CES-468549

Autodesk Civil 3D, Dynamo or the API. Why and When?

Phuc Le Hieu Hong (Ken)

Technical Specialist, Autodesk

Learning Objectives

- Learn how automation can helps us achieve more, better with less in Civil 3D
- Learn how to use Dynamo for Civil 3D to automate repetitive tasks
- Learn how to develop an Autodesk Civil 3D add-in with the .NET API
- Understand which solution is more suitable to do tasks in some typical cases

Description

Are there some tasks you want to automate for more precision and speed while working in AutoCad Civil 3D software? When should you flex Dynamo’s abilities or develop a custom Add-in with .NET API? If you have ever wondered when or why you should use one solution over the other, then this session will help clarify your choices. Some typical scenarios will be discussed, examining two options, and assessing their pros and cons. In this session, you will be inspired and equipped to bring automation to your AutoCAD Civil 3D models.

Speaker

Phuc Le is a Digital Consultant, BIM Advisor, BIM Application Expert & Forge Developer.

He currently serves as a Technical Specialist at Autodesk Asean, supporting firms and organizations in the AEC sector to successfully implement Building Information Modelling, Cloud Collaboration, Computational Design, and Generative Design.

Table of contents

Autodesk Civil 3D

1. Overview

Civil 3D is a civil infrastructure ***design and documentation*** software supports BIM (Building Information Modeling) with integrated features to accelerate design efficiencies, automate construction documentation, and improve project delivery.

Please visit https://www.autodesk.com/products/civil-3d/overview for more: top features and what's new.

2. Automation

Automation is happening. Actually, it's been happening for several centuries, which has been cause for both great rejoicing and great consternation. But it's happening faster now, as more and more of the tasks of our daily and professional lives are handled by machines. That means we'll have to change how we do things—and how we do our jobs. Automation will enable us to do more, do it better, and do it with less time and less effort.

With Automation in Civil 3D, users can increase productivity and enhance project outcomes.

Users can:

- Rapidly test, iterate, and study multiple design options in less time
- Solve complex geometric problems, no programming experience needed
- Automate repetitive tasks for more precision and speed
- Manage risk by exposing tradeoffs and understanding systems and connections at the conceptual phase
- Generate sophisticated designs from simple data, logic, and analysis

When you consider there are typically several ways to do automatic tasks in Civil 3D and that there are other options available such Macro, Dynamo (and Dynamo Player) and the .NET API, you need to consider which solution is the most suitable.

This session aims to review the pros and cons of working with two popular solutions that are integrated with Civil 3D: Dynamo and the .NET API.

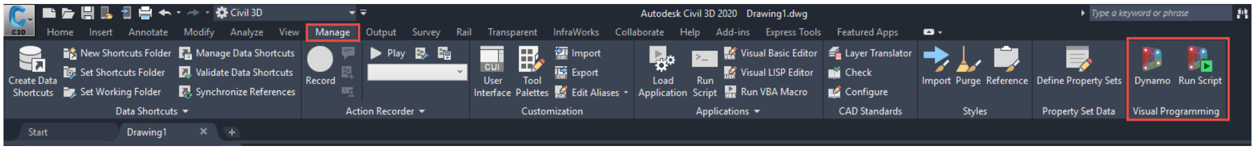
We'll briefly review each option against a common set of considerations:

- Description
- Intended Use
- Benefits & Limitations
- Build Skills & Time
- Flexibility
- Robustness
- Deployment

Dynamo for Civil 3D

1. What is Dynamo

Dynamo for Civil 3D is a visual programming application that can automate tasks in Civil 3D. This was introduced in Civil 3D 2020 as a separate add-on installation. It is an open sourced graphical tool for design and documentation.



2. Why Dynamo

Dynamo is intended to develop a wide range of creative workflows, from processing data to generating geometry.

