the left is a sidebar with a blue bracket grouping the first five items: 'モデルを修正する' (Modify Model), 'サーバを作成する' (Create Server), '基本アプリのUI' (Basic App UI), 'プラグインを準備する' (Prepare Plugin), and 'Activity を定義する' (Define Activity). Below these are 'WorkItem を実行する' (Execute WorkItem), '実行とデバッグ' (Run and Debug), 'ビューアのエクステンション' (Viewer Extensions), and '基本スケルトン' (Basic Skeleton). The main content area has a language selector at the top right with 'EN', 'JA' (selected), 'ZH-CN', and 'ZH-TW'. The title 'モデルを修正する' is prominently displayed. The introductory text explains that the tutorial covers uploading input files, modifying 'width' and 'height' parameters, saving output files, and defining AppBundle and Activity in the user interface. A green vertical bar highlights the bottom section of the page, which contains the text: '入力ファイルと出力ファイルの両方が OSS バケットに保存されます。これらを表示するには、「モデルを表示する」のチュートリアルを使用してください。' (Both input and output files are saved to the OSS bucket. To display them, use the 'Display Model' tutorial.)

モデルを修正する

サーバを作成する

基本アプリのUI

プラグインを準備する

Activity を定義する

WorkItem を実行する

実行とデバッグ

ビューアのエクステンション

基本スケルトン

EN JA ZH-CN ZH-TW

モデルを修正する

このチュートリアルでは、入力ファイルのアップロード、`width` と `height` パラメータの変更、および出力ファイルの保存を実行できる Web アプリの作成方法について説明します。ユーザ インタフェースでは、出力結果を使用して AppBundle と Activity を(初期設定として)定義し、WorkItem を実行することができます。

入力ファイルと出力ファイルの両方が OSS バケットに保存されます。これらを表示するには、「モデルを表示する」のチュートリアルを使用してください。

Forge を始めるには?

- まずは <https://forge.autodesk.com/> へ



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

Applications からのアプリ登録

どのAPIを使うか指定

API Access

Select the APIs you want to use in your app.

Delete this application

21 APIs selected

- ☒ App Framework API
- ☒ Autodesk Construction Cloud API
- ☒ BIM 360 API
- ☒ BIM 360 Client API
- ☒ Compute API
- ☒ Data Exchange API
- ☒ Design Automation API
- ☒ Desktop Connections API
- ☒ Developer Portal API

FOLLOW FORGE

SOLUTIONS

DOCUMENTATION

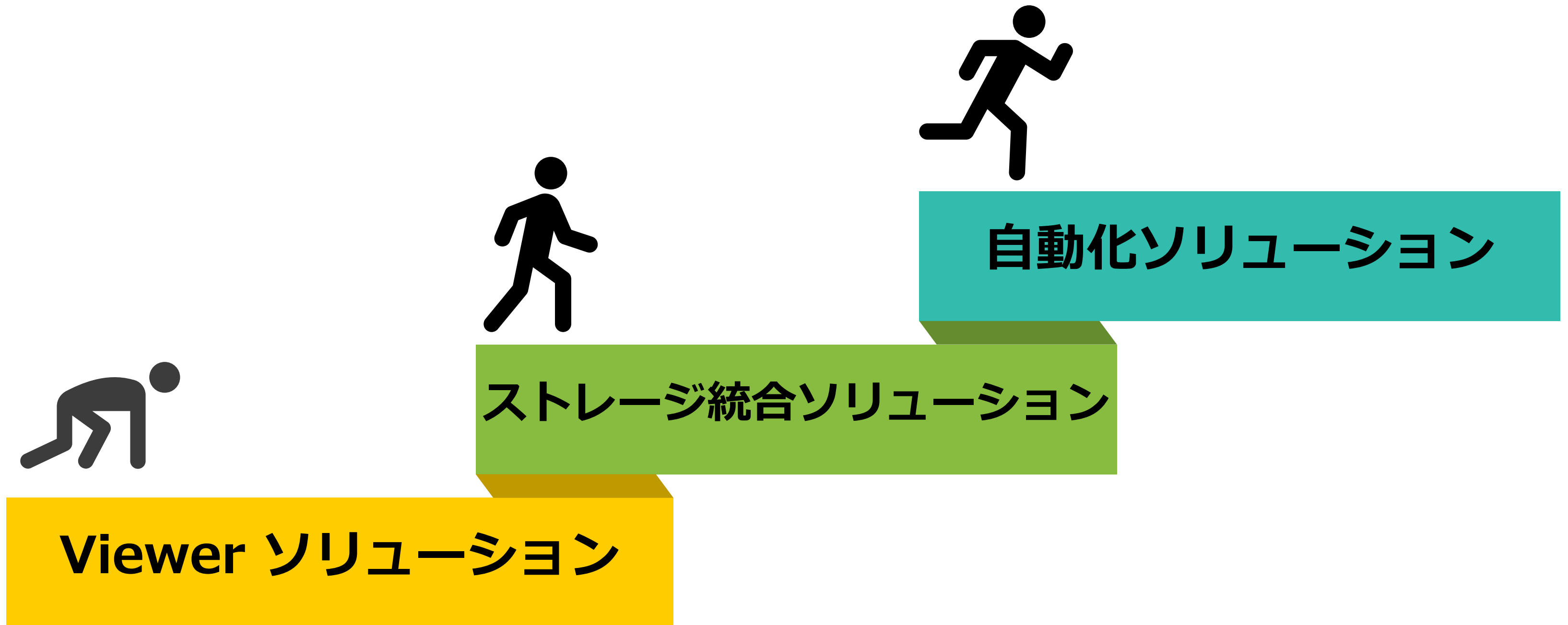
RESOURCES

ABOUT



Design Automation APIの理解

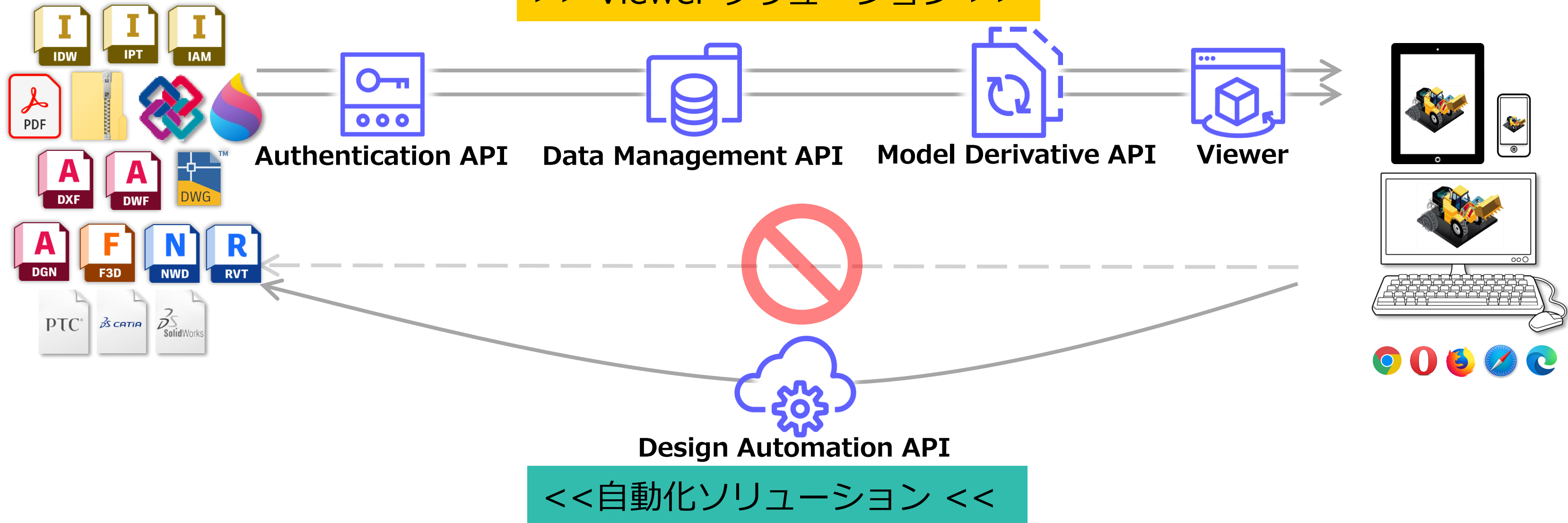
Forge の主なソリューション



Design Automation API の連携

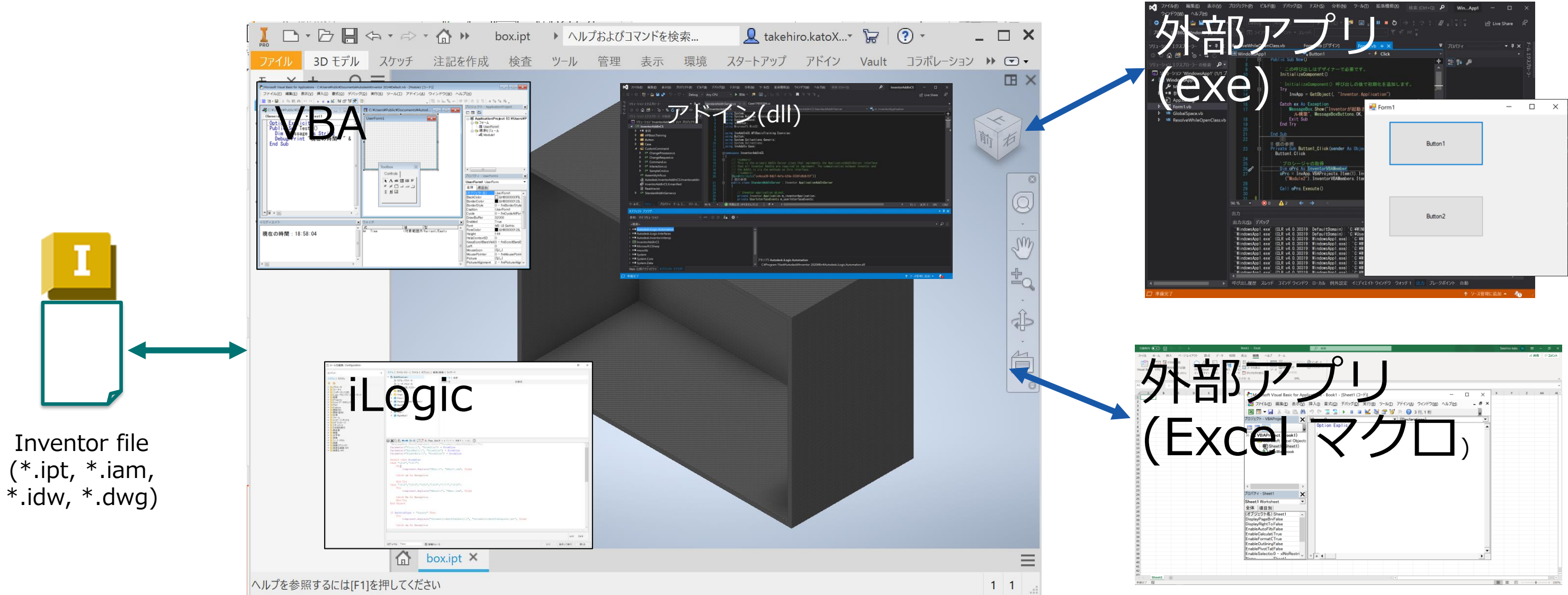
- Forge Viewer からのオリジナルデータへの反映は不可
- Design Automation API の連携は可能

>> Viewer ソリューション >>



Inventor のカスタマイズ API

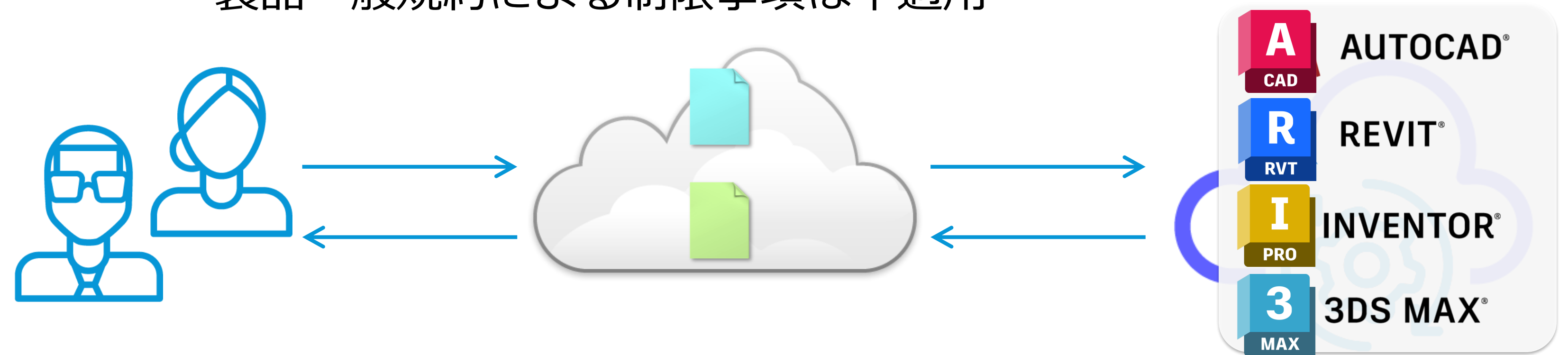
様々な「設計業務（と付随業務）」の自動化による、省力化、生産性の向上



APIの呼び出し方法は色々だが、APIの実行にはローカル環境での
Inventorエディタの起動が必須

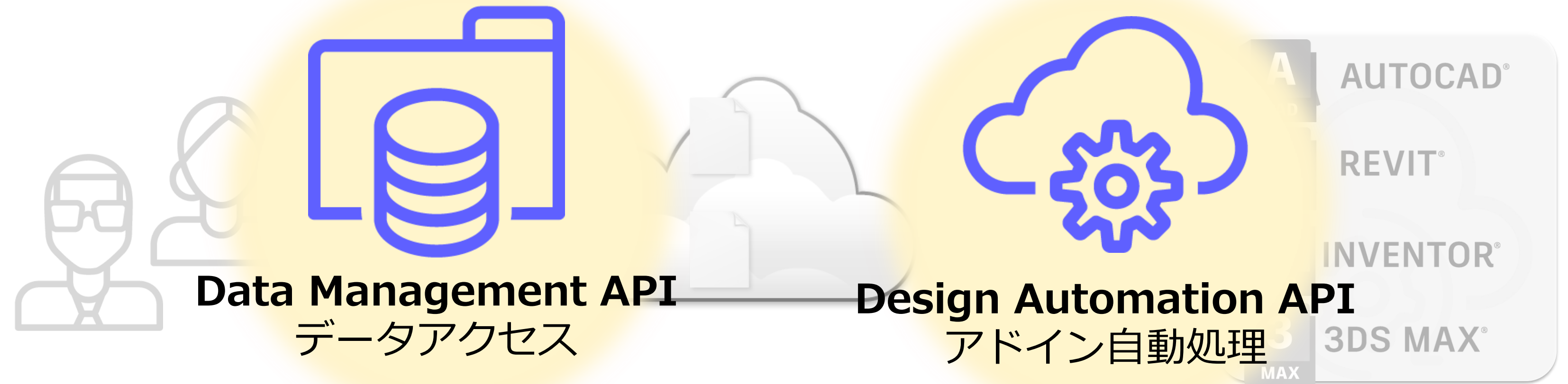
自動化ソリューション

- **Design Automation API**
- CAD エンジンで**アドイン**を実行する仕組み
 - 対面操作を意図した '製品' ではない
 - CAD エンジンは UI を持たないためオーバーヘッドなし
 - 製品一般規約による制限事項は不適用



自動化ソリューション

- Design Automation API
- CAD エンジンでアドインを実行する仕組み
 - 製品一般規約による制限事項は不適用
 - CAD エンジンには UI を持たないためオーバーヘッドなし



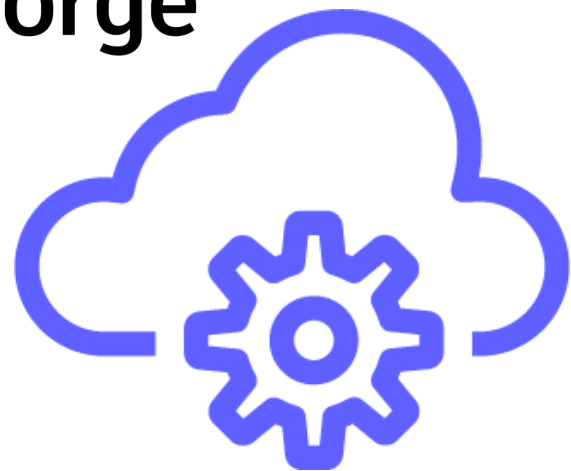
Design Automation API

Web API + デスクトップ API の知識が必要

Forge 開発

製品別アドイン開発

 **AUTODESK**
Forge



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

- AutoCAD Web 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

Design Automationの用語



AppBundle

Inventor プラグイン



Activity

AppBundleとWorkItemのインターフェース



WorkItem

Inventor Serverを実行するJob



Engine

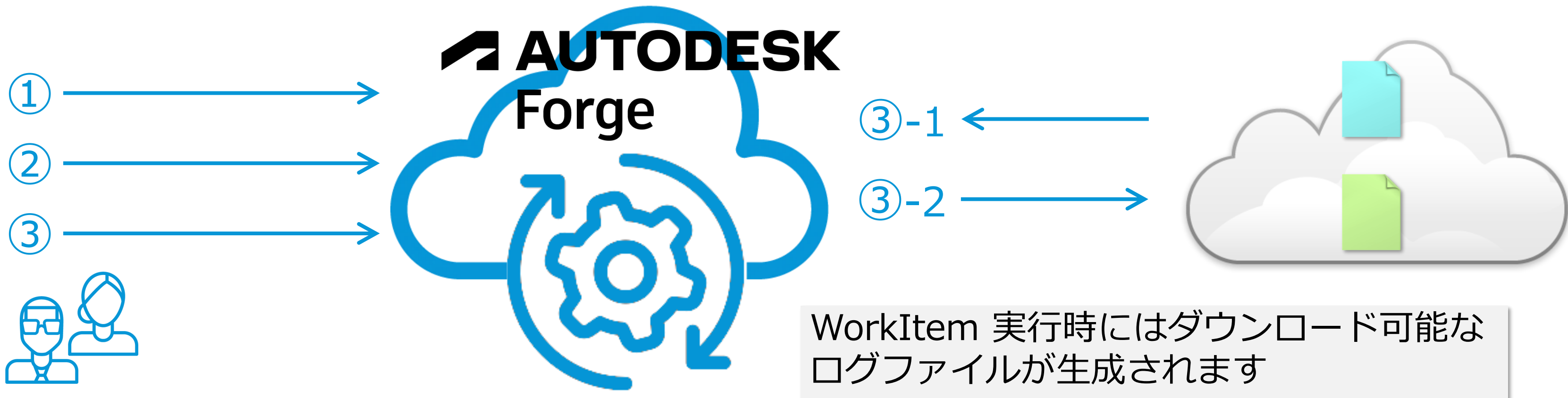
複数バージョンのInventor Server

Design Automation API とは？

- **起動したコアエンジンにアドインをロード/実行**
 - AppBundle（事前登録が必要）
- **成果物としてデザインファイルや関連ファイルの作成してダウンロードして利用**
 - Activityで入出力ファイル/パラメータを宣言（事前登録が必要）
 - WorkItem（ファイル/パラメータ渡しを実行）
- **クラウド上の CAD コアエンジンをリモートで起動**
 - WorkItem（実際の処理実行）

