MFG225552

Automating Fusion 360 with the API

Patrick Rainsberry
Autodesk

Learning Objectives

Get started with the Fusion 360 API
Learn how to write a basic script
Learn how to create a custom add-in or feature
Learn how to find necessary information and troubleshoot a Fusion 360 add-in

Description

The Fusion 360 platform is an extremely powerful tool for design and manufacturing, and with the inclusion of an easy-to-use API, it can become even more powerful to automate user workflows. This class will cover a basic overview of the Fusion 360 API and how to get started. We will introduce students to a simplified method of creating and distributing Fusion 360 scripts and add-ins. Focusing on actual customer problems and sample applications, students will learn how to quickly write simple python scripts to automate tasks and create useful utilities and workflow enhancements.

Speaker

I am a mechanical engineer. I have an undergrad degree from UC Berkeley and a Masters from UCLA. I also have an MBA from the University of La Verne. I have been working in the CAD industry for 15 years as well as some time spent as a design engineer.   \n \nMy responsibilities here at Autodesk are to grow the community of partners that are engaged with Fusion 360. I am interested in discussing all sorts of opportunities for partner applications, software integrations, hardware and software.

Get started with the Fusion 360 API

The Fusion 360 desktop client is extensible through the development of 3rd party add ins. Currently the API supports Python and C++.

Here are several useful resources:

Download Fusion 360:
http://www.autodesk.com/products/fusion-360/overview

Fusion 360 API Documentation:
http://help.autodesk.com/view/NINVFUS/ENU/?guid=GUID-A92A4B10-3781-4925-94C6-47DA85A4F65A

Offline Documentation of the Fusion 360 API:
http://forums.autodesk.com/t5/api-and-scripts/fusion-360-api-reference-manual-for-offline-viewing/m-p/5832190

Fusion 360 GitHub page (with lots of samples):
https://github.com/AutodeskFusion360
http://autodeskfusion360.github.io/

Inserting models into Fusion from an external source such as a web page:
http://help.autodesk.com/view/NINVFUS/ENU/?guid=GUID-8A2C4ECD-7D82-4E56-AFE8-4FA72464AE66

Using the Palette UI component (Web Page) within Fusion 360:
http://help.autodesk.com/view/fusion360/ENU/?guid=GUID-6C0C8148-98D0-4DBC-A4EC-D8E03A8A3B5B

Information on Publishing to Autodesk App Store:
http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=24734968

App submission process overview:
http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=20149658

Python:
I recommend spending a little time getting familiar with the syntax of Python if you are not already familiar. It is such a great and simple language that makes putting together these addins extremely fast.

There is a pretty nice tutorial here that I personally used to get started:
https://www.codecademy.com/learn/python