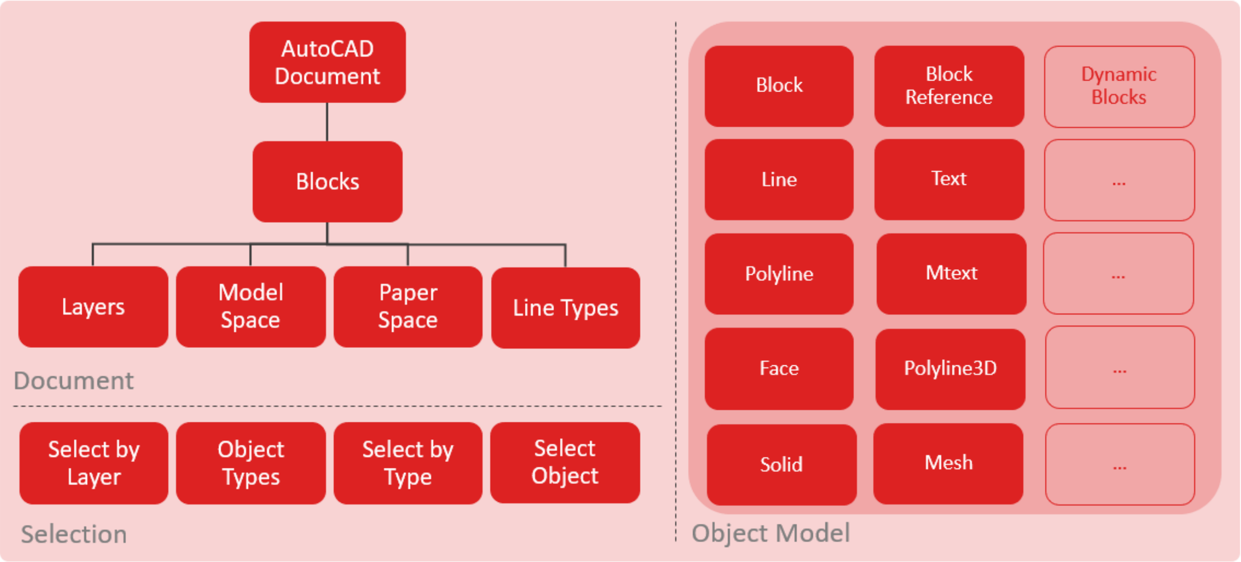
The primary benefit of Dynamo is that it opens up the power of the .NET API without the need to write code.

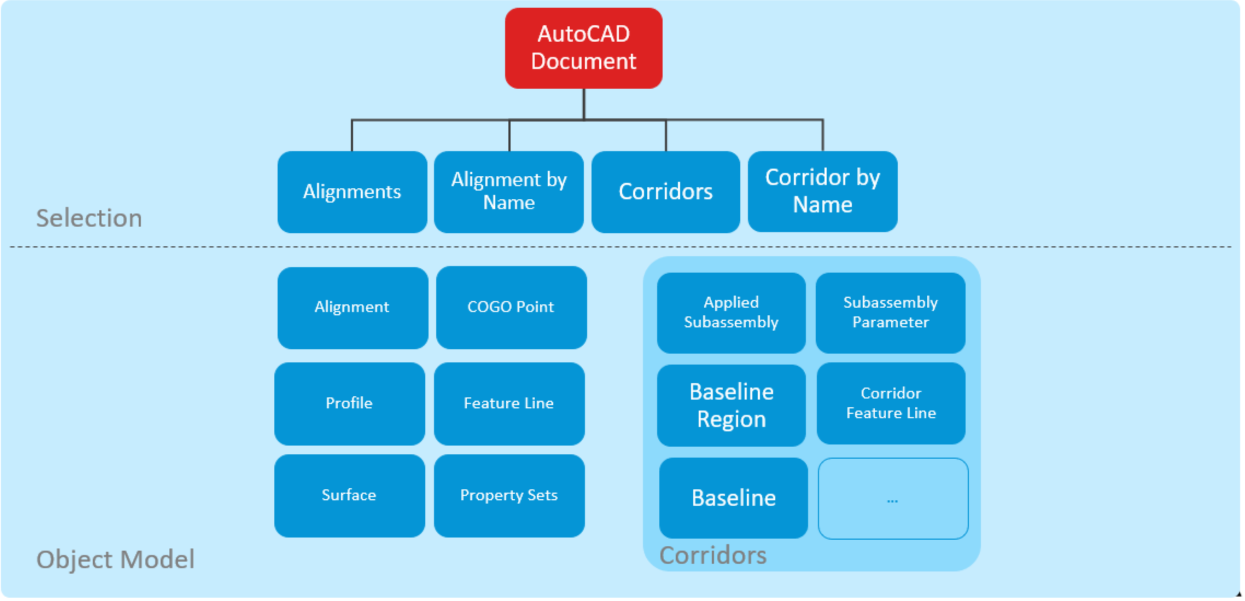
Another key benefit of Dynamo over the other solutions is that Dynamo scripts are flexible. We can be quickly edited to suit the varying needs of different projects.

