Revit カスタマイズ＆クラウドセミナー

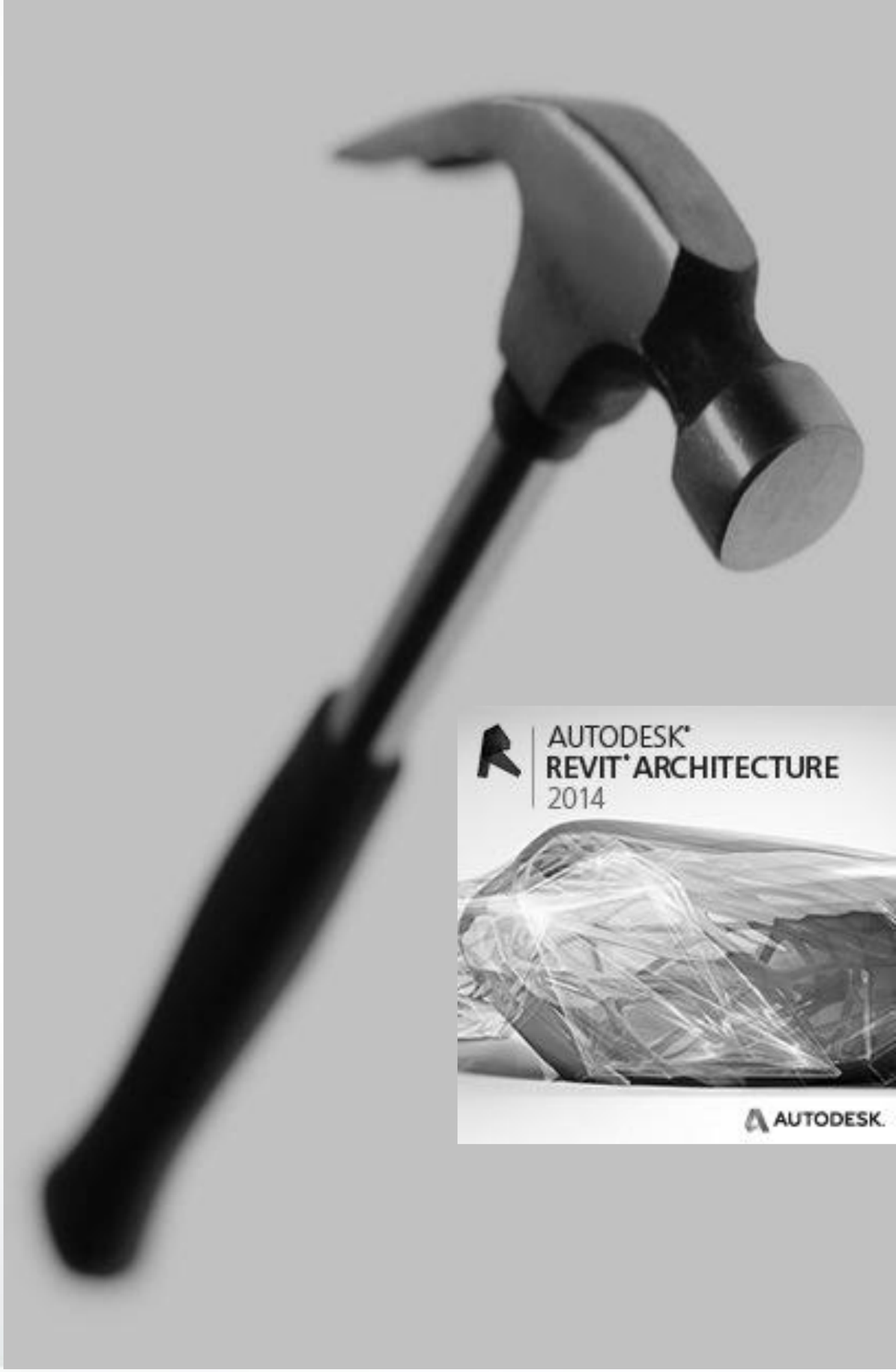
カスタマイズ シナリオと展開

工藤　暁

Developer Technical Services

AUTODESK.

はじめに



カスタマイズで重要なこと

1. 何をカスタマイズしたいかを明確にする

2. どうカスタマイズすればいいか学習する

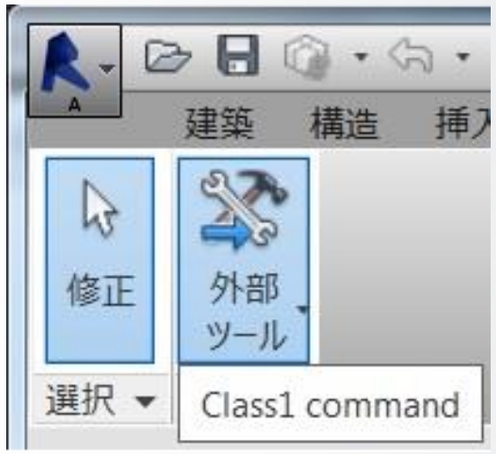
3. カスタマイズがどこに保存、維持されているか知る

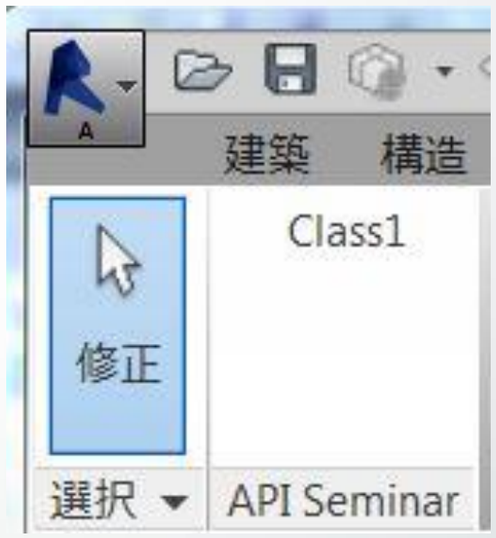
Revitでカスタマイズ可能な領域

- SDKを使用して可能な事
  - モデルデータへのアクセス
  - モデルパラメータへのアクセス
  - 床、壁、柱等のモデル要素の作成、編集、削除
  - タスクを自動化するアドインの作成
  - アプリケーションの統合
    - 例　データベースとの連携
  - プロジェクトのドキュメントを自動作成

一般的なカスタマイズのシナリオ

- コマンドの作成
  - 独自の作図コマンドを作成
- ユーザ インタフェースの登録
  - よく利用するコマンドをボタンとしてリボンに登録
- カスタマイズした定義ファイルのRevit へのロード
  - コマンドを定義した API モジュール
  - ユーザ インタフェース
- カスタマイズした定義ファイルの他PCへの展開
  - 社内設計者への効率的な配布（インストール）



