



Published in CodeX

You have 1 free member-only story left to read



Frank Neugebauer

Follow

Apr 20, 2021 · 10 min read · ✨ · 🎧 Listen



Save



Sign in to Medium with Google



Pablo Maldonado

pablo.maldonado.lopez@gmail.com



Pablo Maldonado

pablo.maldonado@newton.university

Avoid RPA Vendor Lock

Combine UiPath and Python for Separation of Concerns



Photo by [Dimitry Anikin](#) on [Unsplash](#)

Robotic process automation (RPA) tools have come such a long way since “macros,”





The main idea is to very clearly separate UiPath and Python dimensions. The rest of this design, and you'll soon see why it's important.

Disclaimer #1

While I'm going to demonstrate this pattern using Python with UiPath, the same idea can be applied for other programming languages, especially on the .NET platform since UiPath's native programming language is .NET. It's also worth noting that there are other design patterns that can be used (e.g., using serverless functions for programming and connecting them to UiPath via REST services or directly with .NET).

Technology Setup

Before I show you how I built this, here's my technology setup:

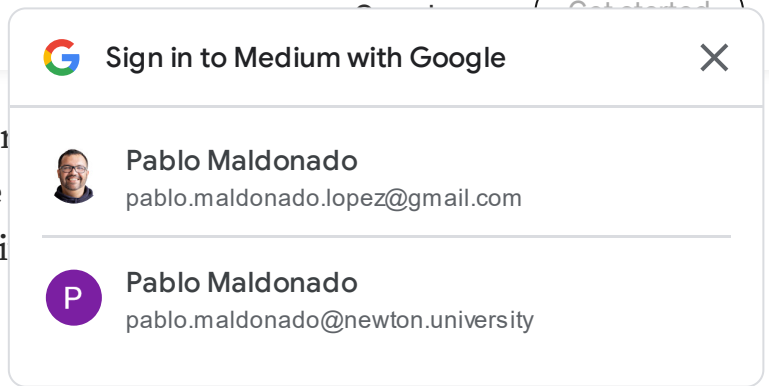
- UiPath Community Edition — [you can get this here](#). (Note the license agreement associated with UiPath Community Edition.)
- Python — I use the Anaconda Individual distribution, [which can be downloaded here](#).

The "Project"

Reading PDF tables seems like a very basic function and UiPath **can** manage it; there's a very robust PDF package to do so, which is one of UiPath's official packages. I'd go so far as to say that for complex PDFs it's one of the most reliable PDF readers I've used (and I've used many in Python). However, if I have to perform a lot of programming on the table prior to using the data in a UI, or after processing data from a UI, it's quite cumbersome to do with UiPath (UiPath is not a very good programming environment in my view).

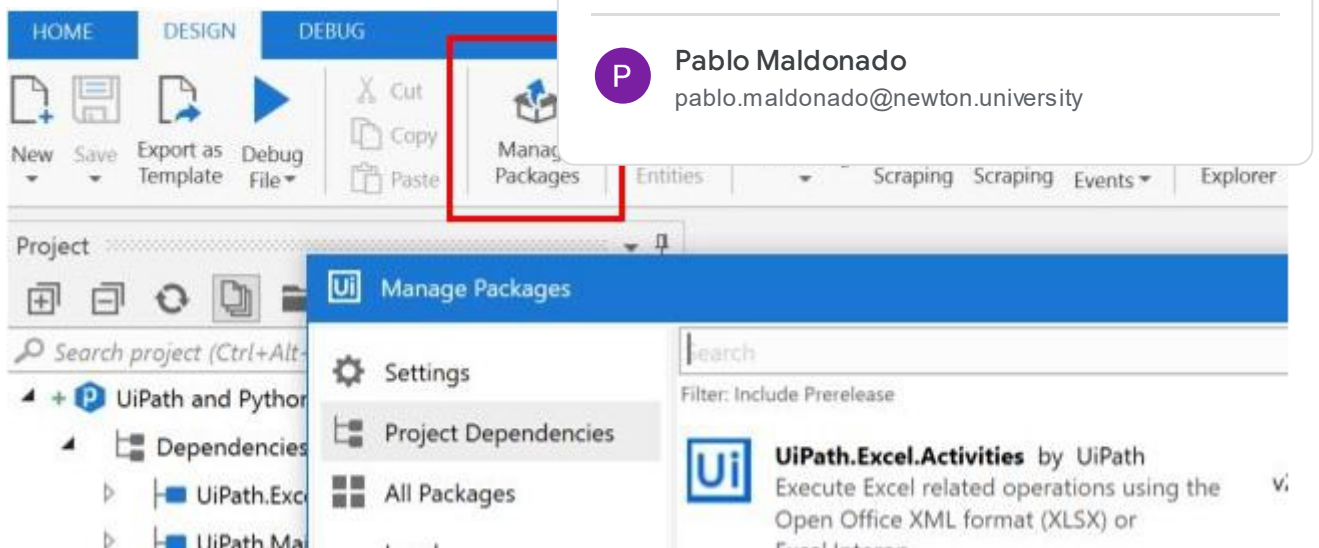
UiPath Setup Steps

The very first thing I did was create a new Process in UiPath and then I created a Sequence. (Please refer to the UiPath documentation if you've never created a UiPath project.)



