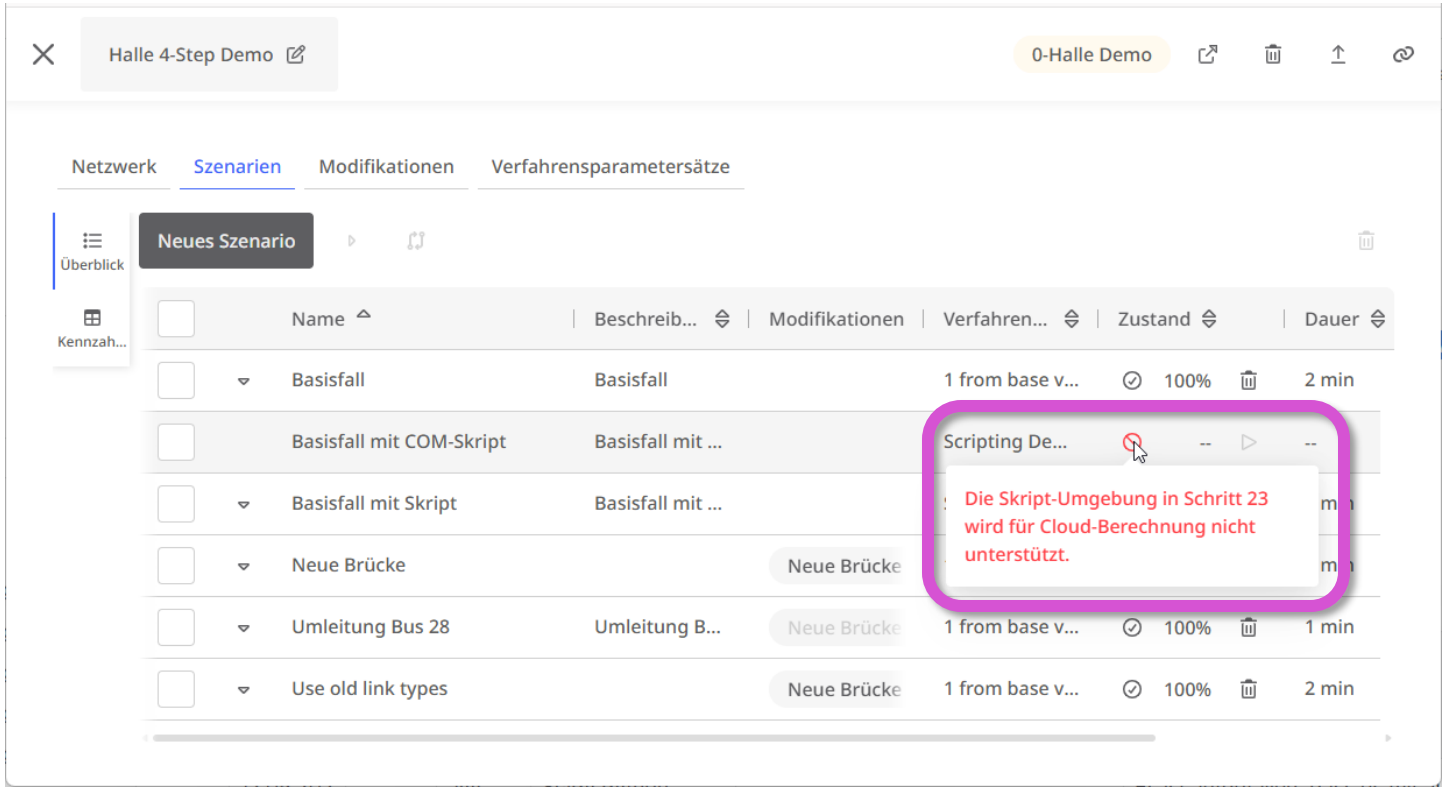
PTV GROUP
part of Umovity

Skripting in PTV Visum & PTV Hub

Motivation: PTV Hub

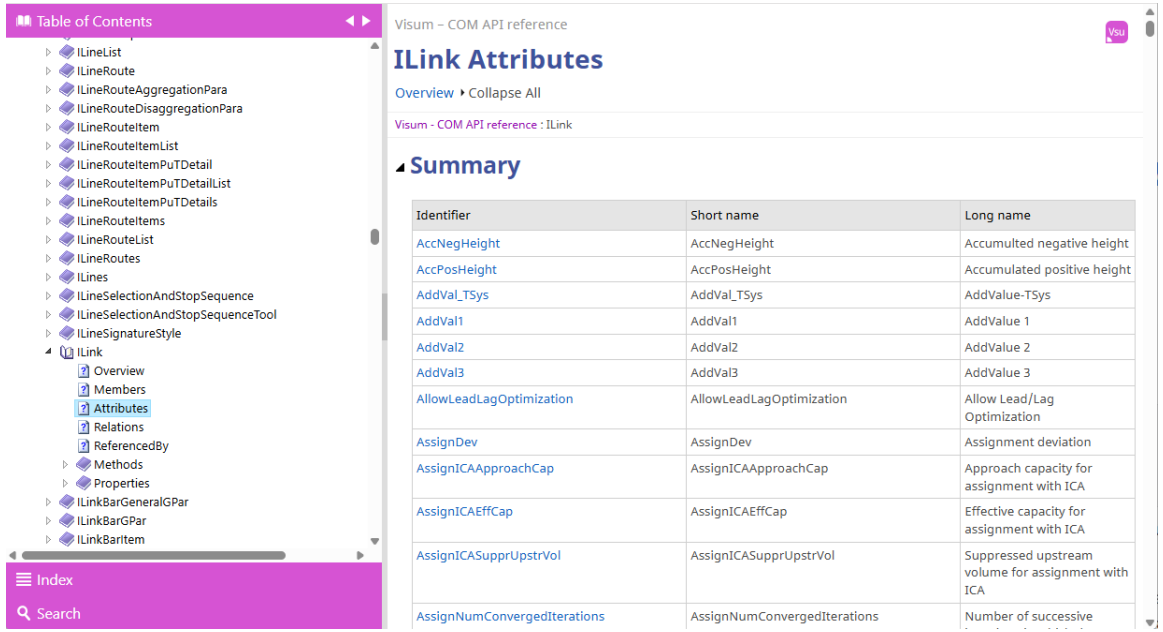


- Cloud computing in PTV Hub basiert auf Linux
- COM-Technologie wird unter Linux / in PTV Hub nicht unterstützt

