



Beyond Scripting

A Beginner's Guide to Accelerating Automation with Executables



www.mainanalytics.de



...on a calm Friday afternoon



...on a calm Friday afternoon



Colleague



Michel

Are you still working?



...on a calm Friday afternoon



Colleague



Michel

Are you still working?

... say no

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Yes, sure.
How can I help you?



...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

Yes, sure.
How can I help you?

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

Yes, sure.
How can I help you?

... say no

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

... and add a ToC?

Yes, sure.
How can I help you?

... say no

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

... and add a ToC?

Yes, sure.
How can I help you?

... say no

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

... and add a ToC?

... and sequential page numbers?

Yes, sure.
How can I help you?

... say **no**

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

... and add a ToC?

... and sequential page numbers?

.. also, not many—just about ~1,000

Yes, sure.
How can I help you?

... say **NO**

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Do you know how to easily merge PDFs?

... and add a ToC?

... and sequential page numbers?

.. also, not many—just about ~1,000

Yes, sure.
How can I help you?

... say **NO**

...on a calm Friday afternoon



Colleague



Michel

Are you still working?

Yes, sure.
How can I help you?

Do you know how to easily merge PDFs?

... and add a ToC?

... and sequential page numbers?

... let me *think* one second about it...

.. also, not many—just about ~1,000

...so, what to do?

... let me *think* one second about it...

What's on the agenda today?

Hi, as a perfect python programmer, how can i combine all pdfs in a folder into one pdf and add a table of content right in the beginning? Add also a hyperlink from the toc entries to the corresponding sides.

+



...so, what to do?

You can absolutely do this in Python—but it's a bit more involved than just merging PDFs because:

1. PDFs don't inherently have a "table of contents page" (that's something you must generate).
2. You need to:
 - Merge PDFs
 - Track page offsets
 - Create a TOC page
 - Add clickable links (annotations)

The best approach is to combine `**PyPDF2` (or newer `pypdf`) + `**ReportLab`.

Overall Strategy

1. Read all PDFs in a folder
2. Merge them while tracking starting page numbers
3. Create a TOC PDF page with ReportLab
4. Merge TOC at the beginning
5. Add link annotations pointing to correct pages

... let me *think* one second about it...

...so, what to do?

You can absolutely do this in Python—but it's a bit more involved than just merging PDFs because:

1. PDFs don't inherently have a "table of contents page" (that's something you must generate).
2. You need to:
 - Merge PDFs
 - Track page offsets
 - Create a TOC page
 - Add clickable links (annotations)

The best approach is to combine `**PyPDF2` (or newer `pypdf`) + `**ReportLab`.

Overall Strategy

1. Read all PDFs in a folder
2. Merge them while tracking starting page numbers
3. Create a TOC PDF page with ReportLab
4. Merge TOC at the beginning
5. Add link annotations pointing to correct pages

... let me *think* one second about it...

Seems
pretty easy

The solution (almost)...

✓ Full Working Example

Install dependencies:

↗ Bash

```
pip install pypdf reportlab
```

Python Script

↗ Python



▶ Run

```
# read all PDFs first
for pdf in pdf_files:
    reader = PdfReader(pdf)
    readers.append(reader)

    num_pages = len(reader.pages)
    title = os.path.basename(pdf)

    toc_entries.append((title, current_page))
    current_page += num_pages

# create TOC PDF
toc_pdf, link_rects = create_toc(toc_entries)

# add TOC pages first
for page in toc_pdf.pages:
    writer.add_page(page)

# then add all PDFs
for reader in readers:
    for page in reader.pages:
```

The solution (almost)...

✓ Full Working Example

Install dependencies:

↗ Bash

```
pip install pypdf reportlab
```

Python Script

↗ Python



▶ Run

```
# read all PDFs first
for pdf in pdf_files:
    reader = PdfReader(pdf)
    readers.append(reader)

    num_pages = len(reader.pages)
    title = os.path.basename(pdf)

    toc_entries.append((title, current_page))
    current_page += num_pages

# create TOC PDF
toc_pdf, link_rects = create_toc(toc_entries)

# add TOC pages first
for page in toc_pdf.pages:
    writer.add_page(page)

# then add all PDFs
for reader in readers:
    for page in reader.pages:
```

... the problem

- pdfs are on a sponsor's system
- only SAS is available

The solution (for real) ...

✓ Full Working Example

Install dependencies:

↗ Bash

```
pip install pypdf reportlab
```

Python Script

↗ Python



▶ Run

```
# read all PDFs first
for pdf in pdf_files:
    reader = PdfReader(pdf)
    readers.append(reader)

    num_pages = len(reader.pages)
    title = os.path.basename(pdf)

    toc_entries.append((title, current_page))
    current_page += num_pages

# create TOC PDF
toc_pdf, link_rects = create_toc(toc_entries)

# add TOC pages first
for page in toc_pdf.pages:
    writer.add_page(page)

# then add all PDFs
for reader in readers:
    for page in reader.pages:
```

1. plug logic into **ma-pytemp**

The solution (for real) ...

✓ Full Working Example

Install dependencies:

<> Bash

```
pip install pypdf reportlab
```

Python Script

<> Python



Run

```
# read all PDFs first
for pdf in pdf_files:
    reader = PdfReader(pdf)
    readers.append(reader)

    num_pages = len(reader.pages)
    title = os.path.basename(pdf)

    toc_entries.append((title, current_page))
    current_page += num_pages

# create TOC PDF
toc_pdf, link_rects = create_toc(toc_entries)

# add TOC pages first
for page in toc_pdf.pages:
    writer.add_page(page)

# then add all PDFs
for reader in readers:
    for page in reader.pages:
```

1. plug logic into **ma-pytemp**
2. build **executable** and deploy

m
ma_pdf-
compiler.exe

...1 hour later



You're the best



I created this executable:
`path/to/ma_pdf-compiler.exe`

just click on it

... on a beautiful Friday evening

A person wearing a dark t-shirt and a black watch is pouring beer from a wooden barrel with a tap. The barrel has the word "SCH" visible on it. The scene is set in a room with wooden paneling and a green lamp. A semi-transparent white box with the text "everything went as planned" is overlaid on the image.

everything went
as planned

Executables – What and Why?



self-sustained programs

no further dependencies
no installation



unlocking the power of

high-level languages
libraries
rest of the world



for all environments

GxP
SCE



complexity overhead

develop
build
deploy



abstract complexity

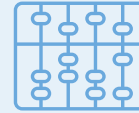
standardize template
prebuild
focus on automation

Executables – What?



