SD473692

Design Automation for Revit: Basics and beyond

Rahul Bhobe
Senior Principal Engineer
Autodesk Inc

Learning Objectives

- Learn how to automate desktop processes to run in the cloud.
- Learn how to develop web applications that utilize the Design Automation for Revit service.
- Learn from examples how to accelerate Forge development.
- Learn how to generate conceptual designs using adaptive components.

Description

At Autodesk, we value the need for our customers to bring their work to the cloud for collaboration and automation at scale. If you're a Revit API developer and would like to use Forge Design Automation for Revit software, this class will cover the basics and dive deeper.

We will explore five different case studies and all the steps necessary to implement them. You will learn to use the power of automation and cloud computing to generate conceptual designs using adaptive components. You will see demos and discover the code needed to implement them. You will learn how to manage your activities and AppBundles to support multiple Revit versions. You will get the links to the speaker’s end-to-end working code samples for you to extend upon. You will learn about Design Automation and how to integrate it with other Forge services, and about new features that have been added since last Autodesk University. You will walk away with solid insight into how to build cloud applications supporting Revit workflows.

Speaker(s)

Rahul Bhobe is a Senior Principal Software Developer for Revit at Autodesk. He holds a B. Tech degree in Naval Architecture from IIT Madras and a masters degree in Software Systems. He has been with Autodesk for 13 years working on several features of Revit. Currently he is working on developing Design Automation for Revit. In 2008, he started working on Revit Conceptual Modeling design where he implemented Adaptive Components, Point Elements and Divided Surface. He has then worked on several Revit features like Family, Groups, Links and Worksharing. He has a total of 21 years of CAD/CAM development experience.

The speaker has experience in teaching Design Automation to customers over emails, slack, workshops, Forge Accelerators and also the morder forums such as Stack Overflow.

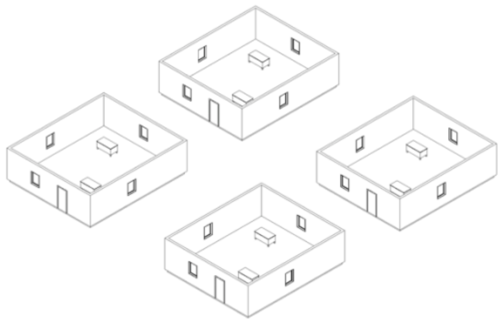
Class Overview

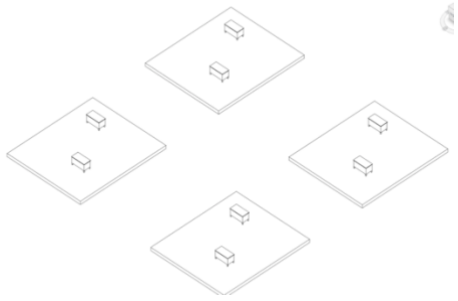
The class will start with high level overview of Design Automation for Revit and recap topics from previous years held by Autodesk and external speakers. Here are some links to prior classes by Autodesk speakers:
SD323658: Getting Started on Design Automation for Revit on Forge
SD125457: Design Automation: Building Web Applications with Revit Data on Forge

The class will then cover technical details of Design Automation application using of tutorial samples. We shall start with basics and gradually dive deeper to understand various options available within the Design Automation service. Currently Design Automation application do not have access to the internet from within the appbundle (aka addin). This class will show how to provide the the data it needs to generate results and upload them.

Delete Walls

This is one of our basic code samples, an “Hello World” application. In this section we will visit details about activity definition to see how to build an application that takes a Revit file as input to modify it and upload it to a cloud location. This section will cover details like command line inputs that run on cloud machine and variable expansion it uses internally. It will show how to setup a relationship between your appbundle and activity definition, wrt filenames that design automation service is expected to download and upload to.





We shall use basic input data sets to understand the workflow. The data sets and source code is provided in the additional materials section. Delete walls sample is used in many of our Forge tutorials including Postman collections and language neutral documentation such as curl based REST API documentation.

Count It

This section we will take another basic sample to understand what goes into implementing an application that takes a Revit file as input and extract information from it. This will also teach how to pass in parameters to the job, in a way that your application conditionally react to those parameters. You will learn to use Newtonsoft, a *nuget* library to serialize and deserialize your data to pass over the network and post it in a workitem argument.