Using Dynamo has both positives and negatives; lots of new features and fixes, but that means lots of updates to both Dynamo version and its packages which have to be managed.

Dynamo Player is a simpler way to interact with Dynamo's *Visual Programming* capabilities from within Civil 3D. Dynamo Player is intended to expose Dynamo's power to lower skilled users through a very simple interface. It is a way to share more powerful workflows with a wider group of users within your company.

The hierarchy of Dynamo for Civil 3D (included Dynamo for AutoCad)



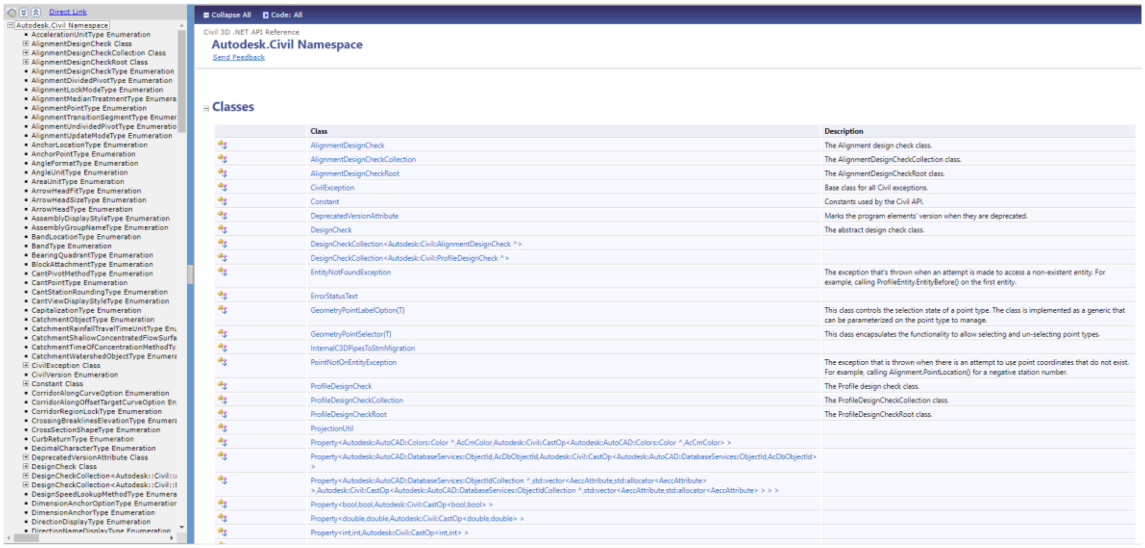


Learn more about Dynamo:
https://dynamobim.org/learn/
https:\\DynamoNodes.com
https:\\Dictionary.dynamobim.com

The .NET API

1. What is Civil 3D API

.NET API allows you to write plug-ins to AutoCAD® Civil 3D® in any .NET language. The Civil 3D.NET API Reference provides critical documentation for your development. This reference contains the exposed namespaces, classes, and members specific to the AutoCAD Civil 3D application. From here, you can understand what is available to plan your solution.