→ Bereitstellung einer plattformunabhängigen Python-API

→ Ermöglicht es Skripte zu erstellen, die in Hub lauffähig sind

2 Motivation: COM vs Python



Visual Basic

```
Public Function AddDemandSegment( _
   ByVal Code  As String, _
   ByVal Mode  As Variant _
) As IDemandSegment
```

- COM-API sehr alt und nicht gut für modernes Programmieren geeignet
  - Nicht durchgängig typisiert (ENUMs, VARIANT, …)
  - Zugriff auf Attribute über ‘Attribut-IDs”

→ Bereitstellung einer modernen Python-API

→ Intuitivere Erstellung von Skripten, Code-Vervollständigung etc.

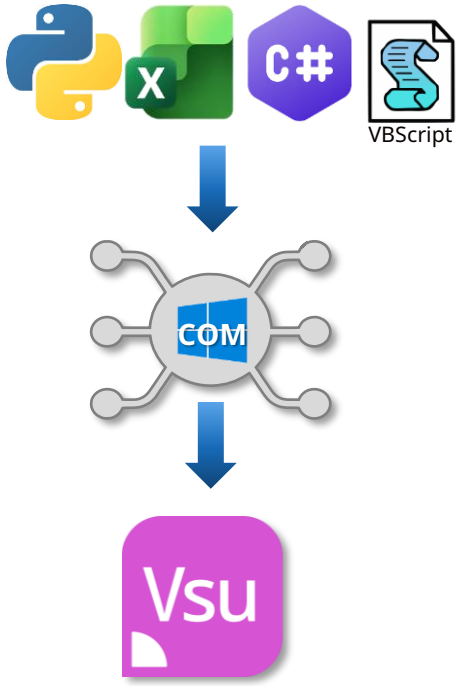
Cloud-kompatible Skripting-Umgebung

- Neue Python-API in Visum
- Ermöglicht die Nutzung von Skripten in Modellen für Cloud-Computing in PTV Hub
- Nutzung im Verfahren “Skript ausführen” d.h. nur im Verfahrensablauf
- Zugriff auf Datenmodell und Matrizen
- Kein Zugriff auf Fenster, Dateien etc.

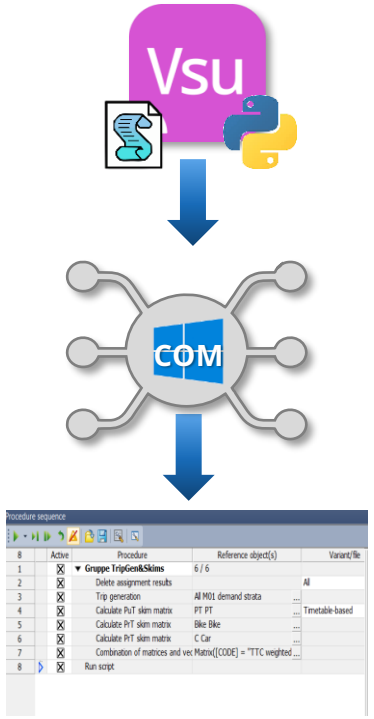
Alternativ: COM-basierte API ist weiterhin verfügbar und wird gepflegt!

