AUTODESK **AUTOCAD Plant 3D**

AutoCAD Plant 3D – Python Scripting

Trainingshandbuch

Skripte für Katalogbauteile

Equipment Skript Format

Nozzle Katalog

Entwicklungsumgebung – Rohrhalterungen - Schwerlastrinnen

AUTODESK **AUTOCAD PLANT 3D**

Plant 3D Python - Skripting

1. Auflage / September 2020

A. Seitz Ingenieur GmbH

Vorbemerkungen

Dieses Handbuch beschreibt die Erstellung von Python Skripten für AutoCAD Plant 3D. Grundlegende Kenntnisse in der Rohrklassen und Katalogerstellung sind Voraussetzung für das Verständnis der Ausführungen. Kenntnisse der Programmiersprache Python sind hilfreich aber nicht unbedingt erforderlich.

Erstellen Sie immer eine Sicherungskopie Ihres aktuellen Projektes, bevor Sie die Beispiele aus dem Handbuch nachvollziehen. Durch fehlerhafte Einträge in den Datenbanken können Sie Ihr Projekt irreparabel beschädigen.

Sonderzeichen in Objektnamen können zu Problemen mit SQL Server führen. Verwenden Sie deshalb für den Projektnamen keine Sonderzeichen und keine Leerzeichen.

Ein „regulärer Bezeichner" ist ein Name der nur A-Z, a-z, 0-9 und einen Unterstrich (_) enthält.

Jede „nicht-reguläre" Kennung sollte in doppelte Anführungszeichen gesetzt werden. Dies funktioniert nicht bei allen Systemen. Beachten Sie diesen Aspekt bei der Formulierung Ihrer SQL – Statements.

Autor: Frank Kümmel

EuKo EDV und Konstruktionsservice

Seckentalstraße 2
66440 Blieskastel
E-Mail: FrankKuemmel@gmx.de

Für Fragen und Anregungen zum Buch steht Ihnen der Autor unter der oben genannten E-Mail Adresse gerne zur Verfügung.

Hinweise

Texte und Screenshots wurden mit größter Sorgfalt erstellt. Trotzdem können Fehler nicht ausgeschlossen werden. Die Autoren übernehmen keine Verantwortung oder Haftung für mögliche fehlerhafte Angaben und deren Folgen.

Benutzerdefinierte Skripte in Plant 3D

Benutzerdefinierte Skripts in Plant 3D

Einführung

Plant 3D enthält einen Katalog mit vordefinierten Komponenten. Um 3D – Darstellungen davon zu erstellen werden „**Scripts**" (Component Scripts) verwendet. Ein Skript ist eine kurze (**Python**) Subroutine, die die Dimensionen einer Komponente als Eingabe verarbeitet und daraus die 3D – Darstellung eines Bauteils generiert.

Etwa 20.000 verschiedene Skripts sind Teil von Plant 3D, zusätzliche Skripte können hinzugefügt werden. Die verfügbaren Skripts decken fast alle Arten von Komponenten ab, die in der Anlage häufig verwendet werden: Rohre, Bögen, Flansche, T-Stücke, Kreuze, Düsen, Böden, verschiedene Ventiltypen und vieles mehr.

Python

Python ist eine interpretierte, interaktive, objektorientierte Programmiersprache. Sie beinhaltet Module, Ausnahmen, dynamische Typisierung und dynamische Datentypen und Klassen. Python hat Schnittstellen zu vielen anderen Systemen und ist in C oder C++ erweiterbar. Ein weiterer Vorteil ist die klare gut verständliche Syntax.

Python ist als Erweiterungssprache für Anwendungen einsetzbar, die eine programmierbare Schnittstelle benötigen. Python ist portierbar: Die Programmiersprache läuft auf vielen Unix-Varianten, auf dem Mac und auf PCs unter Windows.

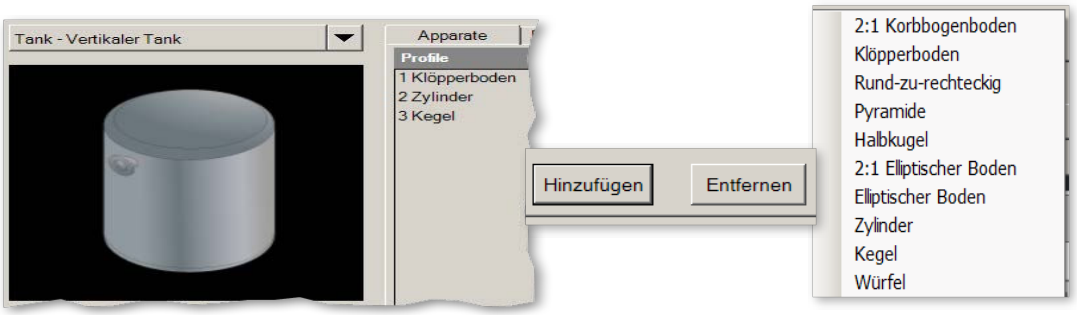
Wie arbeiteten Python und Plant 3D zusammen ?