Direct Link

Autodesk.Civil Namespace
- AccelerationUnitType Enumeration
- AlignmentDesignCheck Class
- AlignmentDesignCheckCollection Class
- AlignmentDesignCheckRoot Class
- AlignmentDesignCheckType Enumeration
- AlignmentDividedPivotType Enumeration
- AlignmentLockModeType Enumeration
- AlignmentMedianTreatmentType Enumera
- AlignmentPointType Enumeration
- AlignmentTransitionSegmentType Enumer
- AlignmentUndividedPivotType Enumeratio
- AlignmentUpdateModeType Enumeration
- AnchorLocationType Enumeration
- AnchorPointType Enumeration
- AngleFormatType Enumeration
- AngleUnitType Enumeration
- AreaUnitType Enumeration
- ArrowHeadFitType Enumeration
- ArrowHeadSizeType Enumeration
- ArrowHeadType Enumeration
- AssemblyDisplayStyleType Enumeration
- AssemblyGroupNameType Enumeration
- BandLocationType Enumeration
- BandType Enumeration
- BearingQuadrantType Enumeration
- BlockAttachmentType Enumeration
- CantPivotMethodType Enumeration
- CantPointType Enumeration
- CantStationRoundingType Enumeration
- CantViewDisplayStyleType Enumeration
- CapitalizationType Enumeration
- CatchmentObjectType Enumeration
- CatchmentRainfallTravelTimeUnitType Enu
- CatchmentShallowConcentratedFlowSurfa
- CatchmentTimeOfConcentrationMethodTy
- CatchmentWatershedObjectType Enumera
- CivilException Class
- CivilVersion Enumeration
- Constant Class
- CorridorAlongCurveOption Enumeration
- CorridorAlongOffsetTargetCurveOption En
- CorridorRegionLockType Enumeration
- CrossingBreaklinesElevationType Enumera
- CrossSectionShapeType Enumeration
- CurbReturnType Enumeration
- DecimalCharacterType Enumeration
- DeprecatedVersionAttribute Class
- DesignCheck Class
- DesignCheckCollection<Autodesk::Civil::
- DesignCheckCollection<Autodesk::Civil::
- DesignSpeedLookupMethodType Enumera
- DimensionAnchorOptionType Enumeration
- DimensionAnchorType Enumeration
- DirectionDisplayType Enumeration
- DirectionNameDisplayType Enumeration