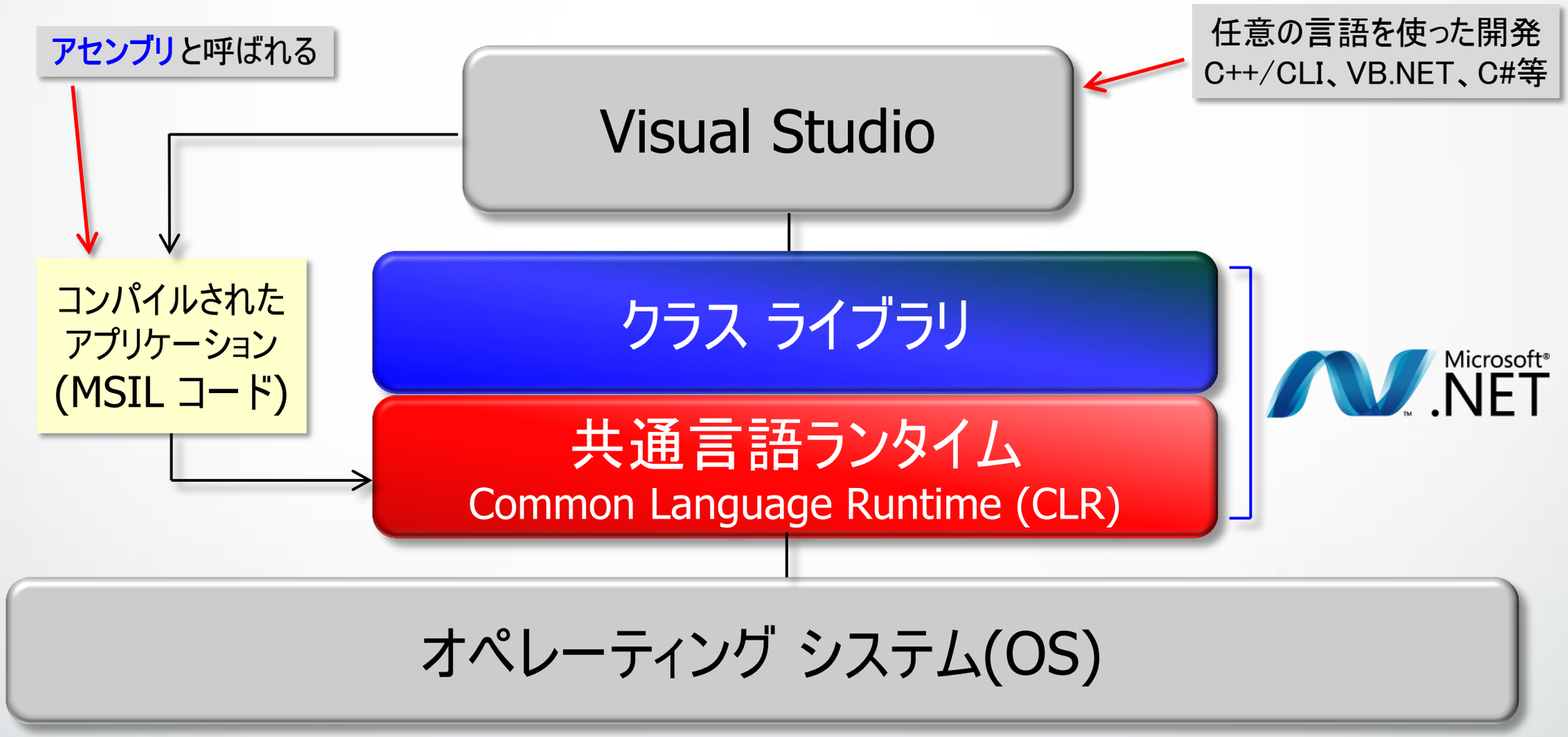


開発を始める前に

- 開発環境
  - Revit Architecture 2014, Revit Structure 2014
  - Microsoft Visual Studio 2010
  - Microsoft .NET Framework 4.0
  - C#,VB.NET等の開発言語
  - Revit Software Developer’s Kit

