

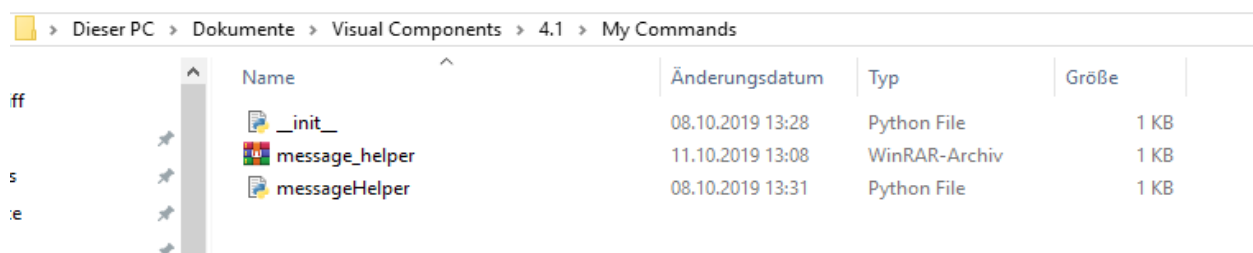
ADDON – MessageHelper für Visual Components

Komponentenbenachrichtigungen, Alarme und bestimmte Reaktionen auf Triggervorgänge werden in der Software „Visual Components V 4.1“ am unteren Bildschirmrand in einer Konsole ausgegeben.



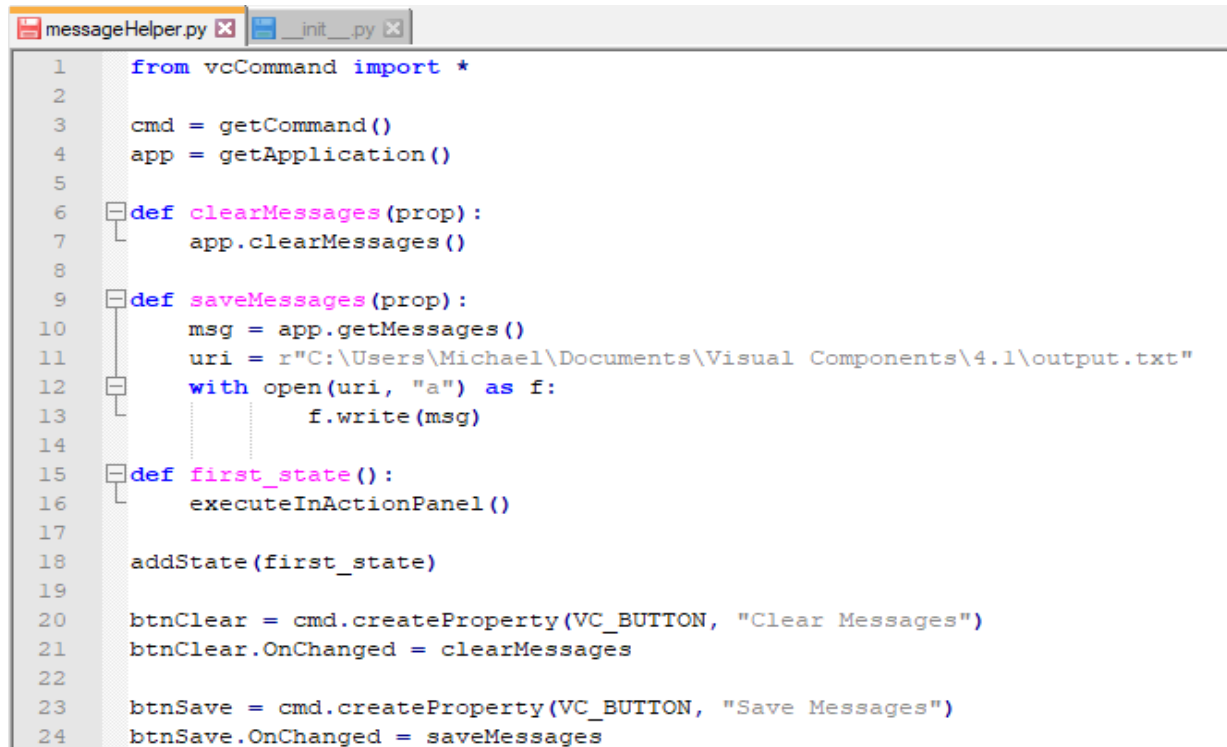
Um Produktionsvorgänge, Sensoren- und Aktorenaktivität verstehen und auswerten zu können, war es von Nöten ein AddOn zu konzipieren, dass dazu in der Lage ist den Inhalt der Kommandoausgabe bei Bedarf zu löschen (**clear**) und in einer separaten Textdatei abspeichern zu können (**save**). Zur Fehleranalyse und -behebung ist es ebenfalls von großer Bedeutung Kontrolle über die Ausgaben zu haben. In *Visual Components* hat man die Möglichkeit die Python-API für eigene Programmiererweiterungen zu nutzen.