Civil 3D .NET API Reference

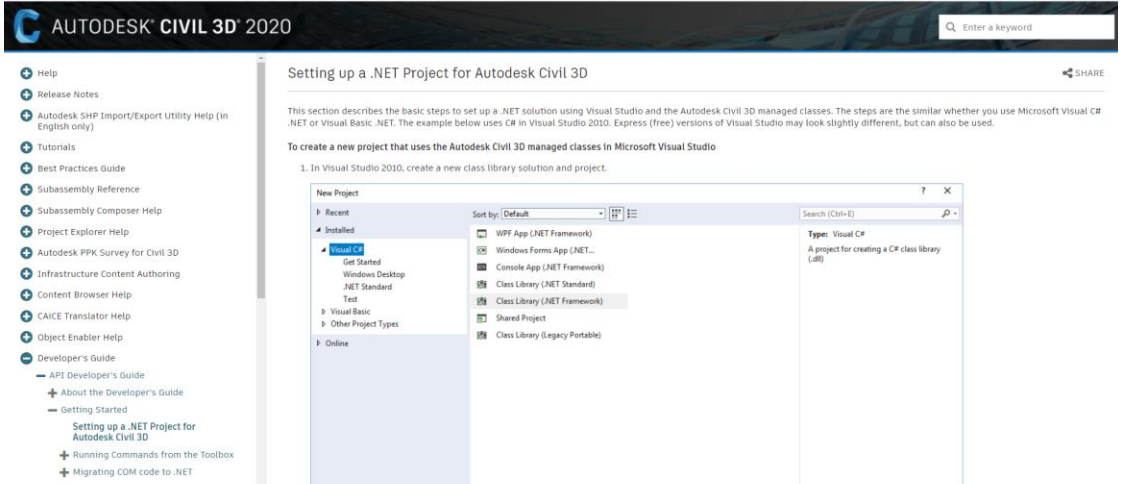
Autodesk.Civil Namespace

Send Feedback

Classes

Class	Description
AlignmentDesignCheck	The Alignment design check class.
AlignmentDesignCheckCollection	The AlignmentDesignCheckCollection class.
AlignmentDesignCheckRoot	The AlignmentDesignCheckRoot class.
CivilException	Base class for all Civil exceptions.
Constant	Constants used by the Civil API.
DeprecatedVersionAttribute	Marks the program elements' version when they are deprecated.
DesignCheck	The abstract design check class.
DesignCheckCollection<Autodesk::Civil::AlignmentDesignCheck ^>	
DesignCheckCollection<Autodesk::Civil::ProfileDesignCheck ^>	
EntityNotFoundException	The exception that's thrown when an attempt is made to access a non-existent entity. For example, calling ProfileEntity.EntityBefore() on the first entity.
ErrorStatusText	
GeometryPointLabelOption(T)	This class controls the selection state of a point type. The class is implemented as a generic that can be parameterized on the point type to manage.
GeometryPointSelector(T)	This class encapsulates the functionality to allow selecting and un-selecting point types.
InternalC3DPipesToStmMigration	
PointNotOnEntityException	The exception that is thrown when there is an attempt to use point coordinates that do not exist. For example, calling Alignment.PointLocation() for a negative station number.
ProfileDesignCheck	The Profile design check class.
ProfileDesignCheckCollection	The ProfileDesignCheckCollection class.
ProfileDesignCheckRoot	The ProfileDesignCheckRoot class.
ProjectionUtil	
Property<Autodesk::AutoCAD::Colors::Color ^,AcCmColor,Autodesk::Civil::CastOp<Autodesk::AutoCAD::Colors::Color ^,AcCmColor> >	
Property<Autodesk::AutoCAD::DatabaseServices::ObjectId,AcDbObjectId,Autodesk::Civil::CastOp<Autodesk::AutoCAD::DatabaseServices::ObjectId,AcDbObjectId> >	
Property<Autodesk::AutoCAD::DatabaseServices::ObjectIdCollection ^,std::vector<AeccAttribute,std::allocator<AeccAttribute> >,Autodesk::Civil::CastOp<Autodesk::AutoCAD::DatabaseServices::ObjectIdCollection ^,std::vector<AeccAttribute,std::allocator<AeccAttribute> > > >	
Property<bool,bool,Autodesk::Civil::CastOp<bool,bool> >	
Property<double,double,Autodesk::Civil::CastOp<double,double> >	
Property<int,int,Autodesk::Civil::CastOp<int,int> >	

In addition, the Autodesk Civil 3D API Developer's Guide contains helpful elaboration on application-specific concepts including how to get started. Microsoft Visual Studio is a powerful (and suggested) environment in which to develop your ideas. This is official Autodesk help material to make sure you have the right tools for the job.