.NET Frameworkの優位性

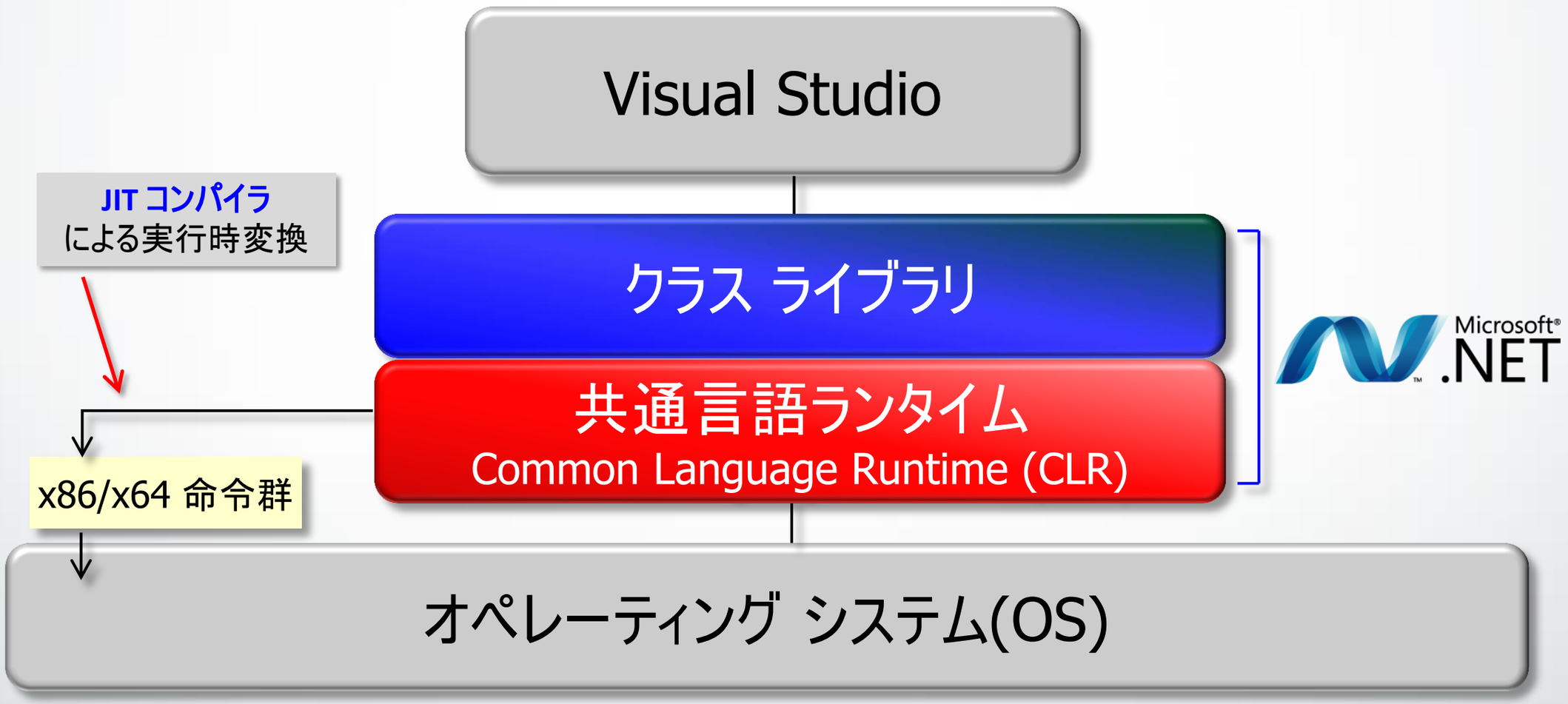
- ビルドされた .NET アプリケーションは中間コードに変換されます…



.NET Frameworkの優位性

- アセンブリは実行時に **x86**、**x64** バイナリにコンパイル、実行される

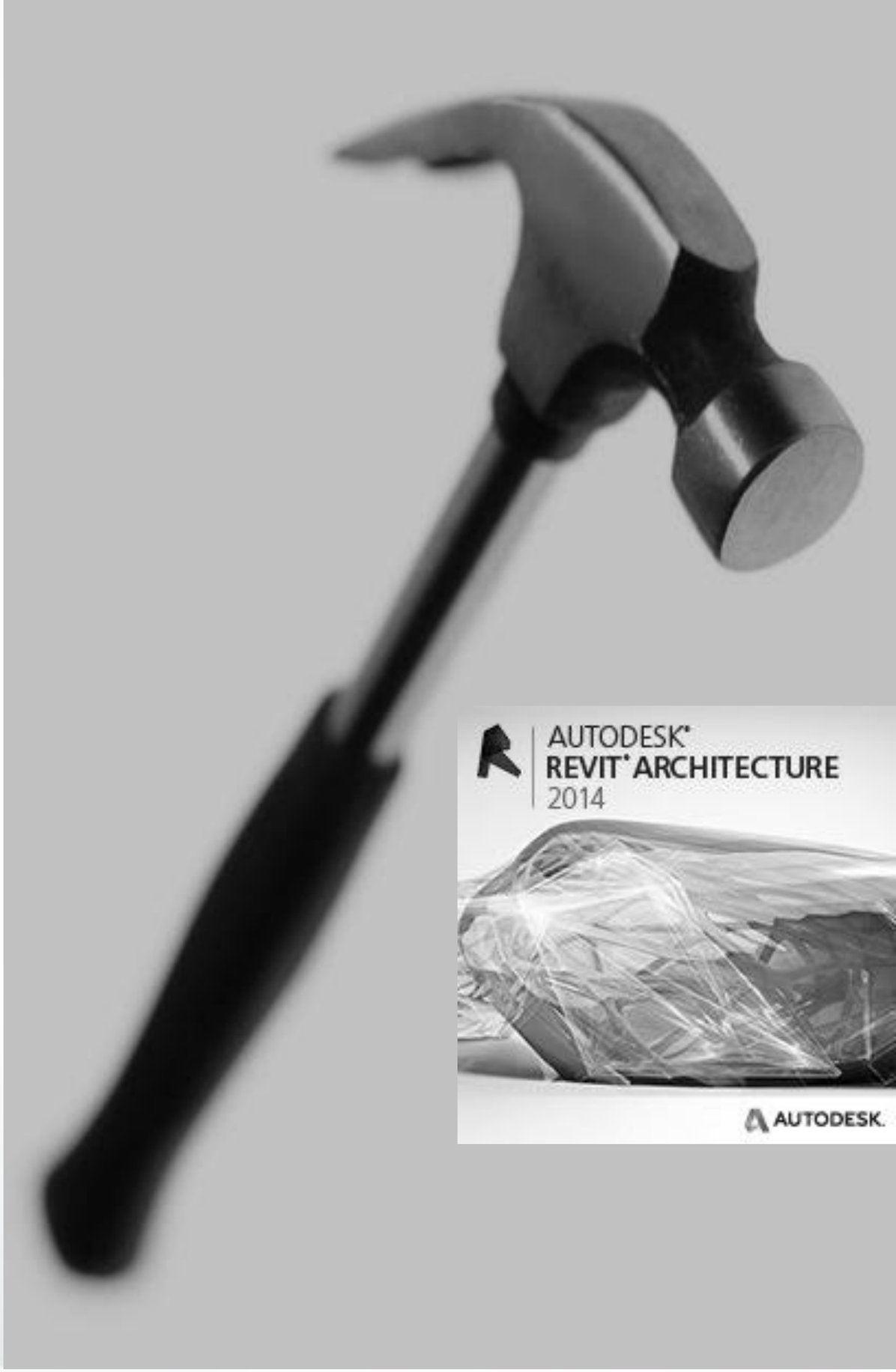
.NET Framework はプラットフォームの違いを吸収



開発言語の選択

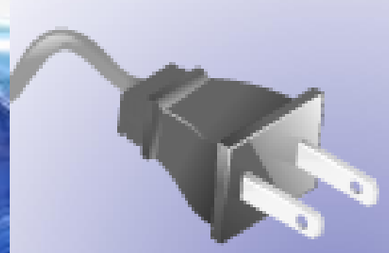
- Microsoft .NET Framework 4.0対応言語を選択可能
  - Visual Basic .NET, C#, C++/CLI等
  - C#又はVisual Basic .NETを推奨
  - SDKサンプル及びブログ等ではC#又はVisual Basic .NETを使用
  - ADNエンジニアでも、その他の言語を使用するのは稀

アプリケーションの作成



Revit Platform API

- 柔軟で強力な Application Programming Interface



.NET API
(.NET)

SDKの構成

- Revit Platform APIはRevitアプリケーション機能の基本
  - 2つのクラスライブラリにて構成
- RevitAPI.dll
  - アプリケーション、ドキュメント、要素及びベータベースレベルのパラメータにアクセスする関数を保持
- RevitAPIUI.dll
  - UIのカスタマイズや操作に関連するAPIを保持
    - 外部コマンド関連インターフェイス
    - 外部アプリケーション関連インターフェイス
    - セレクション
    - リボンパネル,リボンアイテムとサブクラス
    - タスクダイアログ

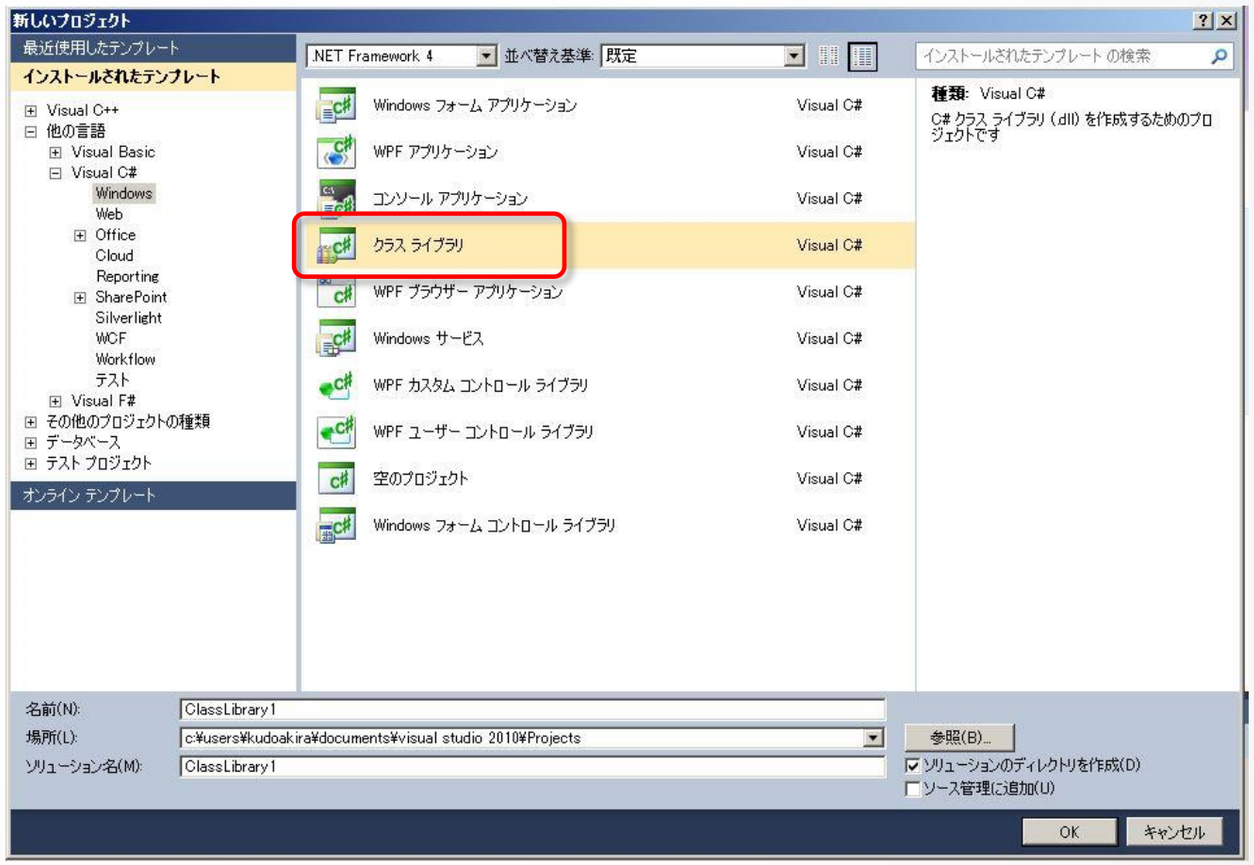
プロジェクトの作成方法

- 手動作成
  - クラス ライブラリ プロジェクト作成
- 自動作成
  - Plug-in Wizardから作成

プロジェクトの作成

1. Visual Studio 2010 で クラス ライブラリ プロジェクト作成

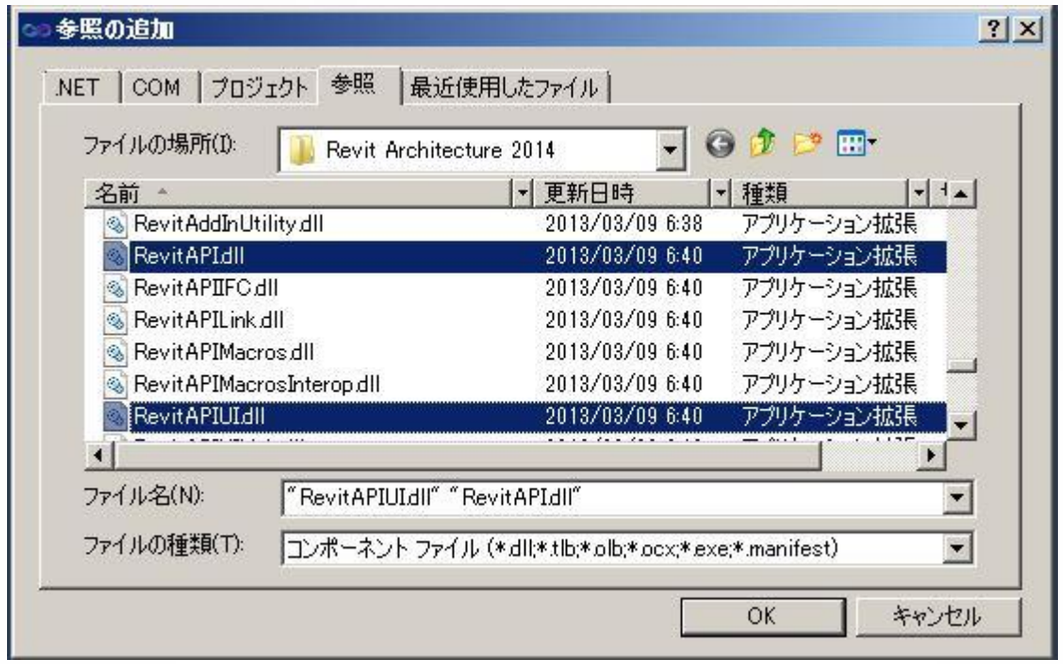
C#プロジェクトの場合 →



2. ソリューション エクスプローラから “参照の追加”を選択

プロジェクトの作成

3. RevitAPI.dllとRevitAPIUI.dll を選択



4. 参照のローカルコピーをFalseに

AUTODESK.