Skripting-Anwendungen und APIs

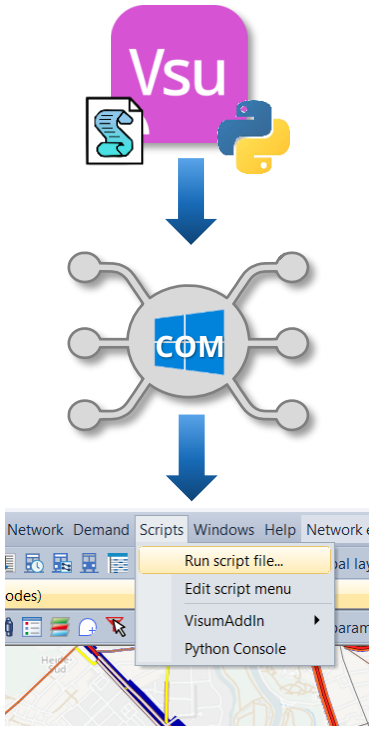
Automation



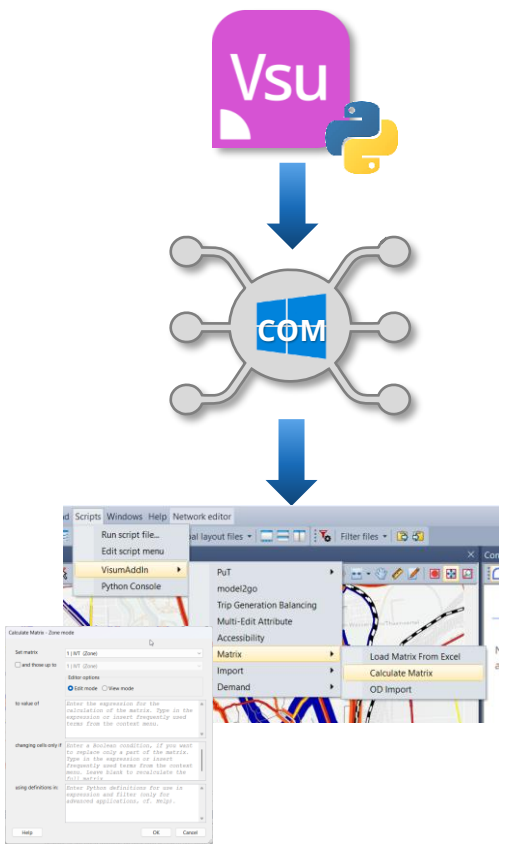
Verfahrensablauf



Menüaktionen



AddIns



Skripting-Anwendungen und APIs

Automation

VBScript

COM

Vsu

Verfahrensablauf

Vsu

COM

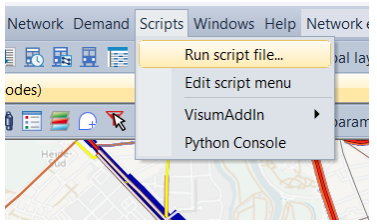
COM

Hub

Menüaktionen

Vsu

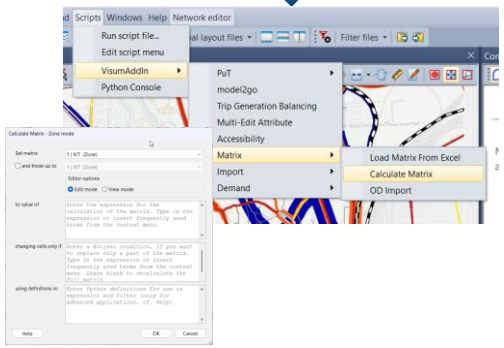
COM



AddIns

Vsu

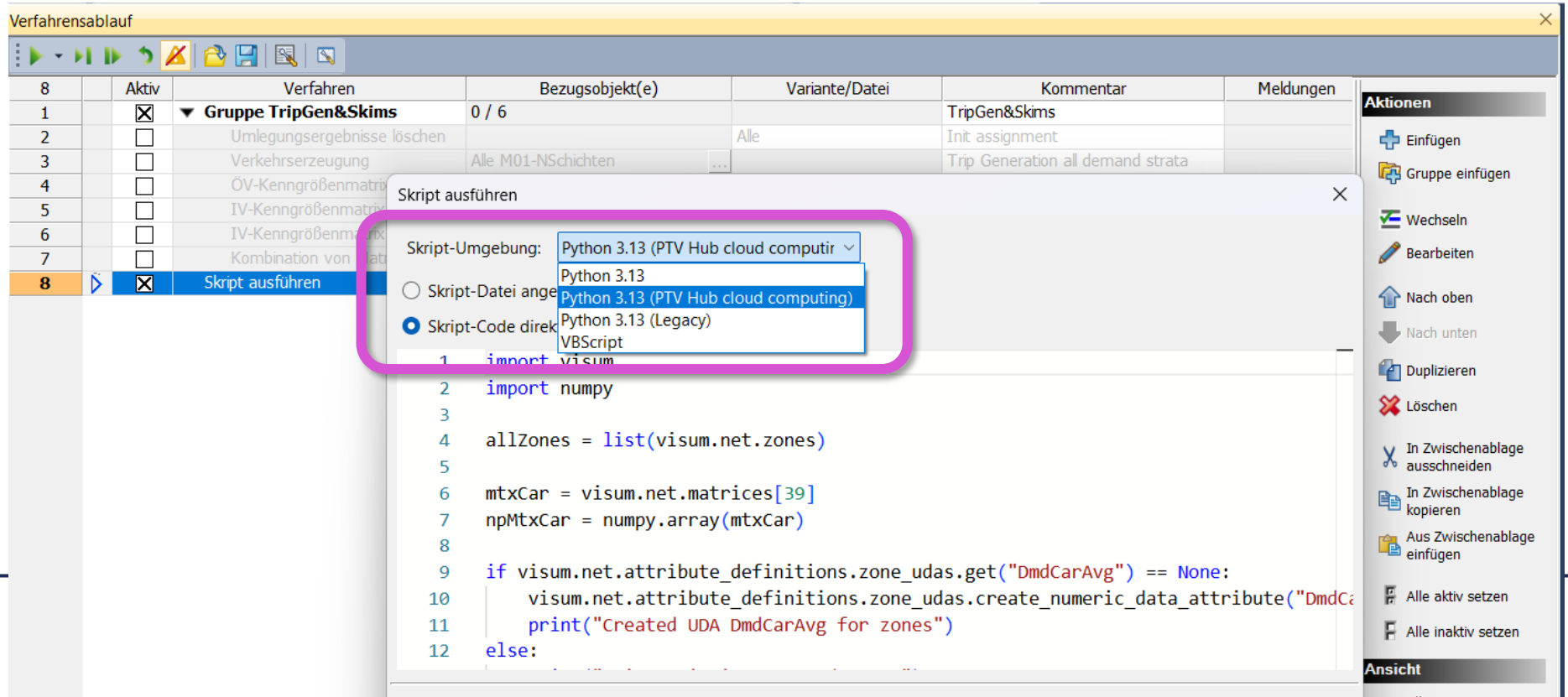
COM



Cloud-kompatible Skripting-Umgebung

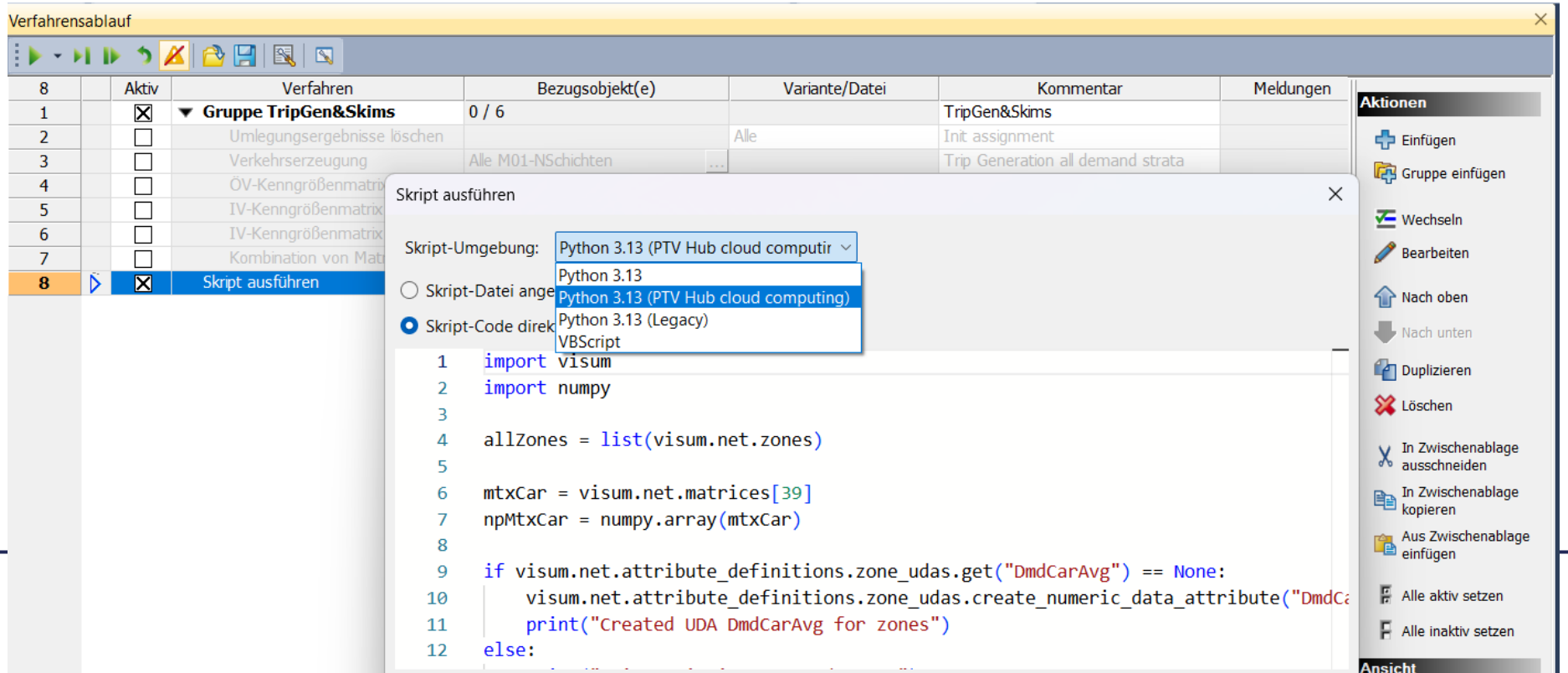
› Verfahren “Skript ausführen” - mehrere Python 3.13 Umgebungen:

- ***Standard: Python 3.13***: COM + Python-API (→ schneller)