- Help
- Release Notes
- Autodesk SHP Import/Export Utility Help (in English only)
- Tutorials
- Best Practices Guide
- Subassembly Reference
- Subassembly Composer Help
- Project Explorer Help
- Autodesk PPK Survey for Civil 3D
- Infrastructure Content Authoring
- Content Browser Help
- CAiCE Translator Help
- Object Enabler Help
- Developer's Guide
  - API Developer's Guide
    - About the Developer's Guide
    - Getting Started
      - Setting up a .NET Project for Autodesk Civil 3D
      - Running Commands from the Toolbox
      - Migrating COM code to .NET

Setting up a .NET Project for Autodesk Civil 3D

This section describes the basic steps to set up a .NET solution using Visual Studio and the Autodesk Civil 3D managed classes. The steps are the similar whether you use Microsoft Visual C# .NET or Visual Basic .NET. The example below uses C# in Visual Studio 2010. Express (free) versions of Visual Studio may look slightly different, but can also be used.

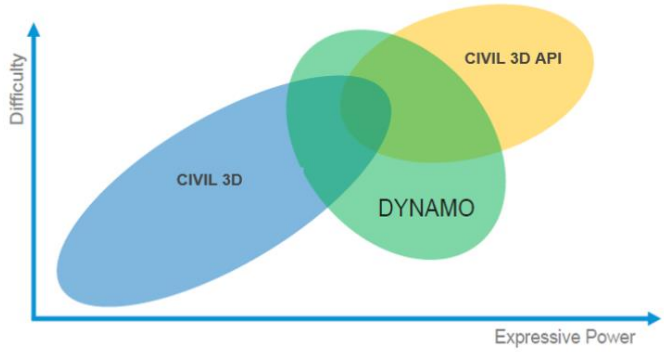
To create a new project that uses the Autodesk Civil 3D managed classes in Microsoft Visual Studio

1. In Visual Studio 2010, create a new class library solution and project.

Type: Visual C#

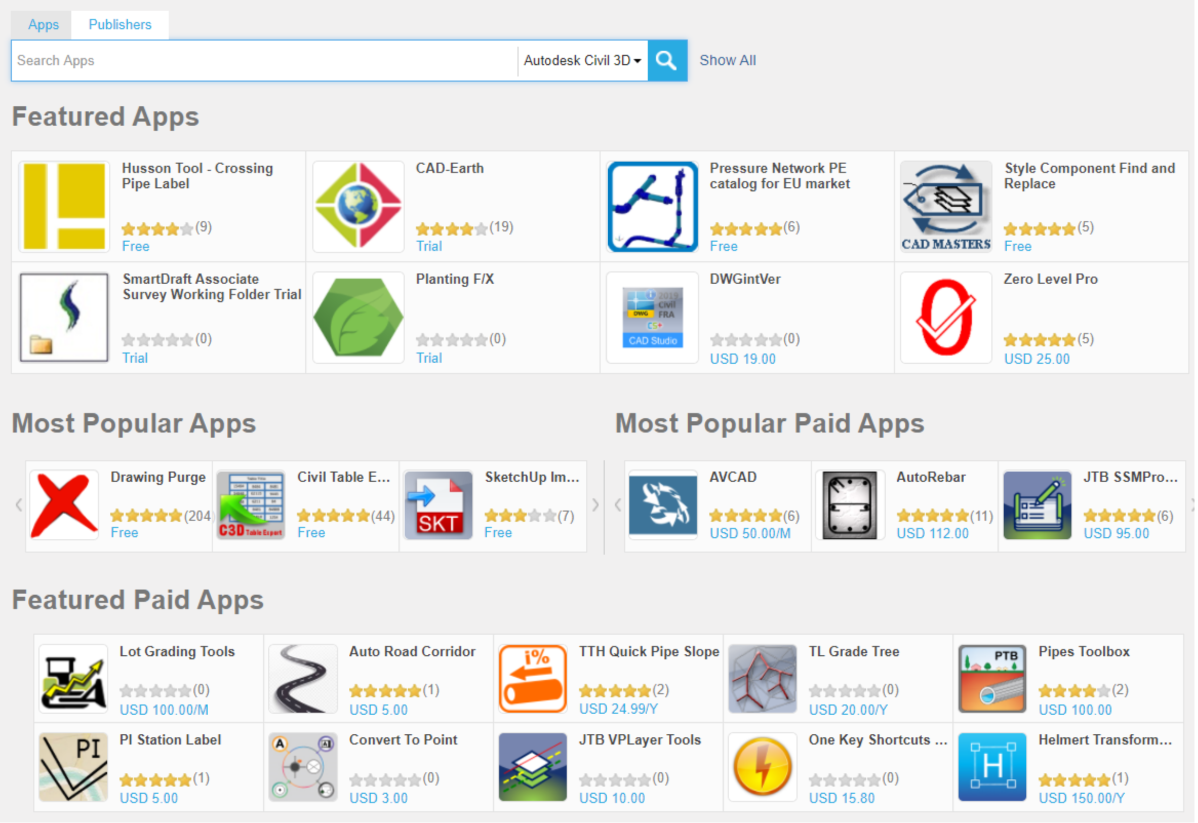
A project for creating a C# class library (.dll)