Simple answer

- .exe
- “click and play”

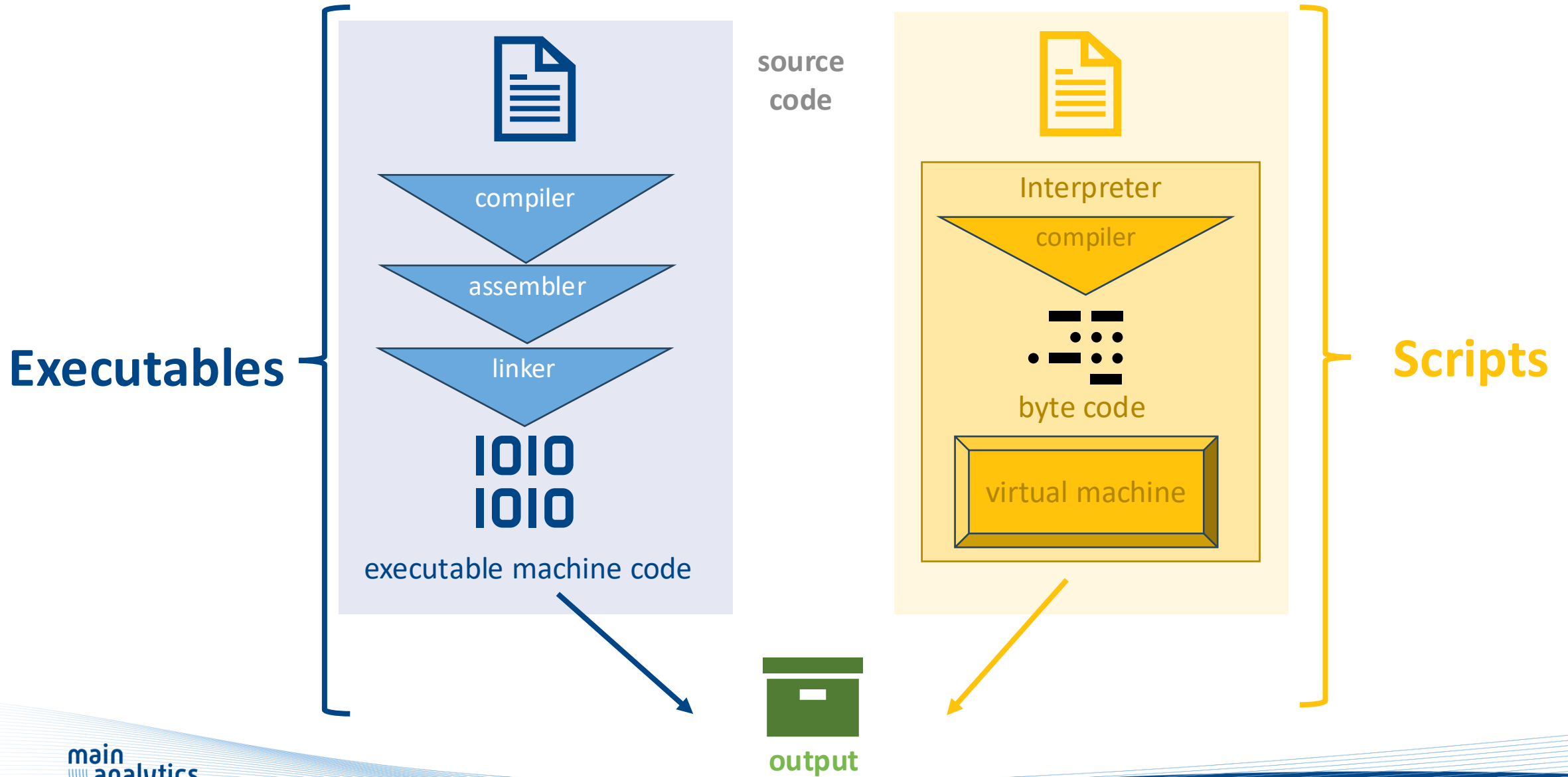


Detailed Answer

- program without needing further compilation
- **machine code** specific to CPU architecture operation system
- **deliverable** of software engineering



Executables vs. Scripts



Executables – Why?

- Many common languages are interpreted
Java, Python, R, C#,...
- Require “extra” software -> often not available

Executables – Why?

- Many common languages are interpreted
Java, Python, R, C#,...
- Require “extra” software -> often not available

Why not using what is already available?

Executables – Why?

- Many common languages are interpreted
Java, Python, R, C#,...
- Require “extra” software -> often not available

Why not using what is already available?



Executables – Why?

- Many common languages are interpreted
Java, Python, R, C#,...
- Require “extra” software -> often not available

Why not using what is already available?

- unleash the internet
- unleash AI



Challenges of Executables

Programming language	Dependency management	User Interface (Framework)	Adaptivity
Hard coded resources	Robustness	Error & Exception handling	IO
User feedback	Logging	Unit Testing	Validation
Version control	Build	Building machine vs. target machine	Deployment

