Introduction

What is CREOSON and Creopyson?

CRESON is an OpenSource initiative by Simplified Logic, Inc. to promote automation of PTC's CREO Parametric. CREOSON uses simple JSON Requests to send commands/functions to CREO, JSON Responses are used to communicate the results of your requests.

Creopyson is a python library that aims to control PTC's CREO Parametric via JLink via CREOSON. CREOSON uses JSON Requests to send commands/functions to CREO, JSON Responses are used to communicate the results of your requests. Creopyson creates a Client to send JSON Requests to CREOSON server.

Benefits of Creopyson over CREOSON

creopyson builds a wrapper for CREOSON functions so that any other user would not have to create their own. Basically to help other python users "get there faster" for Creo Automation using CREOSON. Specifically, load library, initialize, and start using - without having to first build the technical bits for the communication first in python to communicate with the CREOSON server.

CREOSON was developed to be a generic bi-directional service interface into Creo for performing optimized actions via JLINK (e.g. simpler -more obvious- commands, with more options via standard JSON transactions - from ANY language). The core premise of CREOSON was to pass a standard JSON structure over HTTP to a service endpoint - locally or over a network. Because CREOSON uses HTTP as the main interface for transacting requests, ANY language can be used to communicate with it (e.g. VB, Javascript, Java, Python, PHP, etc..) that supports HTTP transactions.

Some languages have HTTP interfaces baked-in, others use libraries to perform the actions as optional packages. It is typically up to the user to define the HTTP interface (simple send/receive function that communicates with the CREOSON server) and then write the function(s) for specific CREOSON requests. Typically users create their own function for say opening a model (e.g. myOpenModel(fileName);) that would take an input argument (like fileName) and build the CREOSON JSON request and execute it, then handing the response. This process would likely be repeated for each function they wanted to use in their application (build a wrapper function that builds and calls CREOSON and responds for their own purpose).

So - creopyson is an open source contribution (from someone in France) who loved CREOSON and wanted to make it even easier to use with python by pre-building all the interfaces in python to call CREOSON.

Getting Started

To get a local copy up and running follow these simple steps.

Prerequisites

- Install Python
- Install CREO Parametric Version 2.0 or Later... 3.0/4.0/5.0/6.0 have most coverage/stability.

Installation

Access to files

Clone the repository (private) to get access to the files, or use the OneDrive link

```
git clone
https://github.com/mrigankdoshy/routing-productivity-improvement.git
```

Creopyson

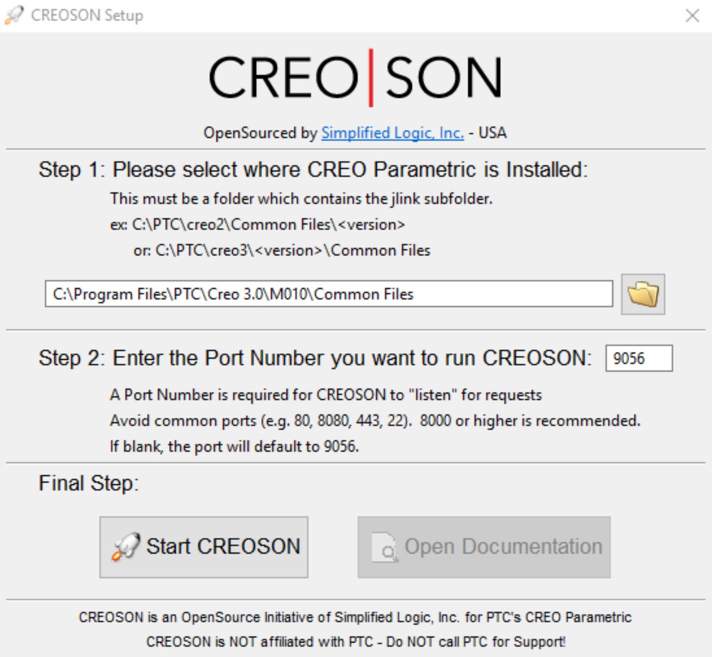
To install Creopyson, run this command in your terminal:

```
pip install creopyson
```

This is the preferred method to install Creopyson, as it will always install the most recent stable release. If you don’t have pip installed, this Python installation guide can guide you through the process.