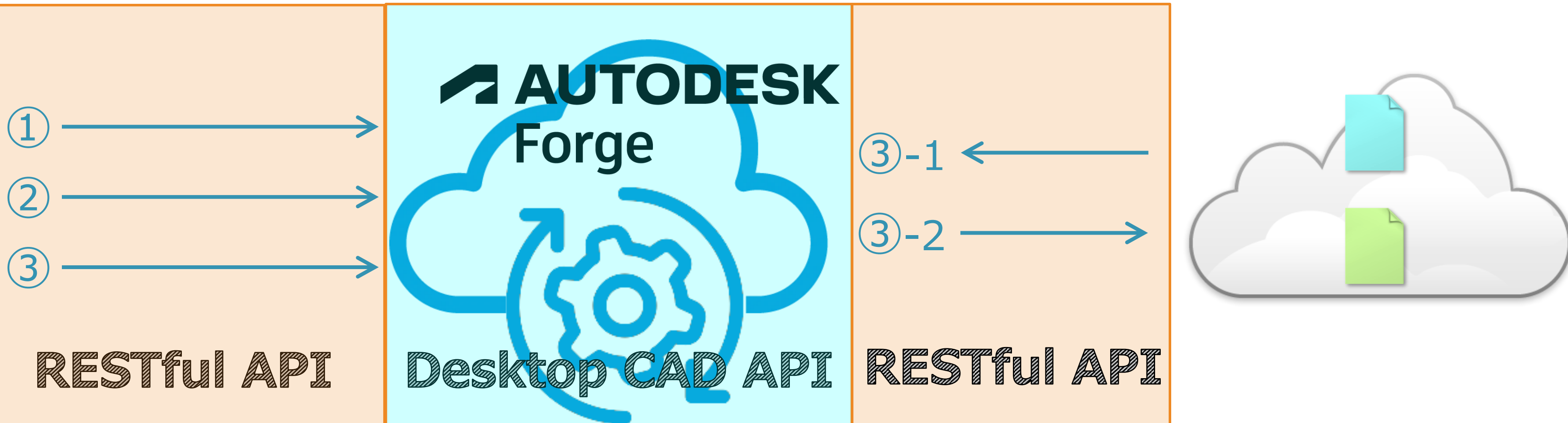
Design Automation API とは

- クラウドのコアエンジン上でアドインを実行する仕組みを提供
 - ① AppBundle(アドイン本体+自動ロード定義) のアップロード&登録
 - ② Activity を登録 (AppBundle が使用する入出力ファイルを指定)
 - ③ WorkItem を指定、実行 (Activity と各種パラメータを指定)



Design Automation API とは

- クラウドのコアエンジン上でアドインを実行する仕組みを提供
 - ① AppBundle(アドイン本体+自動ロード定義) のアップロード&登録
 - ② Activity を登録 (AppBundle が使用する入出力ファイルを指定)
 - ③ WorkItem を指定、実行 (Activity と各種パラメータを指定)



DA4Iで利用可能なコアエンジン バージョン

- デスクトップ製品と同期したコアエンジンバージョン
 - AppPackageはエンジンバージョンに合わせた作成が必須
 - コアエンジンIDの形式はコアエンジン毎に異なる
 - 2022年10月5日現在（Inventor 2018 ～2023が対象）

"Autodesk.Inventor+22" ⇒ Inventor 2018

"Autodesk.Inventor+23" ⇒ Inventor 2019

"Autodesk.Inventor+24" ⇒ Inventor 2020

"Autodesk.Inventor+2021" ⇒ Inventor 2021

"Autodesk.Inventor+2022" ⇒ Inventor 2022

"Autodesk.Inventor+2023" ⇒ Inventor 2023

利用手順と endpoint



※ Nickname の使用は任意

Design Automation API で扱う ID

- AppBundle と Activity で利用

- Client ID に関連付けられる
 - Nickname に置き換えが可能

- 例)

"**piFGJwGRnAxwUznvfSDNRRWqflbyvJAL**.UpdateIPTParamActivity+dev"



- App 名

- AppBundle と Activity 登録時に ID に指定

- Alias 名

- AppBundle と Activity バージョン識別に利用
- 例) **prod**、**dev**

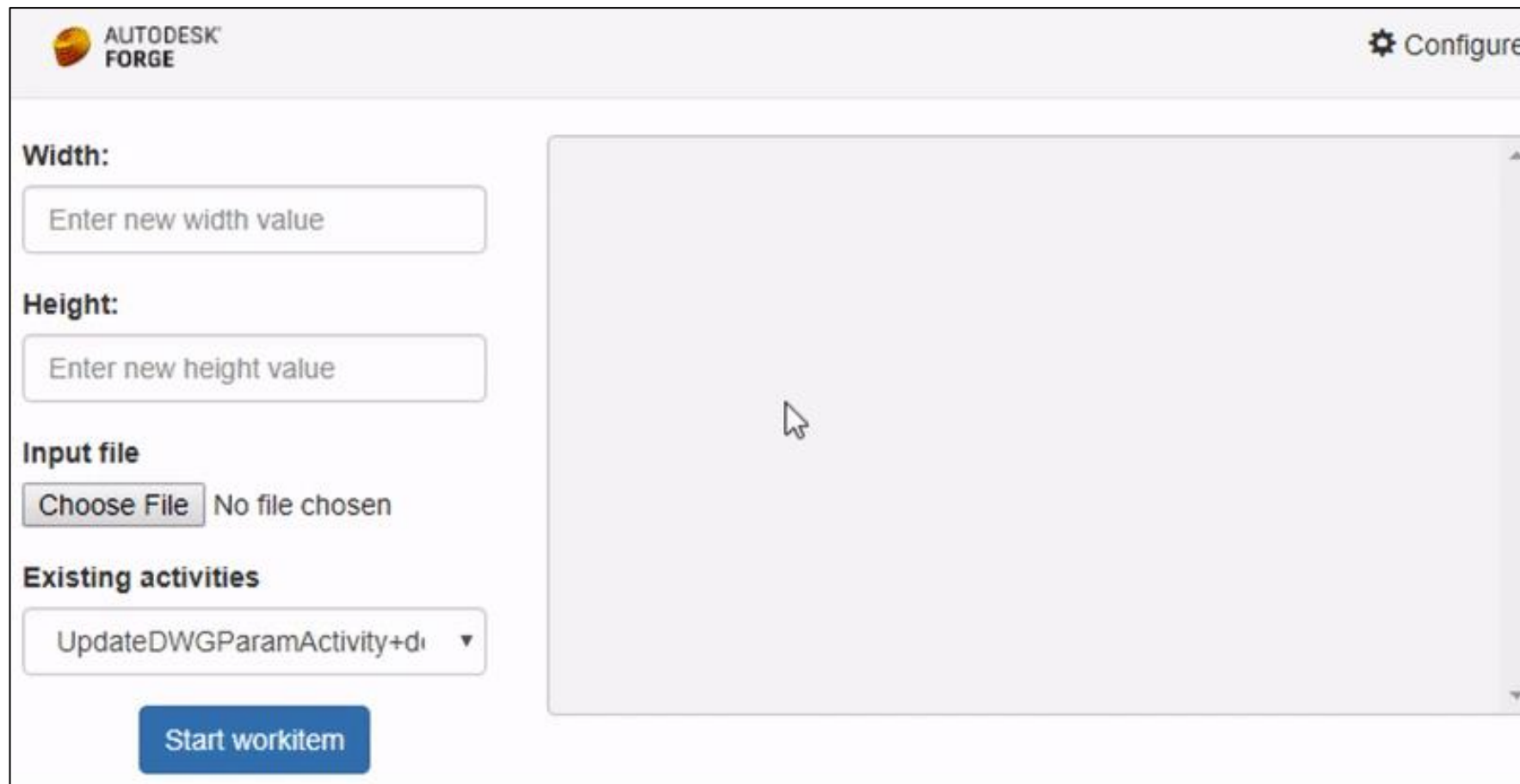
- WorkItem は作成時に個別 ID を割り当て



Learn Forge : モデルを修正する

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



The screenshot shows the Autodesk Forge web interface. At the top left is the Autodesk Forge logo. At the top right is a 'Configure' button with a gear icon. The main form has several sections: 'Width:' with a text input field containing 'Enter new width value'; 'Height:' with a text input field containing 'Enter new height value'; 'Input file' with a 'Choose File' button and the text 'No file chosen'; 'Existing activities' with a dropdown menu showing 'UpdateDWGParamActivity+di'; and a 'Start workitem' button at the bottom left. A large, empty rectangular area is on the right side of the form, with a mouse cursor pointing at it.

言語の選択

- **Node.js**
- .NET Core

今回のチュートリアルで使用する開発環境

- Visual Studio Code
- 7zip
- ngrok ツール
- Google Chrome ブラウザ
- Git for Windows （必須ではありません）

※本トレーニングでは、プラグインのビルドツール（Visual Studio Professional等）は使用せず、ダウンロードしたプラグインモジュールを使用します。

チュートリアルของgoalイメージ

 AUTODESK
FORGE

Configure

Width:

Height:

Input file
 No file chosen

Existing activities

UpdateDWGParamActivity+di ▼

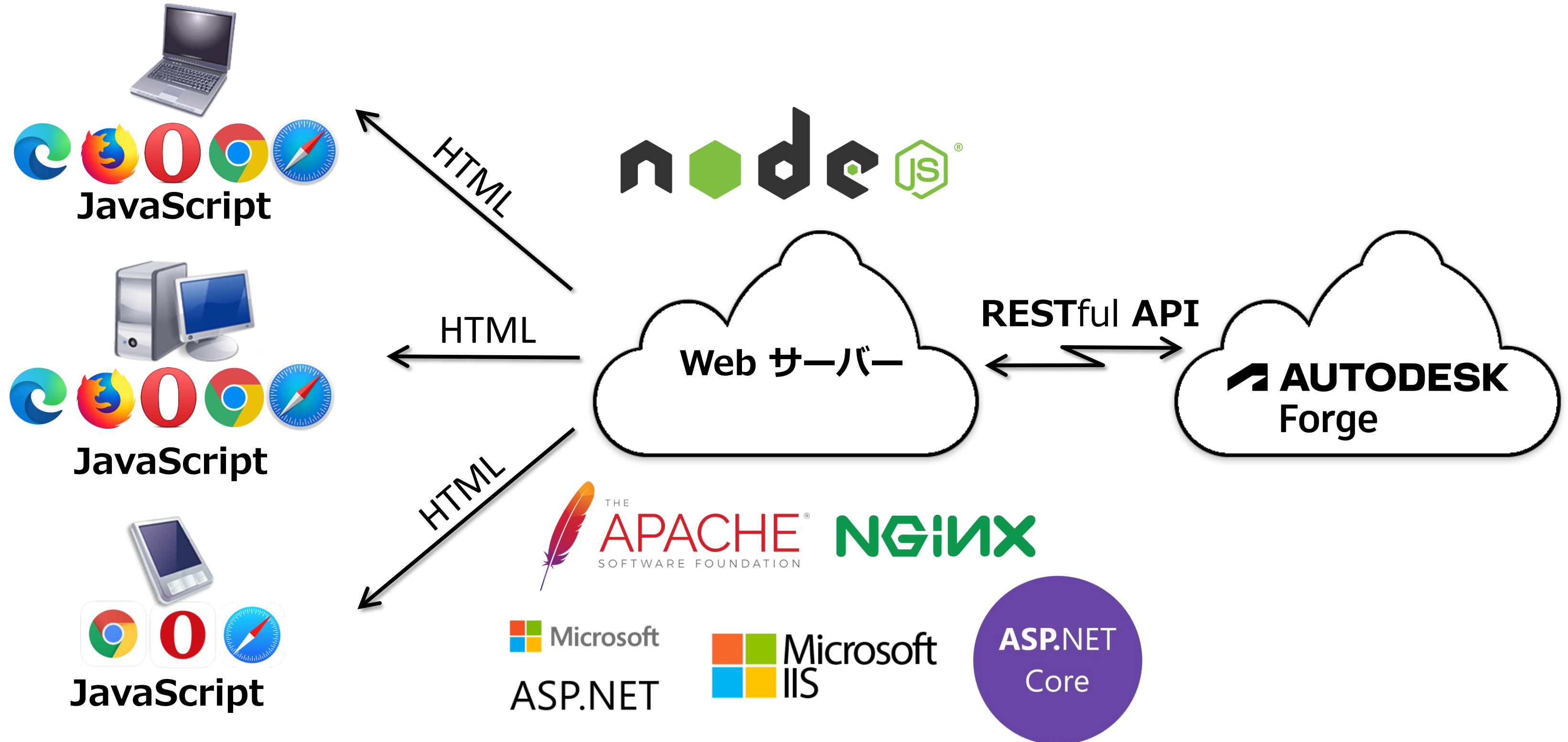
Start workitem

何が必要？

- Webページ：HTML/CSS/JavaScript
- Webサーバ：Webページをホスト、Autodesk Forge APIの実行
- プラグイン：Inventor Plugin (.net アセンブリ)
- 元データ：Inventorのパートファイル
- Autodesk Forge App：Client Idとシークレット

...

Webサーバの役割



- オープンソース
- JavaScript をサーバー上で実行するための環境
 - イベント駆動方式を利用しノンブロッキングI/Oモデルによりリクエストを同時に処理。c10K問題にも対応可能
- Google V8 JavaScript Engine が使用
- Node Package Manager で拡張可能
- 今回使用しているパッケージ(ミドルウェア) :
 - express
 - request
 - serve-favicon
 - ...



サーバを作成する

Node.jsサーバの作成

1. VS Codeでプロジェクトフォルダを開く

- C:¥tempにフォルダforgesampleを作成
- VSCodeのファイルメニューから、C:¥temp¥forgesampleを開く

2. ターミナルでnpm initを実行

- エントリ ポイントにはstart.js を指定

3. 必要なパッケージをインストール

```
出力   ターミナル   JUPYTER   デバッグ コンソール   問題

Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

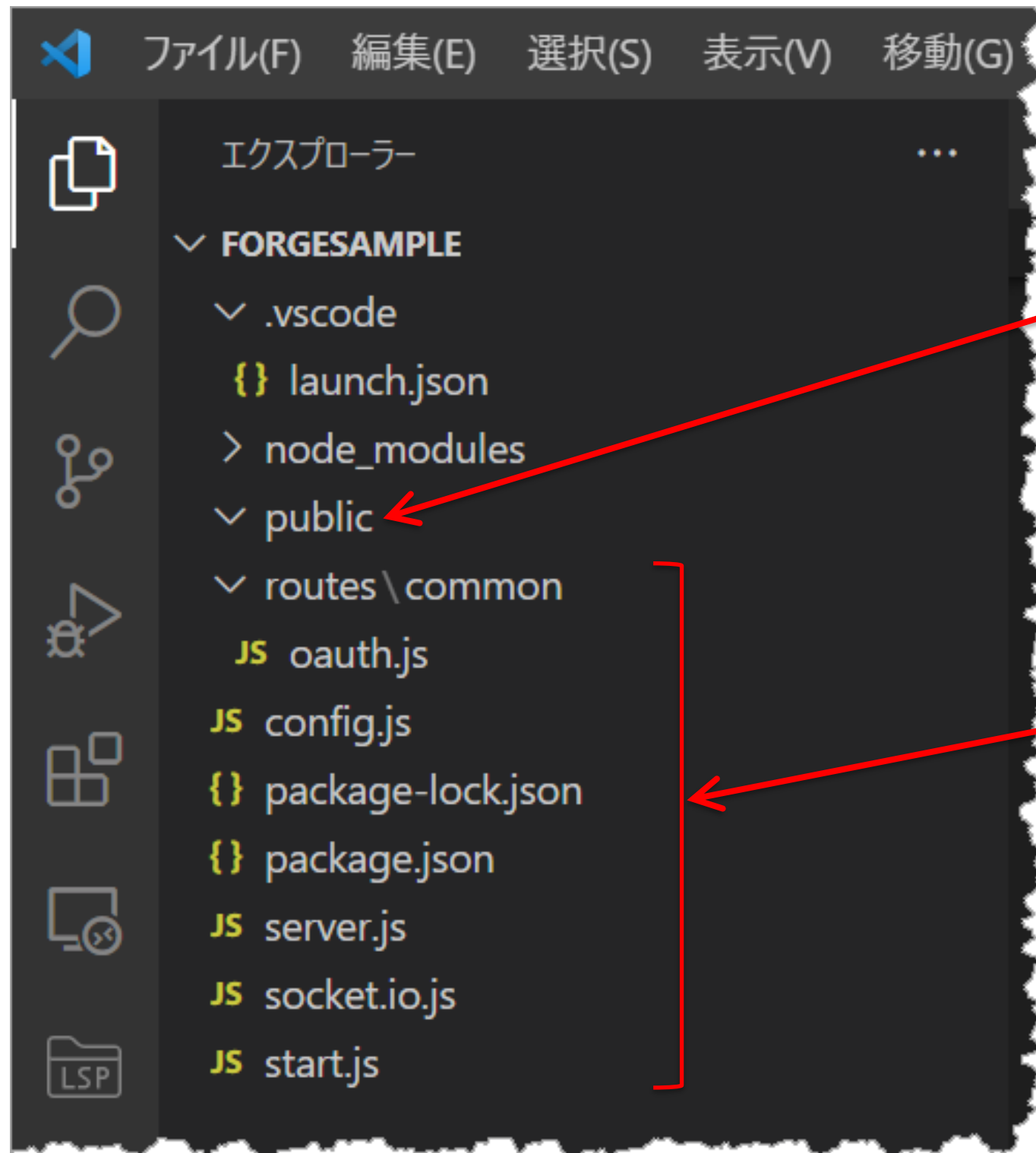
C:\TEMP\forgesample>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (forgesample)
version: (1.0.0)
description:
entry point: (index.js) start.js
test command:
git repository:
keywords:
```

補足：ここまでのプロジェクト構成



クライアントからアクセスするコンテンツ（html, css, js）を配置するフォルダ

Web サーバー側で実行される内容



基本アプリの UI

HTML の構造 メイン画面

index.html

The image shows a screenshot of the Autodesk Forge web interface, specifically the 'index.html' file. The interface is divided into several sections, each annotated with HTML tags and their corresponding visual elements:

- Navigation Bar:** Annotated with `<nav> ~ </nav>`. It contains a 'Width:' input field, a 'Height:' input field, an 'Input file' section with a 'Choose File' button, and an 'Existing activities' dropdown menu.
- Form Inputs:** Two input fields are annotated with `<input> ~ </input>`. The first is labeled '幅の入力' (Width input) and the second is labeled '高さの入力' (Height input). A file input field is annotated with `<input type=file> ~ </input>`.
- Activity Selection:** A dropdown menu is annotated with `<select> ~ </select>`. It is labeled 'Activityの選択' (Activity selection) and has a downward arrow.
- Start Button:** A button labeled 'Start workitem' is annotated with `<button> ~ </button>`.
- Preformatted Text:** A large area on the right is annotated with `<pre> ~ </pre>`.
- Span:** A small area at the top right is annotated with `<span> ~ </span>`.