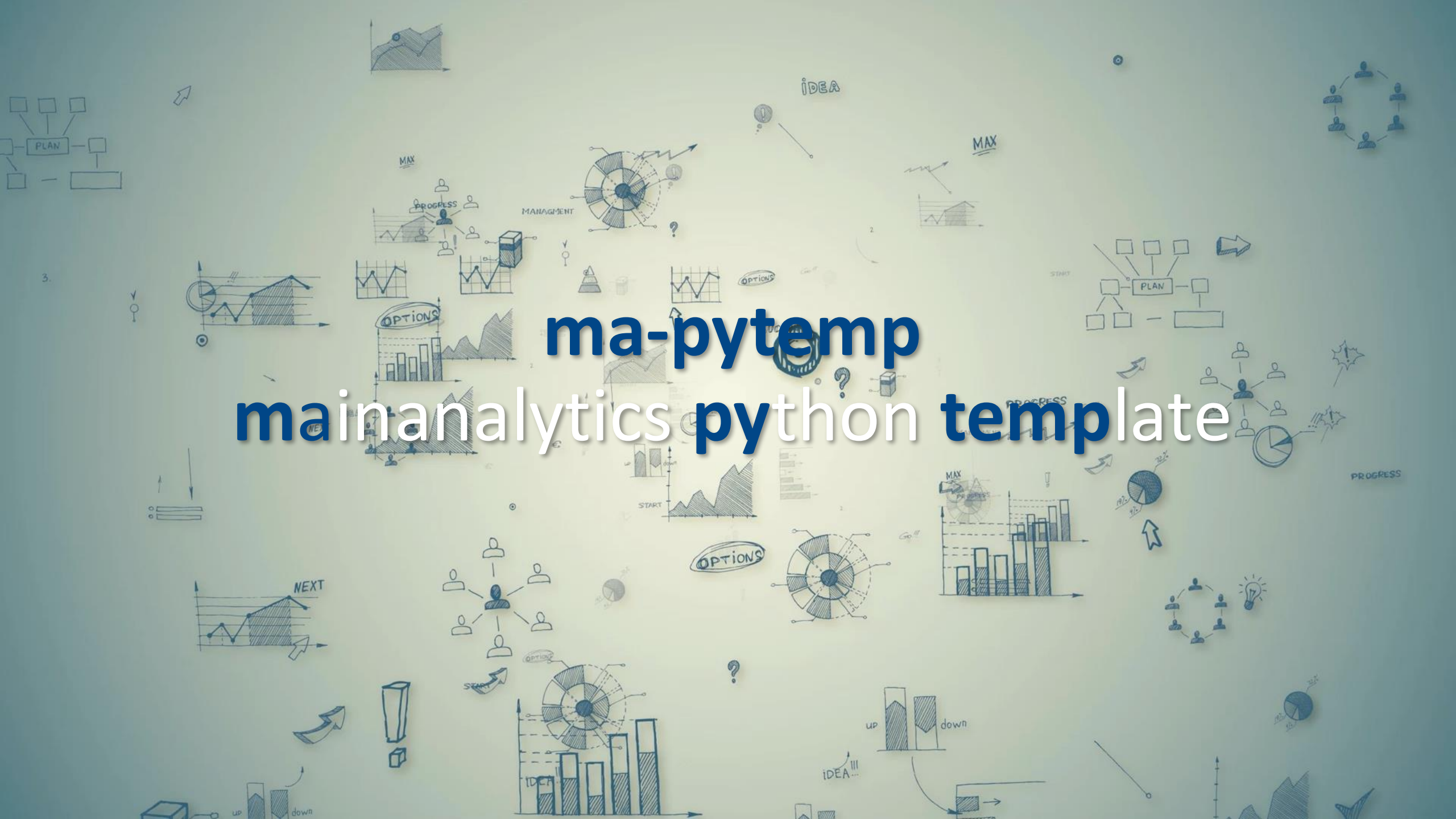
Executables as Tools in Statistical Programming



powerful and modern
even for GxP environments



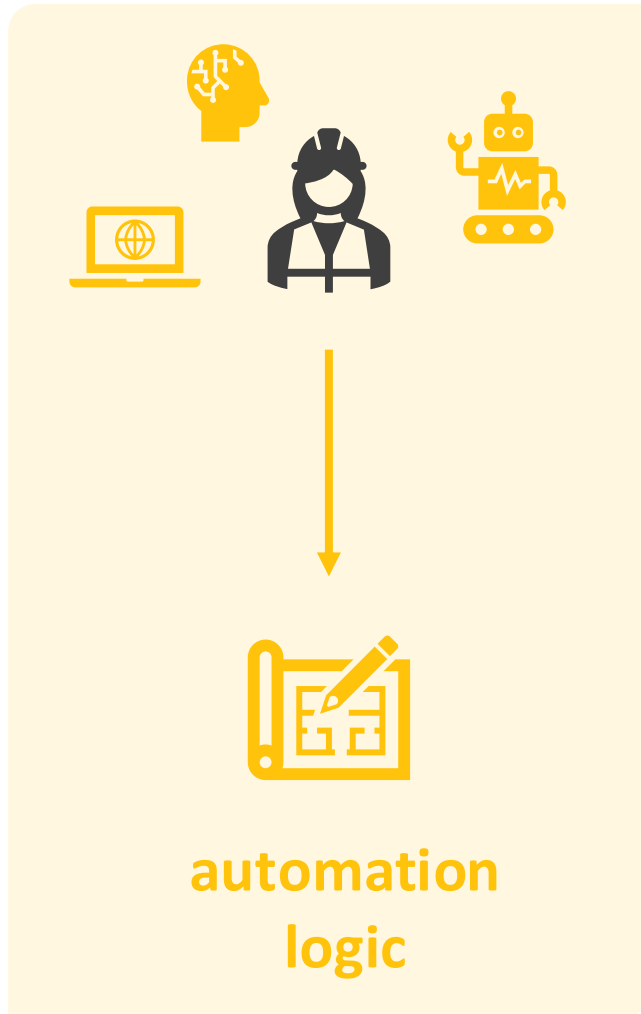
high **development complexity**
for the “box” around although
core logic often **simple**

The background is a light blue surface covered with various hand-drawn sketches in dark blue ink. These sketches include: line graphs with upward trends, bar charts, pie charts, and circular diagrams. Some diagrams feature human figures, suggesting organizational charts or team structures. Text labels like 'PLAN', 'PROGRESS', 'MANAGEMENT', 'IDEA', 'MAX', 'START', 'NEXT', 'OPTIONS', 'UP', and 'DOWN' are scattered throughout. Arrows and question marks are also present, indicating a process of planning, analysis, and decision-making.