- ***Legacy***: COM-Skripte mit Benutzeroberfläche (wxPython etc.)
- ***PTV Hub Cloud Computing***: Nur Python-API, kein COM, nur ‘*numpy*’, keine beliebigen 3rd-Party-Bibliotheken



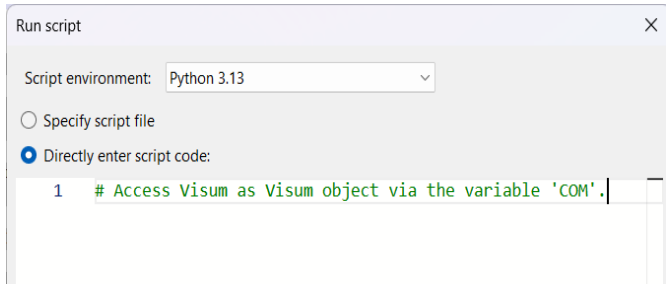
Python-Umgebung allgemein

- Alle Python-Umgebungen basieren auf Python 3.13
- Bibliotheken auf aktuellem Stand
- Weniger AddIns (→ siehe '*Neues In Visum 2026*')
- Diverse nicht mehr benutzte 3rd-Party-Bibliotheken entfernt (Softwaresicherheit)

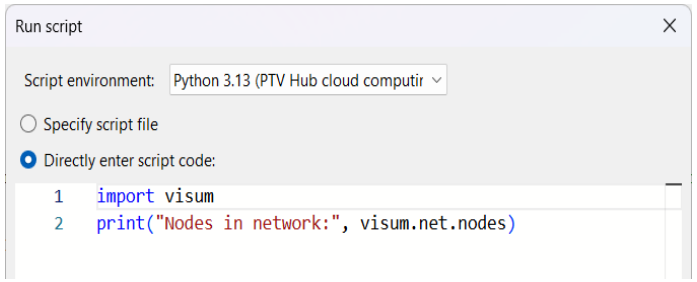


Python-Umgebungen

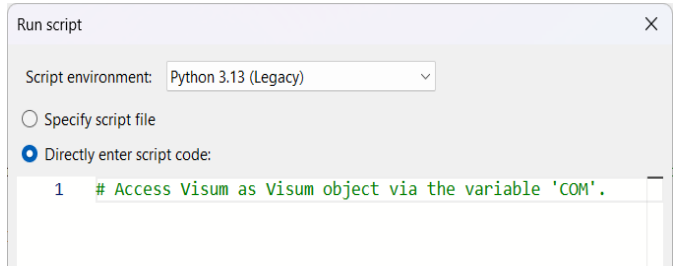
Standard



Hub



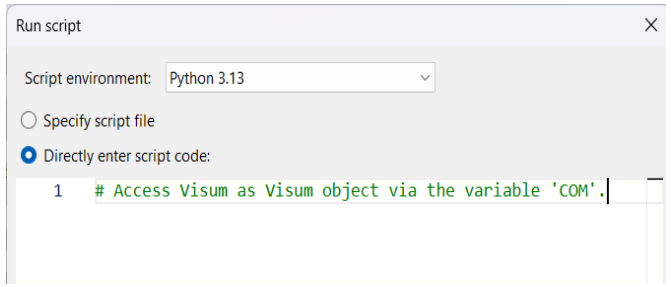
‚Legacy'