Der Python Interpreter ist in die Anwendung durch C++ Routinen als Erweiterungsmodul eingebunden. Dieses Python-Modul wird dann in verschiedenen Content-Skripten verwendet. Diese Skripts erzeugen, mit einem Satz von Parametern, eine Komponente, die als **Content-Adapter** bezeichnet wird, die dann die Geometrie erzeugt

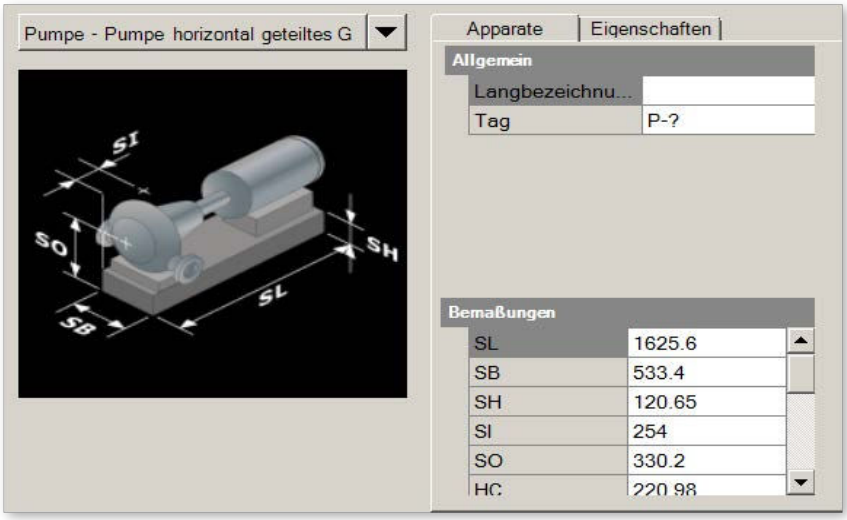
Arten von Python Scripts

Equipment Scripts

Plant 3D verwendet Skripte zu verschiedenen Zwecken. Z.B. für Ausrüstung (Equipment). Im Equipment – Dialog können Sie Geräte aus Grundformen aufbauen. Diese Grundformen sind fix im Programm definiert (**hard - coded**) und lassen sich nicht editieren.



Eine andere Art von Skript ist für die Erzeugung eines kompletten Equipments vorgesehen. Beispielsweise erzeugt das Centrifugal Pump-Skript die gesamte Pumpe, der Benutzer gibt Werte für die Parameter ein.



Die Equipment – Skripts werden in folgendem Ordner gespeichert:

C:\AutoCAD Plant 3D 20XX Content\CPak Common\equipment

Skripte für Katalogbauteile

Die selbsterstellten Skripte für Katalogbauteile wie Fittings, Ventile, Stutzen oder Rohrhalterungen werden im folgenden Ordner gespeichert:

C:\AutoCAD Plant 3D 20XX Content\CPak Common\CustomScripts

Entwicklungsumgebung

Im Gegensatz zur typischen Entwicklung moderner .Net – Sprachen steht Ihnen keine **integrierte Entwicklungsumgebung** (IDE) zum Kompilieren, Testen und Debuggen zur Verfügung. Die Kompilierung erfolgt durch das Ausführen eines speziellen Befehls in Plant 3D. Anschließend werden die Skripte in den Speicher für die AutoCAD – Sitzung geladen.

Zum Testen der Änderungen müssen Sie Plant 3D erneut starten und einen weiteren speziellen Befehl ausführen. Im Anhang, unter Entwicklungsumgebung, finden Sie eine Reihe nützlicher Werkzeuge die Sie beim Erstellen Ihrer Skripte unterstützen.

Als Texteditor eignet sich **Notepad++**. Die Bearbeitung von **Quelltext** wird besonders unterstützt. Für viele Programmiersprachen werden Syntax und Struktur durch typographische Mittel bzw. Code-Faltung hervorgehoben.

Notepad++ unterstützt **Python Script** und **XML**.

Das Kompilieren und Testen der Skripts gestaltet sich durch das Fehlen einer IDE relativ umständlich. Die notwendigen AutoCAD–Befehle können Sie z.B., auf einer Werkzeugpalette platzieren. Reinen Python–Code testen Sie am besten in einer Python Shell.