2. Why .NET API



The API can be used to create custom tools and features that plug directly into Autodesk Civil 3D, extending its functionality.
You can automate repetitive, time-consuming tasks and extend core features without leaving the Civil 3D environment. These add-ins can typically be accessed in Civil 3D from the **Add-Ins** tab of the ribbon.

You can find a lot of Civil 3D add-ins in Autodesk App Store



Featured Apps

Husson Tool - Crossing Pipe Label (9) Free
CAD-Earth (19) Trial
Pressure Network PE catalog for EU market (6) Free
Style Component Find and Replace (5) Free
SmartDraft Associate Survey Working Folder Trial (0) Trial
Planting F/X (0) Trial
DWGintVer (0) USD 19.00
Zero Level Pro (5) USD 25.00

Most Popular Apps

Drawing Purge (204 Free
Civil Table E... (44) Free
SketchUp Im... (7) Free

Most Popular Paid Apps

AVCAD (6) USD 50.00/M
AutoRebar (11) USD 112.00
JTB SSMPro... (6) USD 95.00

Featured Paid Apps

Lot Grading Tools (0) USD 100.00/M
Auto Road Corridor (1) USD 5.00
TTH Quick Pipe Slope (2) USD 24.99/Y
TL Grade Tree (0) USD 20.00/Y
Pipes Toolbox (2) USD 100.00
PI Station Label (1) USD 5.00
Convert To Point (0) USD 3.00
JTB VPLayer Tools (0) USD 10.00
One Key Shortcuts ... (0) USD 15.80
Helmert Transform... (1) USD 150.00/Y

One of the limitations is that the API isn’t as flexible as Dynamo.

And the API requires a higher level of coding with most users typically working in Visual Studio. Either C#, VB.Net, or other .NET compliant languages need to be learned to successfully use the API.

Examples

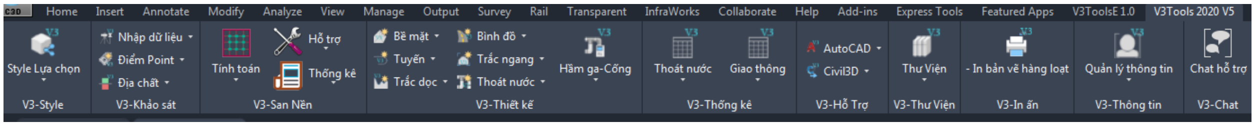
1. Dynamo for Civil 3D

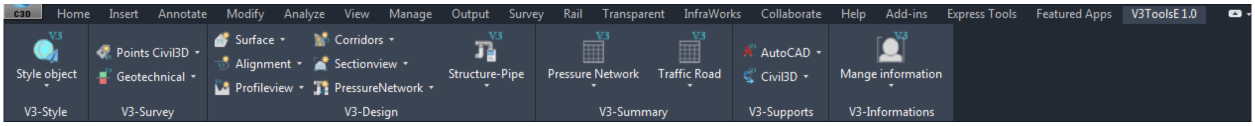
- Road Design
- Site Design
- Rail Design
- Integration
- Civil 3D Toolkit