ma-pytemp

mainanalyticspython template





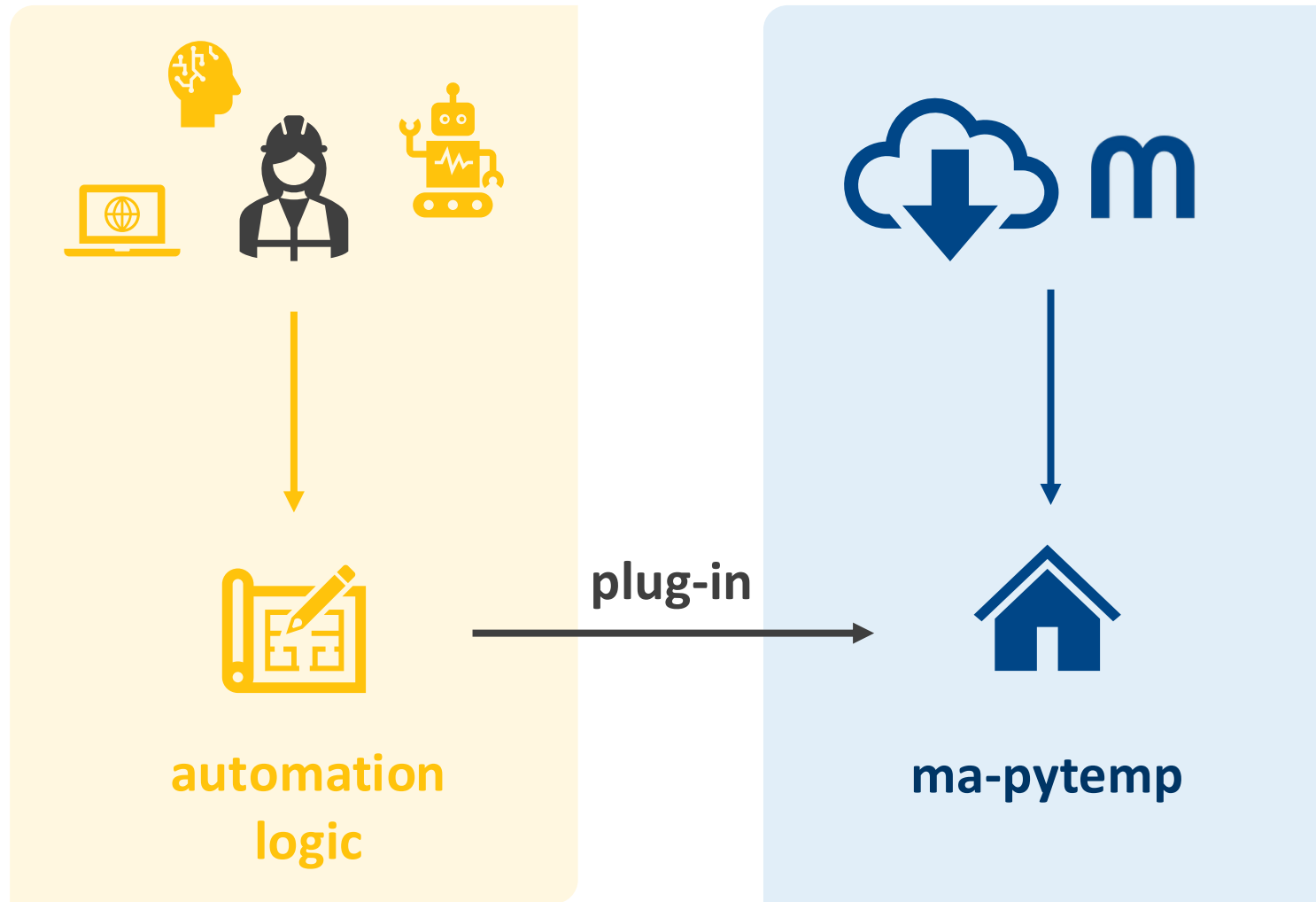
→ **search for solutions**

- Libraries
- Repos
- Internet

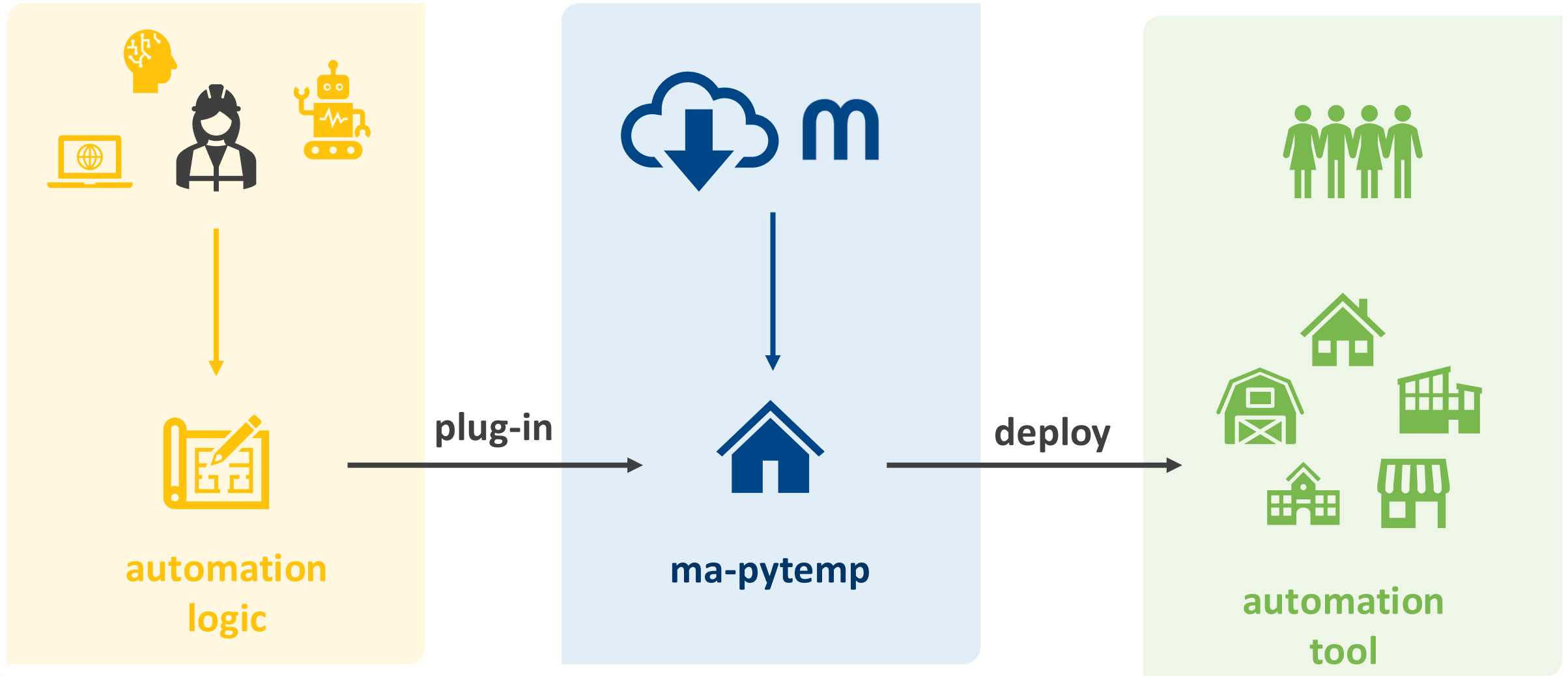
→ **Vibe-coding**

→ **Use your own skills**

ma-pytemp: Idea



ma-pytemp: Idea



**Key
features**

Python

Dependency & Project Management

GUI Framework

Structured architecture

Logging

Adaptability

Validation

CICD

Key features

Python

Dependency & Project Management

GUI Framework

Structured architecture

Logging

Adaptability

Validation

CICD

More features

- Template git repository
- Version control
- Documentation
- Basic code quality
- Customization
- .ico creation
- multi-threading
- ...

how to use ma-pytemp?

1

Use this template ▾