CREOSON

- Download the latest release of Creoson Server for your system.
- Run CreosonSetup and configure it with your Creo's version.



CREOSON Setup

CREO|SON

OpenSourced by Simplified Logic, Inc. - USA

Step 1: Please select where CREO Parametric is Installed:

This must be a folder which contains the jlink subfolder.
ex: C:\PTC\creo2\Common Files\<version>
or: C:\PTC\creo3\<version>\Common Files

C:\Program Files\PTC\Creo 3.0\M010\Common Files

Step 2: Enter the Port Number you want to run CREOSON: 9056

A Port Number is required for CREOSON to "listen" for requests
Avoid common ports (e.g. 80, 8080, 443, 22). 8000 or higher is recommended.
If blank, the port will default to 9056.

Final Step:

CREOSON is an OpenSource Initiative of Simplified Logic, Inc. for PTC's CREO Parametric
CREOSON is NOT affiliated with PTC - Do NOT call PTC for Support!

- If you want to launch Creo with Creoson, please create a nitro_proe_remote.bat file. You can copy C:\Program Files\PTC\Creo x.x\Mxxx\Parametric\bin\parametric.bat and rename it nitro_proe_remote.bat anywhere you want.

Usage:

To use Creopyson in a project:

```
import creopyson
```

Create a Client object and create a connection with Creoson:

```
c = creopyson.Client()
c.connect()
```

Verify is Creo is running:

```
c.is_creo_running() # Return a boolean.
```

Launch Creo:

```
c.start_creo("path to nitro_proe_remote.bat")
```

Basic usage:

```
current_directory = c.creo_pwd()  # return current working directory.
listfiles = c.creo_list_files() # return a list in the working directory.
listdirs = c.creo_list_dirs() # return a list of folders in the working directory.
c.creo_cd("new_folder")  # change working directory.
c.file_exists("my_file.prt")  # verify if `my_file.prt` exists.
c.file_open("my_file.prt", display=True)  # Open `my_file.prt` in Creo.
c.dimension_set("my_file.prt", "diamm", 180)  # Modify `diamm` dimension.
c.file_regenerate("my_file.prt")  # Regenerate file, raise `Warning` if regeneration fails.
```

http://www.creoson.com/functions.html

«Vanilla» Creoson usage

Mostly for debugging:

```
import creopyson
c = creopyson.Client()
c.connect()

# Here you define command/function
# data is a dictionary with data part of the JSON request
# Please refer to Creoson documentation
command = "file"
function = "open"
data ={"file":"my_file.prt", "display": True}
result = c._creoson_post(command, function, data)
```

Result would be the data part of Creoson’s response:

```
{'dirname': 'C:/your/working/path/', 'files': ['my_file.prt'], 'revision': 1}
```

Logging basic usage

If you want see what are the requests to Creoson you should activate logging this way:

```
import logging
logging.basicConfig(level=logging.DEBUG)
# Hide urllib3 logging
logging.getLogger("urllib3").setLevel(logging.WARNING)

import creopyson

c = creopyson.Client()
c.connect()

c.file_open("my_file.prt", display=True)
```

The result in you console would be something like this:

```
DEBUG:creopyson.connection:request: {'sessionId': '', 'command':
'connection', 'function': 'connect', 'data': None}
DEBUG:creopyson.connection:response: {'status': {'error': False},
'sessionId': '-8685569143476874454'}
DEBUG:creopyson.connection:request: {'sessionId':
'-8685569143476874454', 'command': 'file', 'function': 'open',
'data': {'display': True, 'activate': True, 'file': 'my_file.prt'}}
DEBUG:creopyson.connection:response: {'status': {'error': False},
'data': {'revision': 1, 'files': ['MY_FILE.prt'], 'dirname':
'C:/your/working/path/'}}
```

Common Issues

[Insert issues here]

Documentation

Creopyson documentation at: https://creopyson.readthedocs.io/en/latest/ and GitHub Repo https://github.com/Zepmanbc/creopyson