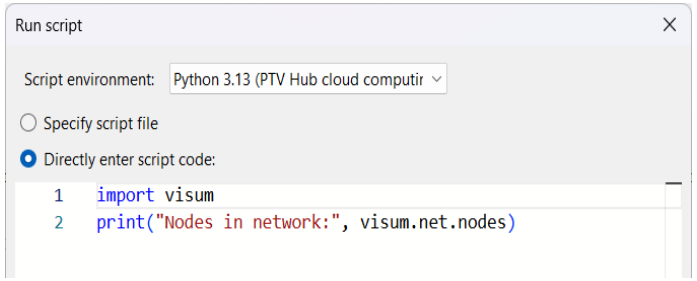
COM	✓	✗	✓
Python-API	✓	✓	✓
GUI (wx, TK)	✗	✗	✓
Verfahrensablauf	✓	✓	✓
PTV Hub	✗	✓	✗
Skriptmenu	✗	✗	✓
AddIns	✗	✗	✓
Vorinstallierte Bibliotheken	numpy, pandas, matplotlib, osgeo, , ...	numpy	numpy, pandas, matplotlib, osgeo, ... **+ wx, VisumPy**
Eigene Bibliotheken	✓	✓	✓

Python-Umgebungen

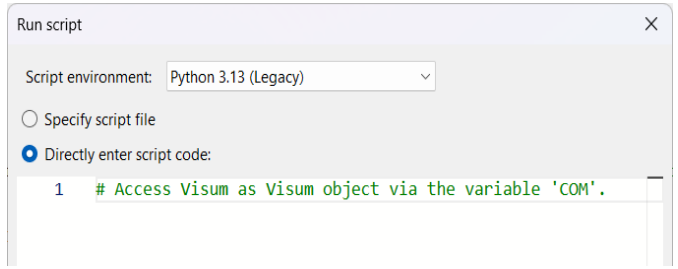
Standard



Hub



‚Legacy‘



	Standard	Hub	‚Legacy‘
COM	✓	✗	✓
Python-API	✓	✓	✓
GUI (wx, TK)	✗	✗	✓
Verfahrensablauf	✓	✓	✓
PTV Hub	✗	✓	✗
Skriptmenu	✗	✗	✓
AddIns	✗	✗	✓
Vorinstallierte Bibliotheken	numpy, pandas, matplotlib, osgeo, , …	numpy	numpy, pandas, matplotlib, osgeo, … **+ wx, VisumPy**
Eigene Bibliotheken	✓	✓	✓

COM vs Python-API

Skript-Code direkt eingeben:

```python
import visum

print("Example 01, using Python-API and direct attribute access")
fromNode = visum.net.nodes[984]
toNode = visum.net.nodes[10000675]

link1 = visum.net.links.find_by_nodes(fromNode,toNode)
reverse_link1 = link1.reverse_link
print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no, link1.to_node_no))
print("Reverse link {} is of type {} with volume {:.2f}". \
format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[visum.network.AHP.AP]))
```

Skript-Code direkt eingeben:


```python
# Greifen Sie über die Variable 'Visum' auf Visum als COM-Objekt zu.
print("Example 01, using COM-API")
fromNode = Visum.Net.Nodes.ItemByKey(984)
toNode = Visum.Net.Nodes.ItemByKey(10000675)

link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.AttValue("FromNodeNo"), link1.AttValue("ToNodeNo")))
link2 = Visum.Net.Links.ItemByKey(10000675, 984)
print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

Skript-Code direkt eingeben:

```python
import visum

print("Example 01, using Python-API and direct attribute access")
fromNode = visum.net.nodes[984]
toNode = visum.net.nodes[10000675]

link1 = visum.net.links.find_by_nodes(fromNode,toNode)
reverse_link1 = link1.reverse_link
print("Link {} connects nodes {} and {}".format(link1 
print("Reverse link {} is of type {} with volume 
format(reverse_link1.no, reverse_link1.type
```

Demo

Skript-Code direkt eingeben:


```com
# Greifen Sie über die Variable 'Visum
print("Example 01, using COM-API")
fromNode = Visum.Net.Nodes.ItemByKey(984)
toNode = Visum.Net.Nodes.ItemByKey(10000675)

link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.AttValue("FromNodeNo"), link1.AttValue("ToNodeNo")))
link2 = Visum.Net.Links.ItemByKey(10000675, 984)
print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

Skript-Code direkt eingeben:

```python
import visum

print("Example 01, using Python-API and direct attribute access")
fromNode = visum.net.nodes[984]
toNode = visum.net.nodes[10000675]

link1 = visum.net.links.find_by_nodes(fromNode,toNode)
reverse_link1 = link1.reverse_link
print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no, link1.to_node_no))
print("Reverse link {} is of type {} with volume {:.2f}". \
format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[visum.network.AHP.AP]))
```

Skript-Code direkt eingeben:


```python
# Greifen Sie über die Variable 'Visum' auf Visum als COM-Objekt zu.
print("Example 01, using COM-API")
fromNode = Visum.Net.Nodes.ItemByKey(984)
toNode = Visum.Net.Nodes.ItemByKey(10000675)

link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.AttValue("FromNodeNo"), link1.AttValue("ToNodeNo")))
link2 = Visum.Net.Links.ItemByKey(10000675, 984)
print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

Skript-Code direkt eingeben:

```
1   import visum
2
3   print("Example 01, using Python-API and direct attribute access")
4   fromNode = visum.net.nodes[984]
5   toNode = visum.net.nodes[10000675]
6
7   link1 = visum.net.links.find_by_nodes(fromNode,toNode)
8   reverse_link1 = link1.reverse_link
9   print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no
10  print("Reverse link {} is of type {} with volume {:.2f}". \
11  format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[vi
```

```
9   print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

Skript-Code direkt eingeben:

```python
import visum

print("Example 01, using Python-API and direct attribute access")
fromNode = visum.net.nodes[984]
toNode = visum.net.nodes[10000675]

link1 = visum.net.links.find_by_nodes(fromNode,toNode)
reverse_link1 = link1.reverse_link
print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no, link1.to_node_no))
print("Reverse link {} is of type {} with volume {:.2f}". \
format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[visum.network.AHP.AP]))
```