The interface also includes a 'Configure' button in the top right corner and a 'Start workitem' button at the bottom left.

HTML の構造 モーダルポップアップ

index.html

Create/Update AppBundle & Activity ×

Define AppBundle & Activity only once.
Redefine only when your plugin code change (creates a new version).

Select a local AppBundle:

Tip: Make sure .ZIP bundles are placed at **/bundles/** folder

Select engine:

For this sample the .ZIP name is used as suffix to define **AppBundle** and **Activity** names.
Activities will have file and params input, and file output.

Clear account Create/Update

<div class="modal fade"> ~ </div>

 ~

<pre> ~ </pre>

<input> ~ </input>

<input> ~ </input>

<input type=file> ~ </input>

<select> ~ </select>

<div class="modal-dialog"> ~ </div>

<select> ~ </select>

Appバンドルの選択 ▼

<select> ~ </select>

DA エンジン選択 ▼

<button> ~ </button>

Clear Account

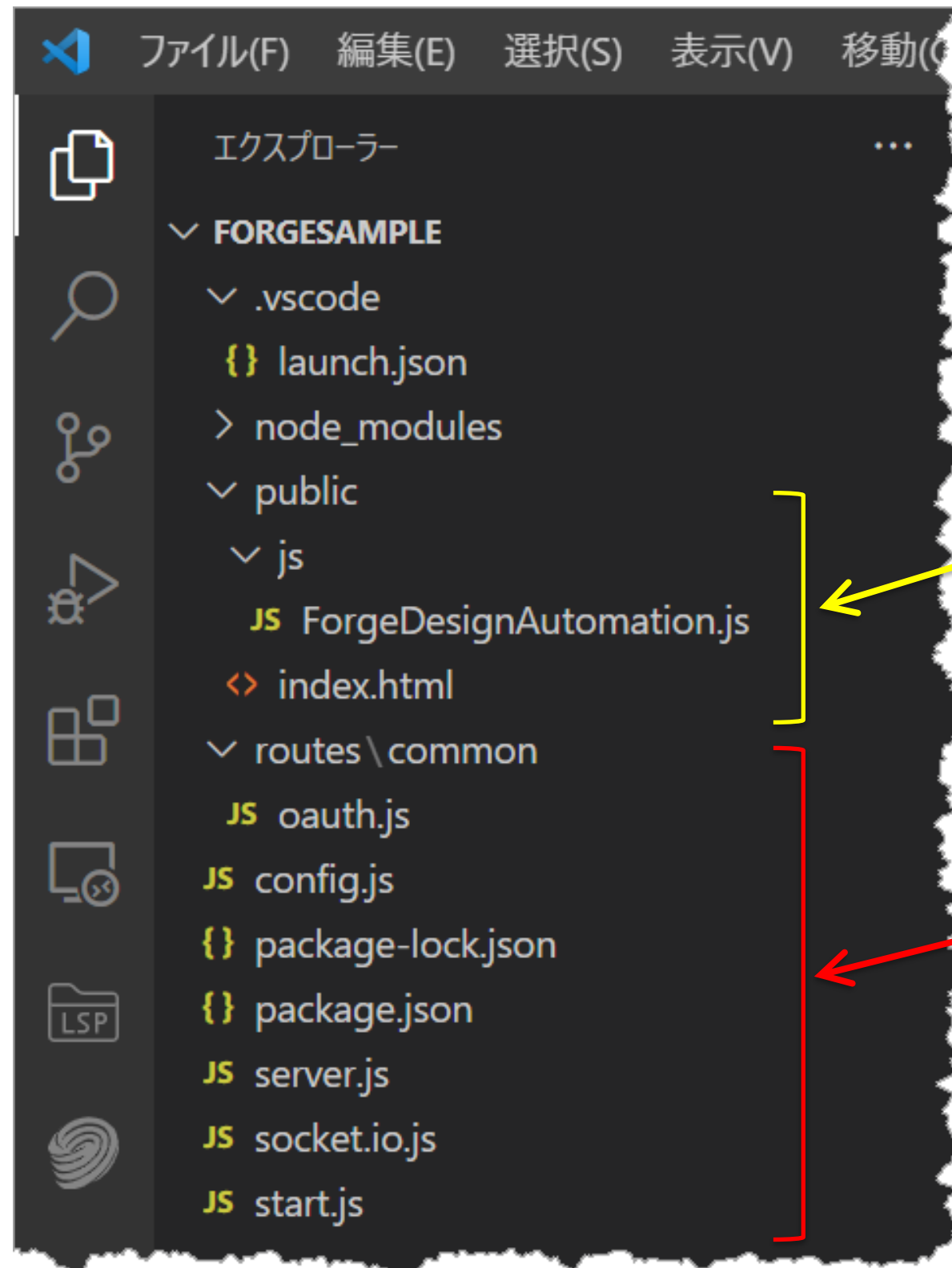
<button> ~ </button>

Create Update

<button> ~ </button>

Start workitem

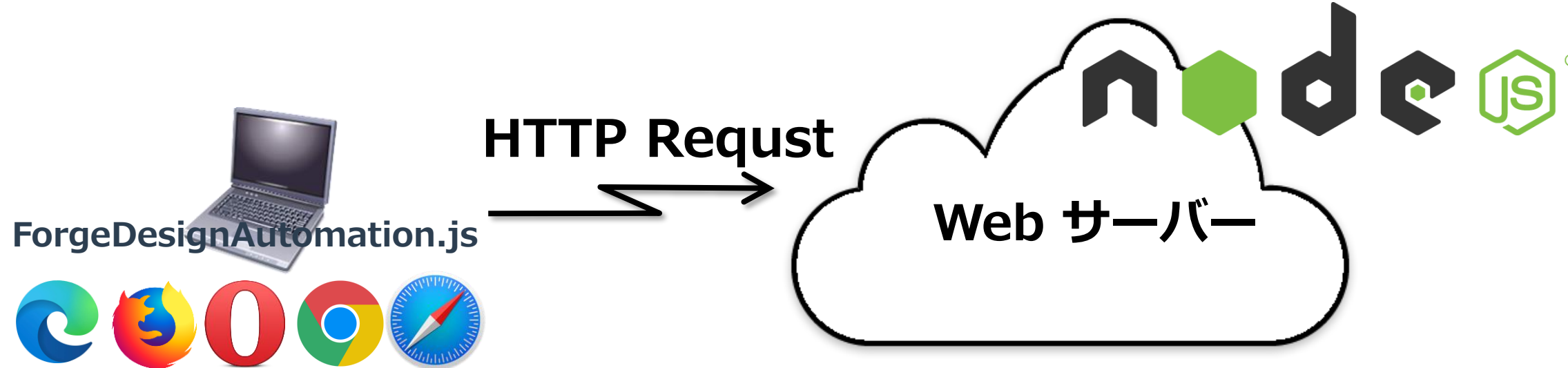
ここまでのプロジェクト構成



ローカルの Web ブラウザ側で
表示、または、実行される内容

Web サーバー側で
実行される内容

クライアント側 JavaScript ~Node.js サーバへのリクエスト

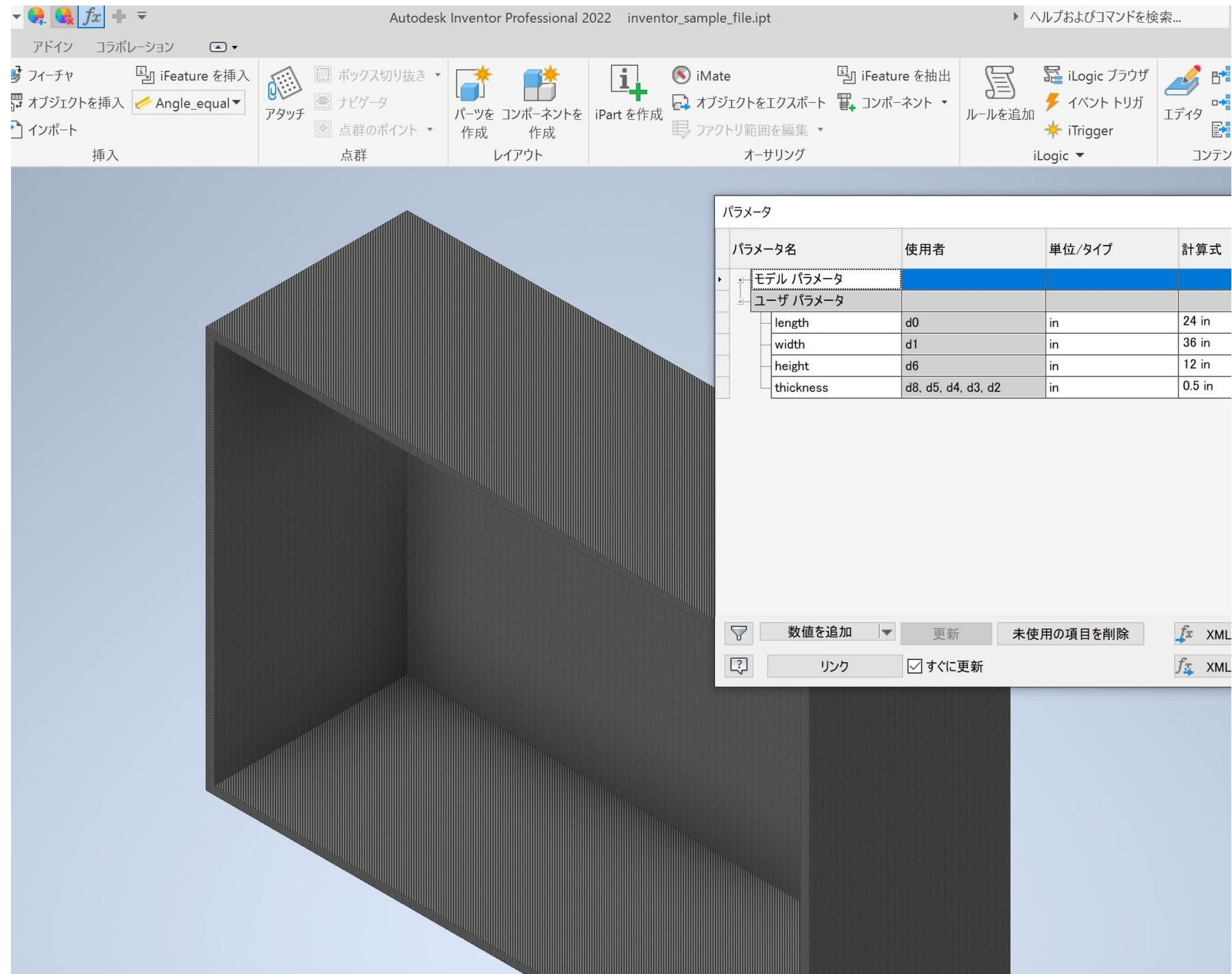


- 各種リストの取得 ~Onload時
 - Activityのリスト : '/api/forge/designautomation/activities'
 - DAエンジンのリスト : '/api/forge/designautomation/engines'
 - ローカルAppBundleのリスト : '/api/appbundles'
- アクション ~ボタンクリック時
 - アカウントのクリア : 'api/forge/designautomation/account'
 - AppBundle作成 : 'api/forge/designautomation/appbundles'
 - Activityの作成 : 'api/forge/designautomation/activities'
 - WorkItemの実行 : 'api/forge/designautomation/workitems'



プラグインを準備する

UpdateIPTParam Plug-In



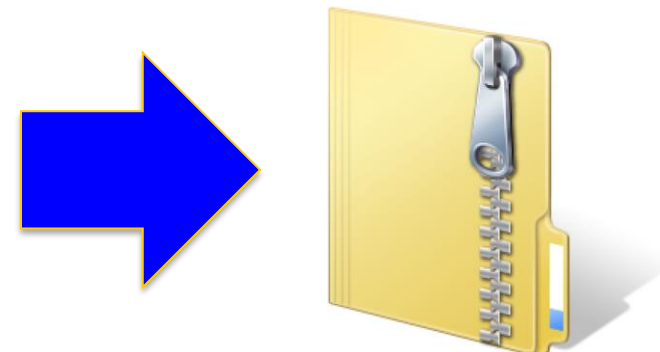
widthとheightパラメータを
指定した値に変更するプラグイン

Inventor プラグインをAppBundleにパッケージ化

- カスタムコマンドを実行するバイナリファイルとサポートファイルをパッケージ化
- AppBundleにアップロードして使用

¥UpdateIPTParam.bundle
+ PackageContents.xml
¥Contents
+ UpdateIPTParam.dll
+ UpdateIPTParam.Inventor.addin

Platform="Inventor"
ModuleName="./Contents/ UpdateIPTParam.Inventor.addin"



App Bundleの作成の手順

1. *.bundleフォルダの作成
2. PackageContents.xmlの作成
3. *.bundle¥contentsフォルダを作成し、アセンブリと.addin
ファイルをコピー
4. .addinファイルを修正
5. Zipアーカイブを作成

1. *.bundleフォルダの作成

UpdateIPTParam			
	☐ 名前	更新日時	種類
   	 UpdateIPTParam.bundle	2022/09/14 10:40	ファイル フォルダー

2. PackageContents.xmlの作成

UpdateIPTParam > UpdateIPTParam.bundle

名前	更新日時	種類
PackageContents.xml	2022/09/14 10:40	XML ドキュメント

```
<?xml version="1.0" encoding="utf-8" ?>
<ApplicationPackage SchemaVersion="1.0" Version="1.0" ProductCode="{c803510f-2486-4dd7-b477-a236127d25ed}"
Name="UpdateIPTParam" Description="UpdateIPTParamPlugin Plugin" Author="Design Automation for Inventor">
  <CompanyDetails Name="Autodesk, Inc" Phone="415.555.5555" Url="learnforge.autodesk.io" Email="forge.help@autodesk.com" />
  <Components>
    <!-- For Inventor Engine, "Platform" attribute must be "Inventor" -->
    <RuntimeRequirements OS="Win64" Platform="Inventor" />
    <!-- For Inventor Plug-in, the "Module" attribute must point to the .addin manifest file. -->
    <ComponentEntry LoadOnAutoCADStartup="False" LoadOnCommandInvocation="False"
      AppDescription="UpdateIPTParam App Package. "
      ModuleName="./Contents/UpdateIPTParam.Inventor.addin" AppName="UpdateIPTParam"/>
  </Components>
  <EnvironmentVariables>
  </EnvironmentVariables>
</ApplicationPackage>
```

3. *.bundle¥contentsフォルダを作成し、アセンブリと.addinファイルをコピー

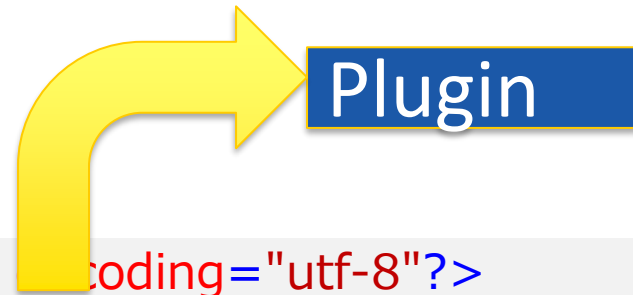
“Contents” フォルダを作成

UpdateIPTParam > UpdateIPTParam.bundle				
☐ 名前	更新日時	種類	サイズ	
☒ Contents	2022/09/14 10:40	ファイル フォルダー		
☐ PackageContents.xml	2022/09/14 10:40	XML ドキュメント	1 KB	