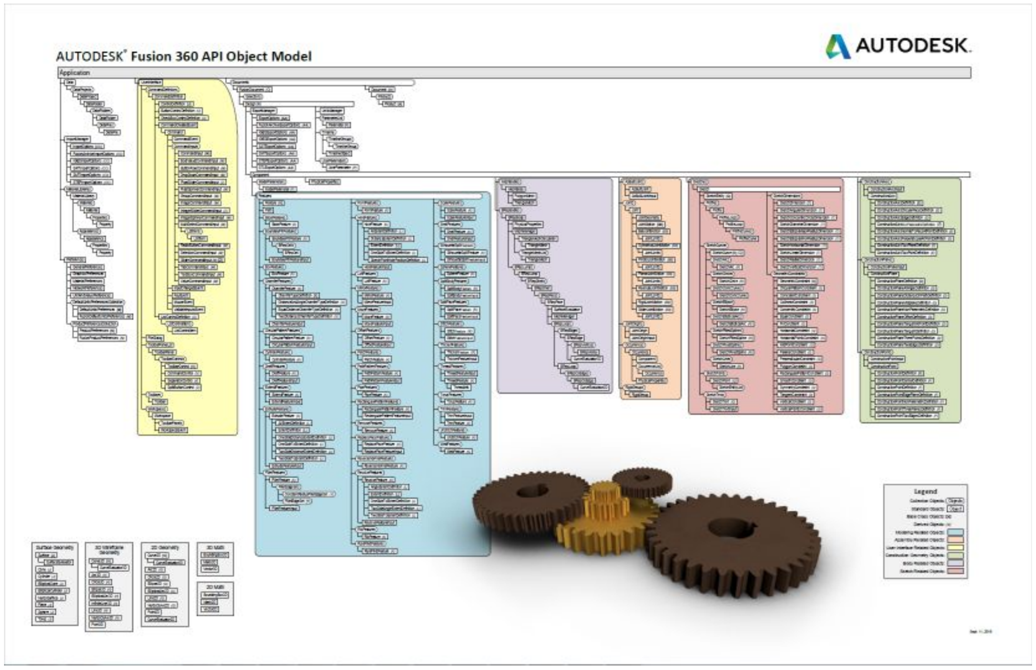
Learn how to write a basic script

Scripts are quick ways to simply run a small bit of code or automation. Next we will introduce add-ins that let you create more complex UI elements and commands.

Basic API Concepts and usage (*from* ***API Documentation***)

The Fusion 360 API is an object oriented API exposed through a set of objects. Many of these objects have a one-to-one correspondence with the things you are already familiar with as a Fusion 360 user. For example, an extrusion in a Fusion 360 model is represented in the API by the ExtrudeFeature object. Through functionality provided by the ExtrudeFeature object you can do the same things you can do through the user-interface. For example, you can create a new extrusion, get and set its name in the timeline, suppress it, delete it, or even access and edit the associated sketch.

One of the basic differences between using the Fusion 360 user interface and the API is how specific objects are accessed. With the user interface, you select (click on) things graphically in the browser,the timeline, and in the graphics window. New objects are created using dedicated commands such as 'Extrude' to create an extrusion or 'Box' to create a new box. With the API, objects are accessed through what is called an "Object Model". The Fusion 360 object model is a hierarchical structure of objects that is represented in the chart shown below. This chart is a useful tool when working with the API. You can download a printable pdf version of the chart here:



https://help.autodesk.com/cloudhelp/ENU/Fusion-360-API/images/Fusion.pdf

Script to create a block

```
import adsk.core, adsk.fusion, adsk.cam, traceback
def run(context):
    ui = None
    try:
        app = adsk.core.Application.get()
        ui  = app.userInterface
        design = app.activeProduct
        rootComp = design.rootComponent
        sketches = rootComp.sketches
        xyPlane = rootComp.xYConstructionPlane
        sketch = sketches.add(xyPlane)
        lines = sketch.sketchCurves.sketchLines
        point0 = adsk.core.Point3D.create(0, 0, 0)
        point1 = adsk.core.Point3D.create(0, 1, 0)
        point2 = adsk.core.Point3D.create(1, 1, 0)
        point3 = adsk.core.Point3D.create(1, 0, 0)
        lines.addByTwoPoints(point0, point1)
        lines.addByTwoPoints(point1, point2)
        lines.addByTwoPoints(point2, point3)
        lines.addByTwoPoints(point3, point0)
        profile = sketch.profiles.item(0)
        extrudes = rootComp.features.extrudeFeatures
        ext_input = extrudes.createInput(profile, adsk.fusion.FeatureOperations.NewBodyFeatureOperation)
        distance = adsk.core.ValueInput.createByReal(1)
        ext_input.setDistanceExtent(False, distance)
        ext_input.isSolid = True
        extrudes.add(ext_input)
    except:
        if ui:
            ui.messageBox('Failed:\n{}'.format(traceback.format_exc()))
```

Learn how to create a custom add-in or feature

To create add-ins, I create a starting template. This is a starting point for creating a Fusion 360 add-in. It simplifies the creation of Fusion 360 add-ins greatly. It hides much of the complexity of handling events and command registration from the user. Refer to the documentation for a more detailed and comprehensive discussion of creating add-ins from scratch.