<https://github.com/mainanalytics>

how to use ma-pytemp?

1

Use this template ▾



<https://github.com/mainanalytics>

2



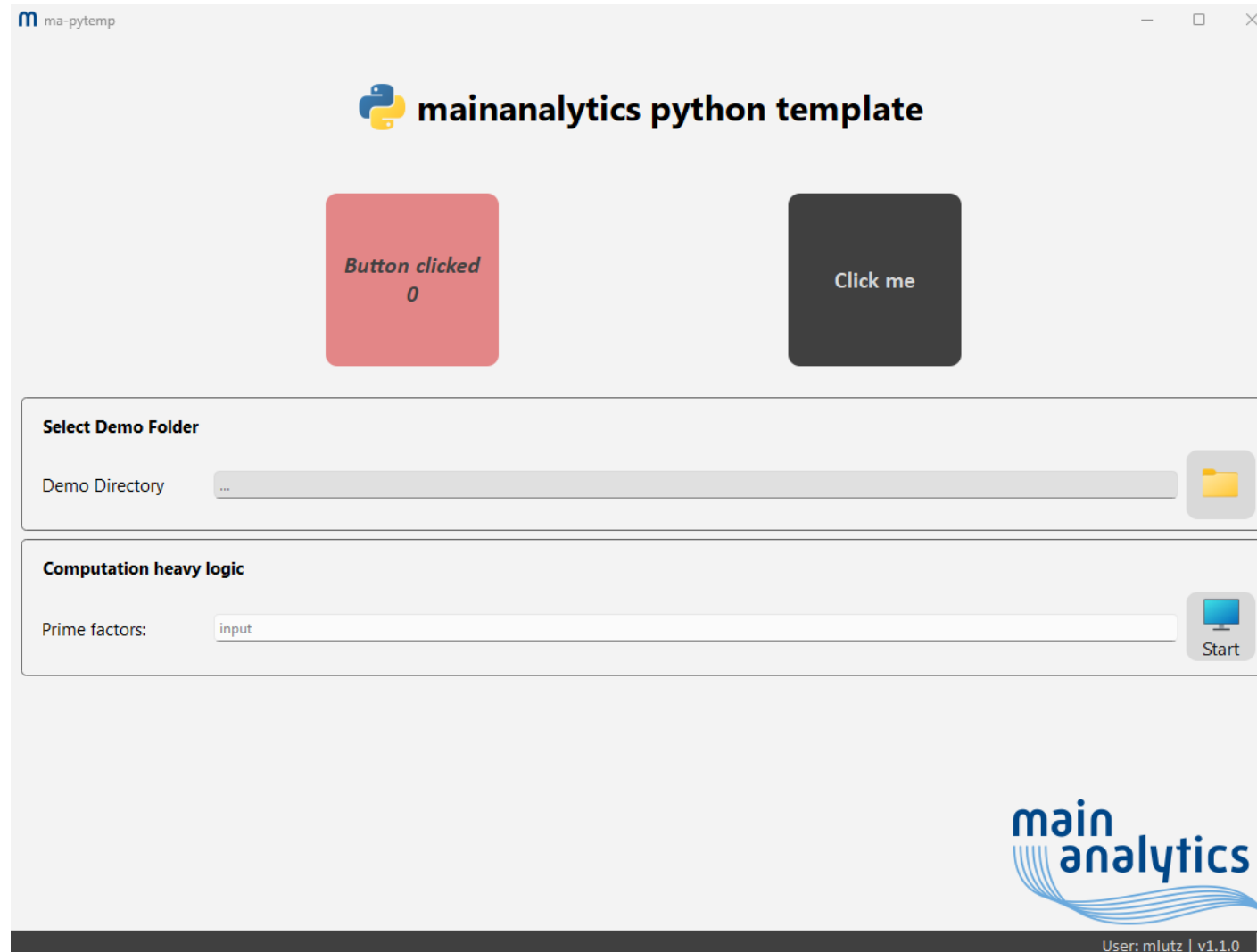
initialize project
create VENV
install dependencies
...

- ▶ C:\Users\mlutz> cd ma-pytemp
- ▶ C:\Users\mlutz> poetry install

The image features three overlapping mobile application wireframes on a light gray background. The wireframes are white rectangles with rounded corners, each containing gray placeholder elements for text, images, and icons. The central wireframe is the most prominent, showing a header with a profile picture and name, a large central image area, and a bottom navigation bar with four icons. The other two wireframes are partially obscured behind it, one to the left and one to the right.