プロジェクトの作成

5. グローバル スコープに名前空間を追加してクラスにコマンドを追加

```
using System;
using System.Collections.Generic;
using System.Text;

using Autodesk.Revit;
using Autodesk.Revit.DB;
using Autodesk.Revit.UI;
using Autodesk.Revit.ApplicationServices;

namespace ClassLibrary1
{
   [Autodesk.Revit.Attributes.Transaction(Autodesk.Revit.Attributes.TransactionMode.Manual)]
  public class    Class1 : IExternalCommand
   {
       public Autodesk.Revit.UI.Result Execute(ExternalCommandData commandData,
         ref string message, Autodesk.Revit.DB.ElementSet elements)
      {
           return Autodesk.Revit.UI.Result.Succeeded;
      }
   }
}
```

名前空間のインポート

コマンド定義記述とコマンド関数

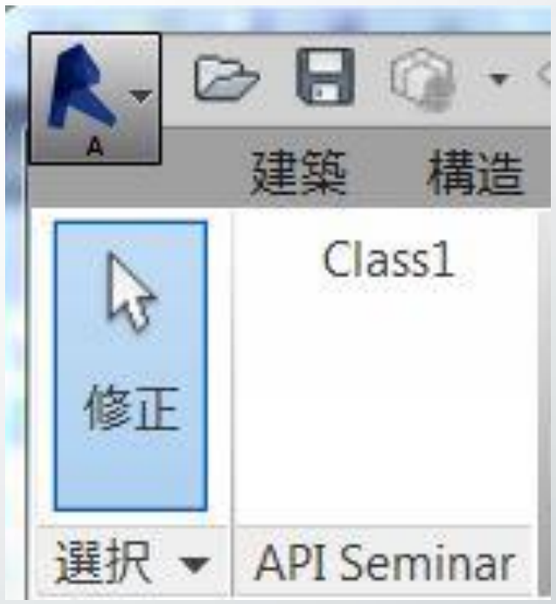
外部コマンド

- 外部ツールメニューより実行
- Execute()
  - ボタンを押すと起動される関数
- IExternalCommandを継承


```
public class Class1 : IExternalCommand
{
      public Autodesk.Revit.UI.Result Execute(
        ExternalCommandData commandData,
        ref string message,
        Autodesk.Revit.DB.ElementSet elements)
      {
          return Autodesk.Revit.UI.Result.Succeeded;
      }
}
```

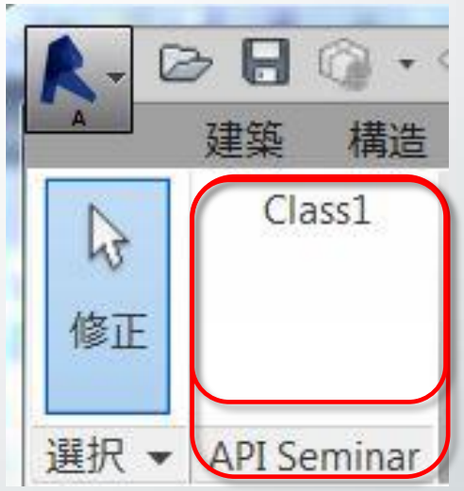
外部アプリケーション

- IExternalApplicationを継承
  - 2つのオーバーライド関数OnStartup()とOnShutdown()
- リボン・タブ、リボン・パネルのカスタマイズに使用
- リボン・パネルのボタンには外部コマンド



外部アプリケーション

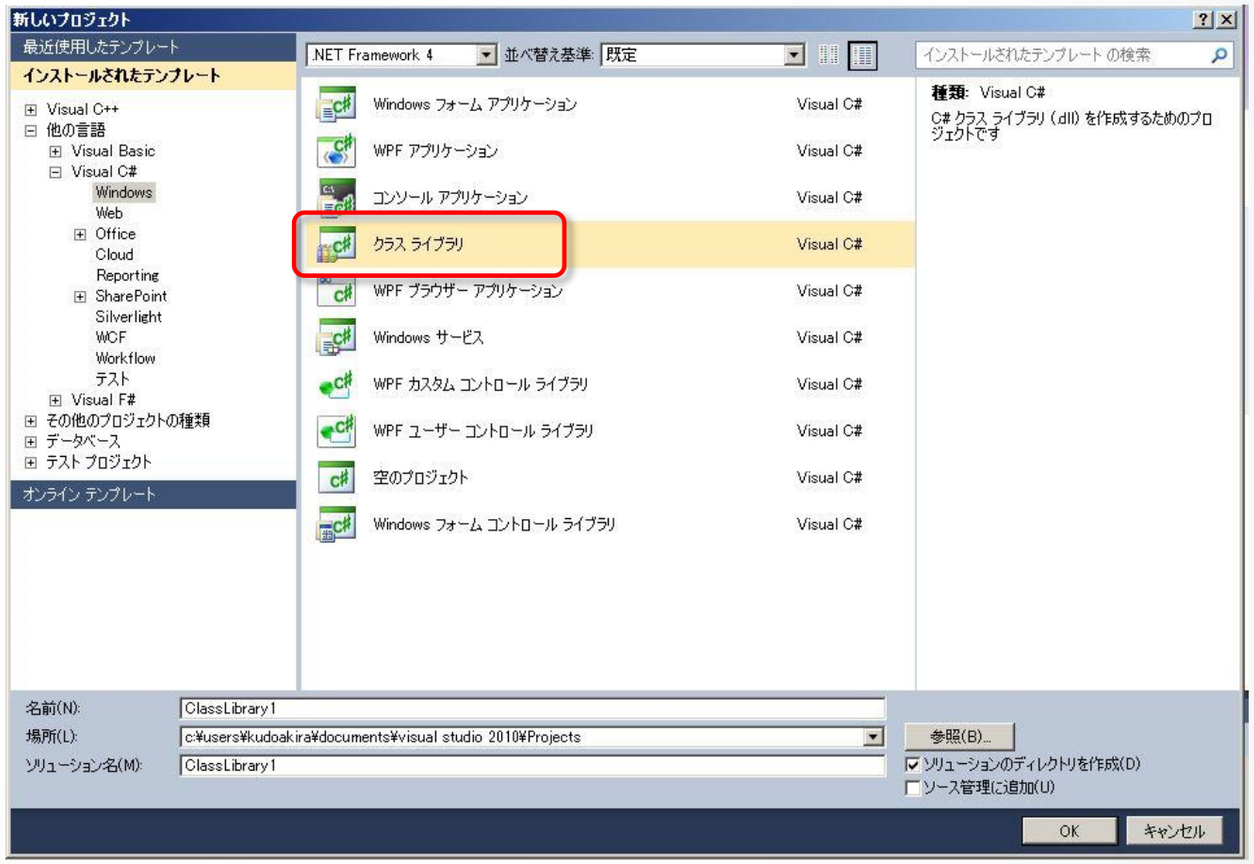
```
public class Class2 : IExternalApplication
{
    // ExternalCommands assembly path
    static string AddInPath = typeof(Class2).Assembly.Location;

    public Autodesk.Revit.UI.Result OnStartup(UIControlledApplication application)
    {
        RibbonPanel ribbonSamplePanel = application.CreateRibbonPanel("API Seminar");
        PushButtonData pushButton = new PushButtonData("ForExCmd", "Class1", AddInPath, "ClassLibrary1.Class1");
        ribbonSamplePanel.AddItem(pushButton);

        return Autodesk.Revit.UI.Result.Succeeded;
    }

    public Autodesk.Revit.UI.Result OnShutdown(UIControlledApplication application)
    {
        //remove events
        return Autodesk.Revit.UI.Result.Succeeded;
    }
}
```