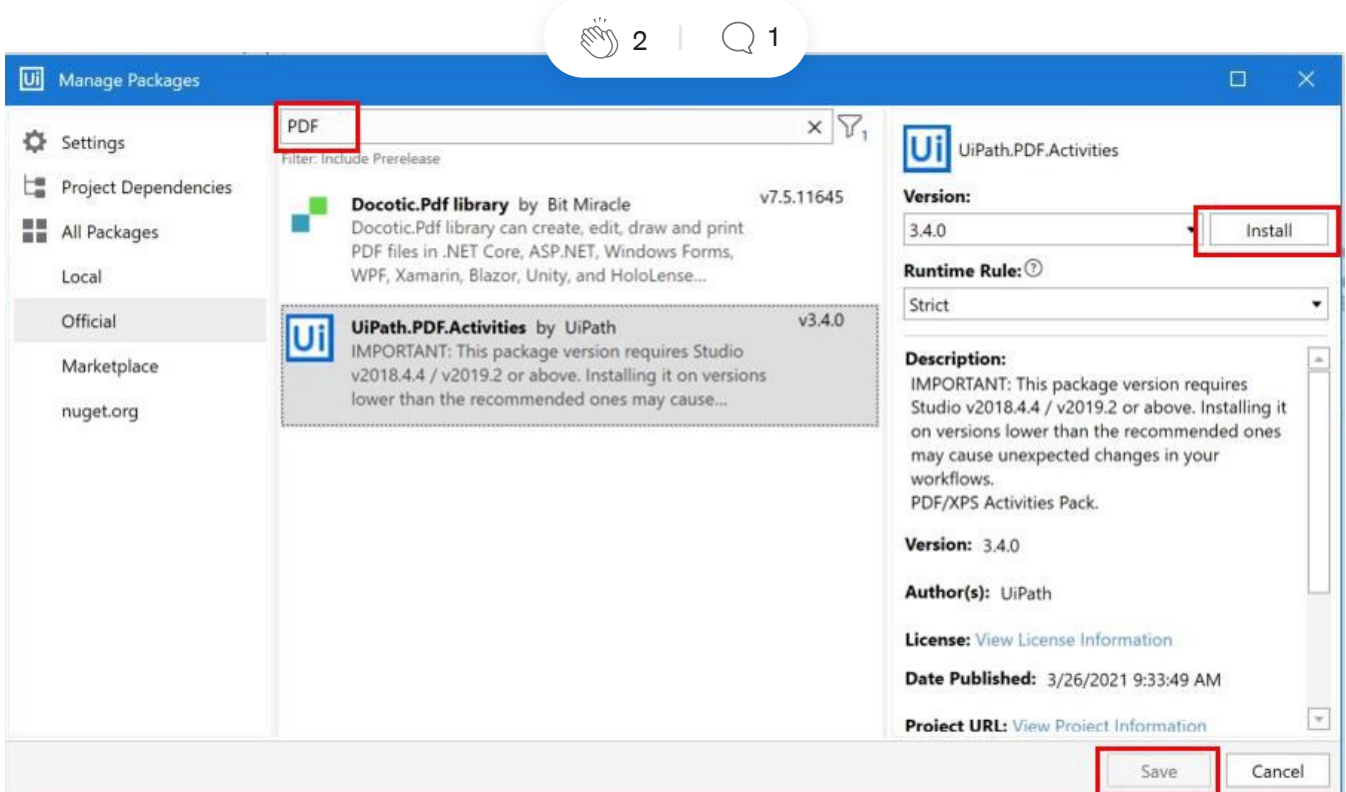


First, I launched Manage Packages.



Manage Packages in UiPath — Image by Author

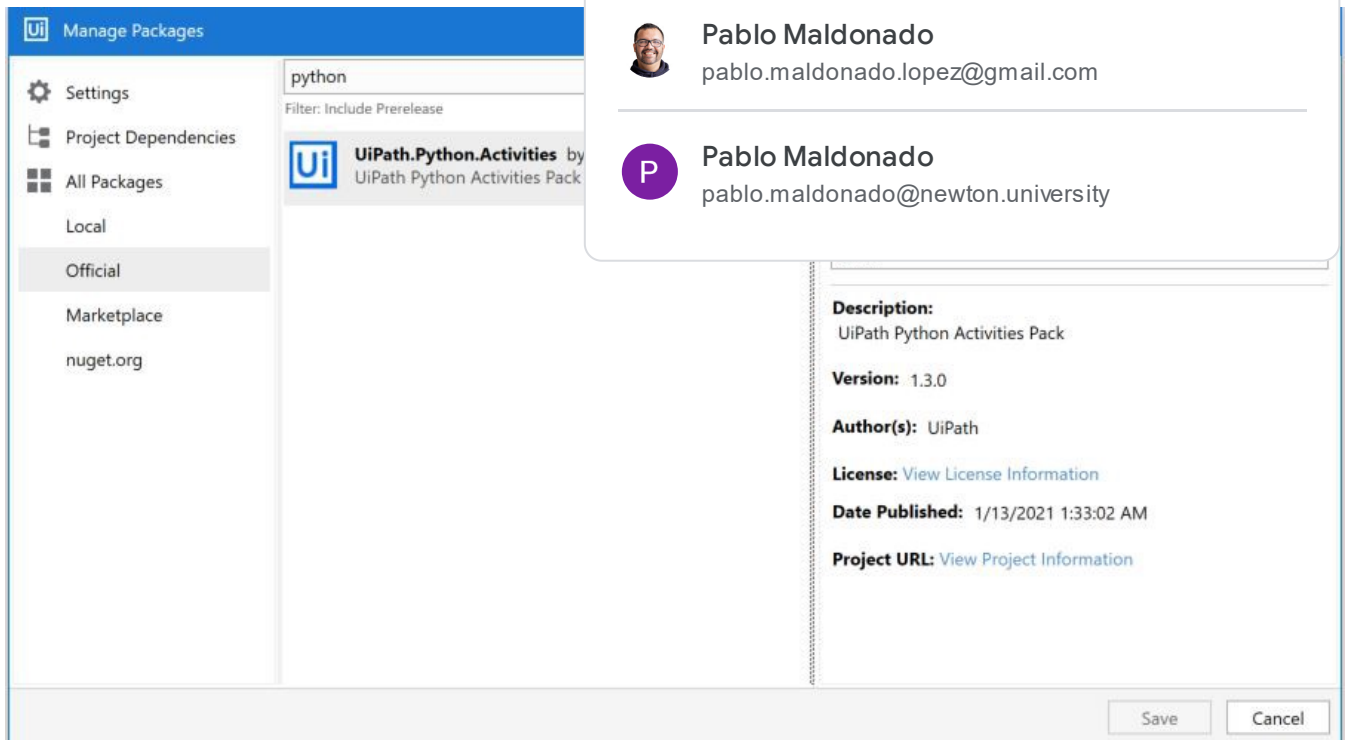
Then I searched for the UiPath.PDF.Activities package—I just searched for “PDF.”