Um das Addon „**messageHelper**“ verwirklichen zu können, ist es nötig zwei Python-Files, nämlich **messageHelper.py** und **__init__.py**, zu erstellen und diese im Dokumentenordner von *Visual Components* zu speichern.

A screenshot of a Windows File Explorer window. The address bar shows the path: "Dieser PC > Dokumente > Visual Components > 4.1 > My Commands". The main area displays a table of files and folders.

	Name	Änderungsdatum	Typ	Größe
ff	__init__	08.10.2019 13:28	Python File	1 KB
s	message_helper	11.10.2019 13:08	WinRAR-Archiv	1 KB
:e	messageHelper	08.10.2019 13:31	Python File	1 KB

Für die Erstellung der beiden Pythonskripte wird ein Texteditor benötigt. Folgender Code befindet sich in der **messageHelper.py**-Datei:



```
1  from vcCommand import *
2
3  cmd = getCommand()
4  app = getApplication()
5
6  def clearMessages(prop):
7      app.clearMessages()
8
9  def saveMessages(prop):
10     msg = app.getMessage()
11     uri = r"C:\Users\Michael\Documents\Visual Components\4.1\output.txt"
12     with open(uri, "a") as f:
13         f.write(msg)
14
15  def first_state():
16     executeInActionPanel()
17
18     addState(first_state)
19
20  btnClear = cmd.createProperty(Vc_BUTTON, "Clear Messages")
21  btnClear.OnChanged = clearMessages
22
23  btnSave = cmd.createProperty(Vc_BUTTON, "Save Messages")
24  btnSave.OnChanged = saveMessages
```

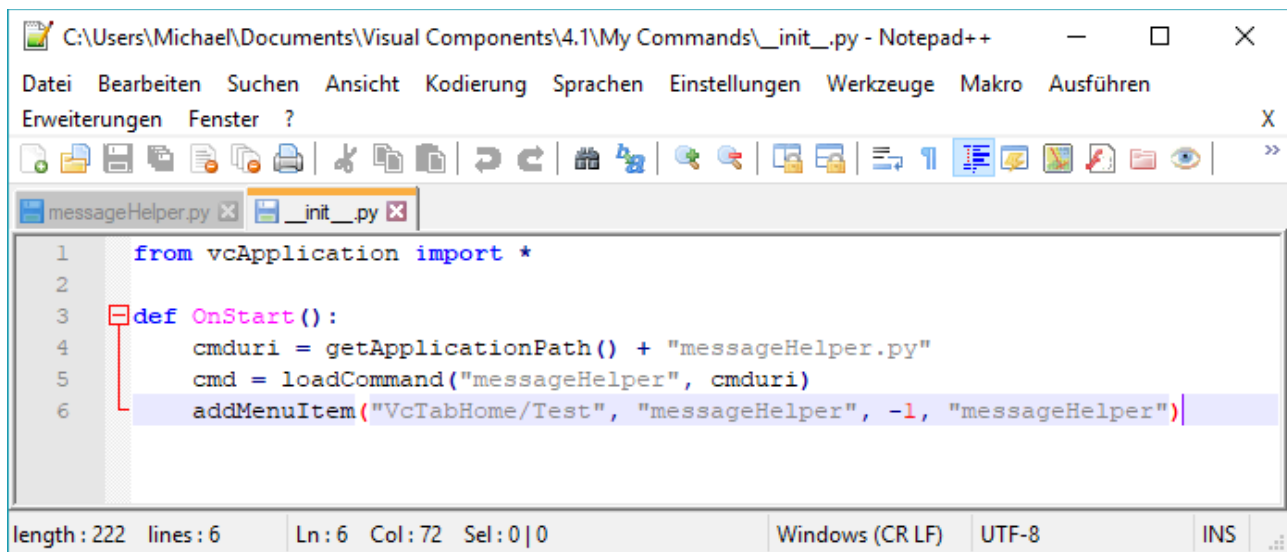
Zum Code:

In Zeile eins werden sämtliche Python-Methoden (*) aus der Bibliothek **vcCommand** in das Skript importiert. Die Variable **cmd** deklariert und mit der Methode **getCommand()** initialisiert. Sie gibt einen Befehl mit einem angegebenen Namen zurück. In der darunterliegenden Zeile wird die Variable **app** mit der Methode **getApplication()** initialisiert. In der Definition des Events **clearMessages** wird festgelegt, dass nach Drücken des Buttons *Clear* die Outputnachrichten im Ausgabefeld gelöscht werden. In der Definition des darunterliegenden Events (**saveMessages**) wird eine Variable **msg** deklariert. Sie wird mit der Methode **getMessage()** initialisiert, welche mit der Variable **app** aufgerufen wird. Die Variable darunter (**uri**) wird mit dem absoluten Pfad, in dem später die Ausgabenachrichten abgespeichert werden und mit der Aufforderung den Inhalt der Variable **msg** dort zu speichern, initialisiert. Damit nach Drücken auf einen Button erkannt wird, dass es sich dabei um eine Statusänderung handelt, ist es nötig ein Event **first_state** zu definieren (Zeile 15).

In Zeile 20 wird noch eine Variable **btnClear** zum löschen des Ausgabetextes angelegt. Nachdem der Button geklickt wurde, also eine Zustandsänderung erfolgt ist (**OnChanged**), wird das ausgeführt, was unter **clearMessages** definiert wurde.

In Zeile 23 wird selbiges noch einmal mit der Variable **btnSave** zum Speichern der Ausgabenachrichten festlegt. Wird hier eine Zustandsänderung festgestellt (Button wird gedrückt), wird das ausgeführt, was in **saveMessages** definiert wurde.

__init__.py ist eine Initialisierungsdatei. Wenn sich eine init-Datei in einem Ordner befindet, wird dieser von Python als Package angesehen und kann importiert werden. Folgender Code befindet sich in der **__init__.py**-Datei:



```
1  from vcApplication import *
2
3  def OnStart():
4      cmduri = getApplicationPath() + "messageHelper.py"
5      cmd = loadCommand("messageHelper", cmduri)
6      addMenuItem("VcTabHome/Test", "messageHelper", -1, "messageHelper")
```

length: 222 lines: 6 Ln: 6 Col: 72 Sel: 0 | 0 Windows (CR LF) UTF-8 INS

Zum Code:

In Zeile eins werden sämtliche Methoden (*) der vcApplication in das Skript importiert. Im definierten **OnStart**-Event wird die Variable **cmduri** mit der Methode **getApplicationPath** (gibt den Pfad der Software zurück) und der Datei **messageHelper.py** initialisiert. In Zeile sechs wird noch mit der Methode **addMenuItem()** der Reiter in der grafischen Benutzeroberfläche festgelegt.