プロジェクトの作成

1. Visual Studio 2010 で クラス ライブラリ プロジェクト作成

C#プロジェクトの場合 →

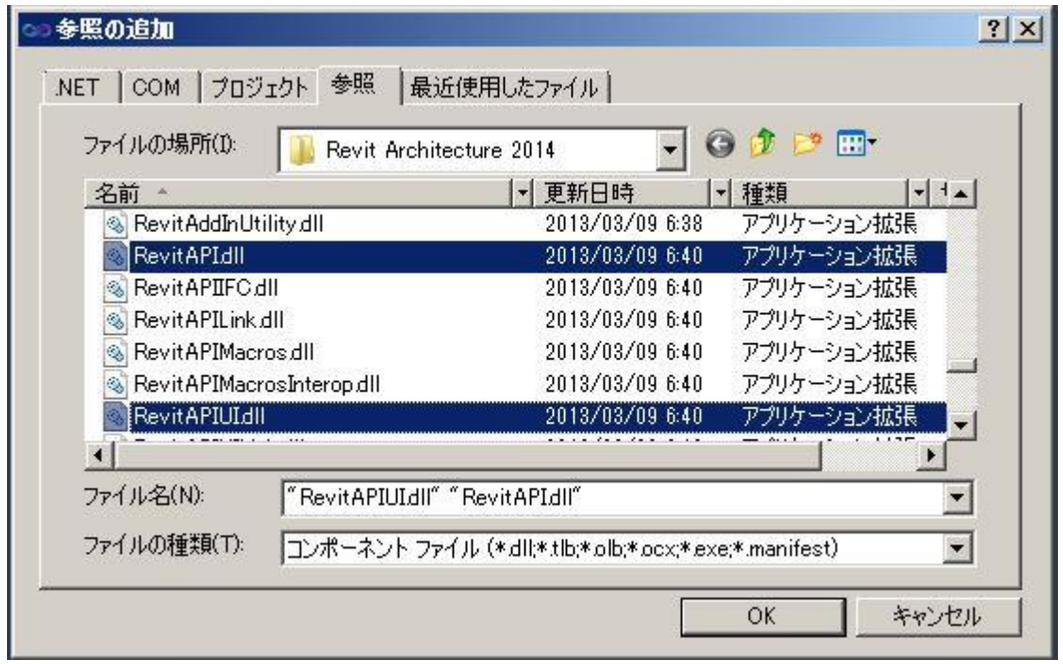


2. ソリューション エクスプローラから “参照の追加”を選択

AUTODESK.

プロジェクトの作成

3. RevitAPI.dllとRevitAPIUI.dll を選択



4. 参照のローカルコピーをFalseに

プロジェクトの作成

5. グローバル スコープに名前空間を追加してクラスにコマンドを追加

```
using System;
using System.Collections.Generic;
using System.Text;

using Autodesk.Revit;
using Autodesk.Revit.DB;
using Autodesk.Revit.UI;
using Autodesk.Revit.ApplicationServices;

namespace ClassLibrary1
{
   [Autodesk.Revit.Attributes.Transaction(Autodesk.Revit.Attributes.TransactionMode.Manual)]
  public class    Class1 : IExternalCommand
   {
       public Autodesk.Revit.UI.Result Execute(ExternalCommandData commandData,
         ref string message, Autodesk.Revit.DB.ElementSet elements)
      {
           return Autodesk.Revit.UI.Result.Succeeded;
      }
   }
}
```

名前空間のインポート

コマンド定義記述とコマンド関数

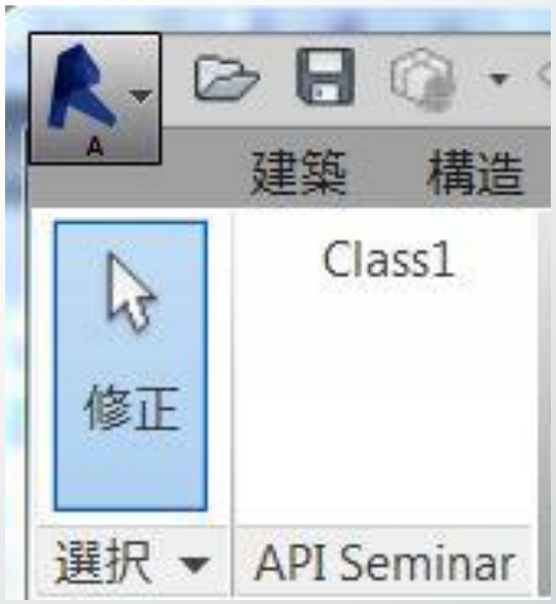
外部コマンド

- 外部ツールメニューより実行
- Execute()
  - ボタンを押すと起動される関数
- IExternalCommandを継承


```
public class Class1 : IExternalCommand
{
      public Autodesk.Revit.UI.Result Execute(
        ExternalCommandData commandData,
        ref string message,
        Autodesk.Revit.DB.ElementSet elements)
      {
          return Autodesk.Revit.UI.Result.Succeeded;
      }
}
```

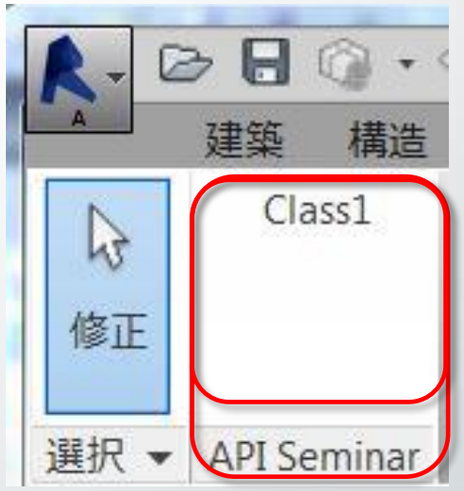
外部アプリケーション

- IExternalApplicationを継承
  - 2つのオーバーライド関数OnStartup()とOnShutdown()
- リボン・タブ、リボン・パネルのカスタマイズに使用
- リボン・パネルのボタンには外部コマンド



外部アプリケーション

```
public class Class2 : IExternalApplication
{
    // ExternalCommands assembly path
    static string AddInPath = typeof(Class2).Assembly.Location;

    public Autodesk.Revit.UI.Result OnStartup(UIControlledApplication application)
    {
        RibbonPanel ribbonSamplePanel = application.CreateRibbonPanel("API Seminar");
        PushButtonData pushButton = new PushButtonData("ForExCmd", "Class1", AddInPath, "ClassLibrary1.Class1");
        ribbonSamplePanel.AddItem(pushButton);

        return Autodesk.Revit.UI.Result.Succeeded;
    }

    public Autodesk.Revit.UI.Result OnShutdown(UIControlledApplication application)
    {
        //remove events
        return Autodesk.Revit.UI.Result.Succeeded;
    }
}
```