Installing the PDF Package in UiPath — Image by Author

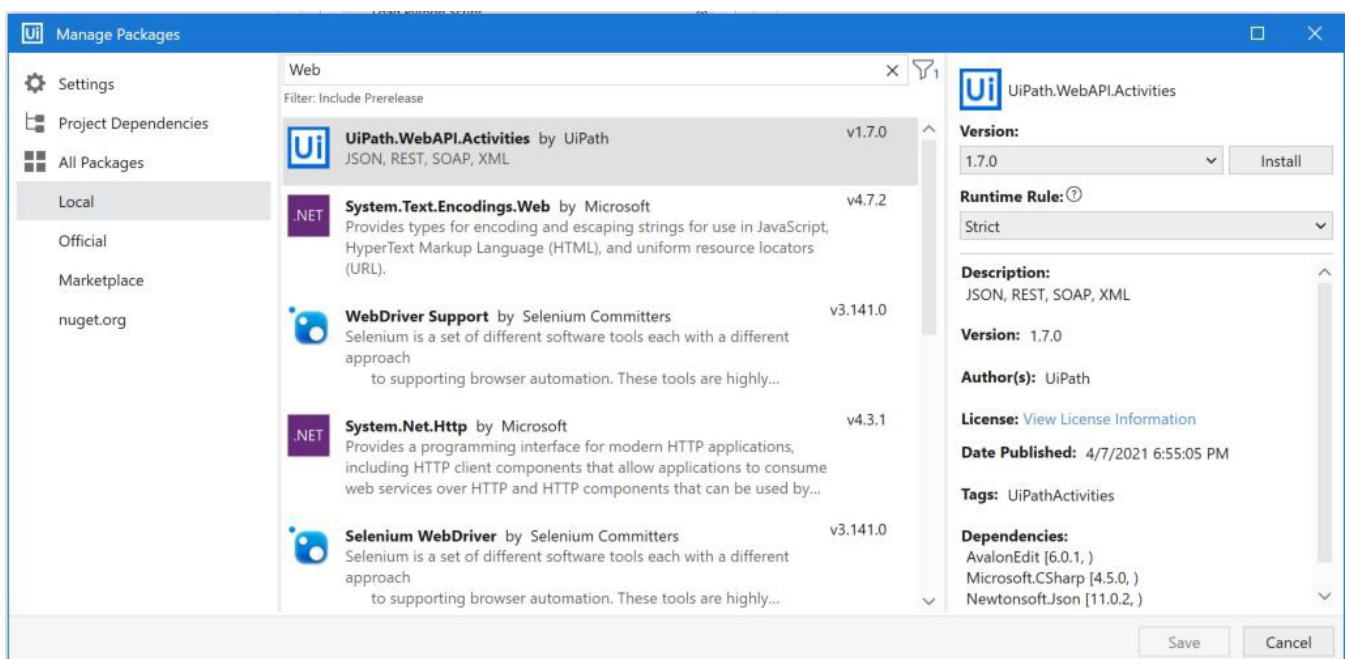
When I found the UiPath.PDF.Activities package, I selected “Install” and then “Save.”






Installing the Python Activities Package — Image by Author


Finally, I'm eventually going to work with JSON. To do that, I need to add the "Local" UiPath.WebAPI.Activities package. This is done the same way as the others, but instead of selecting "Official" on the far-left of the Manage Packages dialog, I selected "Local," then I searched for "Web."






Last Name	First Name
Smith	Albert
Williams	Gene
Jacobs	Sonia

 Sign in to Medium with Google ✕



Pablo Maldonado
pablo.maldonado.lopez@gmail.com



Pablo Maldonado
pablo.maldonado@newton.university

The Data — Image by Author

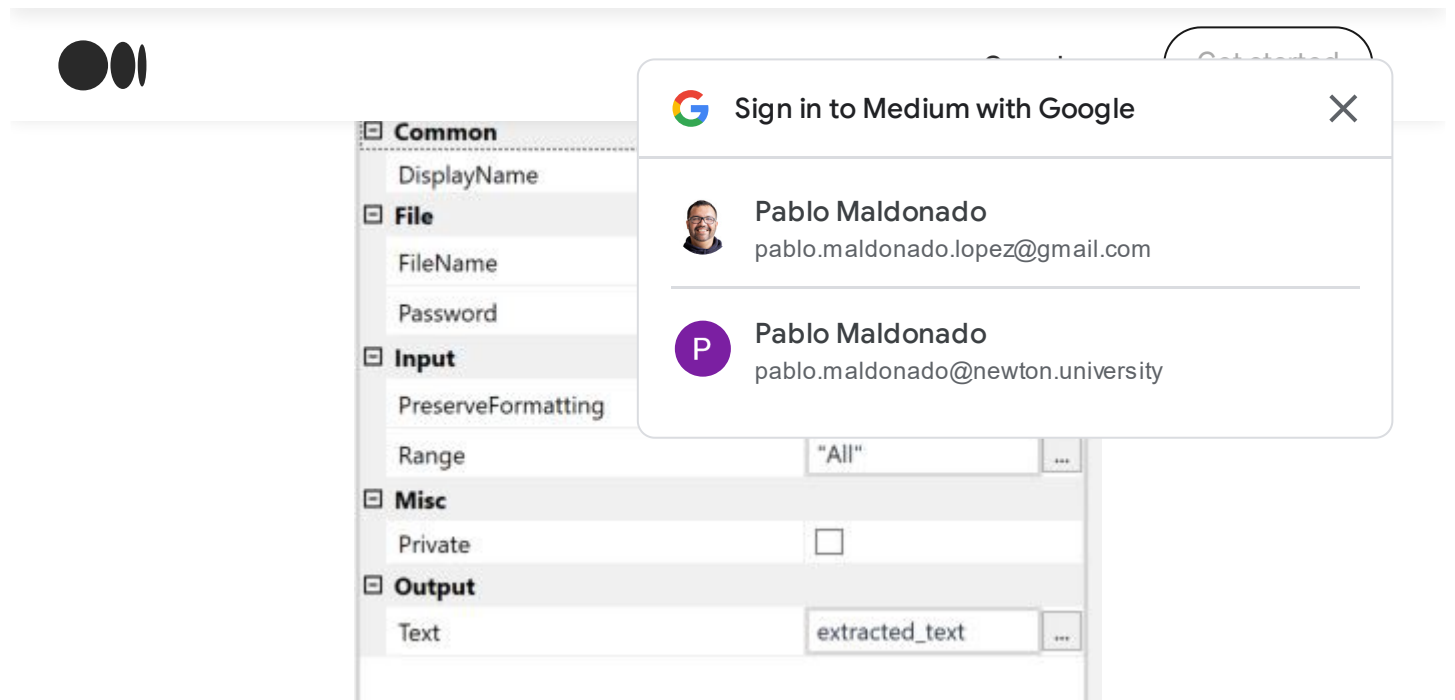
Reading Tables from a PDF the UiPath Way

As I mentioned previously, the PDF can be read well enough by UiPath. In my sequence, I first added the `Read PDF Text` activity.

Add Read PDF Text — Image by Author

There are several options for this activity including the PDF file name and a variable where I put the output. I created a `String` variable called `extracted_text` to hold the result and left the default `Range` set to `All` (this means read the entire PDF).