Skript-Code direkt eingeben:


```python
# Greifen Sie über die Variable 'Visum' auf Visum als COM-Objekt zu.
print("Example 01, using COM-API")
fromNode = Visum.Net.Nodes.ItemByKey(984)
toNode = Visum.Net.Nodes.ItemByKey(10000675)

link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.AttValue("FromNodeNo"), link1.AttValue("ToNodeNo")))
link2 = Visum.Net.Links.ItemByKey(10000675, 984)
print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

Skript-Code direkt eingeben:

```
1   import visum
2
3   print("Example 01, using Python-API and direct attribute access")
4   fromNode = visum.net.nodes[984]
5   toNode = visum.net.nodes[10000675]
6
7   link1 = visum.net.links.find_by_nodes(fromNode,toNode)
8   reverse_link1 = link1.reverse_link
9   print("Link {} connects nodes {} and {}".
10  print("Reverse link {} is of type {} with volume
11  format(reverse_link1.no, reverse_link1.type_no, reve
```

visum **statt** Visum

[id] **statt** ItemByKey(id)

Skript-Code direkt eingeben:

```
1   # Greifen Sie über die Variable 'Visum' auf Visum als COM-Objekt zu.
2   print("Example 01, using COM-API")
3   fromNode = Visum.Net.Nodes.ItemByKey(984)
4   toNode = Visum.Net.Nodes.ItemByKey(10000675)
5
6   link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
7   print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.Att
```

COM vs Python-API

Skript-Code direkt eingeben:

```python
import visum

print("Example 01, using Python-API and direct attribute access")
fromNode = visum.net.nodes[984]
toNode = visum.net.nodes[10000675]

link1 = visum.net.links.find_by_nodes(fromNode,toNode)
reverse_link1 = link1.reverse_link
print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no, link1.to_node_no))
print("Reverse link {} is of type {} with volume {:.2f}". \
format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[visum.network.AHP.AP]))
```

Skript-Code direkt eingeben:


```python
# Greifen Sie über die Variable 'Visum' auf Visum als COM-Objekt zu.
print("Example 01, using COM-API")
fromNode = Visum.Net.Nodes.ItemByKey(984)
toNode = Visum.Net.Nodes.ItemByKey(10000675)

link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.AttValue("FromNodeNo"), link1.AttValue("ToNodeNo")))
link2 = Visum.Net.Links.ItemByKey(10000675, 984)
print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

```
6
7   link1 = visum.net.links.find_by_nodes(fromNode,toNode)
8   reverse_link1 = link1.reverse_link
9   print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no, link1.to_node_no))
10  print("Reverse link {} is of type {} with volume {:.2f}". \
11  format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[visum.network.AHP.AP]))
```

link2.type_no **statt** link2.AttValue(„TypeNo“)

Direkter Zugriff **statt** ‚Attribut-ID‘

[visum.network.AHP.AP] **statt** „(AP)“


```
abl
M-A
Ite
emByKey(10000675)

mByKey(fromNode, toNode)
 nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.Att lue("FromNodeNo"), link1.AttValue
mByKey(10000675, 984)
ype {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

Skript-Code direkt eingeben:

```python
import visum

print("Example 01, using Python-API and direct attribute access")
fromNode = visum.net.nodes[984]
toNode = visum.net.nodes[10000675]

link1 = visum.net.links.find_by_nodes(fromNode,toNode)
reverse_link1 = link1.reverse_link
print("Link {} connects nodes {} and {}".format(link1.no, link1.from_node.no, link1.to_node_no))
print("Reverse link {} is of type {} with volume {:.2f}". \
format(reverse_link1.no, reverse_link1.type_no, reverse_link1.vol_veh_prt[visum.network.AHP.AP]))
```

Skript-Code direkt eingeben:


```python
# Greifen Sie über die Variable 'Visum' auf Visum als COM-Objekt zu.
print("Example 01, using COM-API")
fromNode = Visum.Net.Nodes.ItemByKey(984)
toNode = Visum.Net.Nodes.ItemByKey(10000675)

link1 = Visum.Net.Links.ItemByKey(fromNode, toNode)
print("Link {:.0f} connects nodes {:.0f} and {:.0f})".format(link1.AttValue("No"),link1.AttValue("FromNodeNo"), link1.AttValue("ToNodeNo")))
link2 = Visum.Net.Links.ItemByKey(10000675, 984)
print("Reverse Link is of type {:.0f} with volume {:.2f}".format(link2.AttValue("TypeNo"), link2.AttValue("VolVehPrt(AP)")))
```

COM vs Python-API

```
ing Python-API and accessing attributes through Attribute-ID")
.nodes[984]
odes[10000675]