アセンブリと.addinファイルをコピー

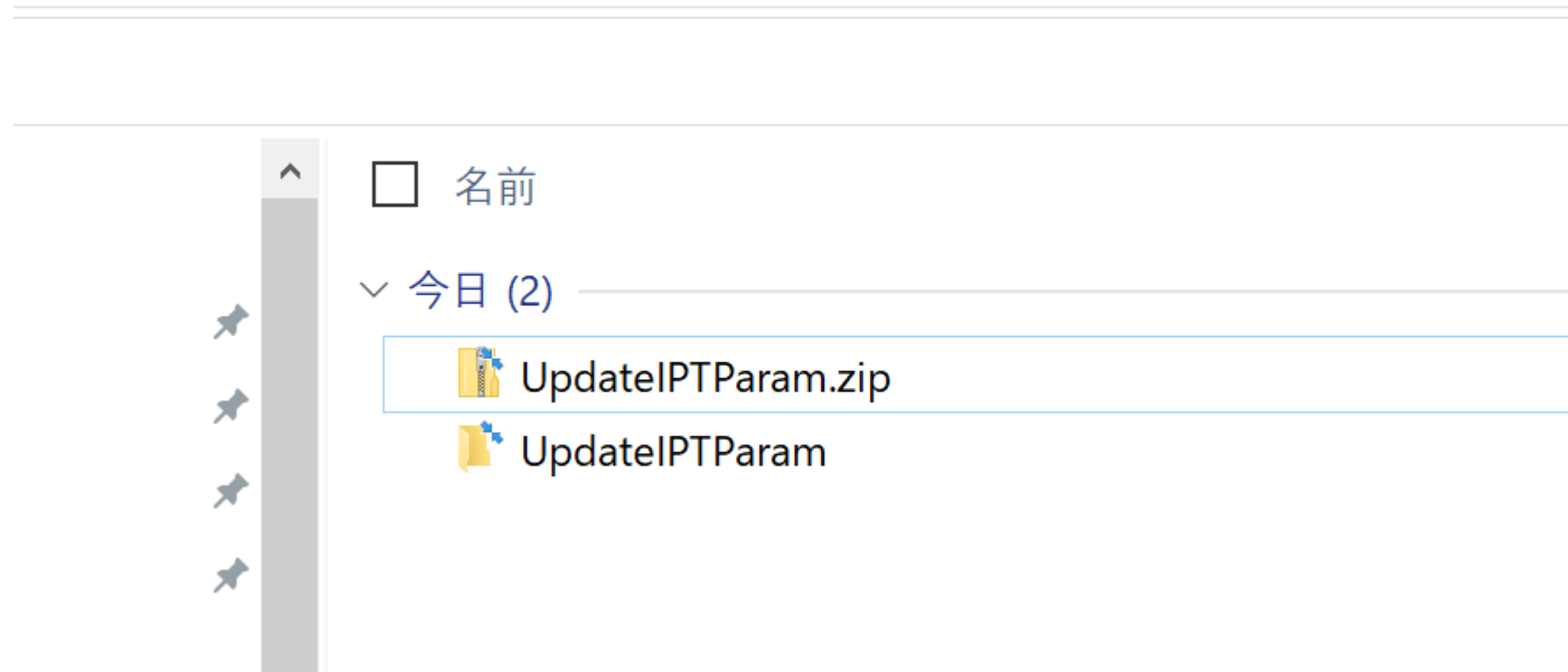
UpdateIPTParam > UpdateIPTParam.bundle > Contents				
☐ 名前	更新日時	種類	サイズ	
☒ UpdateIPTParam.Inventor.addin	2022/09/14 10:40	Visual Studio Add-in ...	1 KB	
☒ UpdateIPTParam.dll	2022/09/14 10:40	アプリケーション拡張	84 KB	
☐ System.Xml.XPath.XDocument.dll	2022/09/14 10:40	アプリケーション拡張	23 KB	
☐ System.Xml.XPath.dll	2022/09/14 10:40	アプリケーション拡張	21 KB	
☐ System.Xml.XmlSerializer.dll	2022/09/14 10:40	アプリケーション拡張	22 KB	
☐ System.Xml.XmlDocument.dll	2022/09/14 10:40	アプリケーション拡張	22 KB	
☐ System.Xml.XDocument.dll	2022/09/14 10:40	アプリケーション拡張	22 KB	
☐ System.Xml.ReaderWriter.dll	2022/09/14 10:40	アプリケーション拡張	22 KB	
☐ System.ValueTuple.dll	2022/09/14 10:40	アプリケーション拡張	24 KB	
☐ System.Threading.Timer.dll	2022/09/14 10:40	アプリケーション拡張	21 KB	
☐ System.Threading.ThreadPool.dll	2022/09/14 10:40	アプリケーション拡張	21 KB	

4. .addinファイルを修正



```
<?xml version="1.0" encoding="utf-8"?>
  <!-- Type attribute is same as Type registry key (Standard, Translator, Plugin (Server only) -->
  <Addin Type="Standard">
    <ClassId>{c803510f-2486-4dd7-b477-a236127d25ed}</ClassId>
    <ClientId>{c803510f-2486-4dd7-b477-a236127d25ed}</ClientId>
    <!-- Both of the following fields should be translated. NO OTHER FIELDS SHOULD BE TRANSLATED! -->
    <DisplayName>UpdateIPTParam</DisplayName>
    <Description>UpdateIPTParam</Description>
    <!-- Assumes that SimpleAddIn.dll is underneath Inventor¥bin -->
    <Assembly>UpdateIPTParam.dll</Assembly>
    <SupportedSoftwareVersionGreaterThan>17.</SupportedSoftwareVersionGreaterThan>
    <LoadOnStartUp>1</LoadOnStartUp>
    <Hidden>0</Hidden>
  </Addin>
```

5. Zipアーカイブを作成



Inventor アドインとプラグイン

*Inventor*アドイン

- Inventor SDK .NET Wizard でプロジェクト作成
- Inventor エディタにロードして、GUIからカスタム処理を実行
- コマンドを作成し、GUI（リボンボタン等）のイベントハンドラに登録

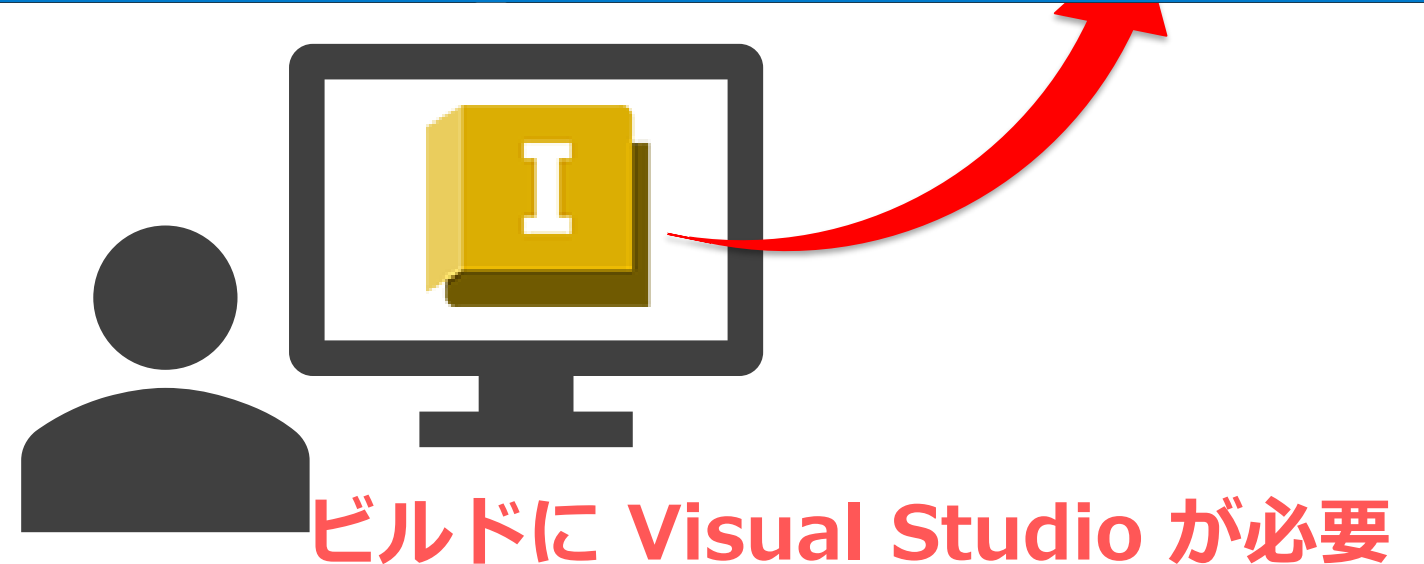
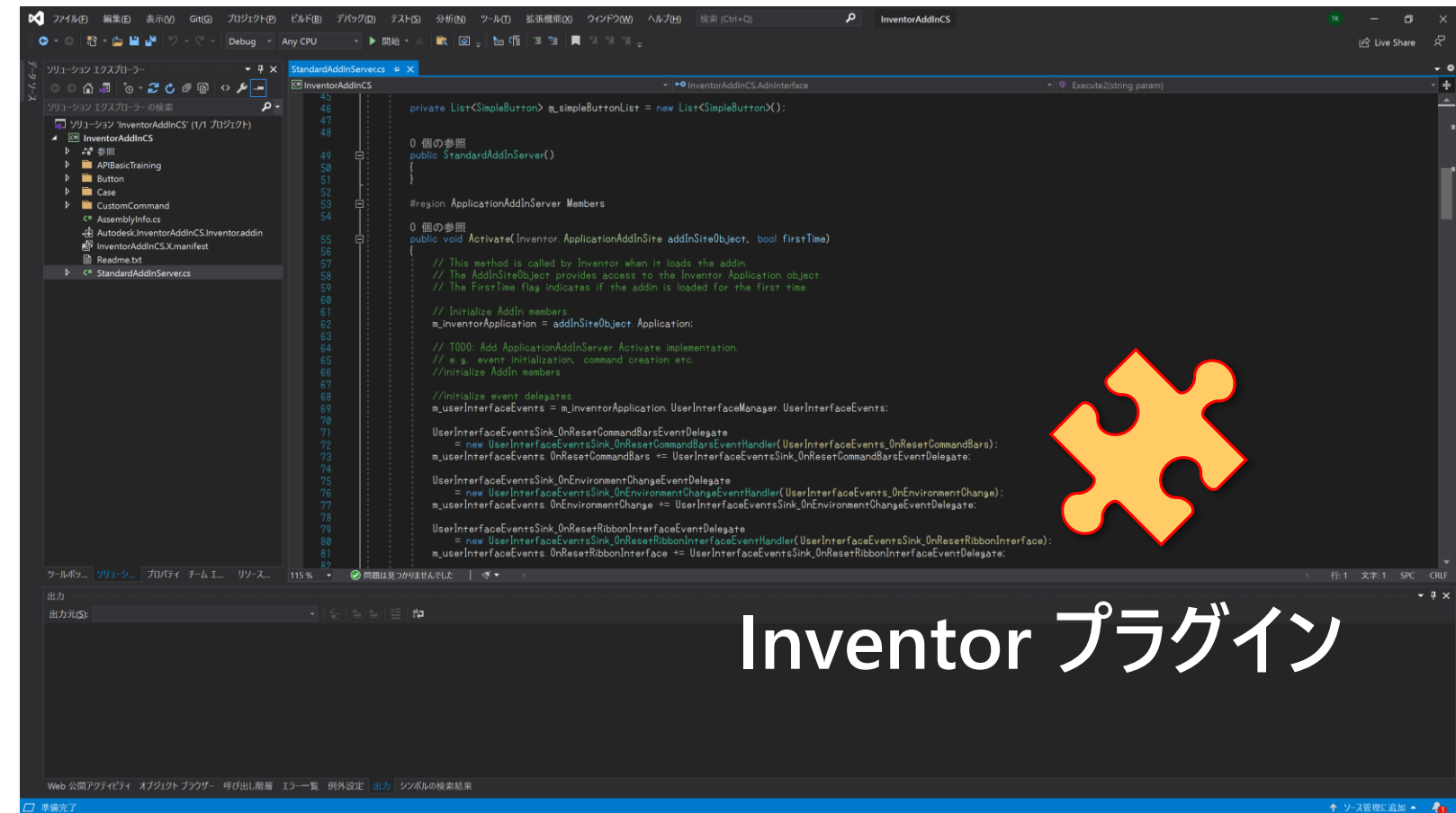
*Inventor*プラグイン

- Design Automation for Inventor テンプレートでプロジェクト作成
- Inventorコアエンジンにロードして、GUIなしでカスタム処理を実行
- Run()または、RunWithArguments()メソッドに処理を記述。処理はコアエンジンから起動される

- Visual Studio (Visual Studio 2019 professional 等) を使用
- Inventor SDKのAPIを用いてInventor機能を実行

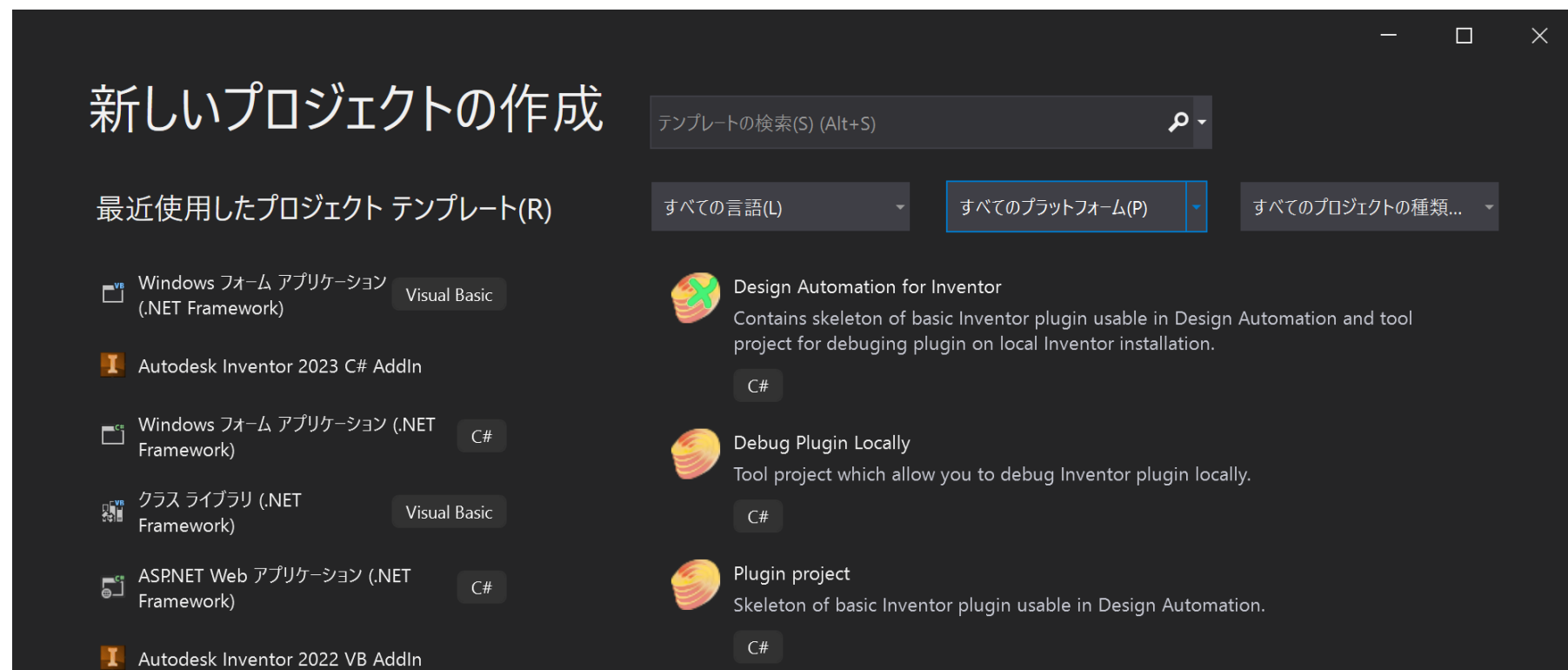
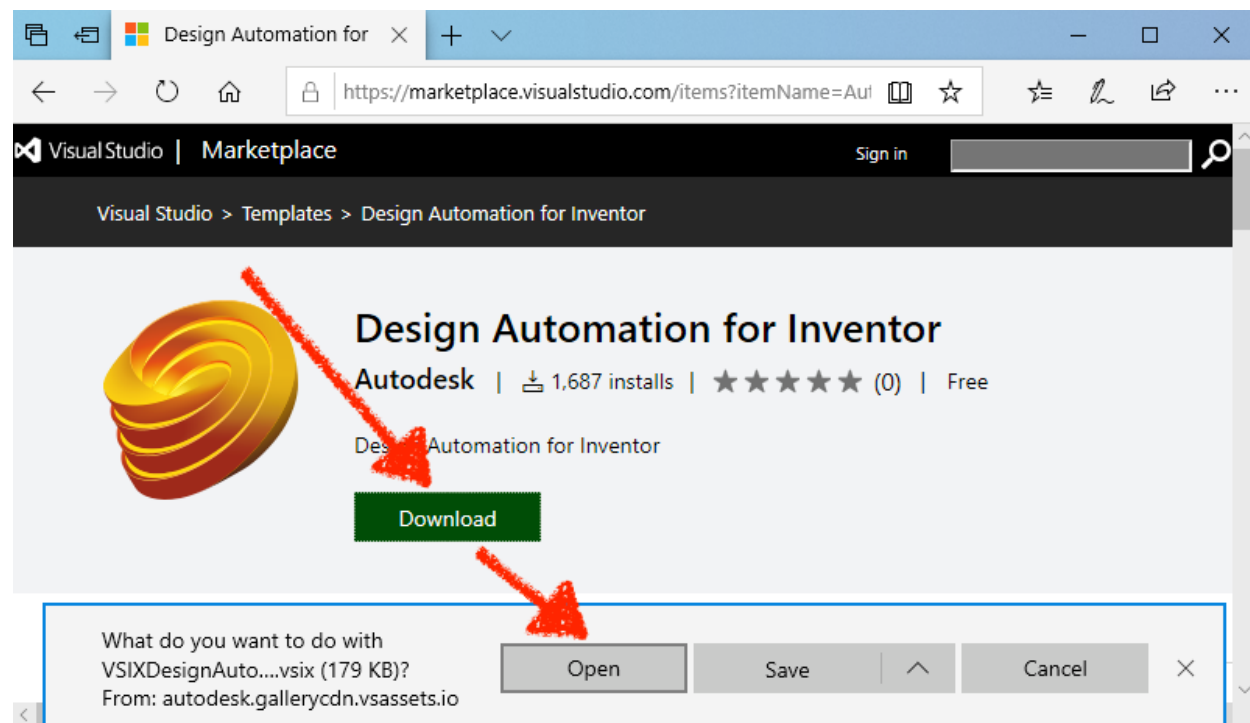
Design Automation for Inventor プラグインの開発環境

- Inventor SDK
 - C#、VB.NET
- .NET Framework
 - .NET Framework 4.6~4.8
- 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
- 参照ライブラリ
 - autodesk.inventor.interop.dll
 - Nuget パッケージ
 - Autodesk.Forge.DesignAutomation.Inventor.Utils
 - Newtonsoft.Json (JSONファイルのR/W)



Inventor プラグインの開発

- Inventor SDKを用いたカスタマイズ
- Visual Studioを使用して、.netアセンブリを作成
- Design Automation for Inventor テンプレートでプロジェクト作成