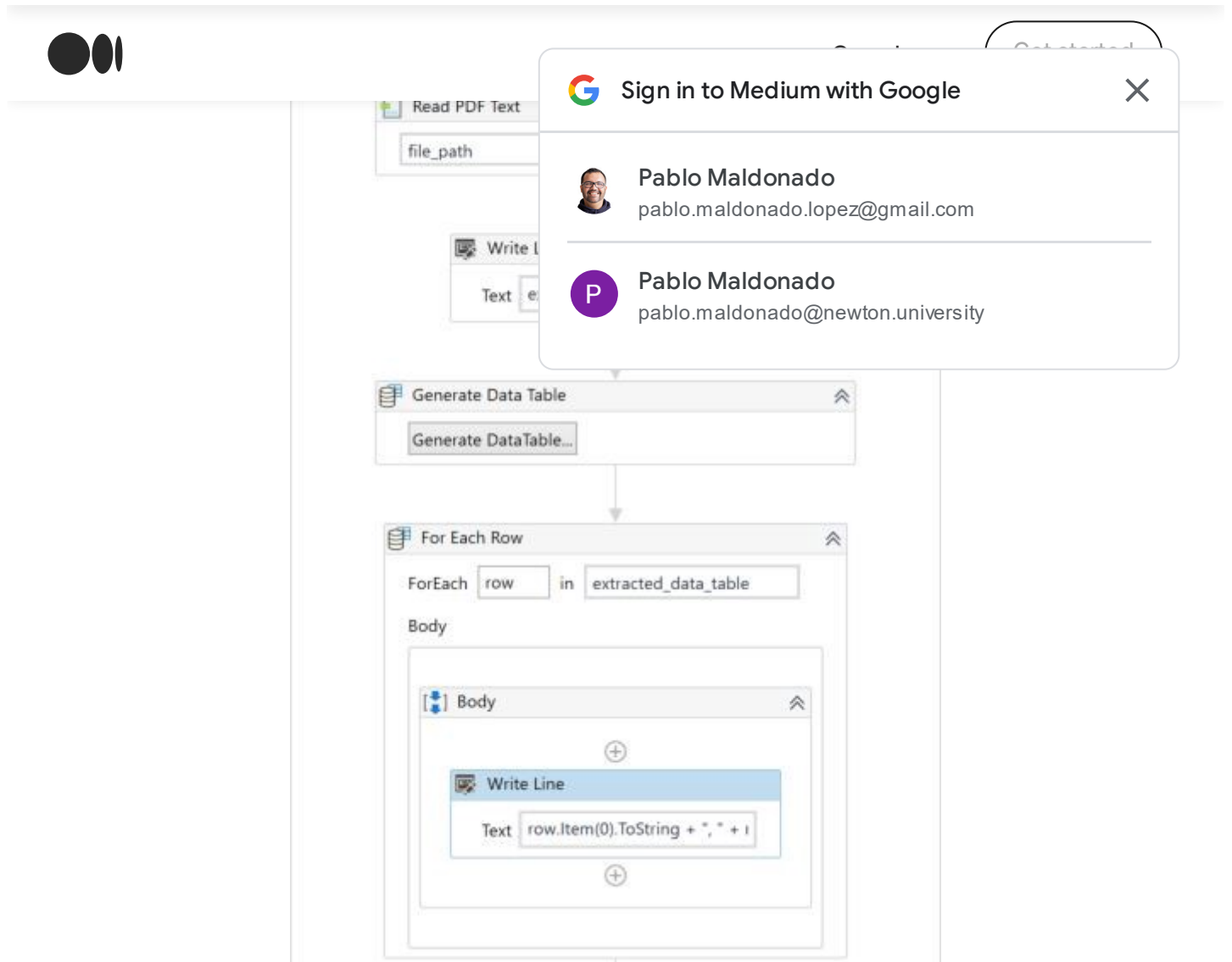


Read PDF Text Options — Image by Author

Then I added a `Write Line` Activity in UiPath and entered `extracted_text` as the Text (to write). Running this sequence gives me the table contents as a string. I can then use the `Generate Table` activity to convert this to a `DataTable`. From there, I can iterate through the `DataTable` in UiPath.

Here's the sequence:





Full Read PDF Sequence — Image by Author

The output shows the table first as a string and then as DataTable rows (with just the last and first names).

Output

⌚ ⚠ 0 🛑 0 ⓘ 1 ⌚ 5 ✅ 0 ✔ 0

Search

🔍 UiPath and Python execution started

⊙	Last Name	First Name	DOB	Income
	Smith	Albert	1/22/1970	\$125,000.00
	Williams	Gene	5/12/1966	\$ 88,000.00
	Jacobs	Sonia	11/22/1977	\$220,000.00

⊙ Smith ,Albert

⊙ Williams ,Gene

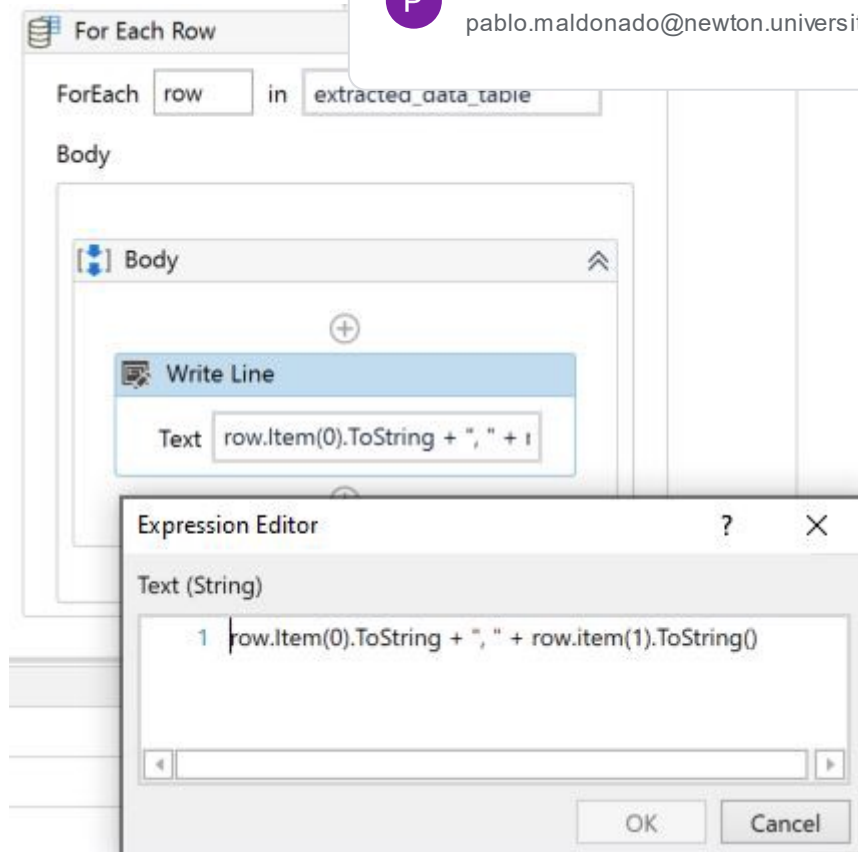
⊙ Jacobs ,Sonia

PDF Activity Output — Image by Author





the UI and to me, the UiPath way is d
that UiPath really isn't a programmin
is cumbersome.



Using a DataTable in UiPath — Image by Author

There has to be a better way to separate the programming from the UI interactions...