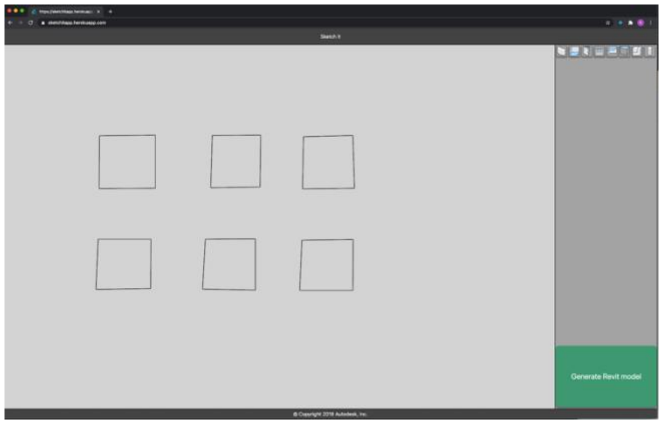
```
{
  "walls": 16,
  "floors": 4,
  "doors": 4,
  "windows": 16,
  "total": 40
}
```

This application is simple to understand and uses very basic Revit API routines to count number of elements of an input category list. You will learn about how to provide inputs and generate outputs that are both not Revit files.

This is a great start for anyone looking to build data extraction from Revit models. It should be as easy as use same or similar activity definitions and replace the appbundle with your complicated workflows.

Sketch It

Sketch It application was one of the early simple to understand end to end web application. This application allows a user to draw walls and floors on a web browser and generate Revit file with walls and floors corresponding to the lines drawn on the browser. This application, behind the scenes sends the corrdinate locations of points/lines drawn on the browser to Design Automation, and generate viewables so the user can view the results on the browser.



In this class we shall learn and understand how to implement Design Automation activity, appbundle and workitem necessary to build this application. While building on what we learnt about json serialization in previous section, this section will cover more complicated and large json input workflows. Unlike previous sections this section covers a Revit file creation workflow, since there is no input file to start with in this example.

Application: https://sketchitapp.herokuapp.com/
Source code: https://github.com/Autodesk-Forge/forge-sketchit-revit

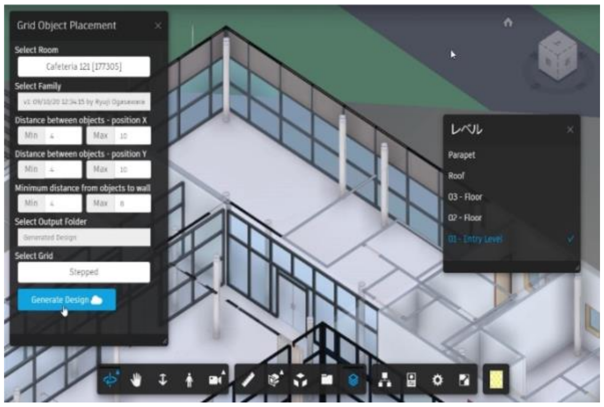
Stadium Panels

This section will teach how to combine the power of Revit API and Design Automation to generate conceptual design using Adaptive Components and Family Parameters. Adaptive components are PointElement based families, that allow you to parametrize your families with Adaptive Point elements. Like Family Instance parameters, Adaptive point elements can be modified at the Family instance level to generate flexible geometry and reuse family definition.

It is often tough to visualize the design upfront. Design automation for Revit offers a platform to automate generation of large number conceptual design files, and then see them all in a browser before deciding to chose one or tweak it further. In this section, I shall use the beauty in randomness to select a panel family and randomly vary its instance parameter to generate a complicated and beautiful structure. It is fascinating to see, how we start from so little: 2 family files, less than 200 lines of code and end up with a complicated and yet beautiful structure.

Grid Object Placement

In this section, I will show how to use Genetic Algorithm to generate multiple DWG files and upload them to BIM 360 as a single zipped file. A stepped or rectangular grid pattern will be created using parametric constraints as inputs for minimum and maximum distance between furniture families.



Grid Object Placement

Select Room: Cafeteria 121 [177305]

Select Family: v1 09/10/20 12:34:15 by Ryuji Ogasawara

Distance between objects - position X: Min 4, Max 10

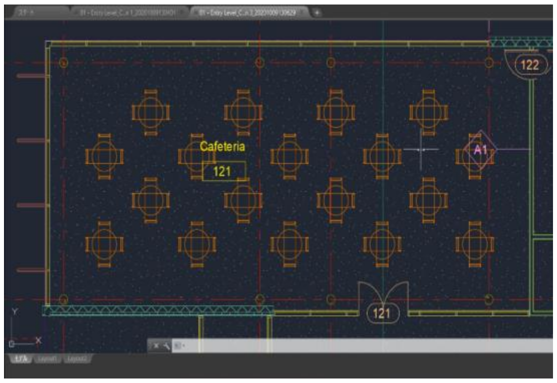
Distance between objects - position Y: Min 4, Max 10

Minimum distance from objects to wall: Min 4, Max 8

Select Output Folder: Generated Design

Select Grid: Stepped

Generate Design



In an iterative process each “generation” will export and save its best fit layout DWG that has created most number of furniture instances using the given constraints. A full end to end working sample that integrates with BIM 360 will be shared for you to extend upon

Class References

Source code: https://github.com/rahulbhobe/autodesk-university-2020-sd473692