User Interface

Graphical User Interface

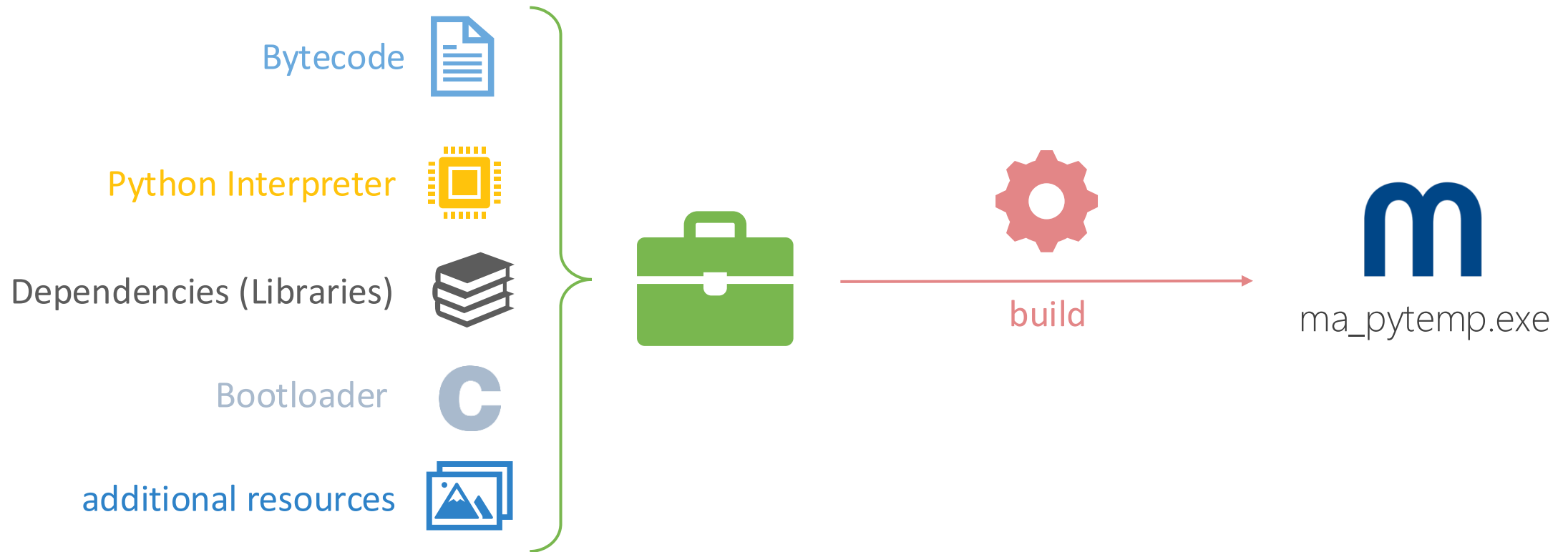




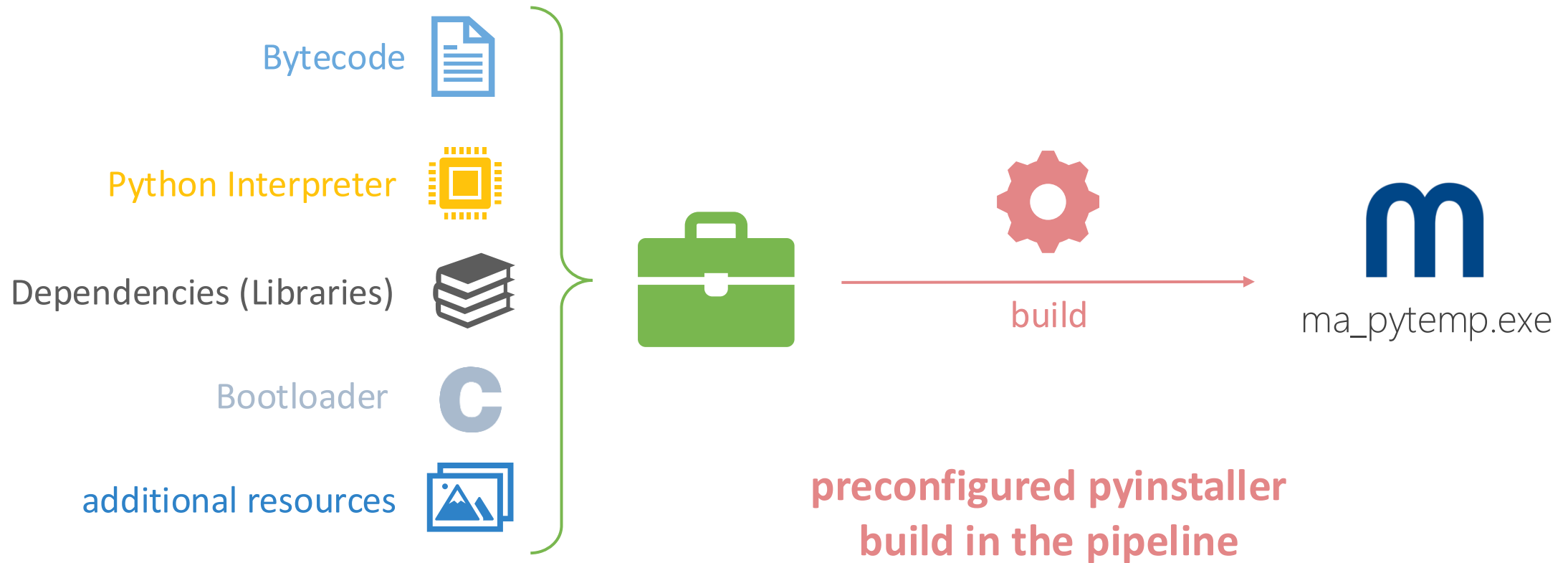
creating the executable



I'm packing my suitcase

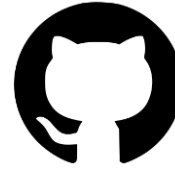


I'm packing my suitcase

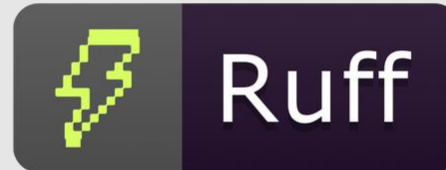


Deployment

A 3D rendering of a warehouse floor. The floor is a light blue-grey color with a grid of red lines. Several cardboard boxes are scattered across the floor. Some boxes are on the grid, while others are off to the side. The boxes are brown and have some labels, including one that says 'FRAGILE'. The word 'Deployment' is written in a large, bold, blue font across the center of the image. In the top right corner, there is a small blue 'm' logo.