Get the template and create your add-in directory

Download or clone this repo: https://github.com/tapnair/Fusion360AddinSkeleton

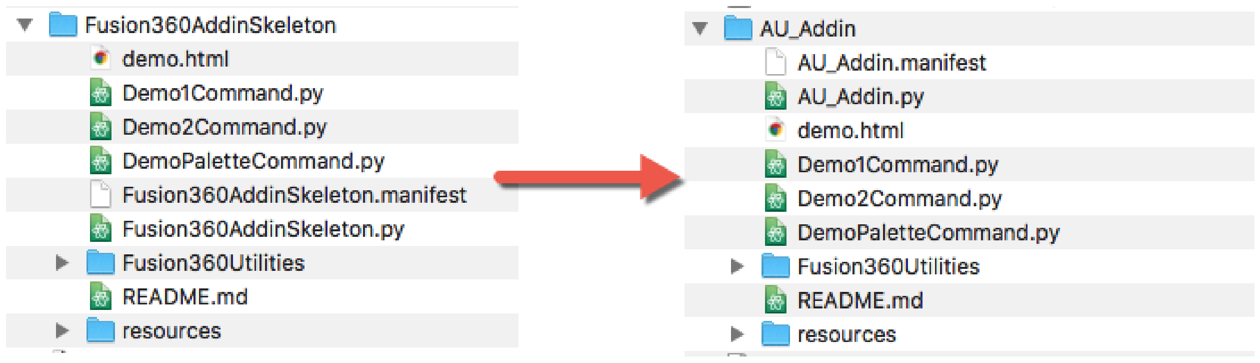
Move the folder into your add-ins directory. Look here for more information: https://tapnair.github.io/installation.html

Files in the Fusion360Utilities folder should not be modified.

Rename the following items to your desired addin name:

- The top level folder
- Fusion360AddinSkeleton.py
- Fusion360AddinSkeleton.manifest

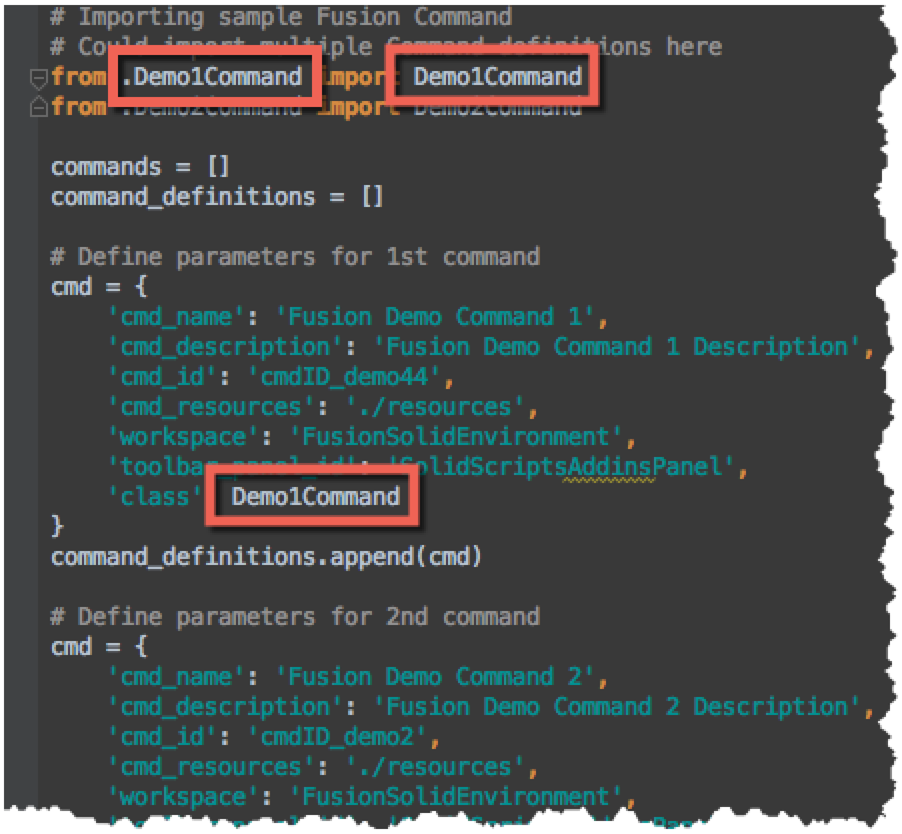
Edit the manifest file and update the fields accordingly