…

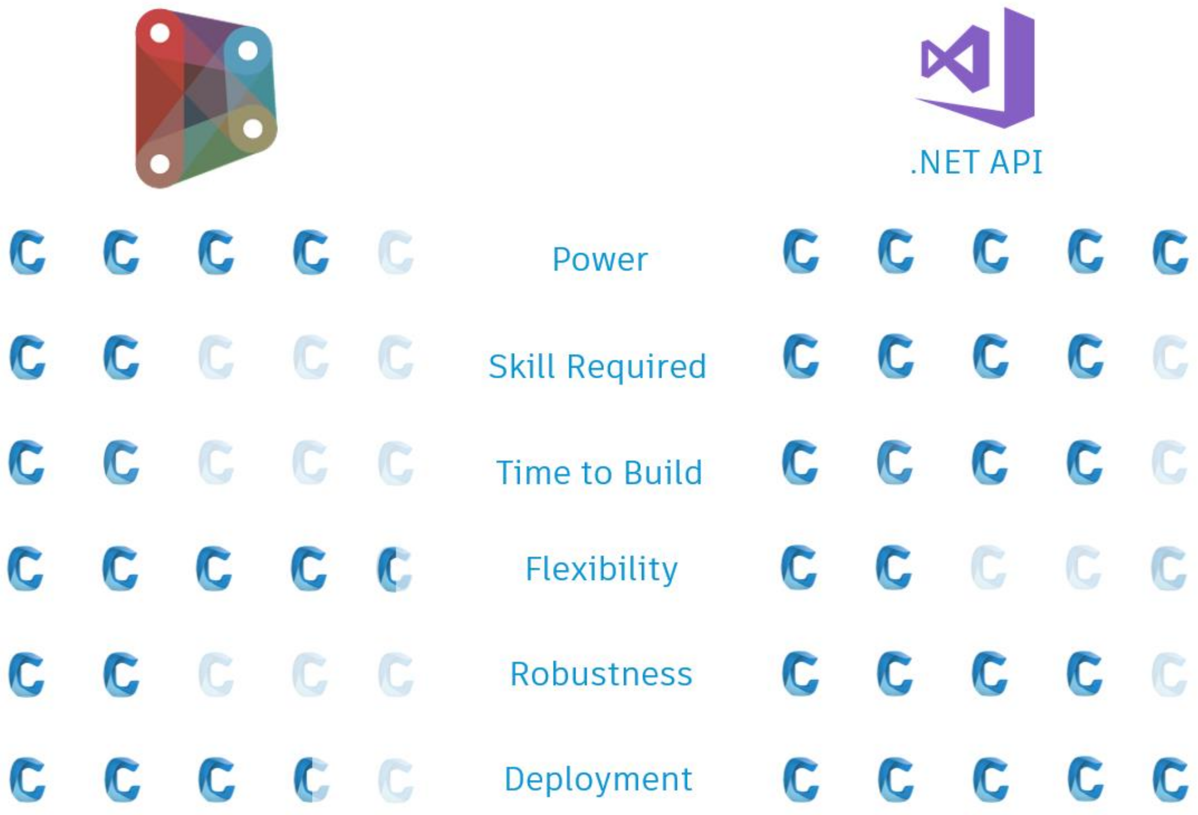
2. .NET API

V3 Tools by Trung Truong





Conclusion



The goals and complexity of your task, along with many of the factors discussed will help you decide which option (Dynamo or .NET API) is most suitable for your team and situation.

If one solution can do what you want, it doesn't mean it is the best solution, it is just a suitable solution for you at that time.

"Begin with the end in mind" is a good way to let you know what you really want.

And don't forget to **KEEP LEARNING** to improve knowledge and skills.

Let's extend Civil 3D's capability with the power of Dynamo and the API!