The Python Approach

The pattern for this approach is to use the UiPath Python Activity to call a Python script (optionally passing one or more values to it), then the Python script (optionally) returns a value. Notably, with Python, I can now manage the entire PDF table in an actual programming environment, test it without any UI interaction, and then return anything I want (at any level of granularity).

There are many ways this pattern can be used:

1. Setting up data for use by UiPath in entering svstems.



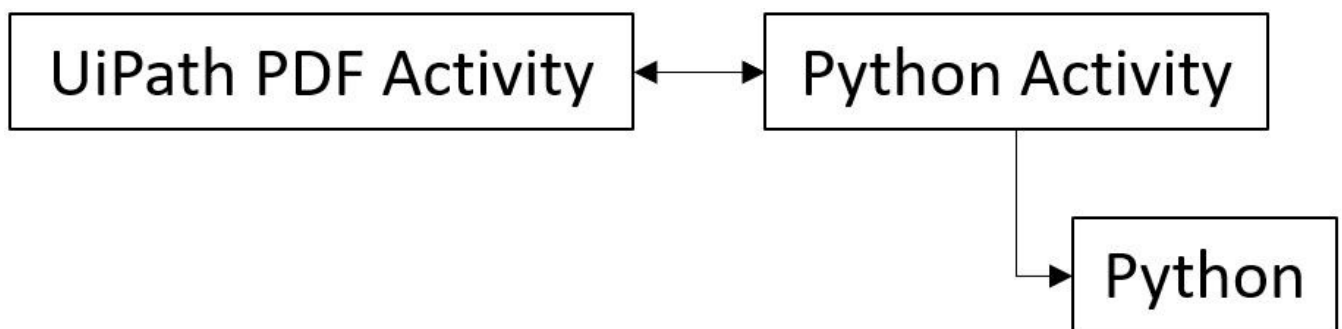
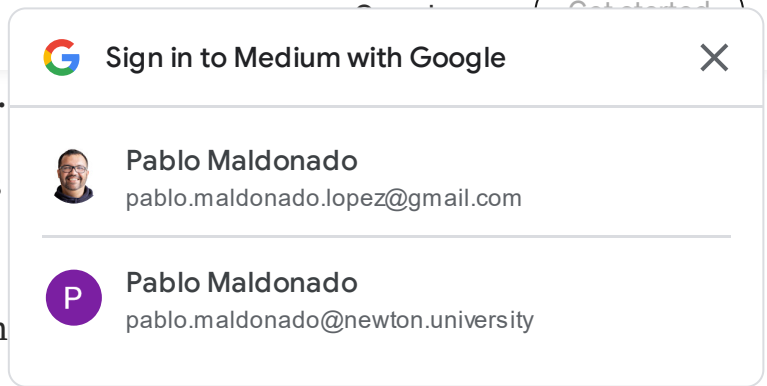


system using a UI-based wizard).

3. Using Python's REST capabilities interaction.

4. To create Python components th outside of the UiPath context.

5. Using a unique package from Python for something UiPath can't do, or doesn't do as well — e.g., reading a PDF table.



The Pattern with UiPath and Python — Image by Author

Tabula for PDF Tables

One way of reading PDF tables efficiently is by using the Tabula Python package. Tabula can read structured tables within PDF files as one or more Pandas DataFrames, which effectively allows me to manage tabular data as a table.

I added Tabula to my Python environment using `pip install tabula-py` (notice it's not called tabula, it's tabula-py).

With Tabula now installed, I created a very simple Python script that reads a passed-in file name using Tabula and returns a Python `list` containing all the tables within the PDF (each item in the list is a DataFrame). I then built a simple “local” (i.e., within the script) test to make sure everything is working properly.





Sign in to Medium with Google



Pablo Maldonado

pablo.maldonado.lopez@gmail.com



Pablo Maldonado

pablo.maldonado@newton.university

Here's the output:

```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE

RangeIndex: 3 entries, 0 to 2
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   Last Name   3 non-null      object
1   First Name  3 non-null      object
2   DOB         3 non-null      object
3   Income      3 non-null      object
dtypes: object(4)
memory usage: 224.0+ bytes
0    $ 125,000.00
1    $  88,000.00
2    $ 220,000.00
Name: Income, dtype: object
```

The Python Output — Image by Author

I can manipulate this table (as a table) within Python at will without complex text







you built all this with the UiPath PDF
worth noting that you *can* call UiPath
idea to use an RPA robot (at thousand
calling out to a web service, which ca

Connecting UiPath and Python


The last step here is to connect UiPath and Python. First, I'm going to comment out the local testing in my Python script — that testing code will run in UiPath, which is not what I want.

Since the file name is going to be passed into the Python script from UiPath, and I'm using Windows, I'm passing Windows path separators (i.e., “\”), which need to be converted to “/” for Python (there are a number of ways to accomplish this task). I also added some logging to give me visibility into the Python as it's running within UiPath (I'll demonstrate this soon enough). Here's the new script.