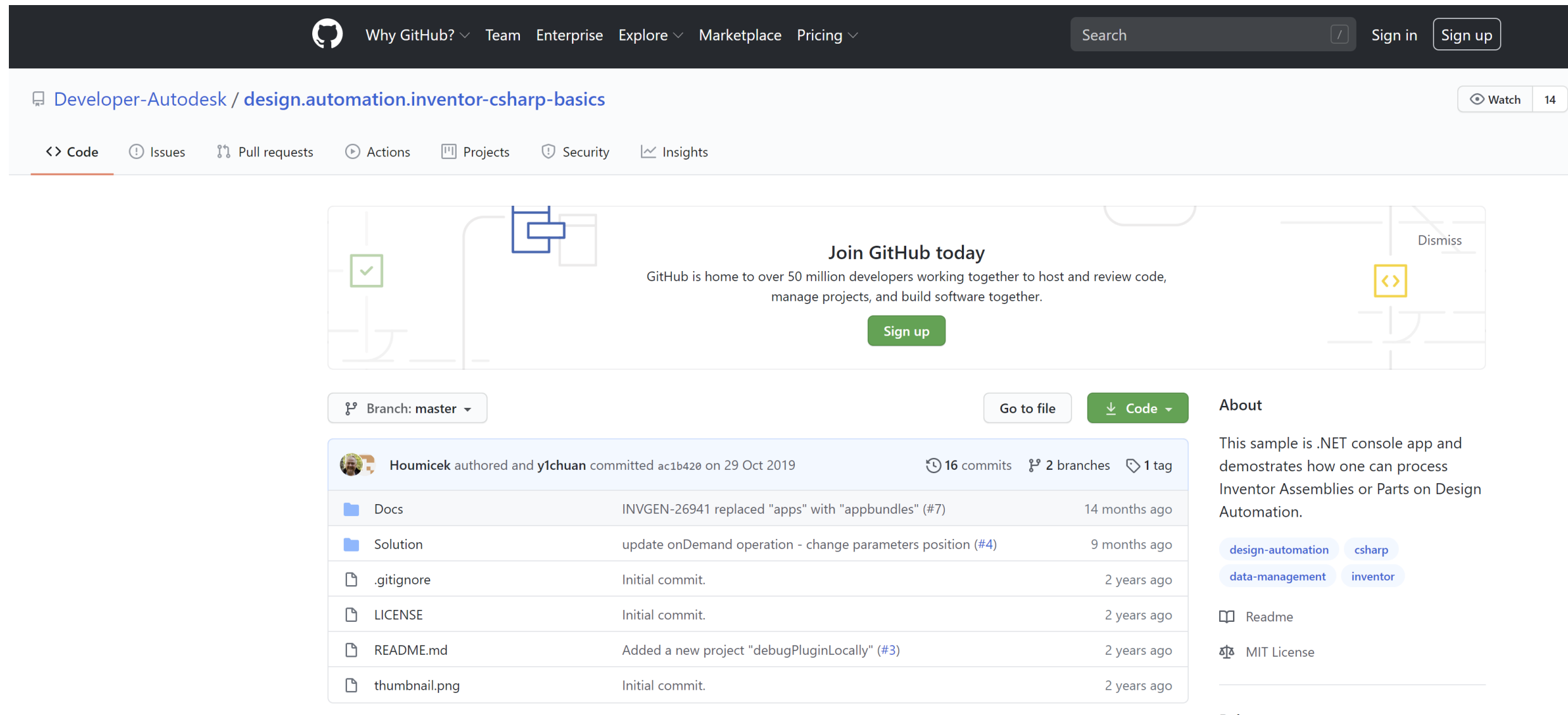
Inventor アドインのプラグイン化のポイント

1. GUIに依存した処理を除去
2. *Inventor.Application* の参照を*InventorServer*に変更
3. *InventorServer*を引数に取るコンストラクタの実装
4. *Run*、*RunWithArguments*メソッドに自動化処理を実装
5. COM Automationのサポート

プラグインのローカルでのデバッグ用プロジェクト

<https://github.com/Developer-Autodesk/design.automation.inventor-csharp-basics.git>

■ DebugPluginLocallyプロジェクト



Developer-Autodesk / [design.automation.inventor-csharp-basics](#) Watch 14

<> Code Issues Pull requests Actions Projects Security Insights

Join GitHub today
GitHub is home to over 50 million developers working together to host and review code, manage projects, and build software together.
[Sign up](#)

Branch: master Go to file Code

Houmicek authored and y1chuan committed ac1b420 on 29 Oct 2019 16 commits 2 branches 1 tag

Docs	INVGEN-26941 replaced "apps" with "appbundles" (#7)	14 months ago
Solution	update onDemand operation - change parameters position (#4)	9 months ago
.gitignore	Initial commit.	2 years ago
LICENSE	Initial commit.	2 years ago
README.md	Added a new project "debugPluginLocally" (#3)	2 years ago
thumbnail.png	Initial commit.	2 years ago

About
This sample is .NET console app and demonstrates how one can process Inventor Assemblies or Parts on Design Automation.
[design-automation](#) [csharp](#)
[data-management](#) [inventor](#)

Readme MIT License

Releases

Inventor APIの学習 ～Inventor 2023 API トレーニング マテリアル

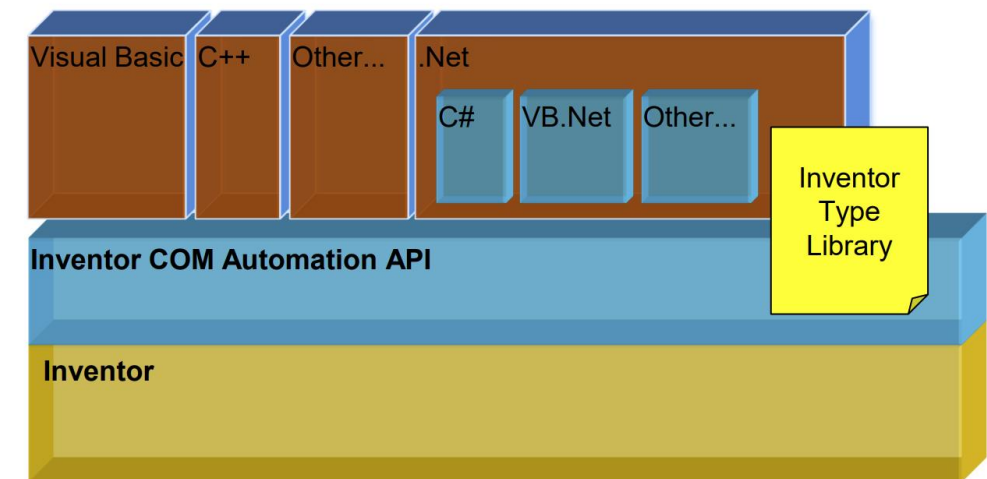
AUTODESK

Inventor 2023 API 基礎ト レーニング

Inventor APIの概要

COM API Model

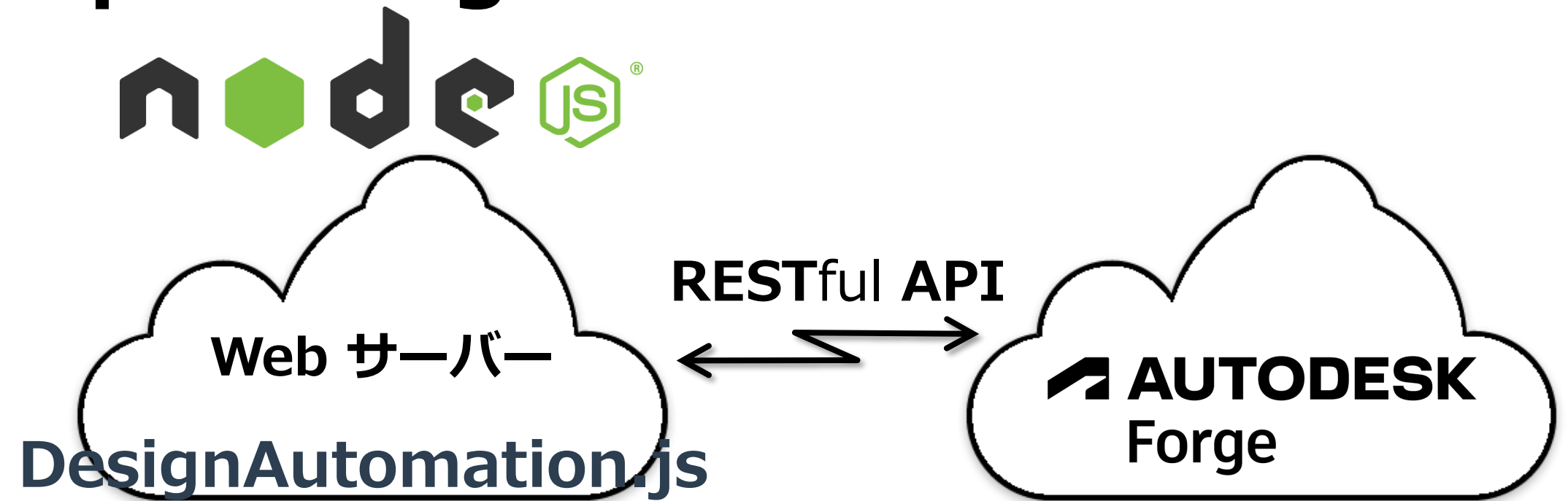
InventorのAPIは、COM (Component Object Model) Automation interfaceで公開される。





**AppBundle をアップロードする
Activity を定義する**

サーバ側 JavaScript ~Forgeエンドポイントの実行



■ 各種リストの取得 ~Onload時

- Activityのリスト : '/api/forge/designautomation/activities'
⇒ AutodeskForgeDesignAutomationClient.getActivities():GET '/v3/activities'
- DAエンジンのリスト : '/api/forge/designautomation/engines'
⇒ AutodeskForgeDesignAutomationClient.getEngines(): GET '/v3/engines'

DA4Iで利用可能なコアエンジン バージョン

- デスクトップ製品と同期したコアエンジンバージョン
 - AppPackageはエンジンバージョンに合わせた作成が必須
 - コアエンジンIDの形式はコアエンジン毎に異なる
 - 2022年10月6日現在（Inventor 2018 ～2023が対象）

"Autodesk.Inventor+22" ⇒ Inventor 2018

"Autodesk.Inventor+23" ⇒ Inventor 2019

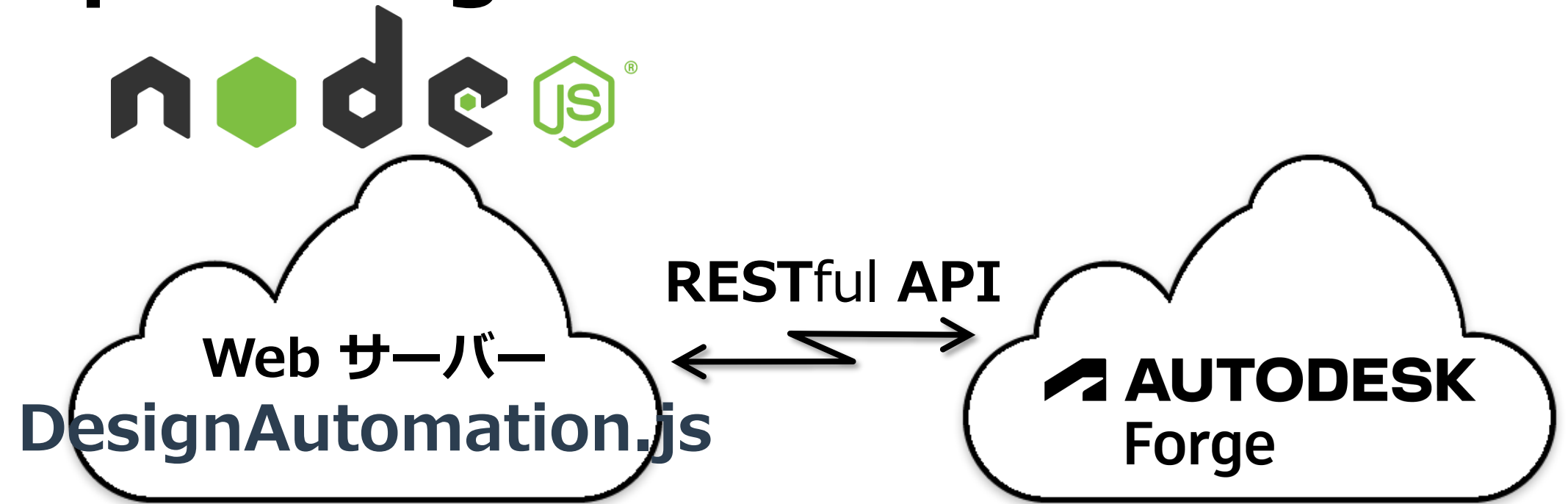
"Autodesk.Inventor+24" ⇒ Inventor 2020

"Autodesk.Inventor+2021" ⇒ Inventor 2021

"Autodesk.Inventor+2022" ⇒ Inventor 2022

"Autodesk.Inventor+2023" ⇒ Inventor 2023

サーバ側 JavaScript ~Forgeエンドポイントの実行



■ アクション ~ボタンクリック時

- Appbundle作成 : 'api/forge/designautomation/appbundles'
⇒ AutodeskForgeDesignAutomationClient.createAppBundle(): POST '/v3/appbundles'
- Activityの作成 : 'api/forge/designautomation/activities'
⇒ AutodeskForgeDesignAutomationClient.createActivity(): POST '/v3/activities'

Inventorコアエンジン

InventorCoreConsole.exe

- ✓ Design Automation for Inventor用のコアエンジン
- ✓ クライアント版Inventorには同梱されていない
- ✓ 起動引数で指定されたAppBundleをロードし、AppBundle内のメソッド（Run, RunWithArgumentsのどちらか）を実行
 - Run(Document doc) : 所定起動引数のみ場合
 - RunWithArguments(Document doc, NameValueMap map) : 所定引数以外を指定した場合。所定以外の引数はNameValueMapにキー“-1”, “-2”...で順に格納される
- ✓ InventorCoreConsole.exeの所定起動引数
 - /a : ロードするAppBundleのパス
 - /i : ロードするドキュメントのパス
 - /s : 実行するスクリプトファイルへのパス

Activityでコアエンジンの起動パラメータを指定

```

const qualifiedActivityId = `${Utils.NickName}.${activityName}+${Utils.Alias}`;
if (!activities.data.includes(qualifiedActivityId)) {
  // define the activity
  // ToDo: parametrize for different engines...
  const engineAttributes = Utils.EngineAttributes(engineName);
  const commandLine = engineAttributes.commandLine.replace('{0}', appBundleName);
  const activitySpec = {
    id: activityName,
    appbundles: [`${Utils.NickName}.${appBundleName}+${Utils.Alias}`],
    commandLine: [commandLine],
    engine: engineName,
    parameters: {
      inputFile: {
        description: 'input file',
        localName: `${inputFile}`,
        ondemand: false,
        required: true,
        verb: dav3.Verb.get,
        zip: false
      },
      inputJson: {
        description: 'input json',
        localName: 'params.json',
        ondemand: false,
        required: false,
        verb: dav3.Verb.get,
        zip: false
      }
    }
  };

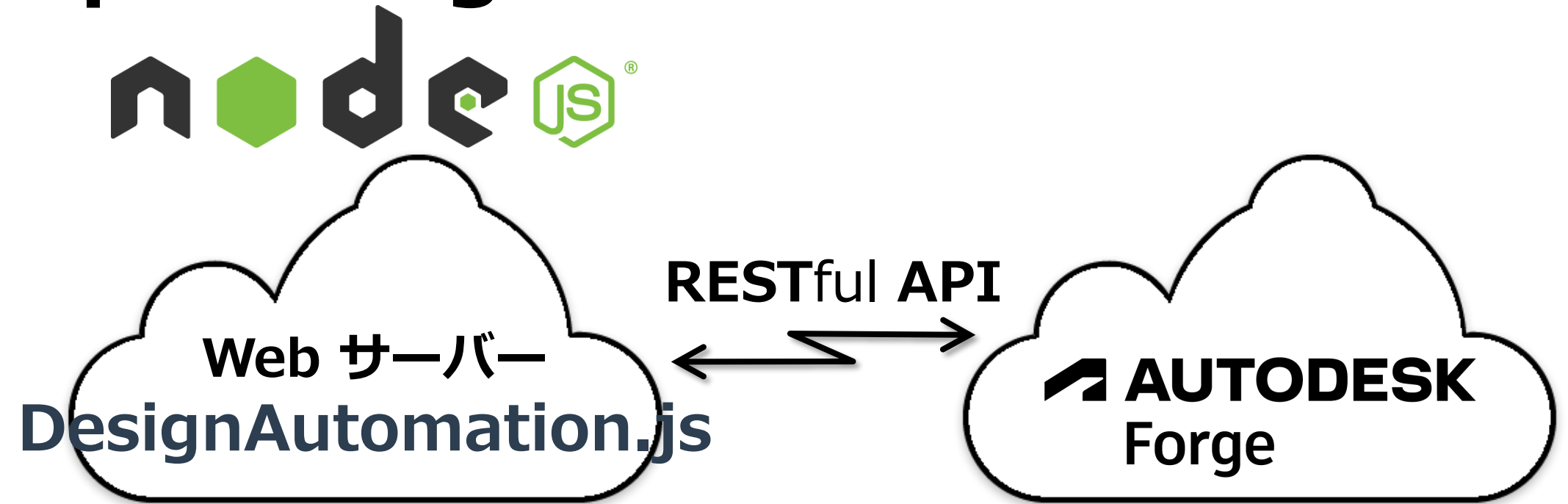
  /// <summary>
  /// Helps identify the engine
  /// </summary>
  static EngineAttributes (engine) {
    if (engine.includes('3dsMax'))
      return ({
        commandLine: `${engine.path}\\3dsmaxbatch.exe -sceneFile "${args[inputFile].path}" "${settings[script].path}"`,
        extension: 'max',
        script: "da = dotNetClass('Autodesk.Forge.Sample.DesignAutomation.Max.RuntimeExecute')\nda.ModifyWindowWidthHeight()\n"
      });
    if (engine.includes('AutoCAD'))
      return ({
        commandLine: `${engine.path}\\accoreconsole.exe /i "${args[inputFile].path}" /al "${appbundles[{0}].path}" /s "${settings[script].path}"`,
        extension: 'dwg',
        script: "UpdateParam\n"
      });
    if (engine.includes('Inventor'))
      return ({
        commandLine: `${engine.path}\\InventorCoreConsole.exe /i "${args[inputFile].path}" /al "${appbundles[{0}].path}"`,
        extension: 'ipt',
        script: ''
      });
    if (engine.includes('Revit'))
      return ({

```



WorkItem を実行する

サーバ側 JavaScript ~Forgeエンドポイントの実行



■ アクション ~ボタンクリック時

- WorkItemの実行 : 'api/forge/designautomation/workitems'
⇒ AutodeskForgeDesignAutomationClient.createWorkItem(): POST '/v3/workitems'

WorkItem 実行時に起こること

- AppBundle (アドイン) は実行時に作成の作業フォルダに展開
- DA API は WorkItem で指定された値を利用
 - 指定のクラウド ストレージから ipt をダウンロード (任意)
 - **指定の値を持つ JSON ファイルを作成**