predefined CI/CD workflow



ma-pytemp: Workflow



Workflow file for this run

[github/workflows/base-pipe.yml](#) at 84d487c

```
1 # This workflow will install Python dependencies, run tests and lint with a single version of Python
2 # For more information see: https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-python
3
4 name: Python application
5
6 on:
7   push:
8     branches:
9       - main
10      - dev
11   pull_request:
12     branches:
13       - main
14      - dev
15
16 permissions:
17   contents: read
18
19 jobs:
20   build:
21     strategy:
22       fail-fast: false
23
24   runs-on: windows-latest
25
26   steps:
27     - name: Checkout code
28       uses: actions/checkout@v5
29
30     - name: Set up Python 3.13.12
31       uses: actions/setup-python@v6
32       with:
33         python-version: "3.13.12"
34
35     - name: Install poetry
36       run: |
37         python -m ensurepip
38         python -m pip install --upgrade pip
39         python -m pip install poetry
```



Workflow runs ·
[mainanalytics/ma_pytemp](#)

```
42 - name: Install dependencies with Poetry
43   run: poetry install --with dev
44
45 - name: Set PYTHONPATH to include the source directory
46   run: echo "PYTHONPATH=$PWD/src/ma_pytemp" >> $GITHUB_ENV
47
48 - name: Formatter
49   run: poetry run ruff format
50
51 - name: Lint with pylint (fail on warnings/errors)
52   run: poetry run pylint src/
53
54 - name: Unit Test and report creation
55   run: poetry run pytest
56
57 - name: Build executable
58   run: poetry run pyinstaller -y main_standard.spec
59
60 # Upload the compiled executable
61 - name: Upload artifacts
62   uses: actions/upload-artifact@v7
63   with:
64     name: ma_pytemp
65     path: |
66       dist/
67       config/
68       validation_report/
69     retention-days: 10
70
71 # Show artifact id and store in environment variable
72 - name: Output artifact ID
73   run: |
74     echo 'artefact_id=${{ steps.artifact-upload-step.outputs.artifact-id }}' >> $GITHUB_ENV
75     echo 'Artifact ID is ${{ steps.artifact-upload-step.outputs.artifact-id }}'
76
77 deploy:
78   runs-on: windows-latest
79   needs: build
80
81   steps:
82     - name: Download artifact
83       uses: actions/download-artifact@v7
84       with:
85         name: ma_pytemp
86         path: /opt
87
88 - name: List Artifacts
89   run: ls /opt
90
91
```


ma-pytemp: Deployment



mainanalytics / ma_pytemp

CodeIssuesPull requestsActionsProjectsSecurity and qualityInsightsSettings

Python application

fix: small adjustments #15

Re-run all jobs

Summary

All jobs

build

deploy

Run details

Usage

Workflow file

Triggered via push 16 minutes ago

Success

4m 17s

1

base-pipe.yml

on: push

build3m 53s

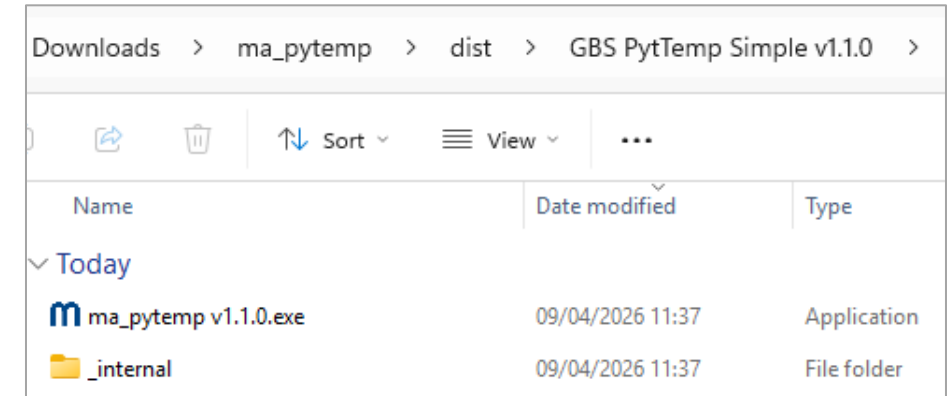
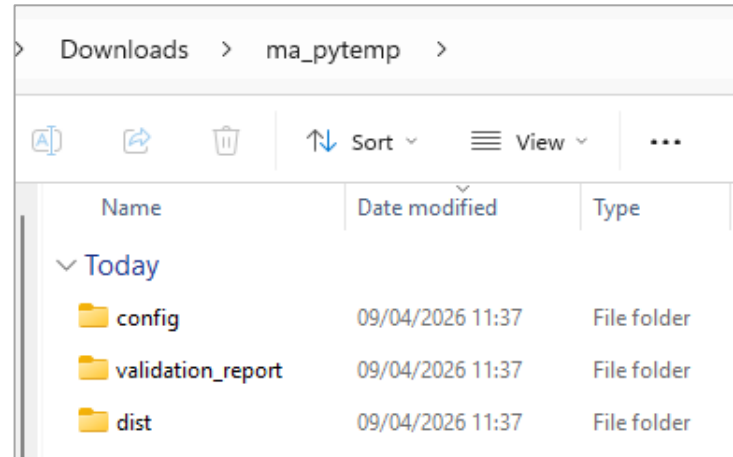
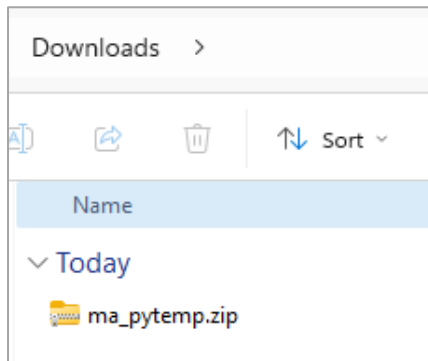
deploy14s