 Sign in to Medium with Google ✕

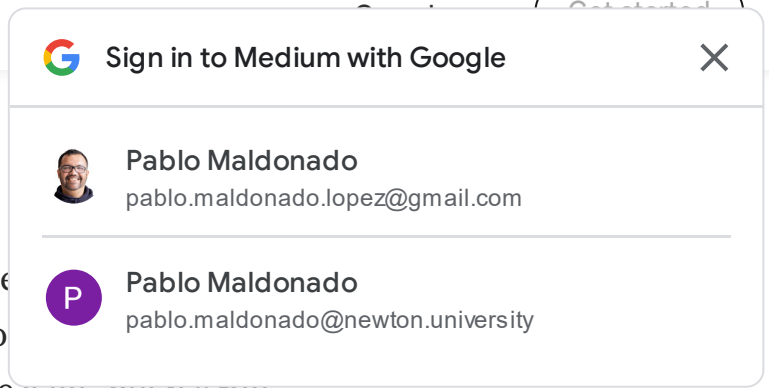


Pablo Maldonado
pablo.maldonado.lopez@gmail.com



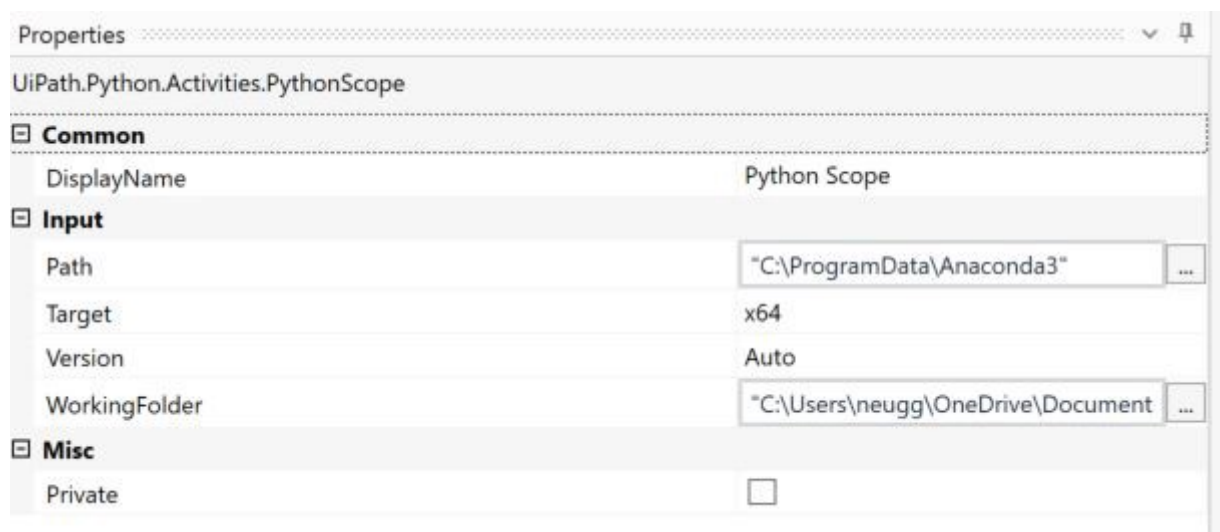
Pablo Maldonado
pablo.maldonado@newton.university





Next, I need to create a Python Scope to invoke a Python method, get a Python JSON to a .NET DataTable. Seems like a lot, but it's not.

1. I added a Python Scope Activity to my sequence. Here is where I specified my Python runtime:

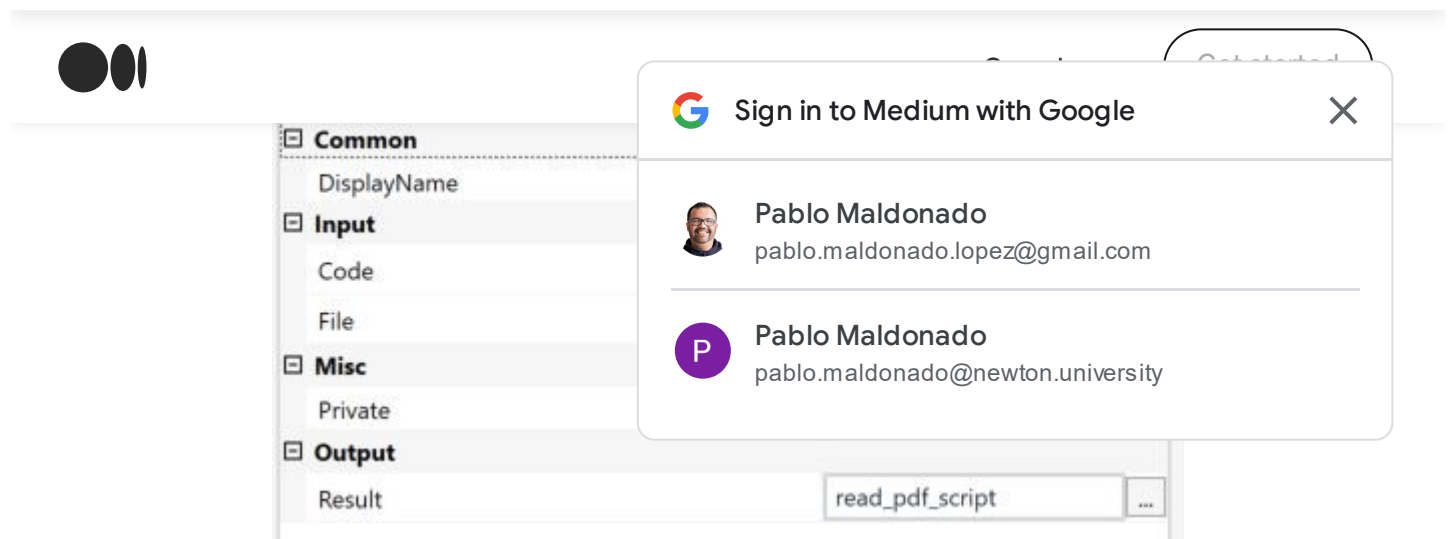


Python Scope Activity Properties — Image by Author

The `Path` is where my `python.exe` is located and the `WorkingFolder` (which is optional) is like the `PYTHONPATH` and tells UiPath where to find `.py` files that are not part of the Python installation. (I'm simplifying that last sentence just a little — in short, that folder is where the custom Python goes.)

2. Within the Python Scope, I added a Load Python Script Activity.

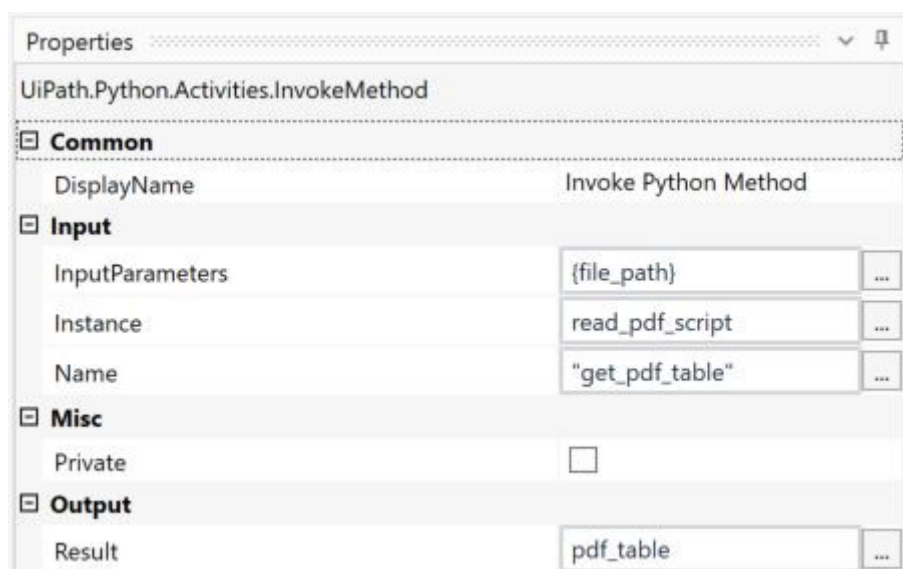




Load Python Script Properties — Image by Author

The `File` property specifies the location of my `read_pdf.py` Python script. The `Result`, which is type `PyObject`, is where the script is kept (in a variable so it doesn't have to be reloaded if I want to run multiple functions with it). Simple enough so far.

3. I added an `Invoke Python Method` Activity. As the name implies, this is the name of my method (or function if you want to be technical about it since my script isn't a class).