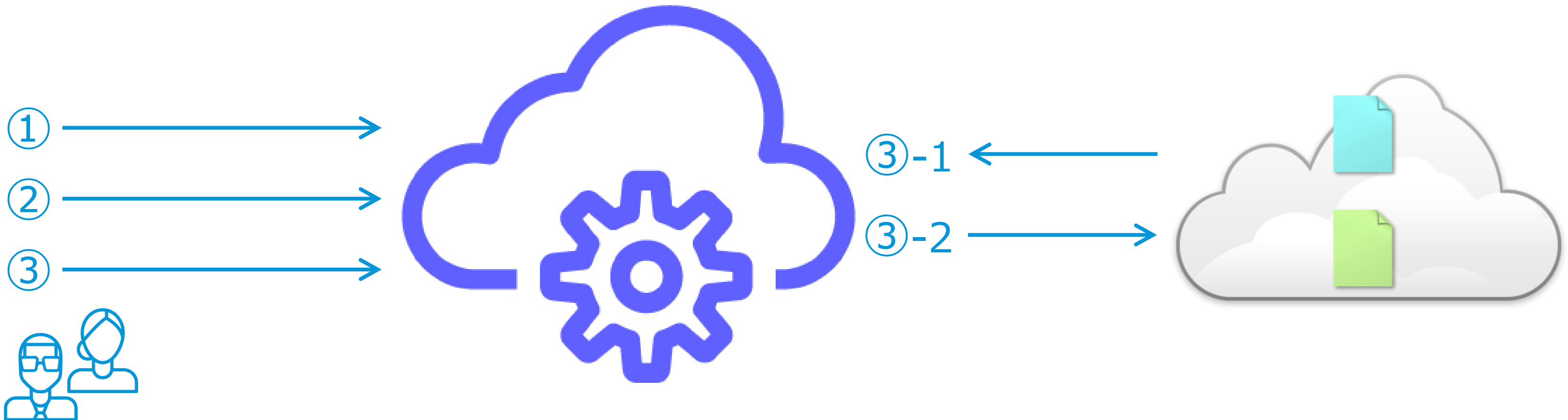
```
{  
  'width':999,  
  'height':555  
}
```



- 成果ファイルを指定のクラウド ストレージへ保存 (アップロード)
 - ipt や PDF など
- WorkItem 終了後には AppBundle、使用ファイルは削除
 - キャッシュして後日の WorkItem で使用することは不可

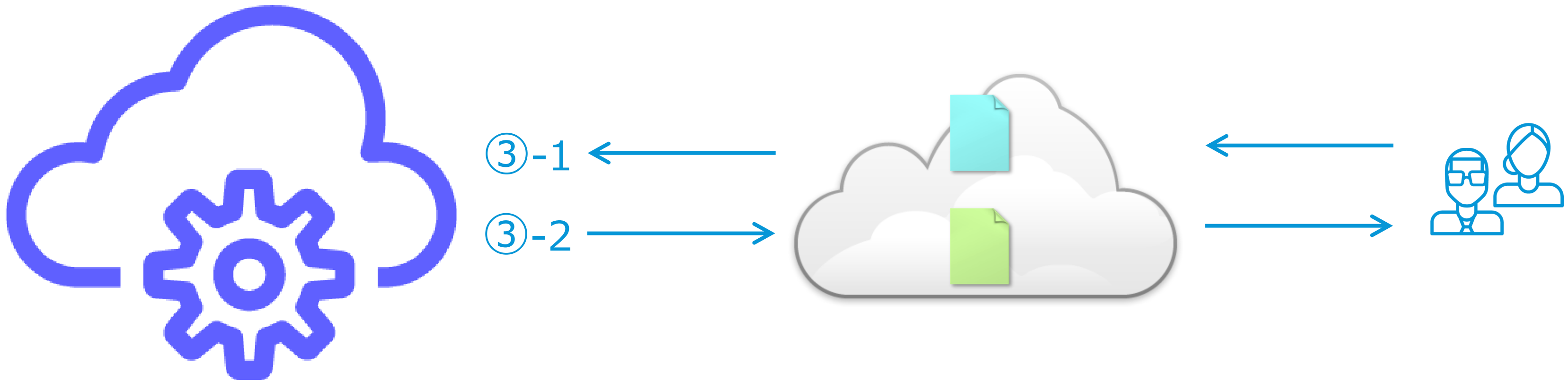
入力ファイルと出力ファイルの扱い

- セキュリティ上、次のようなファイル操作の自動化は不可
 - ローカル PC から DA API 作業領域に参照ファイルを保存
 - DA API 作業領域から成果ファイルをローカル PC に保存
- Web セキュリティ上の制限：ユーザ操作が必要



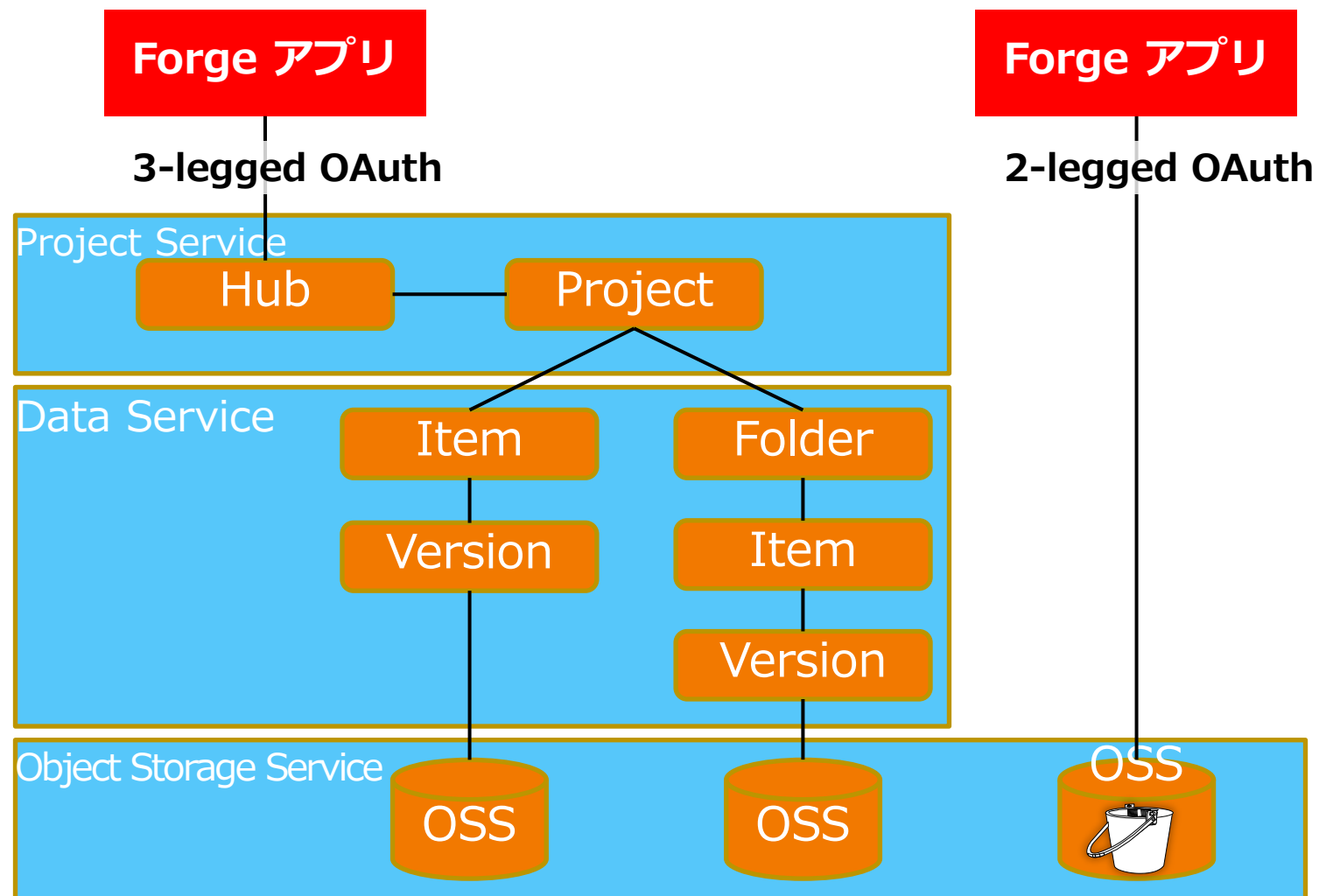
入出力ファイル指定は署名付き URL で

- ユーザにリソース アクセス権を有効期限付きで一時的に提供
 - クラウド ストレージから DA API 作業領域に参照ファイルを保存
 - DA API 作業領域から成果ファイルをクラウド ストレージに保存

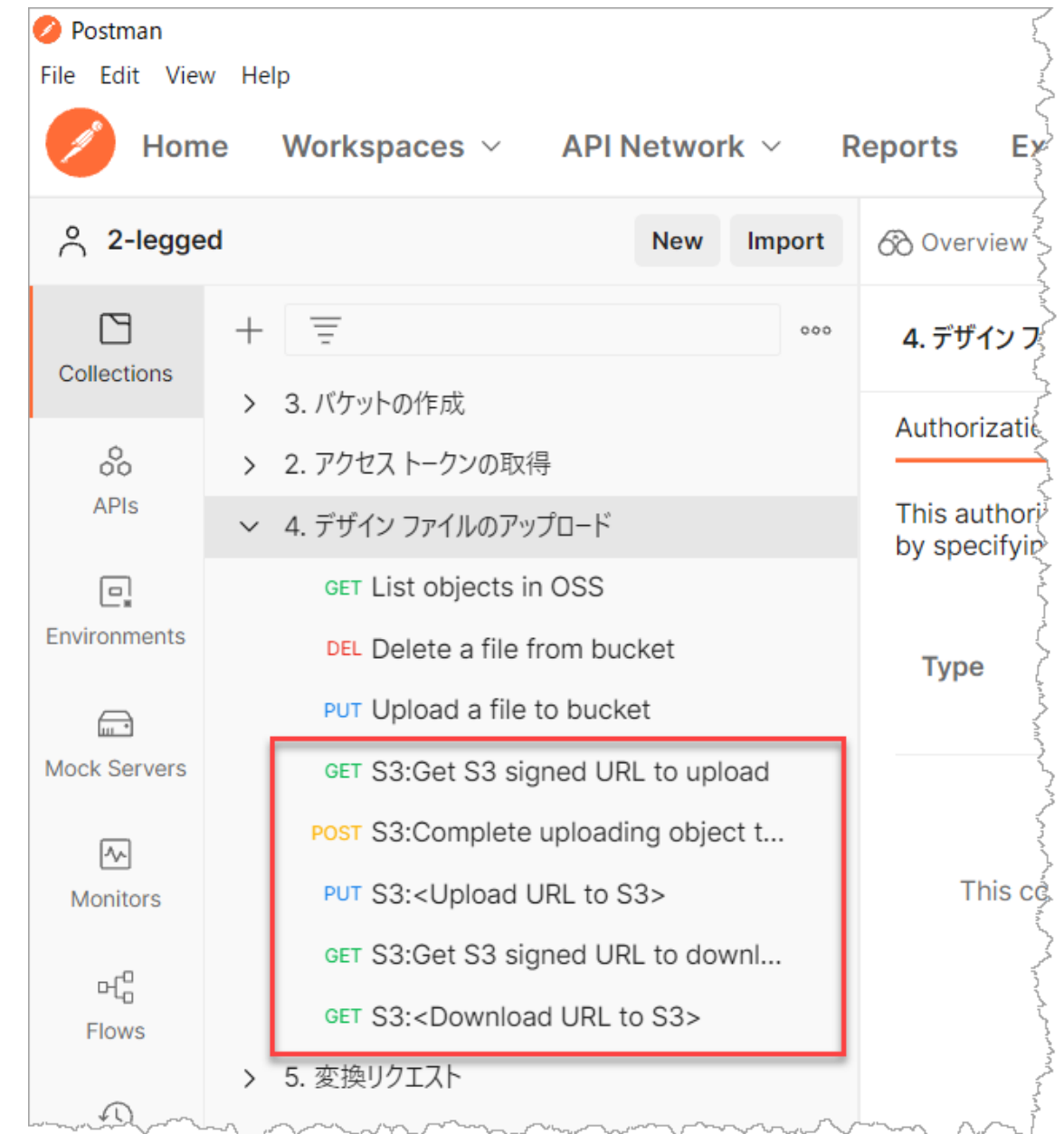


OSS の Direct-to-S3 アプローチへの移行

- アップロードとダウンロードで AWS S3 に直接アクセス

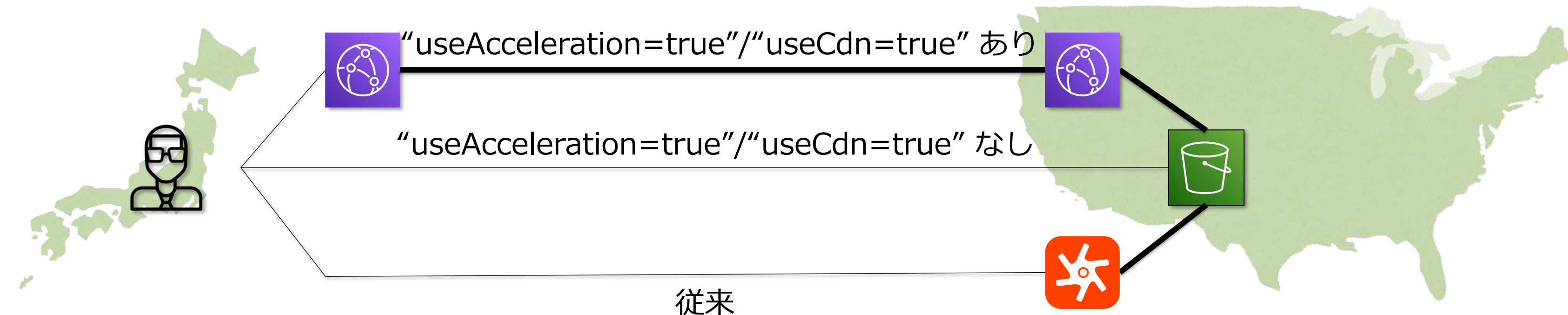


- DA チュートリアルが利用可能

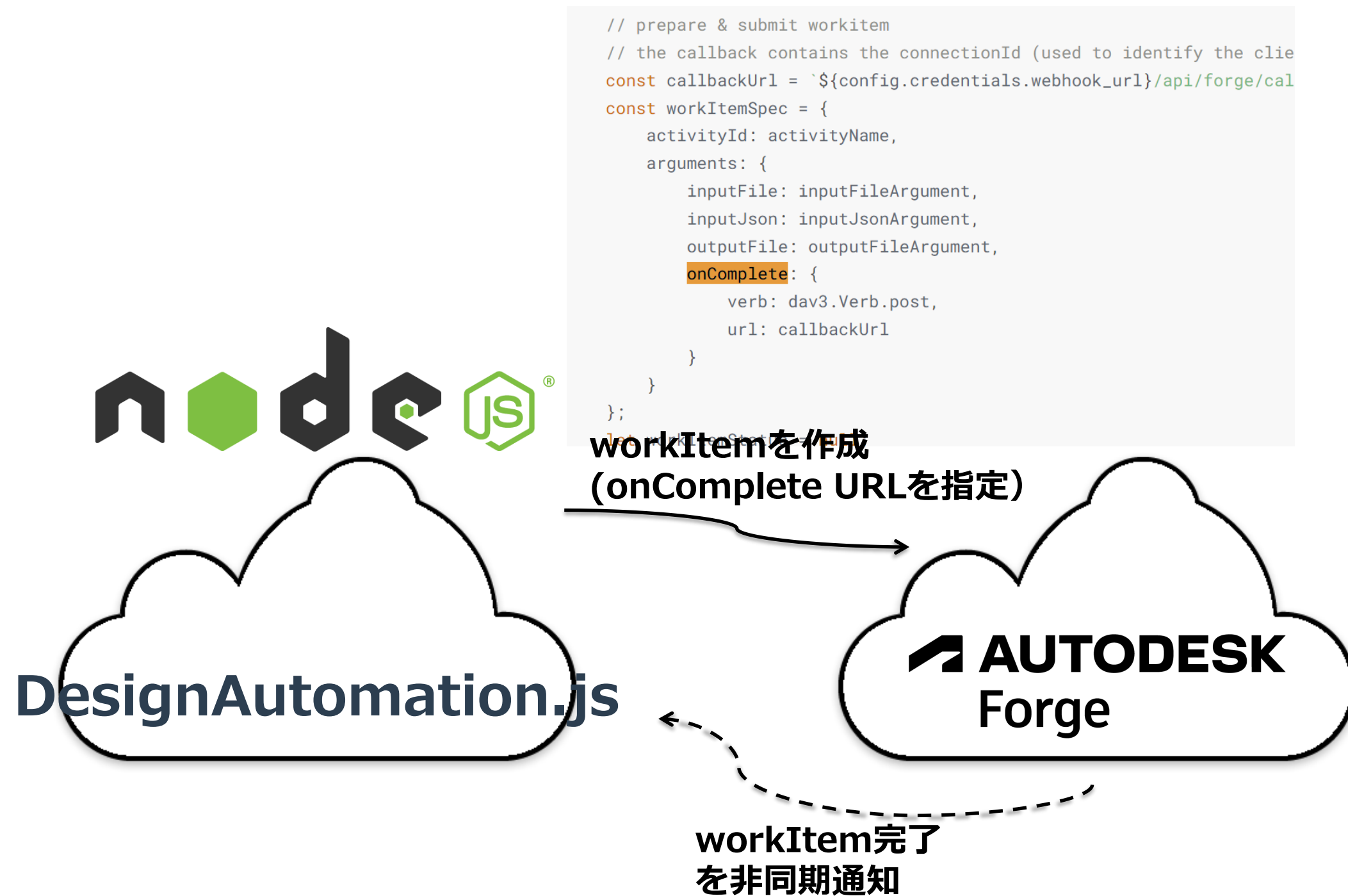


Direct-to-S3 アプローチの署名付き URL 要求

- 日本での利用時次のクエリーパラメータ指定を推奨
 - [GET buckets/:bucketKey/objects/:objectKey/signed3upload](#)
 - “useAcceleration=true”
 - [GET buckets/:bucketKey/objects/:objectKey/signed3download](#) :
 - “useCdn=true”



onCompleteハンドラへのWorkItemの完了通知



DNS 解決出来ない為、開発環境の
localhost:3000 を呼び出せない！



ローカル開発環境で通知を受けるために



- OnComplete コールバック URL はトンネル化が必要

- <https://ngrok.com/download>
- コマンド プロンプト上で **ngrok.exe http <ポート番号>**
- 例) **ngrok http 3000**