visum.net.links.get(5115)
= visum.net.links[5110]
nks.find_by_nodes(fromNode,toNode)
1.reverse_link
ts node {} and node {}".format(link1.no, link1.from_node_no, link1.to_node_no))
s of type {} with volume {}".format(reverse_link1.attributes["TypeNo"], reverse_link1.attributes["VolVehPrt(AP)"])
```

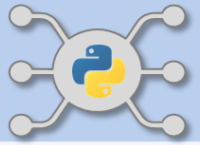
```
2  print("Example 01, using COM-API")
3  fromNode
4  toNode =
5
6  link1 =
7  print("L                                                    ue("ToNodeNo")))
8  link2 =
9  print("R
```

Alternativ:

`Link2.attributes(„TypeNo“)` **statt** `link2.AttValue(„TypeNo“)`

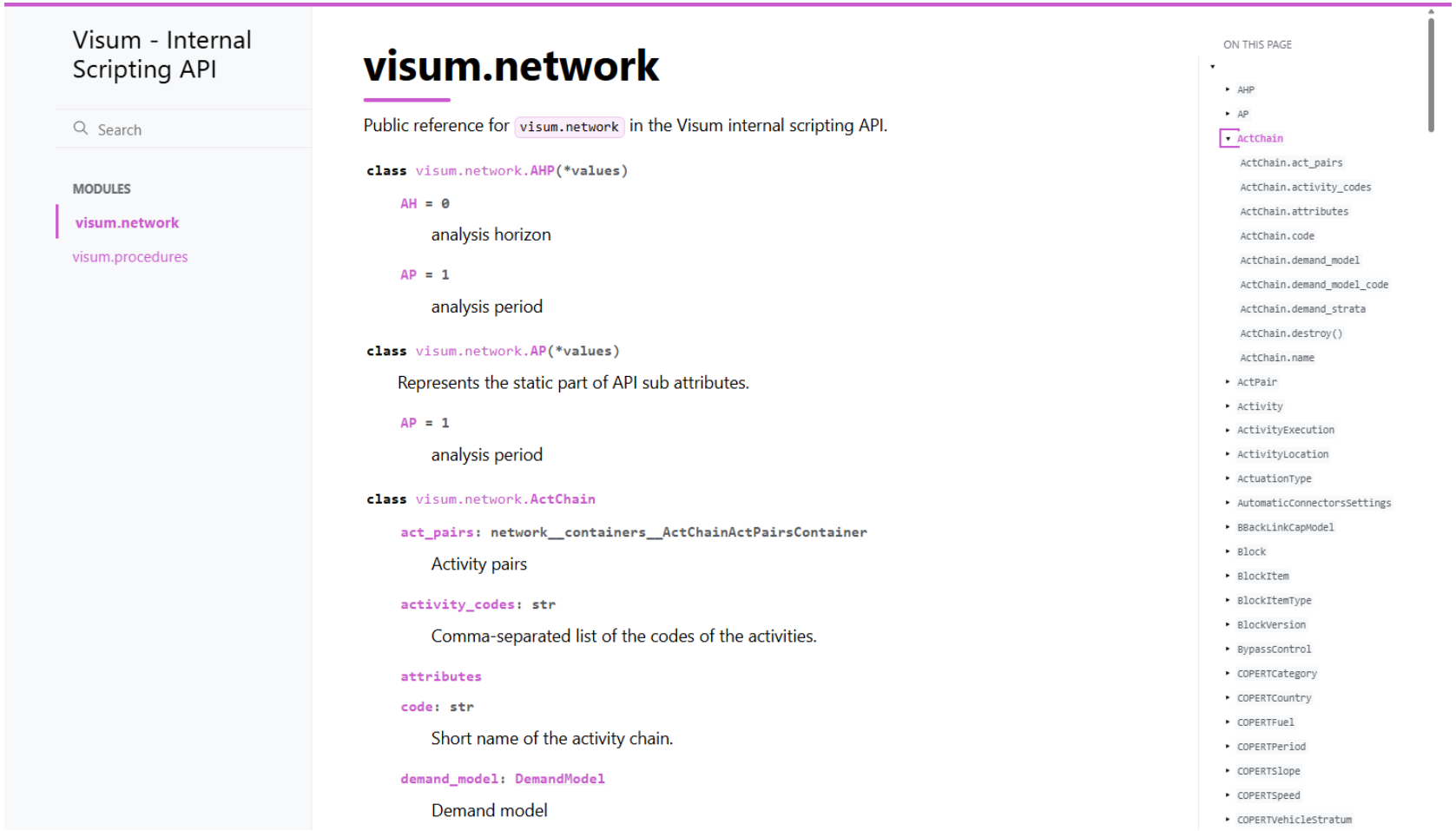
`visum.links.get(#)` **statt** `find_by_nodes()`

COM vs Python-API – Funktionsumfang(Stand 2026)

		
Netzobjekte	✓	✓
Attribute, Subattribute	✓	✓