Einfaches Skript für Katalogbauteile

Um schnell zu Ergebnissen zu kommen werden wir, zu Beginn, ein einfaches Skript erstellen. Detaillierte Erklärungen folgen im weiteren Verlauf dieser Unterlage.

Stellen Sie sicher das ein Ordner namens **CustomScripts** im Verzeichnis:

"**C:\AutoCAD Plant 3D 20XXContent\CPak Common**",

existiert.

Erstellen Sie dort ein Python Script mit dem Namen: **TestScript.py**

```
from aqa.math import *
from varmain.primitiv import *
from varmain.custom import *
@activate(Group="Support",
 TooltipShort="Test script",
 TooltipLong="This is a custom Testscript",
 LengthUnit="in",
 Ports=1)
@group("MainDimensions")
@param(D=LENGTH, TooltipShort="Cylinder Diameter", Ask4Dist=True)
@param(L=LENGTH, TooltipLong="Lenth of the Cylinder")
@param(OF=LENGTH0)
@group(Name="meaningless enum")
@param(K=ENUM)
@enum(1, "align X")
@enum(2, "align Y")
@enum(3, "align Z")
def TESTSCRIPT(s, D=80.0, L=150.0, OF=-1, K=1, **kw):
    CYLINDER(s, R=D/2, H=L, O=0.0).rotateY(90)

```

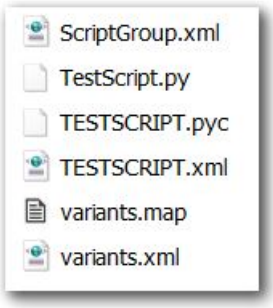
Nachdem Sie das Skript erstellt haben, müssen Sie es in Plant 3D registrieren.

Skript registrieren

Geben Sie die folgenden Kommandos in der Befehlszeile von Plant 3D ein:

(arxload "PnP3dACPAdapter.arx")

PLANTREGISTERCUSTOMSCRIPTS



Bei erfolgreicher Registrierung werden, die im Bild gezeigten Dateien, im Verzeichnis CustomScripts erzeugt.

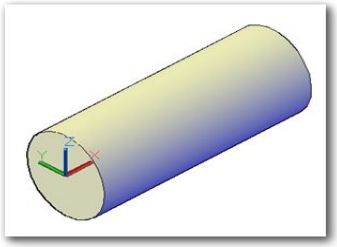
Skript testen

Nach der Registrierung schließen Sie Plant 3D und starten das Programm erneut.

Geben Sie die folgenden Kommandos in der Befehlszeile von Plant ein:

(arxload "PnP3dACPAdapter.arx")

(testacpscript "TESTSCRIPT")



Das Skript erzeugt einen Zylinder im Koordinatenursprung.

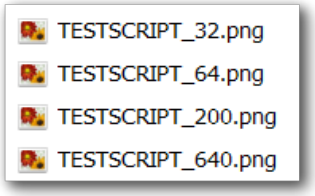
Vorschaubilder erstellen

Erstellen Sie 4 Bilder Ihres Bauteils. Die Bilder müssen folgendes Format haben:

32x32 Pixel / 64x64 Pixel / 240x240 Pixel

640x640 Pixel mit Bemaßungen.

Verwenden Sie den Befehl **PLANTSNAPSHOT**, um Bilder mit 32, 64 und 200 Pixel zu erstellen. Speichen Sie Ihre Bilder im ***.png** Format.

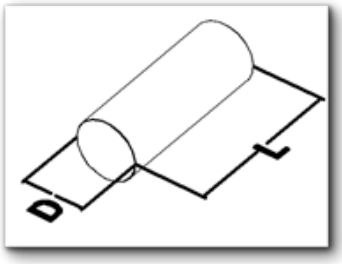


Leider entsprechen diese Grafiken nicht genau dem OOTB-Inhalt, da sich die Grafik-Engine geändert hat.

Speichern Sie Ihre Bilder im Ordner CustomScripts. Geben Sie den Dateien einen Namen, der dem Namen der Skriptdatei und der Größe in Pixel entspricht.

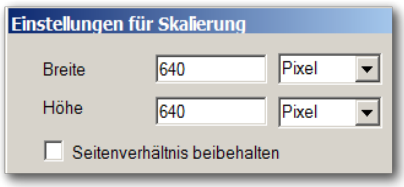
Zum Erstellen der Bilder können Sie jedes beliebige Programm benutzen das *.png – Dateien erzeugen kann. Die Bilder in diesem Handbuch wurden mit **Greenshot** erstellt.