Invoke Python Method Properties — Image by Author

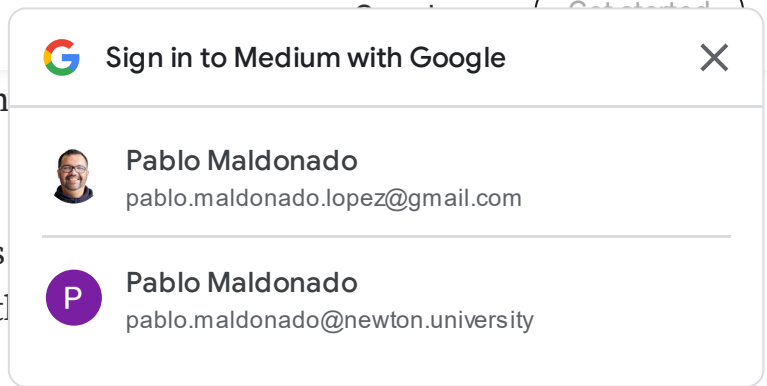
The `InputParameters` property uses squiggle notation (i.e., `{}`) that represent the data I'm passing into the method (recall that the method/function is called `get_pdf_table`). The argument for that function is the file path where the PDF is and



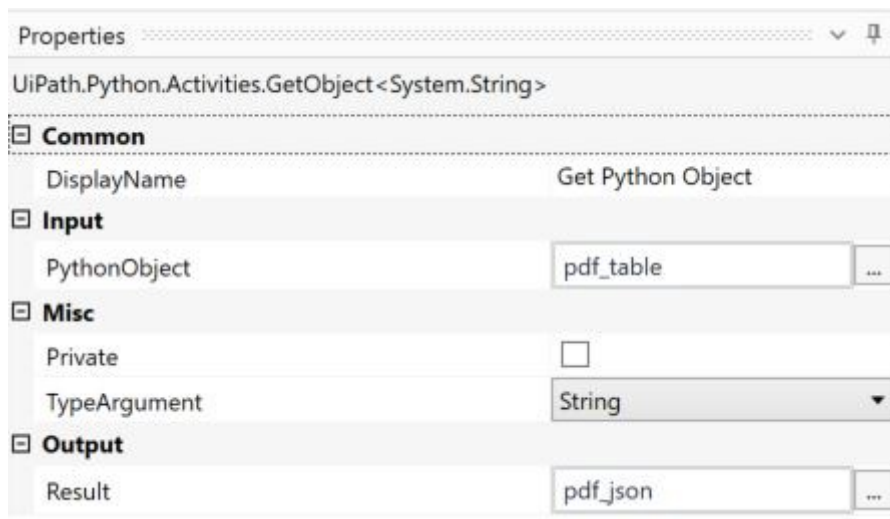


any return from the Python function data type in UiPath.

That pdf_table object, as tempting as UiPath as-is. That's because it's a Python without a little help.



4. I added a Get Python Object Activity to my sequence. This is where the PythonObject is turned into something more useful to UiPath. Recall that the get_pdf_table function returns a JSON string of the PDF table contents.

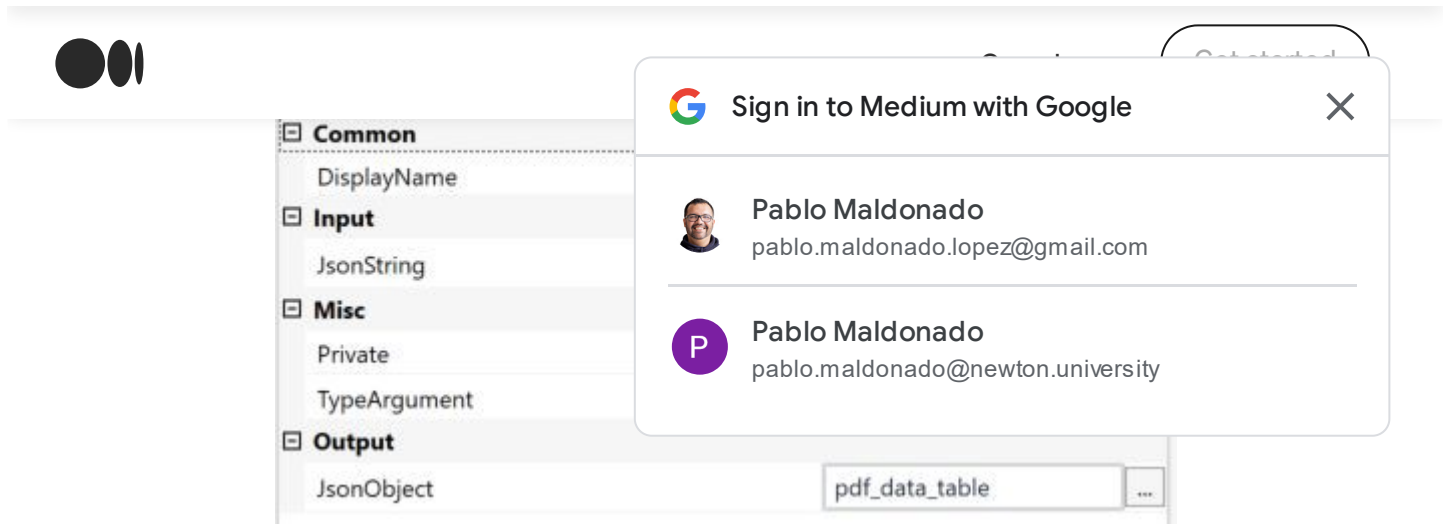


Get Python Object Properties — Image by Author

The PythonObject is the name of the object to convert (the object from step #3). The TypeArgument property specifies the data type of the PythonObject object in UiPath terms (a String). The Result property is where the String goes once it's gotten.

5. Finally, I want to convert the String (JSON) to a DataTable. To do so, I added a Deserialize JSON Activity to my sequence.





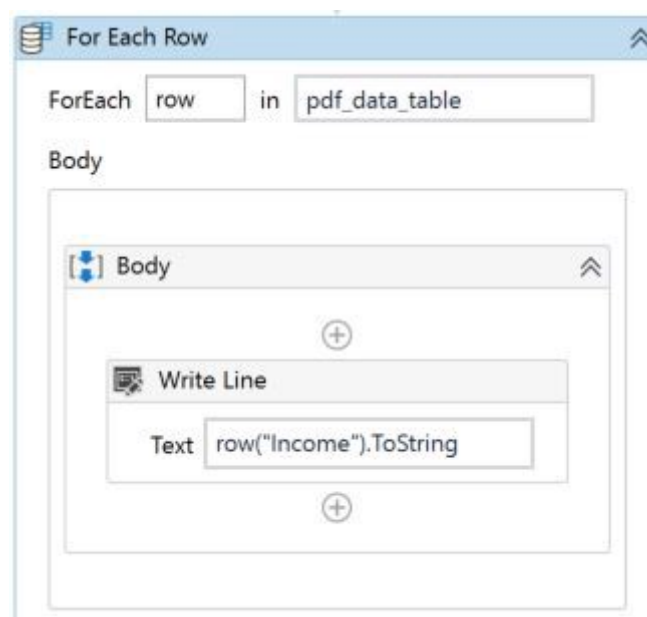
Deserialize JSON Properties — Image by Author

The `JsonString` is pretty self-explanatory and is the JSON (as a `String`) result from step #4. The `TypeArgument` is what I want to convert that JSON String to—a `DataTable`. Finally, the `JsonObject` property is where the `DataTable` is going to go.