Creating an add-in: Step 1

- Open the newly renamed python file
- The current file will create two commands in the Fusion 360 UI in the Addins Drop Down
- Change the names and description strings here to your desired naming conventions

Currently each command relies on a separate file called Demo1Command.py and demo2Command.py. If you want to rename the files that define the names of the commands you must do it for each one in 3 places:



```python
# Importing sample Fusion Command
# Could import multiple Command definitions here
from .Demo1Command import Demo1Command
from .Demo2Command import Demo2Command

commands = []
command_definitions = []

# Define parameters for 1st command
cmd = {
    'cmd_name': 'Fusion Demo Command 1',
    'cmd_description': 'Fusion Demo Command 1 Description',
    'cmd_id': 'cmdID_demo44',
    'cmd_resources': './resources',
    'workspace': 'FusionSolidEnvironment',
    'toolbar_panel_id': 'SolidScriptsAddinsPanel',
    'class': Demo1Command
}
command_definitions.append(cmd)

# Define parameters for 2nd command
cmd = {
    'cmd_name': 'Fusion Demo Command 2',
    'cmd_description': 'Fusion Demo Command 2 Description',
    'cmd_id': 'cmdID_demo2',
    'cmd_resources': './resources',
    'workspace': 'FusionSolidEnvironment',
```

Creating an add-in: Step 2

- Edit Demo1Command.py and add functionality to the desired methods.
- onCreate: Build your UI components here
- onExecute: Will be executed when user selects OK in command dialog.
- DemoCommand1 creates a very basic UI and then accesses the input parameters.

Other useful components of Add-in Skeleton:

input_values

In the on_execute, on_preview, on_input_changed methods there is a parameter called "input_values"

This parameter is a dictionary containing the relevant values for all of the user inputs.

The key is the name of the input.

The value is dependant on the type input:

- Value type inputs will have their actual value stored (string or number depending)
- List type inputs (drop downs, etc) will have the name of the selected item as the value (string)
- Selection inputs regardless of whether they contain one or more selections will be returned as an array of the selected objects

Note: you can still access the raw command inputs object with the "inputs" variable. This would behave similar to any of the examples in the API documentation.

AppObjects

This is a helper class that can be used to easily access of many useful fusion 360 objects.

It contains many properties:

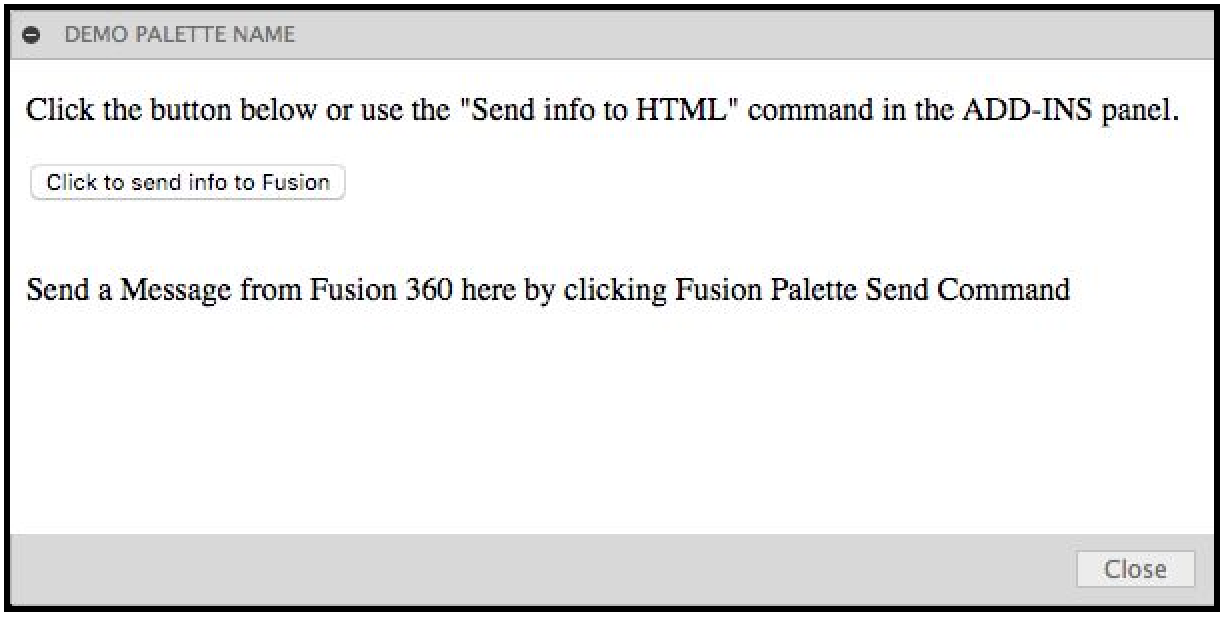
- app - Application Object
- document - Active Document
- product - Active Product
- design - Design Product (if it exists)
- cam - CAM Product (if it exists)
- ui - User Interface
- import_manager - Application Import Manager
- export_manager - Export Manager (if the active product is Design)
- units_manager - Fusion Units Manager (if the active product is design) or Units Manager
- root_comp - Root Component (if the active product is design)
- time_line - (if the active product is design and the type os Parametric Design Type)

Sample Usage:

```
from .Fusion360Utilities.Fusion360Utilities import AppObjects
ao = AppObjects()
ao.ui.messageBox('Hello World!')
```

HTML Palettes

A useful user interface component in the Fusion 360 API is a Palette. This gives you the ability to display a web page in Fusion 360 and interact with the application from it. You can send events from the Palette to Fusion and from Fusion to the web page.



Click the button below or use the "Send info to HTML" command in the ADD-INS panel.

Send a Message from Fusion 360 here by clicking Fusion Palette Send Command

To use this functionality open DemoPaletteCommand.py

There are three methods to work with:

- on_palette_execute: Run when the command executes and launches the Palette.
- on_html_event: Run every time an event is fired from the HTML Page
- on_palette_close: Run when the user closes the Palette

Learn how to find necessary information and troubleshoot an add-in

The best place to get help is the Fusion 360 forum. Otherwise I find an infinite resource in places like stack exchange. Most of the challenges I come across are really python questions more than anything.

Useful Links:

Forum to ask questions. (The API architects watch this forum constantly):
https://forums.autodesk.com/t5/api-and-scripts/bd-p/22

For more detaled information about editing and debugging your scripts and add-ins see the language specific topics (Python or C++) because the process is different depending on which programming language you're using.

Python Specific Issues

C++ Specific Issues

Samples:

My main page for these projects: https://tapnair.github.io/index.html

ventMaker - Create custom vent features in Fusion 360. Circular, Slot and rectangular vents.

HelixGenerator - Generate Helical Curves in Fusion 360

Dogbone - Create dog-bone fillets. Can create individual or automatically for entire assembly.

ParamEdit - Quick editor to make changes to user parameters with real time update.

stateSaver - Save the current state of: hide/show, suppress/unsuppress, and user parameter values.

ShowHidden - Display utilities for Fusion 360. Show hidden or all: bodies, components and planes.

Project-Archiver - Automate the export of all designs in a project to a local archive directory.

copyPaste - Copy and paste bodies between documents in Fusion 360, explicitly breaking references

NESTER - Semi automated nesting of sheet/flat parts in Fusion 360.

OctoFusion - Automate the process of exporting a file and sending it to Octoprint.

UGS_Fusion - Automate the process of posting a file and opening it in Universal G-code Sender