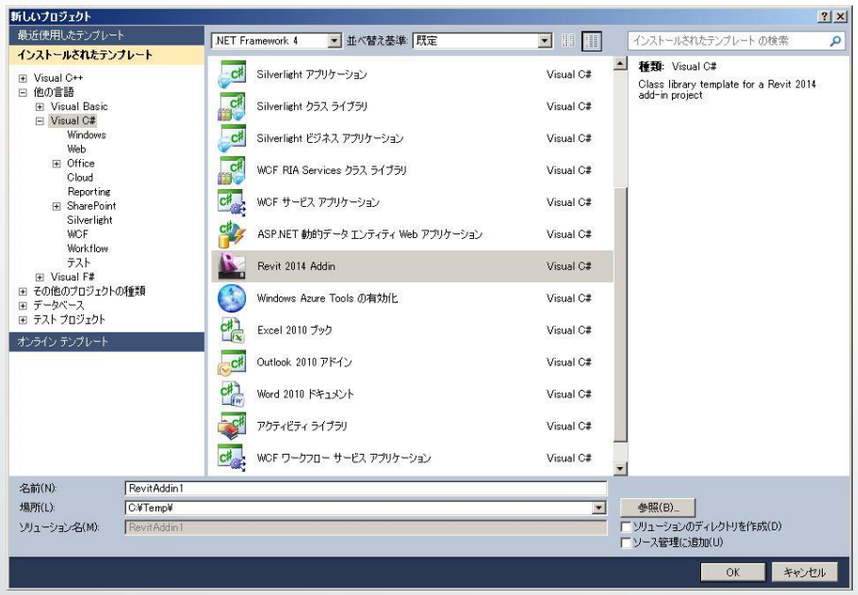
アプリケーションの提供形式

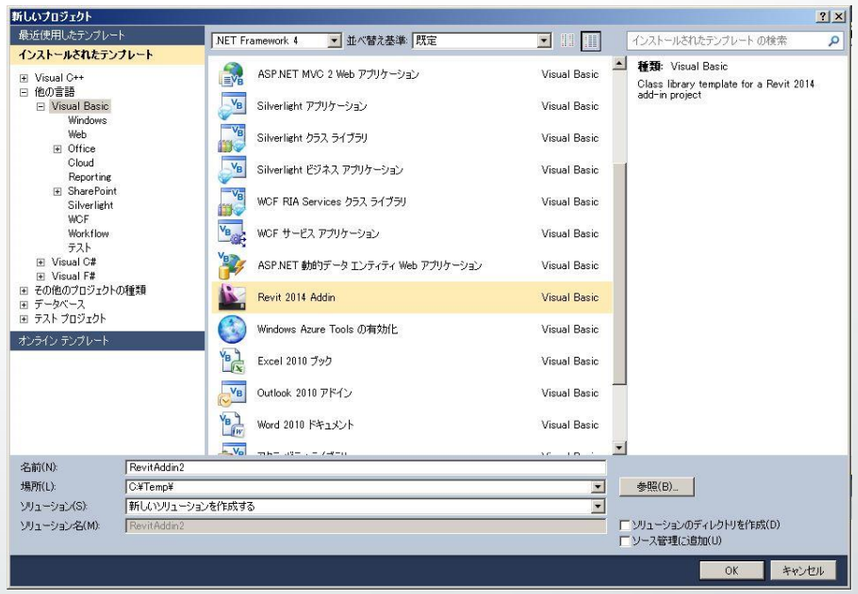
- API のアセンブリ
  - .NET API : .dll ファイル 
- 実行にはアセンブリを Revitにロード
  - ロード後に定義した機能が利用可能に

Revit

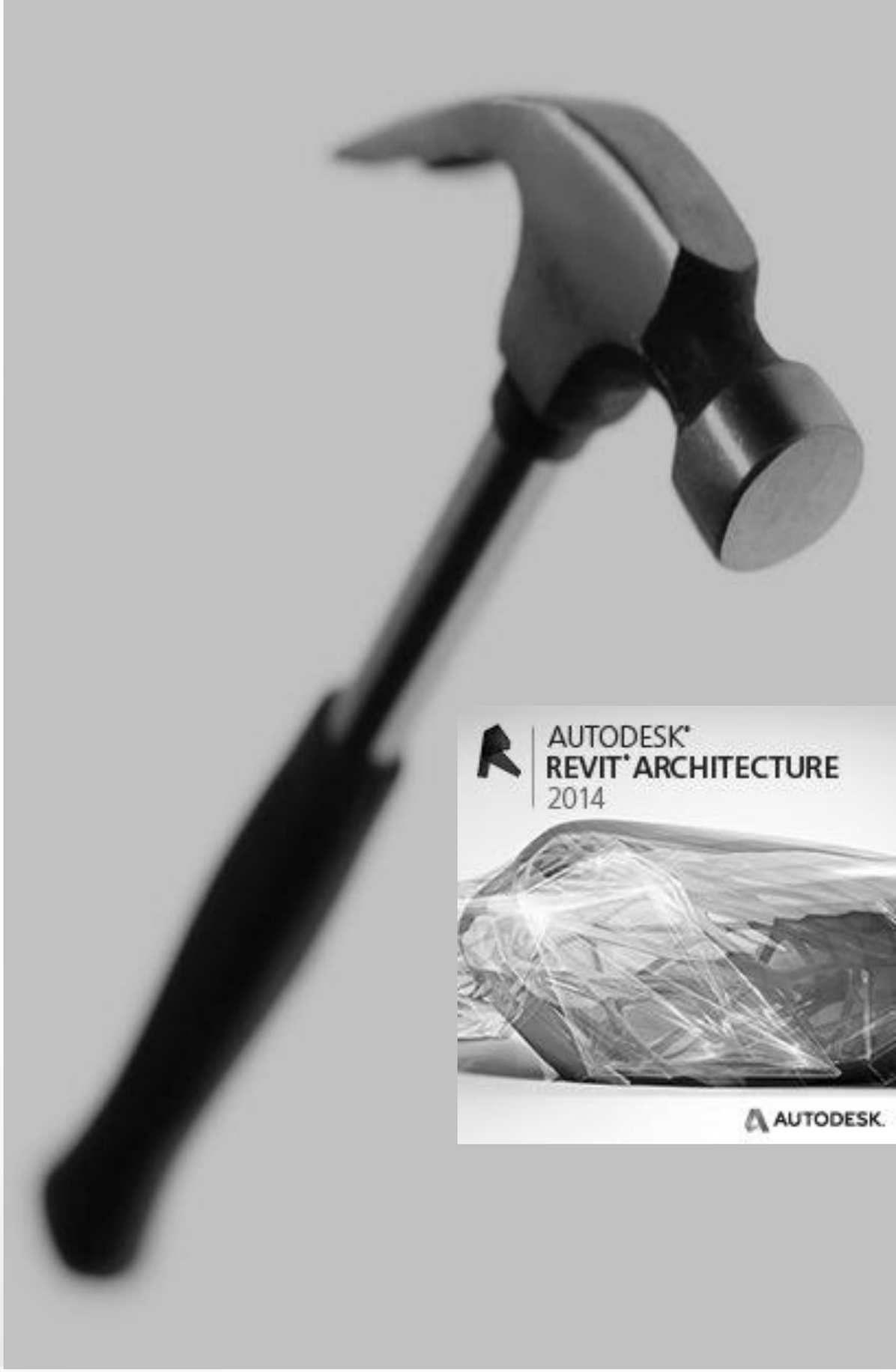
プロジェクトの作成方法

- Plug-in Wizard
  - Jeremy’s Blog “The Building Coder”
    - May 09, 2013 “Add-In Wizards for Revit 2014”
    - Revit2014AddinWizardCs.zip を
      [マイ ドキュメント]\Visual Studio 2010\Templates\ProjectTemplates\Visual C# へ配置
    - Revit2014AddinWizardVb.zip を
      [マイ ドキュメント]\Visual Studio 2010\Templates\ProjectTemplates\Visual Basic へ配置





アプリケーションのロード

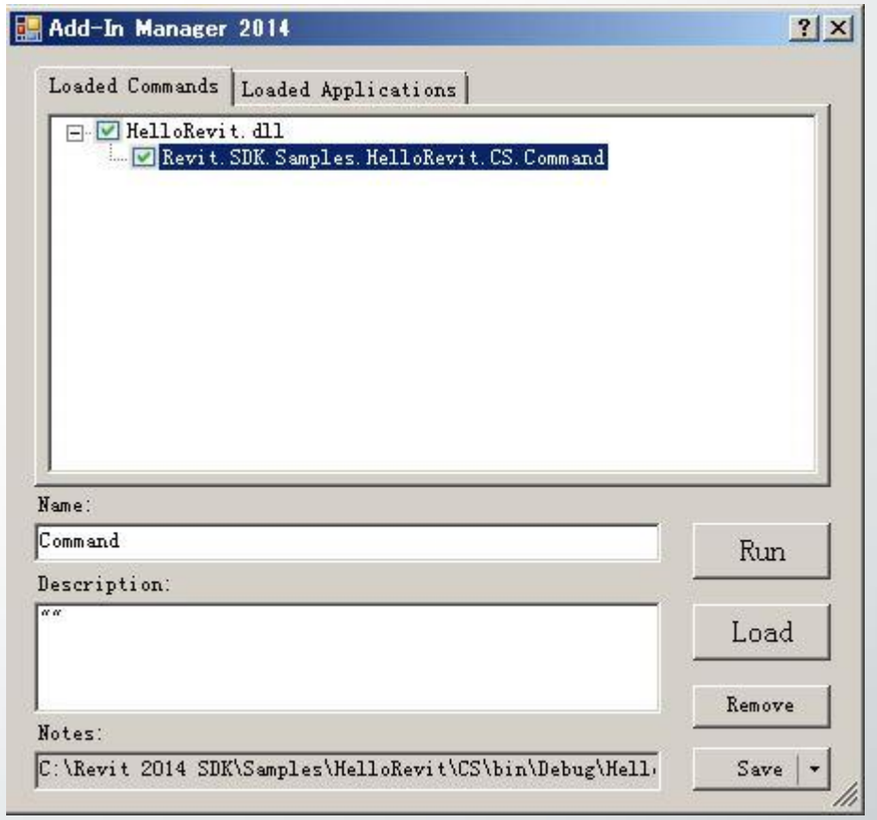


アプリケーションのロード方法

- 手動ロード
  - Add-In Manager
    - Revit上にてアセンブリ（DLL）を指定
      - コマンド実行
      - .addinマニフェストファイルの作成
- 自動ロード
  - アドインファイルを使った方法
    - .addinマニュフェストファイルによる自動ロード
  - 自動ローダーを使った方法
    - XML パッケージファイルでのロード