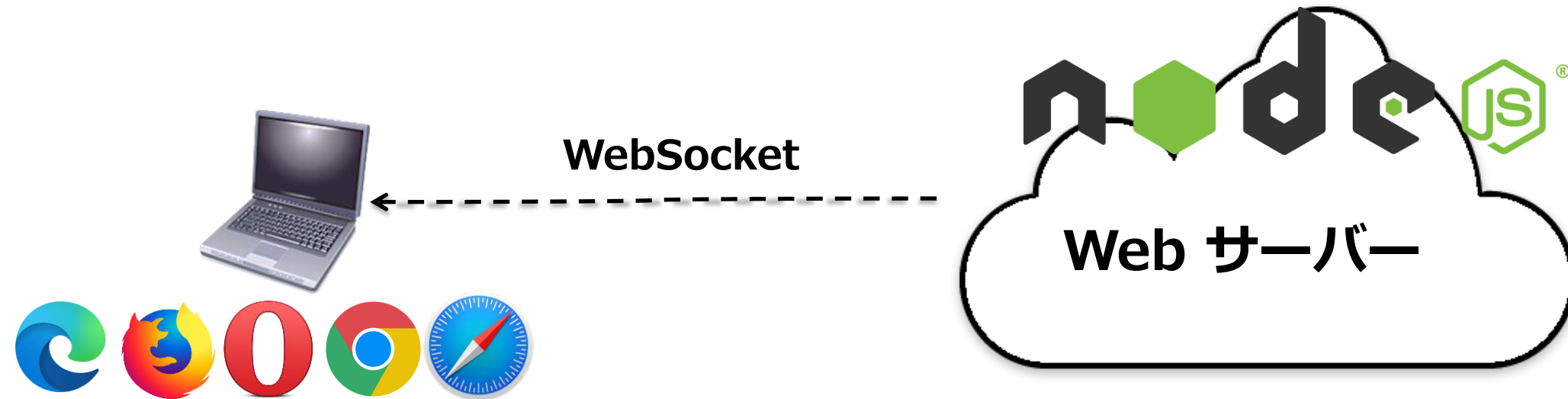
`http://localhost:3000 >> http://1e7e-115-162-81-167.ngrok.io`

```
選択コマンド プロンプト - ngrok http 3000 -host-header="localhost:3000"
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Session Expires     1 hour, 11 minutes
Update              update available (version 2.3.40, Ctrl-U to update)
Version              2.2.8
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://1e7e-115-162-81-167.ngrok.io -> localhost:3000
Forwarding           https://1e7e-115-162-81-167.ngrok.io -> localhost:3000

Connections          ttl      opn      rt1      rt5      p50      p90
                    1        0        0.00     0.00     5.18     5.18
```

WorkItemの実行結果をクライアントに通知



ForgeDesignAutomation.js

```
function startConnection(onReady) {
  if (connection && connection.connected) {
    if (onReady)
      onReady();
    return;
  }
  connection = io();
  connection.on('connect', function () {
    connectionId = connection.id;
    if (onReady)
      onReady();
  });

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

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

  connection.on('onComplete', function (message) {
    if (typeof message === 'object')
      message = JSON.stringify(message, null, 2);
    writeLog(message);
  });
}
```

Work Itemの完了通知

DA reportの内容

iptファイルのDownload URL

DesignAutomation.js

```
router.post('/forge/callback/designautomation', async /*OnCallback*/ (req, res) => {
  // your webhook should return immediately! we could use Hangfire to schedule a job instead
  // ALWAYS return ok (200)
  res.status(200).end();

  try {
    const socketIO = require('../server').io;

    // your webhook should return immediately! we can use Hangfire to schedule a job
    const bodyJson = req.body;
    socketIO.to(req.query.id).emit('onComplete', bodyJson);

    http.get(
      bodyJson.reportUrl,
      response => {
        //socketIO.to(req.query.id).emit('onComplete', response);
        response.setEncoding('utf8');
        let rawData = '';
        response.on('data', (chunk) => {
          rawData += chunk;
        });
        response.on('end', () => {
          socketIO.to(req.query.id).emit('onComplete', rawData);
        });
      }
    );
    //socketIO.to(req.query.id).emit('downloadReport', bodyJson.reportUrl);

    const objectsApi = new ForgeAPI.ObjectsApi();
    const bucketKey = Utils.NickName.toLowerCase() + '-designautomation';
    if (bodyJson.status === 'success') {
      try {
        //Complete the upload to S3
        await objectsApi.completeS3Upload(bucketKey, req.query.outputFileName,
          { uploadKey: req.query.uploadKey },
          { useAcceleration: false, minutesExpiration: 2 },
          req.oauth_client, req.oauth_token);
        //create a S3 presigned URL and send to client
        let response = await objectsApi.getS3DownloadURL(bucketKey, req.query.outputFileName,
          { useAcceleration: false, minutesExpiration: 15 },
          req.oauth_client, req.oauth_token);
        socketIO.to(req.query.id).emit('downloadResult', response.body.url);
      } catch (ex) {
        console.error(ex);
        socketIO.to(req.query.id).emit('onComplete', 'Failed to create presigned URL');
      }
    }
  }
});
```



Forge Design Automation でiLogicを実行する

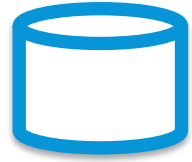
Inventorコアエンジンの/s オプション

InventorCoreConsole.exe

- ✓ Design Automation for Inventor用のコアエンジン
- ✓ クライアント版Inventorには同梱されていない
- ✓ 起動引数で指定されたAppBundleをロードし、AppBundleのメソッド（のどちらか）を実行
 - Run(Document doc) : 所定起動引数のみ場合
 - RunWithArguments(Document doc, NameValueMap map) : 所定引数以外を指定した場合。所定以外の引数はNameValueMapにキー“-1”, “-2”...で順に格納される
- ✓ InventorCoreConsole.exeの所定起動引数
 - /a : ロードするAppBundleのパス
 - /i : ロードするドキュメントのパス
 - /s : 実行するスクリプトファイルへのパス

iLogicファイルの指定方法

場所



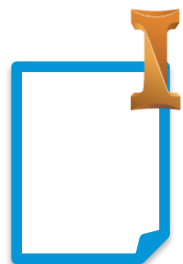
Cloud Storage



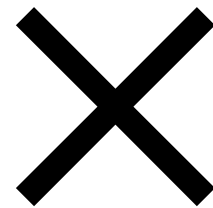
AppBundle

```
Parameter("height") = "2"  
in "Trace.TraceInformation"  
Updating document  
InventorVb.D ...
```

Parameter/
Argument



Inventor File



設定



Activity

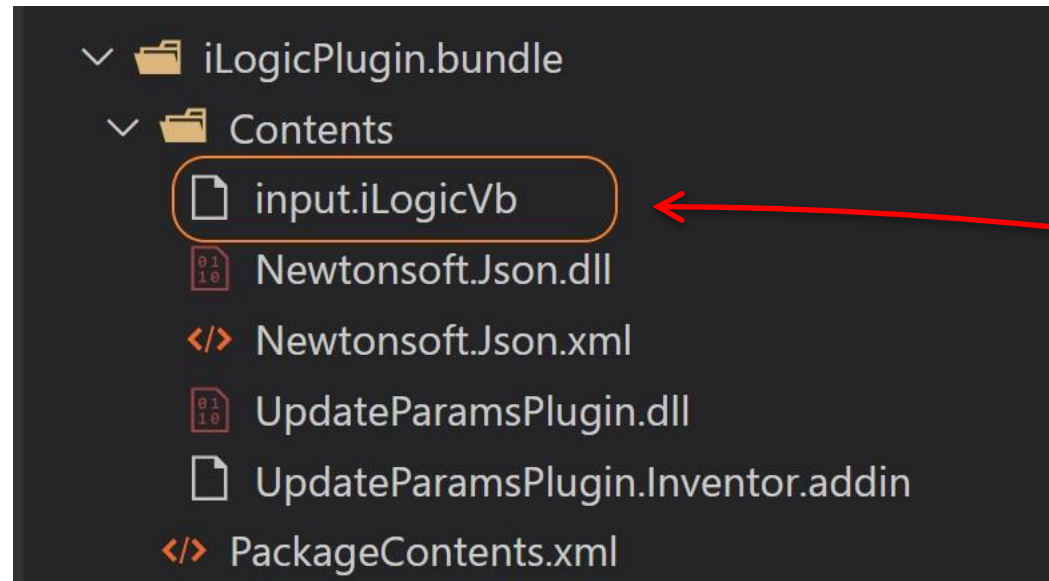


WorkItem

AppBundle内のiLogicの実行



AppBundle



Activity

```
{
  "id": "{ActivityId}",
  "commandLine": [
    "$(engine.path)¥¥InventorCoreConsole.exe /i ¥"$(args[InventorDoc].path)¥" /s ¥"$(appbundles[{AppBundle
Name}].path)¥¥iLogicPlugin.bundle¥¥Contents¥¥input.iLogicVb¥""
    ...
  ],
  "appbundles": [{AppBundleId + AppBundleAlias}],
  ...
}
```

Parameter/Argumentによる指定



Activity

```
{
  "id": "{ActivityId}",
  "commandLine": [
    "$(engine.path)¥¥InventorCoreConsole.exe /i ¥"$(args[InventorDoc].path)¥" /s  $(args[iLogicName].path)"
  ]
  ...
}
```



WorkItem

```
{
  "activityId": "{specify activity id + alias}",
  "arguments": {
    "InventorDoc": {
      "url": "{specify input file url}"
    },
    "iLogicName": {
      "value": "iLogicVb.RunRule(¥"MyRule¥")"
    },
    ...
  }
}
```

```
[01/27/2021 06:37:20] Starting work item 06d287cc9e7248259f08e6050499a75f
[01/27/2021 06:37:20] Start download phase.
[01/27/2021 06:37:20] Start downloading input: verb - 'GET', url -
'https://developer.api.autodesk.com/oss/v2/signedresources/f46ee514-9332-4e96-8c9a-53a55dd71fce?region=US'
[01/27/2021 06:37:20] '84480' bytes have been written to
T:¥Aces¥Jobs¥06d287cc9e7248259f08e6050499a75f¥Input.ipt.
[01/27/2021 06:37:20] End downloading file 'https://developer.api.autodesk.com/oss/v2/signedresources/f46ee514-
9332-4e96-8c9a-53a55dd71fce?region=US'.
[01/27/2021 06:37:20] End download phase successfully.
[01/27/2021 06:37:21] Start preparing script and command line parameters.
[01/27/2021 06:37:21] Command line: [ /i "T:¥Aces¥Jobs¥06d287cc9e7248259f08e6050499a75f¥Input.ipt" /s
"T:¥Aces¥Jobs¥06d287cc9e7248259f08e6050499a75f¥arg_iLogic.dat" ]
[01/27/2021 06:37:21] End preparing script and command line parameters.
```

WorkItem実行時に、指定したvalue値が記載された
ファイルとしてCoreエンジンに渡され、実行される

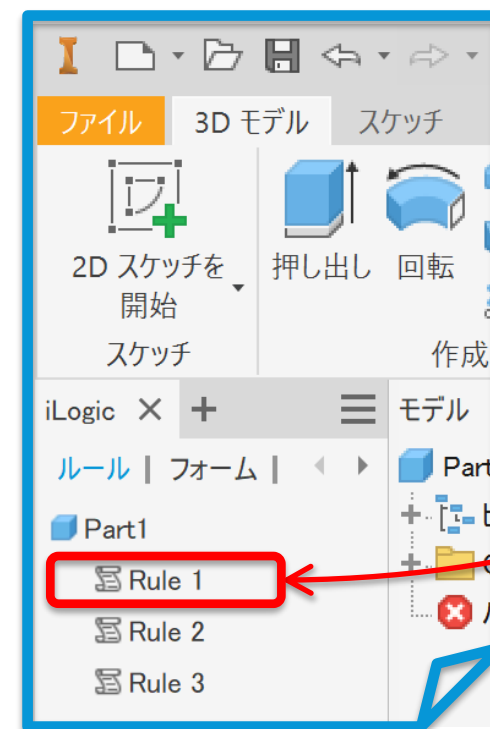
Inventorファイル内のiLogicの実行

- iLogicVb.RunRule(name As String)メソッド
 - ✓ name 実行するiLogic ルール名

InventorCoreConsole.exe

/i <*inventor file*>

/s <*script file*>



```
...  
iLogicVb.RunRule("Rule 1")  
...
```

Design Automation iLogicの記述の注意点

1. ThisApplicationではなくThisServerを
ThisApplication.ActiveDocumentではなく、
ThisDoc.Documentを使用する
2. UI表示を行う処理は記述しない
3. ログ出力はLoggerではなくTraceを使用する

iLogicのGoExcel関連機能について

DA4Iで、iLogicのGoExcel 関連の関数は利用できません

- Design Automationはクラウド上の仮想化された英語 Windows 環境でコアエンジンを起動して実行される。
- InventorのExcel形式でのBOM出力等の“Excelファイルの出力”関連の機能やiLogicのGoExcel 関連の関数は、実行環境にインストールされているExcel の機能を使用して実現。
- MS Officeのライセンス規約およびExcelの実装テクノロジー上、サーバでExcel機能は利用出来ない。