Das System lädt diese Dateien und verarbeitet Sie entsprechend.



Das Bild mit den Bemaßungen erfordert etwas mehr Aufwand.

Erzeugen Sie z.B., mit den AutoCAD Werkzeugen, nebenstehendes Bild.



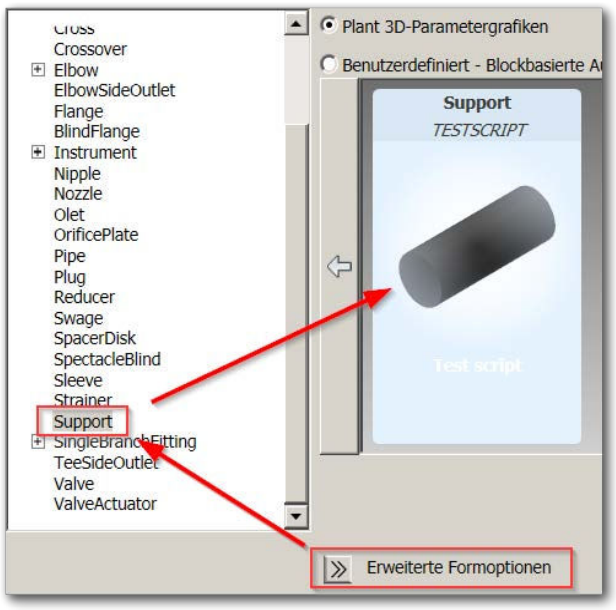
Ein geeignetes Werkzeug zum Zuschneiden des Bildes ist z.B., **Greenshot** (ein Freeware – Programm).

Egal welche Software Sie benutzen, Sie benötigen eine **Zuschnittfunktion** und Ihr Programm muss ***.png – Dateien** erzeugen können.

Erstellen Sie eine Datei mit dem Namen: **TESTSCRIPT_640.png**. Speichern Sie die Datei im Verzeichnis **CustomScripts**.

Anwendung im Spec - Editor

Starten Sie den Spec Editor, wählen Sie einen Katalog und klicken Sie: **Neue Komponente erstellen**.



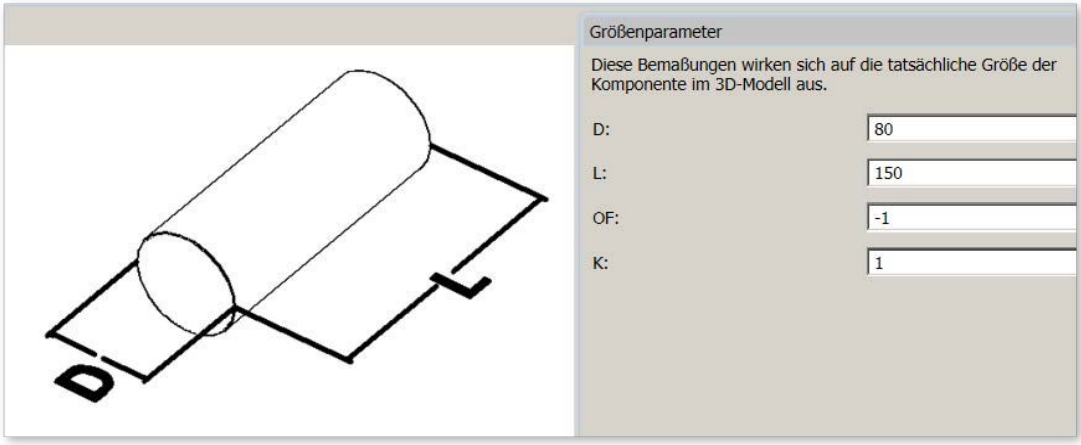
Klicken Sie:

Erweiterte Formoptionen.

Wählen Sie:

Support.

Unser neues Bauteil wird angezeigt und kann ausgewählt werden.



Das Bauteil kann, wie die vordefinierten Bauteile, konfiguriert und einer Rohrklasse hinzugefügt werden.

Die Funktion setpoint

Die Funktion **.setPoint** fügt einen Punkt an der Position **p** ein und definiert die Richtung **v** auf das nächste Teil in diesem Punkt.

Alternativ können wir auch einen Rotationswinkel **a** einstellen (für elliptische Flansche, um die richtige Ausrichtung einzustellen).

Das Argument **p** und **v** kann ein **mPoint**, **mVector** oder ein **3-Tupel (x, y, z)** sein. Der Drehwinkel **a** wird in Grad angegeben.

Aufruf aus Python: obj.setPoint(p,v) **oder** obj.setPoint(p,v,a)