アドインの登録

- Add-In Managerの使用
  - SDK　Add-In Managerフォルダにインストーラが存在
  - DLLを指定しRunボタンにて外部コマンドを実行
  - 外部アプリケーションの登録
  - Add-inファイルの作成と配置



アドインの登録

- 外部コマンド登録例

```
<?xml version="1.0" encoding="utf-8" ?>
- <RevitAddIns>
  - <AddIn Type="Command">
      <Assembly>C:\Temp\ClassLibrary1\bin\Debug\ClassLibrary1.dll</Assembly>
      <ClientId>893C7BCF-1F25-435F-AF8B-B2B5D29C2A0A</ClientId>
      <FullClassName>ClassLibrary1.Class1</FullClassName>
      <Text>Class1 command</Text>
      <Description>API Seminar sample</Description>
      <VendorId>ADSK</VendorId>
      <VendorDescription>Autodesk, www.autodesk.com</VendorDescription>
    </AddIn>
</RevitAddIns>
```

ベンダーIDが必要な際は…
www.autodesk.com/symbolreg

アドインの登録

- 外部アプリケーション登録例

```
<?xml version="1.0" encoding="utf-8" ?>
- <RevitAddIns>
  - <AddIn Type="Application">
      <Name>External Tool</Name>
      <Assembly>C:\Temp\ClassLibrary1\bin\Debug\ClassLibrary1.dll</Assembly>
      <ClientId>94F1AEC4-BD3B-4A77-82CA-170AA2F452E1</ClientId>
      <FullClassName>ClassLibrary1.Class2</FullClassName>
      <VendorId>ADSK</VendorId>
      <VendorDescription>Autodesk, www.autodesk.com</VendorDescription>
    </AddIn>
</RevitAddIns>
```

詳しくは… Autodesk WikiHelp - Add-in Registration

http://wikihelp.autodesk.com/Revit/enu/2014/Help/3661-Developers/0001-Introduc1/0018-Add-In_I18/0022-Add-in_R22

アドインの登録

- 外部アプリケーション、コマンドをRevitに認識させる
- .addinマニュフェストファイルに登録
- Revitの起動時にロード
- マニュフェストファイルの配置場所
  - 全ユーザーが使用
    - Windows 7/8 - C:\ProgramData\Autodesk\Revit\Addins\2014\
  - ユーザー別に使用する
    - Windows 7/8 - C:\ユーザー\<user>\AppData\Roaming\Autodesk\Revit\Addins\2014\

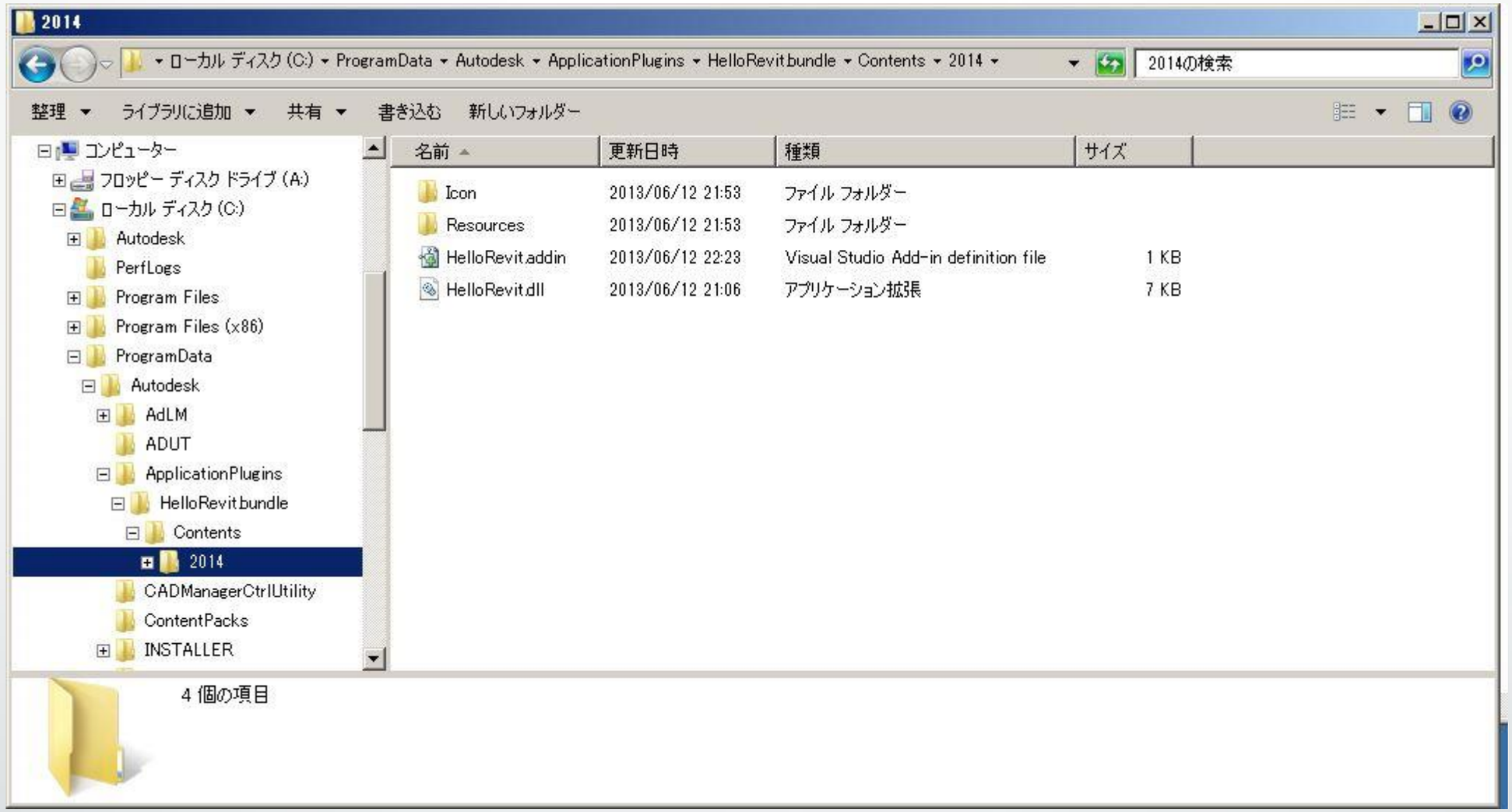
自動ローダーの利点

- 必要なリソースと API モジュールの一括ロードが可能
  - Autodesk Exchange Apps のロード機能
  - 購入&ダウンロードですぐに利用できるようにするため
- 準備
  - API カスタマイズ ファイル（.NET API のアセンブリ ファイル）
  - 部分カスタマイズ ファイルの用意
    - ビットマップ等やヘルプファイル等のリソース
  - PackageContents.xml ファイル
    - アプリケーション情報やファイルのありかを記した XML ファイル

詳しくは… Autodesk Exchange Apps - Autodesk Revit デベロッパ向けの情報
http://www.autodesk.co.jp/adsk/servlet/item?siteID=1169823&id=21691942