BDAs, Indirekte Attribute	✓	✓
Matrizen	✓	✓
Filter (laden / speichern)	✓	✓
Analyse (Spinne, Isochronen)	✓	✗
Verfahrensparameter	✓	✗ / (✓)
Import / Export	✓	✗
Listen, Fenster	✓	✗
Screenshots, Drucken	✓	✗
…	✓	✗

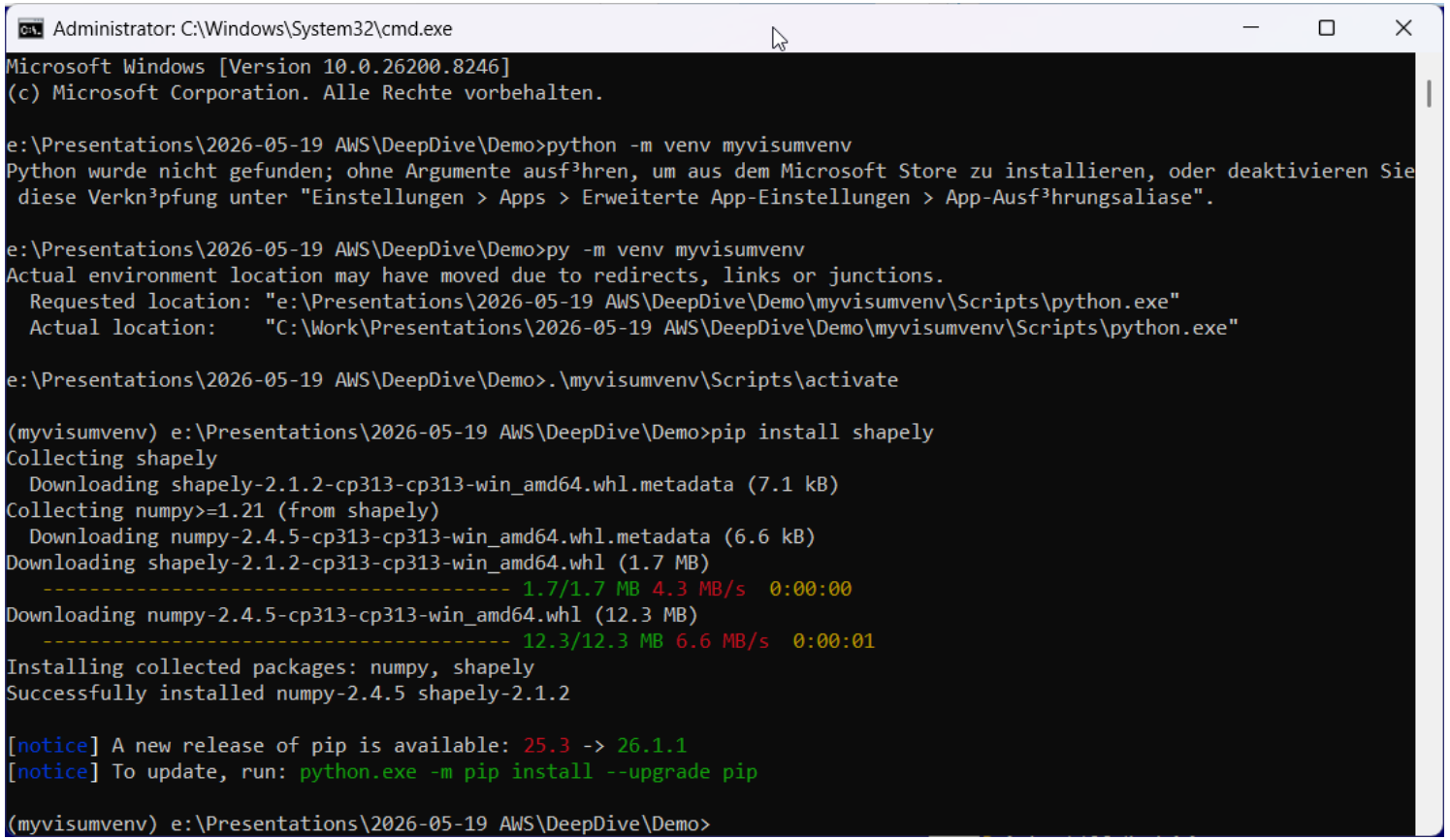
Dokumentation

- API-Referenz
  - in Arbeit ☺
- Beispiele
  - In Arbeit ☺
- Code-Bervolsständigung im Skript-Editor
  - Visum 2027



Zusätzliche Bibliotheken nutzen
(in Standard-Umgebung, nicht für PTV Hub)

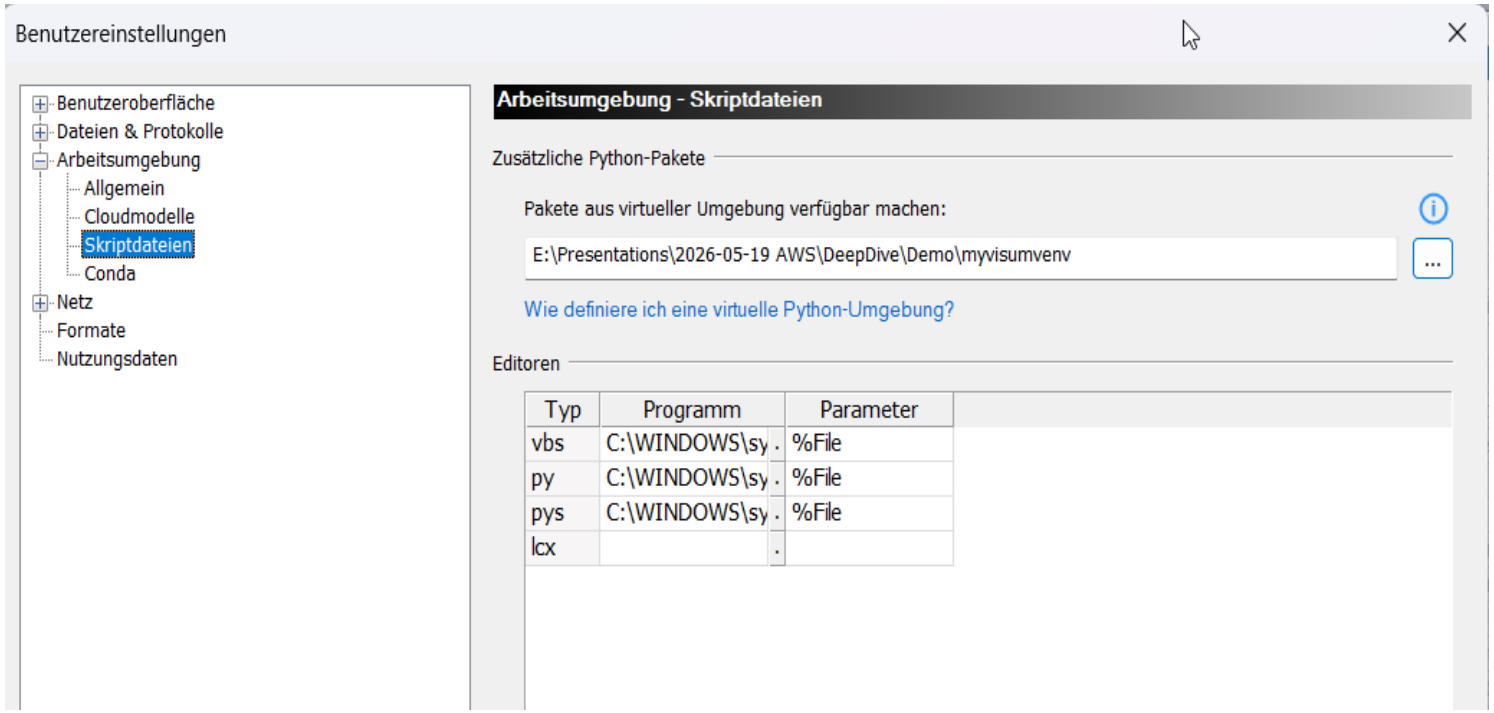
- Virtuelle Python-Umgebung (venv) erstellen
- Virtuelle Umgebung aktivieren
- Bibliotheken installieren mit *pip*



Zusätzliche Bibliotheken nutzen
(in Standard-Umgebung, nicht für PTV Hub)

- Virtuelle Umgebung
  in Visum-Benutzereinstellungen
  auswählen

- Ggf. neu starten




PTV GROUP
part of Umovity