iLogic 活用のサンプル

- iLogicをWebブラウザからクラウド上で実行
- Inventorのインストール不要、Webブラウザがあればロケーションに関係なく利用可能



Tiers

3 Tier ▼

Kettle Size ☒ Small ☐ Large

Burner Type ☒ Banjo ☐ Small

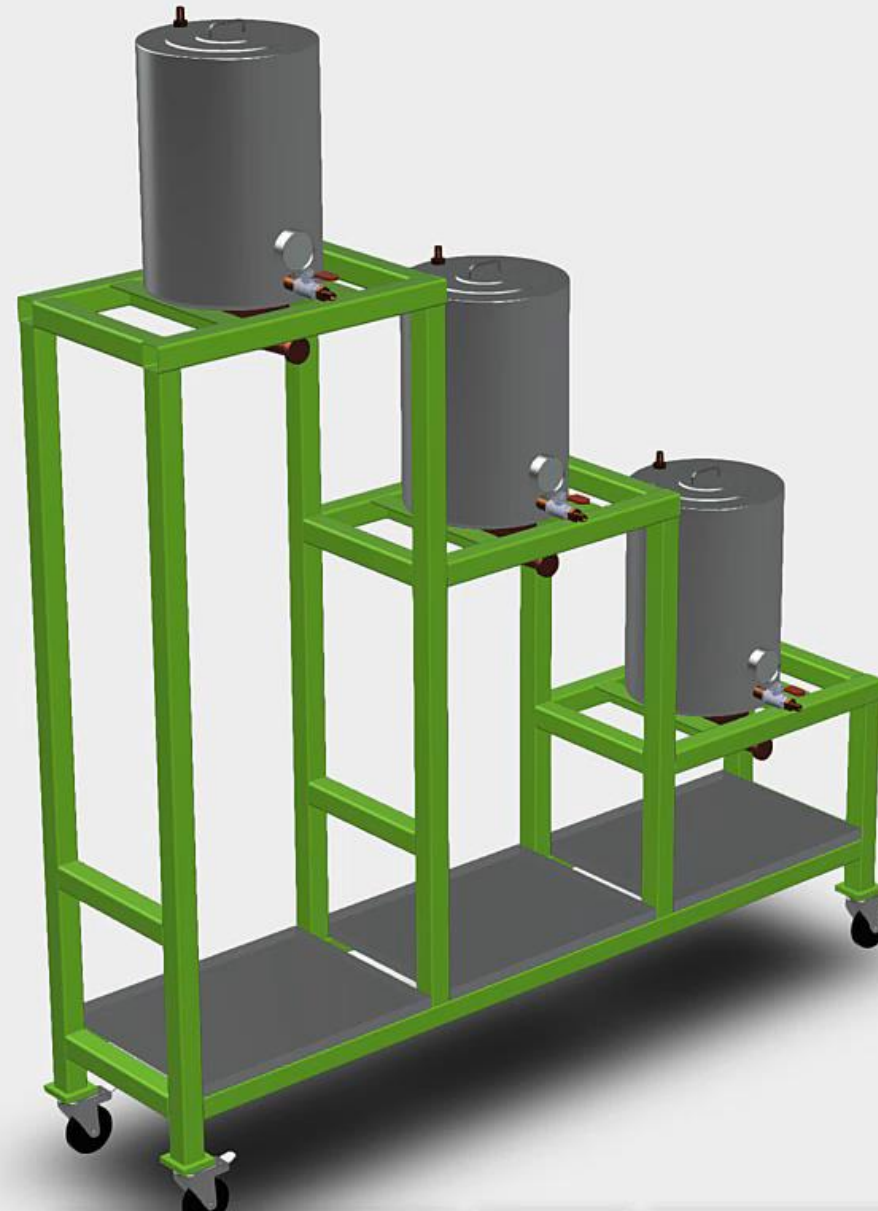
Boilover Trays ☒

Casters

Drawing

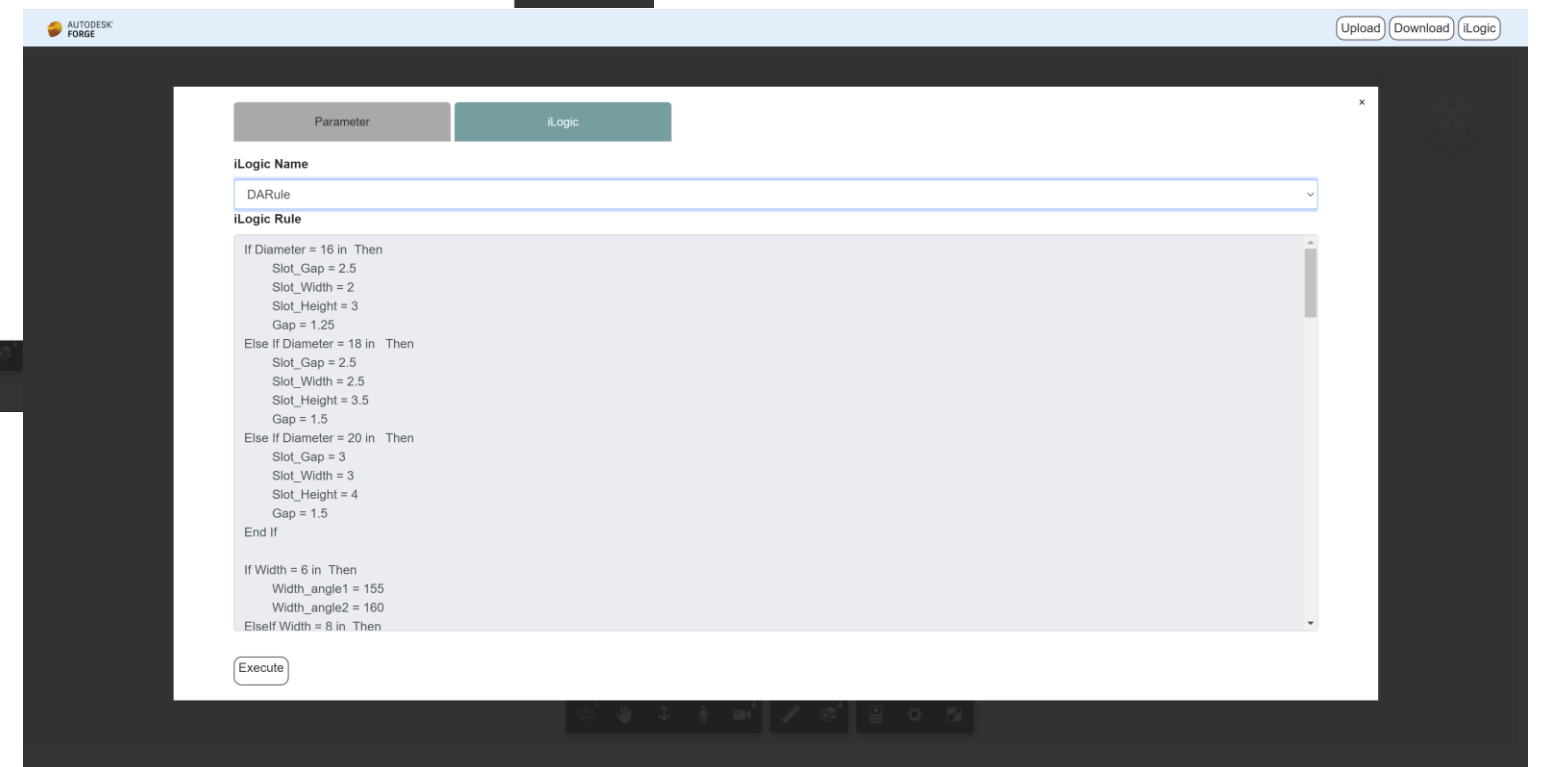
Update Assembly Drawing 

適用



Design Automation for Inventor iLogic実行のサンプルアプリケーション

<https://forge-ilogic.herokuapp.com/>



Design Automation for Inventorでアセンブリ、図面ファイルを利用する場合

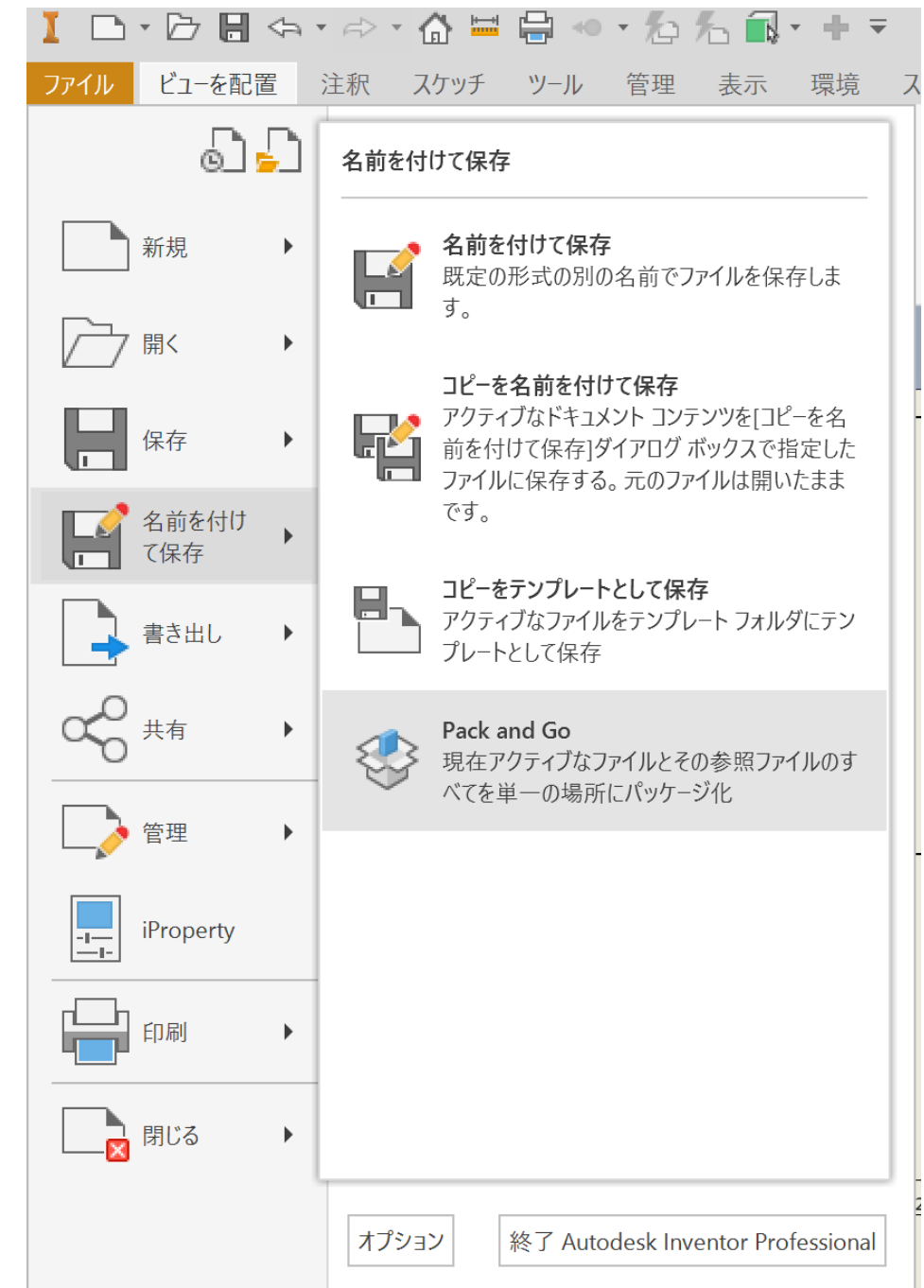
- Inventor の図面、アセンブリは複数の Inventor ファイルで構成されている

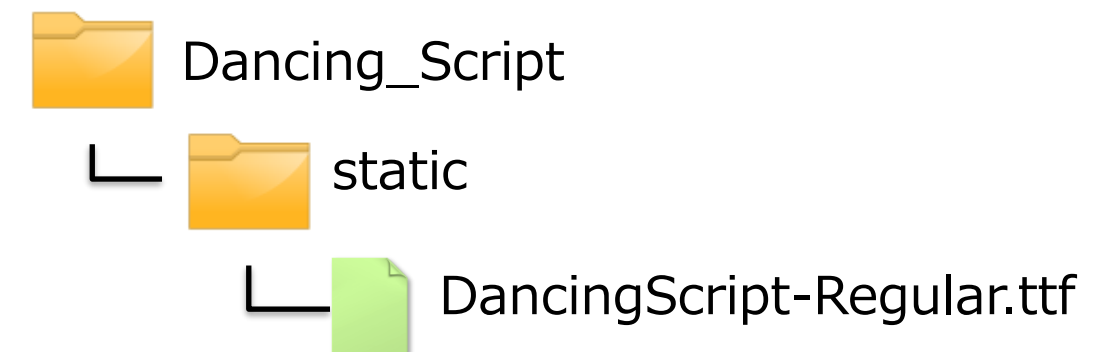
⇒親ファイル内に、参照しているファイルのパス情報を持っており、Inventorで開く際にパス情報を元にロード

- Design Automationの処理は、クラウド上にアップロードしたデータを、Design Automation実行環境（サーバ）にダウンロードして実行する

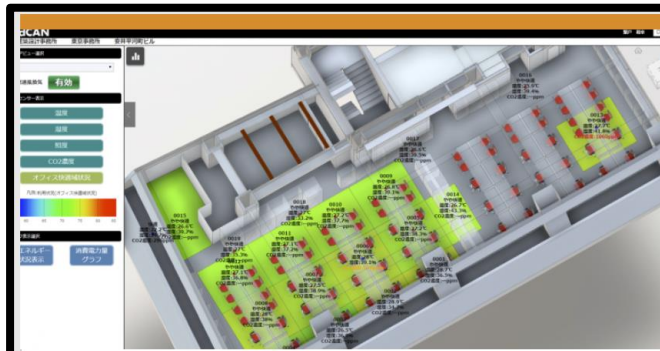
⇒アセンブリ、図面の場合、参照する全てのファイルをフォルダ配下に配置し、ZIPでまとめてアップロード

InventorのPack and Goを活用





Forge活用事例

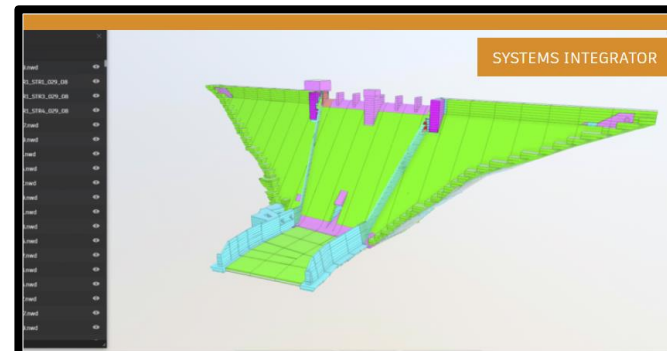


建設, 施工

株式会社 安井建築設計事務所

IoT センサー情報を活用する建築情報マネジメントシステムを Forge で開発し、ビルの運用と保守管理を最適化

→ Read story

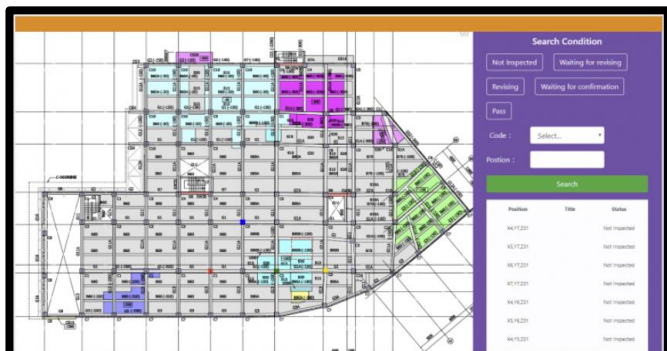


Construction, Engineering, Manufacturing

CAPA

CAPA uses Forge to help OBAYASHI streamline multiteam collaboration across massive dam construction sites.

→ Read story

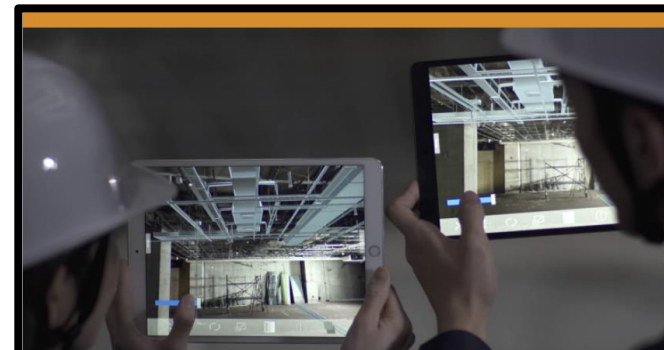


施工

応用技術株式会社

応用技術は、データのデジタル化と BIM データの共有に Forge を活用し、大成建設の現場検査を効率化

→ Read story

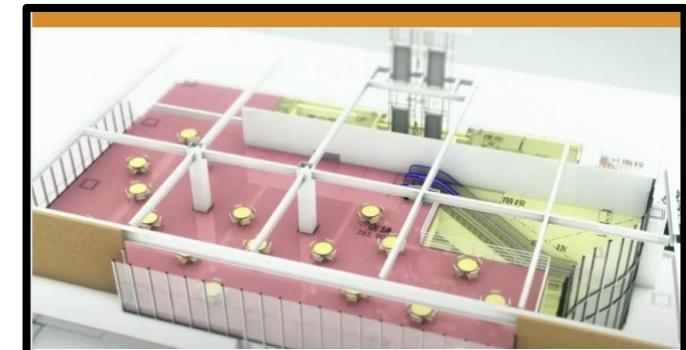


ソフトウェア開発

HoloLab

ホロラボは、建設業と製造業の専門家に向けたソリューションに Forge プラットフォームを利用して、拡張現実（AR）および複合現実（MR）ファイルへのアクセス性を高めることに成功しました。

→ Read story



施工, 設計, 製造業, 運営, メンテナンス

東芝エレベータ

東芝エレベータ社は、デジタル化によってワークフローを改善し、コラボレーションを円滑化し、新たな価値を創出しています。Forge プラットフォームは、そのデジタル化ソリューションにおける重要な原動力となっています。

→ Read story



施工, 設計, 製造, エンジニアリング

JFE エンジニアリング株式会社

JFE エンジニアリングは Forge API を導入することで、関係者が場所を問わずにさまざまなデバイスで 3D モデルの検証とコラボレーションを容易に行えるようにして、時間とリソースを節約しています。

→ Read story

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) GET workitem/:id endpoint の Rate Limit 変更について
- クラウド開発にはメンテナンスが必須
- 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！！

参考ブログ記事

- Forge Online - Design Automation : タスクの自動化
https://adndevblog.typepad.com/technology_perspective/2020/07/forg-online-design-automation-basics.html
- Forge Online - Design Automation : Inventor タスクの自動化
https://adndevblog.typepad.com/technology_perspective/2020/07/forg-online-design-automation-api-for-inventor.html
- Design Automation API for Inventor - iLogicを利用したDesign Automationの実行方法
https://adndevblog.typepad.com/technology_perspective/2020/11/design-automation-api-for-inventor-various-way-to-run-illogic.html
- Design Automation API for Inventor - Inventor iLogic利用時の注意点
https://adndevblog.typepad.com/technology_perspective/2020/12/design-automation-api-for-inventor-note-writing-an-illogic-for-da.html
- Run iLogic Rule without AppBundle
<https://forge.autodesk.com/blog/run-illogic-rule-without-appbundle>
- Different ways to run iLogic code
<https://forge.autodesk.com/ja/node/1801>
- Prepare iLogic Rules for Design Automation
<https://forge.autodesk.com/blog/prepare-illogic-rules-design-automation>



付録： コストについて

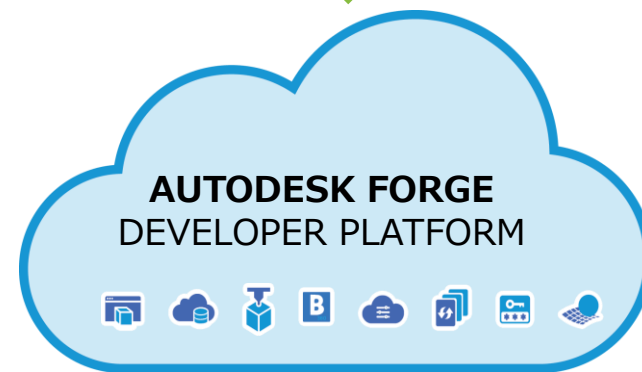
Forge はデベロッパー所有者へ '課金'

Forge アプリ開発者



Forge アプリに対する課金方針は
Forge アプリ 開発者の自由裁量

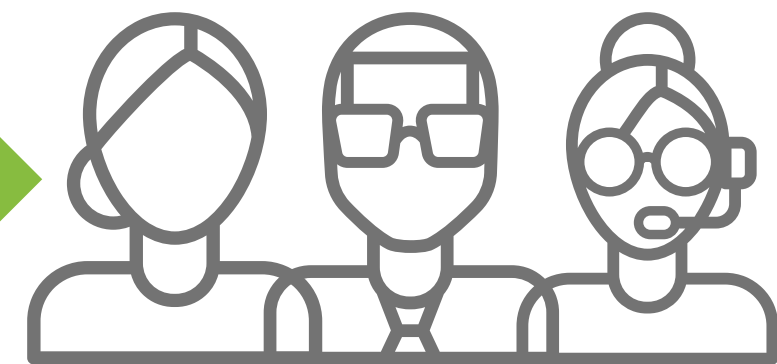
デベロッパ キーを取得した開発者に
クラウドクレジットで従量課金



オートデスク



Forge アプリ



Forge アプリの
エンドユーザ

Forge 利用に対する '課金' とは？

- クラウド クレジットによる従量 '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
 - クレジット カードによる '通貨' 課金ではない
 - <https://forge.autodesk.com/pricing>
 - 別途 Forgeサブスクリプションが必要

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	写真画像 50 枚	3.0ドル – 480円

Design Automation API のコスト算出

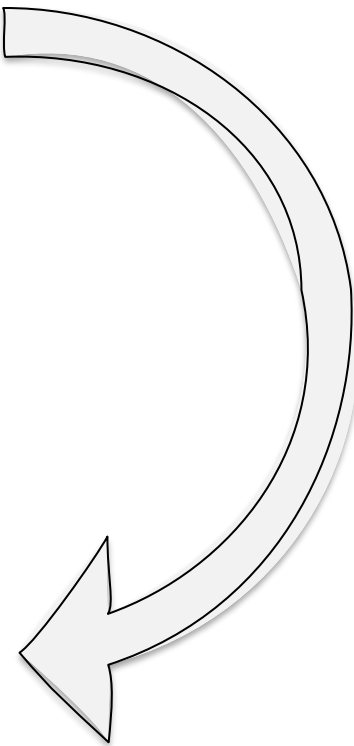
- WorkItem の実行時間に課金
 - ファイルのダウンロードとアップロード時間を含む
 - ストレージから実行環境への素材ファイルのダウンロード
 - 実行環境からストレージへの成果ファイルのアップロード
 - ログ上の *timeDownloadStarted* から *timeUploadEnded* まで
 - “timeQueued”: “2019-11-13T20:22:26.1972643Z”,*
 - “timeDownloadStarted”: “2019-11-13T20:22:25.587368Z”,*
 - “timeInstructionsStarted”: “2019-11-13T20:24:01.3217437Z”,*
 - “timeInstructionsEnded”: “2019-11-13T20:24:35.4545907Z”,*
 - “timeUploadEnded”: “2019-11-13T20:26:03.3608278Z”,*

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）
 - 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11月7日）
 - （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

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）



11月7日に Autodesk Flex に移行

<https://www.autodesk.co.jp/benefits/flex>

- サブスクリプション制を廃止
- クラウド クレジットから Flex トークンによる従量課金

API	消費 Flexトークン	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 Flex トークン	変換	1.5ドル – 180円
Model Derivative – シンプル ジョブ（Revit、Navisworks、IFC 以外）	0.1 Flex トークン	変換	0.3ドル – 108円
Design Automation – すべて	2 Flex トークン	処理時間	6.0ドル – 720円
Reality Capture	1 Flex トークン	写真画像 50 枚	3.0ドル – 360円

- 購入は特定の認定リセラーまで
 - [Autodesk Customer Success Hub](#)

クラウドクレジットは当面存続

- 未使用の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