Artifacts

Produced during runtime

Name	Size	Digest
ma_pytemp	54.1 MB	sha256:9320908f0f5498ef74cf4ae05a1d1d70f5fff27b3fb6df29d6c361b63e6752fb

artifact is a .zip folder





into production

into production



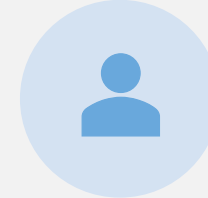
COMPILE PDFS



ANNOTATE CRF



**DIR &
PERMISSIONS**



PARSE SAS LOG



RFT -> PDF



PDF BOOKMARKS



SAS BATCH RUNS



TOOL LAUNCHER



...

CRF Bookmarks

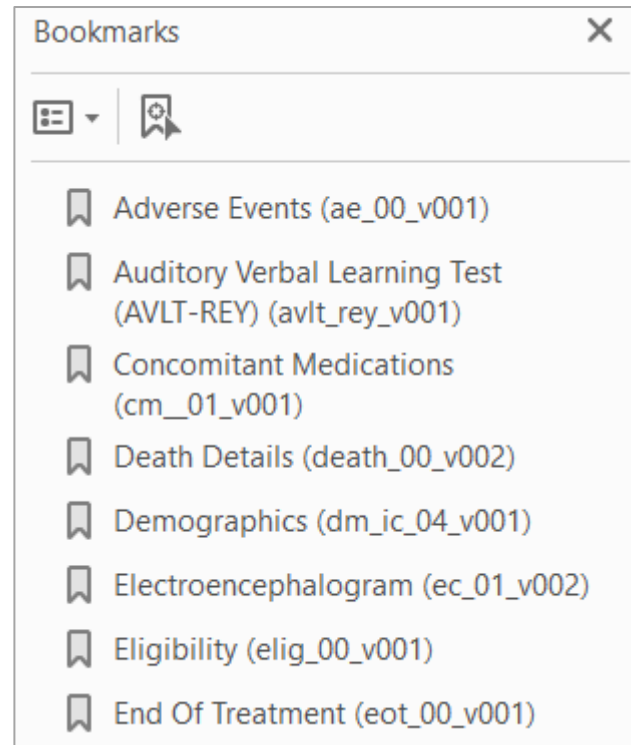
Situation

- a **software** provider cannot extract CRFs with bookmarks by form and visit
- >> 20 studies
- unclear when or even if this will be fixed

Current solution

- TDMs create bookmarks by hand
- painful and error prone

CRF Bookmarks



- first level bookmarks
<form_label> (<form_name>)
- no visit information
- Study Definition Specs (SDS)
contains visit information

CRF Bookmarks

crf_raw.pdf

Bookmarks		
		
	Adverse Events (ae_00_v001)	
	Auditory Verbal Learning Test (AVLT-REY) (avlt_rey_v001)	
	Concomitant Medications (cm_01_v001)	
	Death Details (death_00_v002)	
	Demographics (dm_ic_04_v001)	
	Electroencephalogram (ec_01_v002)	
	Eligibility (elig_00_v001)	
	End Of Treatment (eot_00_v001)	

sds.xlsx

Form Label	Form Name	Running Records	Screening		Baseline	
		Running Records	Screening 1	Screening 2	Baseline	
Adverse Events	ae_00_v001	X				
Concomitant Medications	cm_01_v001	X				
Death Details	death_00_v002	X				
Electroencephalogram	ec_01_v002	X				
Exposure	exp_000	X				
Injection Site Reactions	inj_site_00_v001	X				
Demographics	dm_ic_04_v001		X			
Eligibility	elig_00_v001		X			
Informed Consent	ic_01_v001		X			
Medical History	mh_00_v001		X			
Vital Signs (Weight, Height And Temperature)	vs_wgh_hgt_temp_000		X			
Ophthalmic Examination	ophth_exm_v001		X	X	X	
Auditory Verbal Learning Test (AVLT-REY)	avlt_rey_v001			X		
Vital Signs (Temperature)	vs_temp_000			X		
Patient Health Questionnaire-9 (PHQ-9)	phq_9_v001				X	
Satisfaction With Life Survey (SWLS)	swls_000				X	
Hamilton Depression Rating Scale - 17 (HAMD-17)	hamd_17_00_v001				X	
Vital Signs (Weight And Temperature)	vs_wgh_temp_000				X	
Patient Summary	pat_sum_000					
End Of Treatment	eot_00_v001					

CRF Bookmarks

Solution

1. Read in crf_raw.pdf with pymupdf
2. Extract input bookmarks
3. Read in sds.xlsx with pandas
4. Match Form Label and Form Names from original bookmarks with sds
5. Create new bookmarks
 - by Form
 - by Visit
6. Save new file

Deployment

1. Plug logic into **ma_pytemp**
2. Add 3 IO Buttons
3. Add Start button

Example: CRF Bookmarks

m GBS CRF Bookmark

CRF Bookmark Tool

CRF

select CRF

Study Definition Spec (SDS)

select SDS

Output CRF

output

 Create Bookmarks



User: mlutz | v1.0.0

Executables – What and Why?



self-sustained programs

no further dependencies
no installation



unlocking the power of

high-level languages
libraries
rest of the world



for all environments

GxP
SCE



complexity overhead

develop
build
deploy



abstract complexity

standardize template
prebuild
focus on automation

give it a try



https://github.com/mainanalytics/ma_pytemp

References

- [1] – Image taken from: [File:Bud Spencer 1974.png - Wikimedia Commons](#) **22-Apr-2026: it is in the public domain**
- [2] – Image taken taken from: Michel Lutz – “An enquiry of southeastern Swabia traditions”. All rights beerserved ©