自動ローダーを利用したカスタマイズの配置と構造

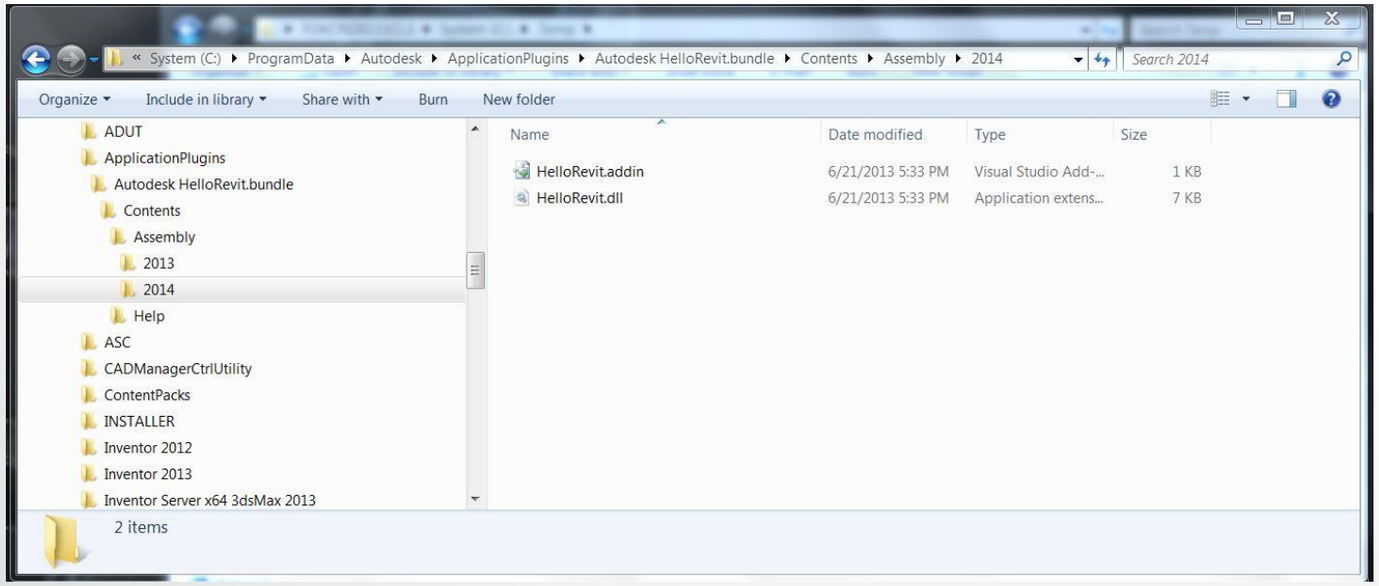
- C:\ProgramData\Autodesk\ApplicationPlugins フォルダ
  - *XXXXX.bundle* フォルダ配下に各ファイルを配置



PackageContents.xml ファイルの記述

- アプリケーション パッケージ

```
<?xml version="1.0" encoding="utf-8" ?>
- <ApplicationPackage SchemaVersion="1.0" AutodeskProduct="Revit" Name="HelloRevit" Description="HelloRevit" AppVersion="1.0.0" FriendlyVersion="1.0.0"
    ProductType="Application" HelpFile="./Contents/Help/help.txt" SupportedLocales="Enu" AppNameSpace="appstore.exchange.autodesk.com"
    Author="Akira Kudo" ProductCode="{23EFF01B-2F95-49CC-8B92-F3017F37BFA1}" UpgradeCode="{56B68FB5-622F-4956-9454-8BC58A0A3A26}"
    OnlineDocumentation="www.autodesk.com">
    <CompanyDetails Name="Autodesk" Email="akira.kudo@autodesk.com" Url="www.autodesk.com" />
    <RuntimeRequirements OS="Win32|Win64" Platform="Revit" SeriesMin="R2013" SeriesMax="R2014" />
  - <Components Description="Assembly parts">
      <RuntimeRequirements OS="Win32|Win64" Platform="Revit" SeriesMin="R2013" SeriesMax="R2013" />
      <ComponentEntry AppName="HelloRevit" Version="1.0.0" ModuleName="./Contents/Assembly/2013/HelloRevit.addin" />
    </Components>
  - <Components Description="Assembly parts">
      <RuntimeRequirements OS="Win32|Win64" Platform="Revit" SeriesMin="R2014" SeriesMax="R2014" />
      <ComponentEntry AppName="HelloRevit" Version="1.0.0" ModuleName="./Contents/Assembly/2014/HelloRevit.addin" />
    </Components>
</ApplicationPackage>
```



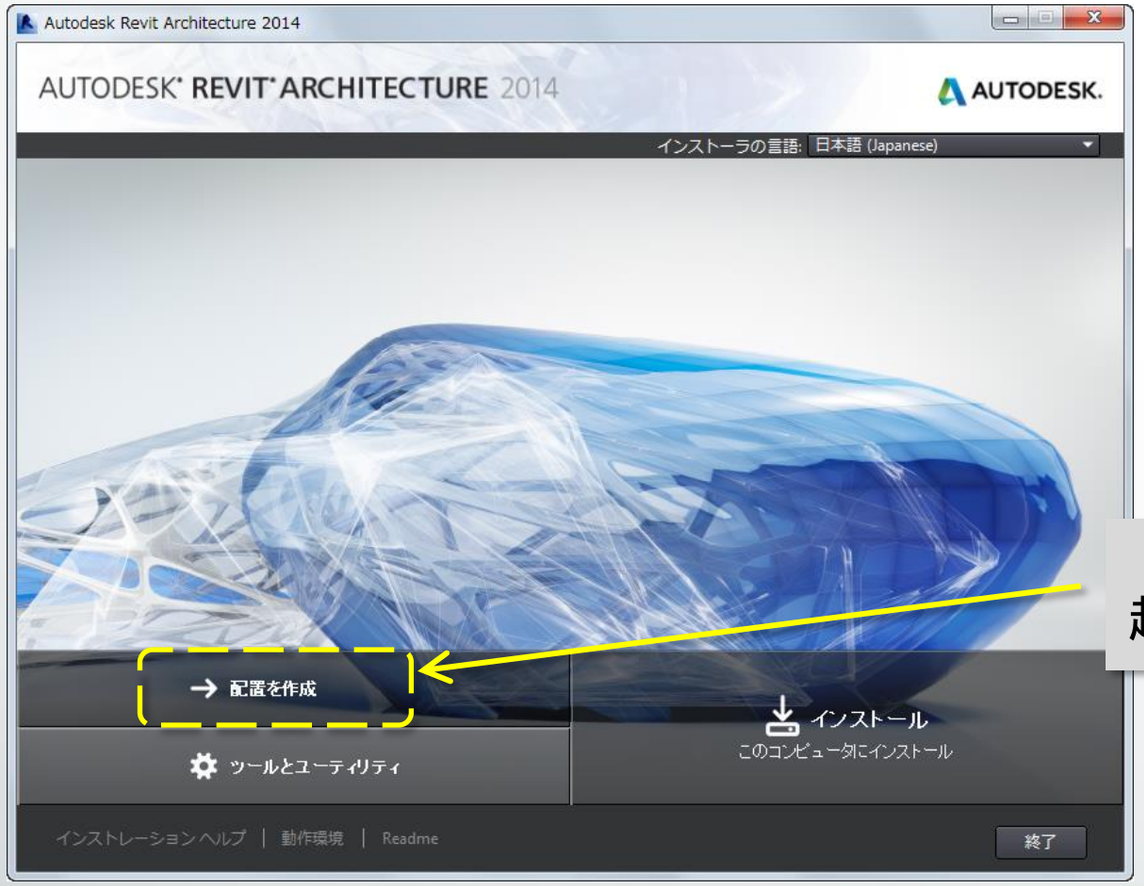
詳しくは… PackageContents.xml 形式リファレンス

http://docs.autodesk.com/ACD/2014/JPN/index.html?url=files/GUID-BC76355D-682B-46ED-B9B7-66C95EEF2BD0.htm,topicNumber=d30e492496

AUTODESK.

Revit とカスタマイズ ファイルをインストール

- 配置イメージを利用してカスタマイズ込みの Revit をインストール
- 配置イメージとは？
  - サーバーの共有フォルダにインストール イメージを作成する（DVD-ROMの代わり）
  - 元々はネットワーク ライセンスのクライアントに Revit をインストールする手法



Revit の DVD-ROM から
起動したインストーラから作成します

Autodesk Revit Architecture 2014

Autodesk Revit Architecture 2014

Autodesk Revit Architecture 2014 - 展開モジュール

Autodesk Revit Architecture 2014 - 展開モジュール

Autodesk Revit Architecture 2014 - 展開モジュール

Autodesk Revit Architecture 2014 - 展開モジュール

Autodesk Revit Architecture 2014 - 展開モジュール

AUTODESK® REVIT® ARCHITECTURE 2014

AUTODESK.

配置を作成 > 配置環境設定の完了

選択した製品の配置環境設定が正常に完了しました。

展開モジュール

配置環境設定に含めた全製品をインストールするには、次の項目を使用:

\\TOKSGH025PXL1\Users\isezakt\Desktop\revit installer\展開モジュール.lnk

配置環境設定を修正するには、次の項目を使用:

\\TOKSGH025PXL1\Users\isezakt\Desktop\revit installer\Tools\配置を作成または修正.lnk

すべての製品を登録 下記のリンクを使用して製品のオンライン登録を行ってください。インターネットに接続していない場合は、アクティベーションを行う際に登録を行うこともできます。

オンライン製品登録

インストレーション ヘルプ | 動作環境 | Readme

完了

AUTODESK.