It seems like a lot, but in having done this quite a few times, it's very fast after the first time.

Last but not Least

The last thing I'm going to do is iterate through the `DataTable` and write out the income for each row. This last part is done the same way as I showed previously in the UiPath way.



Iterating Through a `DataTable` in UiPath — Image by Author

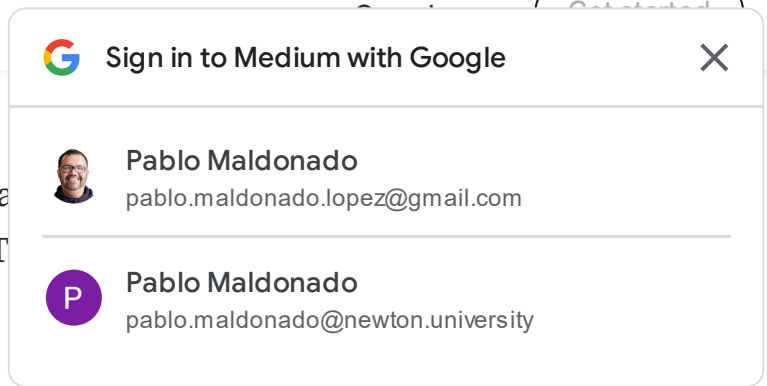




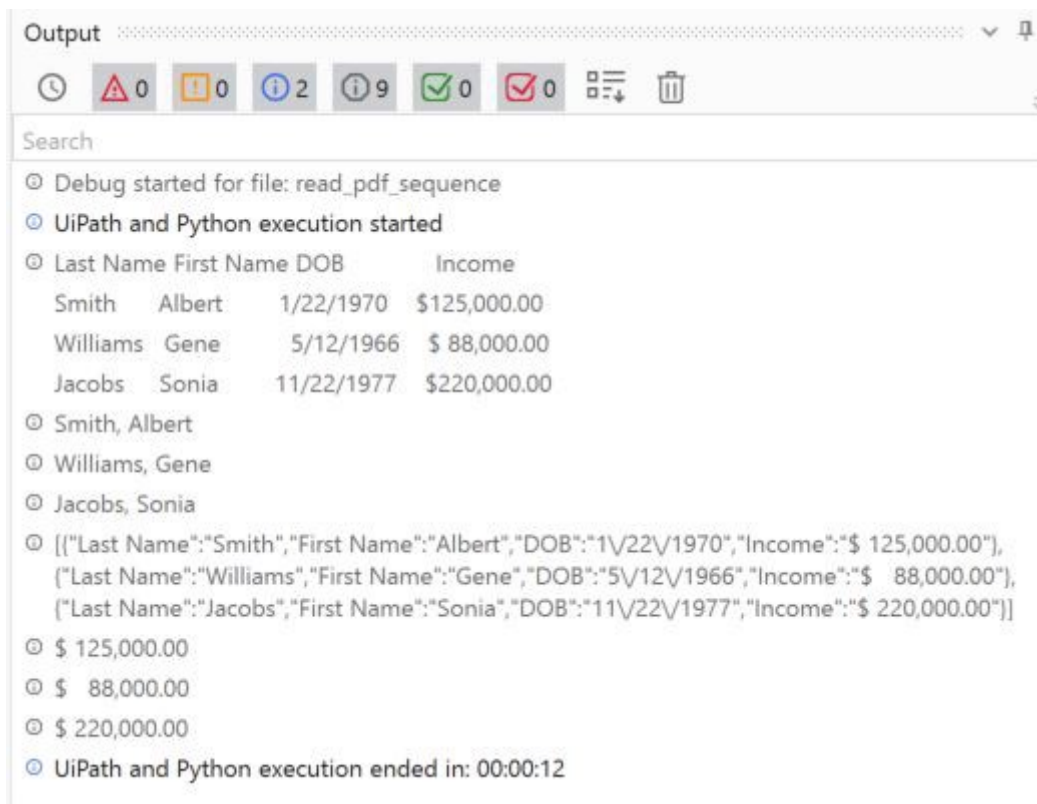
from step #5.

The Body is where processing (for each row) is done. This is done with some .NET code:

```
row("Income").ToString .
```



Here's the output from the entire UiPath automation:



UiPath and Python Full Output — Image by Author

You can see from the end of the output that the incomes are each written out. You can also see the JSON String just prior to that (which I also put in a Write Line Activity).

Remember that logging I added to the Python script? It's purpose is to help with debugging because when the script is running in UiPath, it's like having it on the dark side of the moon; there's no way to communicate with it in UiPath. Having logging enabled is critical for this design. Here's the output from my `logging.txt` file:

```
INFO:root:C:/Users/neugg/.../simple_data.pdf
```





```
"DOB": "5/12/1966", "Income": "$ 88,  
Name": "Sonia", "DOB": "11/22/1977"
```

If something had gone wrong, I could have
passed into the Python script correctly.

Disclaimer #2: I recognize that this result is the same as what I showed using only UiPath (i.e., a DataTable). However, it's all the pre and post UiPath processing that I can do with Python that makes this separation **SO** important. Not only that, but I could also have only given back the income values (as an array).

The Right Tool for the Job and Keeping Your Independence

I believe UiPath is great at what it's great at, but I don't think it's great at everything. By using Python with UiPath, the right tool can be used for the job (just like Tabula for reading PDF tables as Pandas DataFrames). Moreover, by avoiding the programmatic capabilities of RPA tools, it's significantly easier to remain independent from them, which creates not only leverage (at contract time) but may also enable you to create or use the components you build outside of the UiPath context.



Sign in to Medium with Google



Pablo Maldonado

pablo.maldonado.lopez@gmail.com



Pablo Maldonado

pablo.maldonado@newton.university

Sign up for CrunchX

By CodeX

A weekly newsletter on what's going on around the tech and programming space [Take a look.](#)

Your email



Get this newsletter





[About](#) [Help](#) [Terms](#) [Privacy](#)

Get the Medium app



Sign in to Medium with Google



Pablo Maldonado

pablo.maldonado.lopez@gmail.com



Pablo Maldonado

pablo.maldonado@newton.university

