CS LAB s.c.
ElectronicLaboratory

simCNC
software de control

Guía de scripts Python

Indice

I. Python - información básica 3

II. SimCNC y Python, ¿qué merece la pena saber? 4

III. Arranque y descripción del editor de script Python 5

IV. Ctrl + Space – sugerencias en el editor de script Python 6

V. Puertos digitales y analógicos - acceso directo 8

VI. Leer y guardar coordenadas de eje 17

VII. Desplazamiento de los ejes de la máquina 21

VIII. Sondeo 25

VIII. Revista de herramientas 29

IX. Husillo, refrigerante y niebla. 39

X. Desplazamiento de trabajo - bases materiales. 48

I. Python - información básica

El software de control SimCNC ha sido equipado con soporte de script que le permite automatizar actividades como el cambio de herramienta o la medición de la longitud de la herramienta. Sin embargo, la aplicación de scripts no se limita a estas actividades y, a veces, existe la necesidad de crear scripts mucho más complejos. Para facilitar a los usuarios de software la creación de sus propios scripts, simCNC utiliza Python como lenguaje de scripts.

Python es un lenguaje de programación de nivel alto para fines generales y con un amplio paquete de bibliotecas estándar, cuya característica principal es la legibilidad y claridad del código fuente. Por otro lado, la sintaxis del lenguaje es clara y concisa.

Tanto los principiantes como los profesionales usan Python. La gran popularidad de este lenguaje contribuye a la creación de muchos tutoriales y bibliotecas que se pueden encontrar en innumerables sitios web. Este estado de cosas provoca que la única barrera que pueda interponerse en nuestro camino para escribir nuestro propio guión sean nuestra propia voluntad e imaginación.

II. SimCNC y Python, ¿qué merece la pena saber?

Al instalar simCNC en el disco duro, Python se instala como un software separado. El software SimCNC se comunica con Python a través del protocolo de Internet UDP utilizando una biblioteca especialmente diseñada "___ COMM.pyc". Además de esta biblioteca, los desarrolladores también han preparado la biblioteca "___ DEVICE.pyc", que contiene un conjunto de funciones que le permiten controlar el software simCNC desde macros.

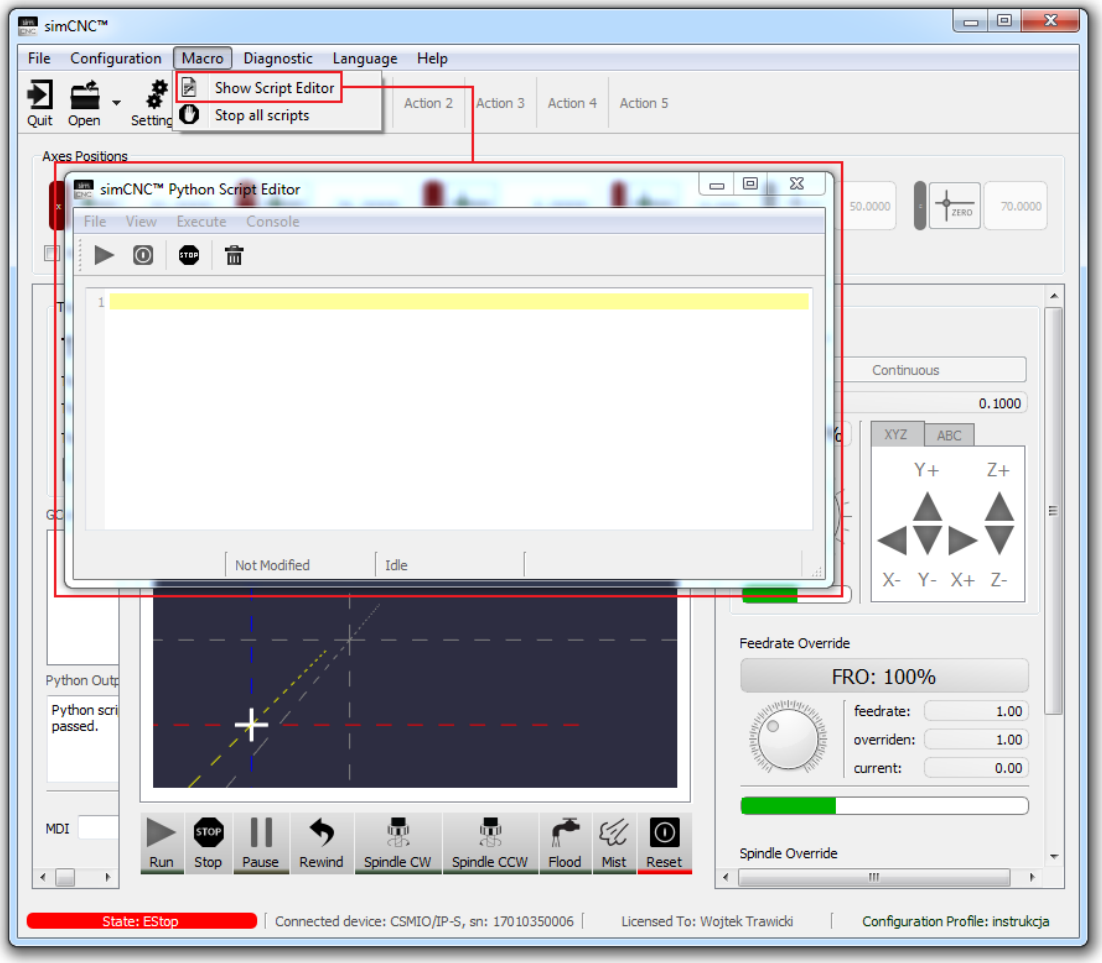
Esta solución le permite:

- utilizar un editor de programación externo avanzado con la opción de depuración, por ejemplo, Visual Studio Code.
- controlar el software simCNC a través de UDP en su red local desde otro PC.

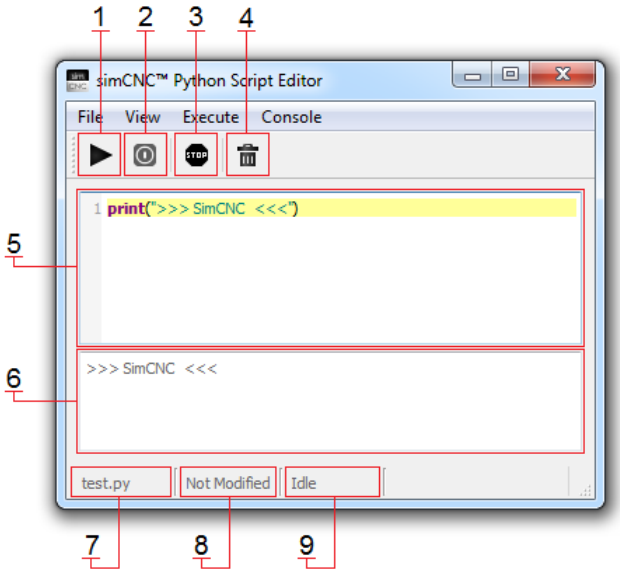
Las dos soluciones anteriores se analizarán con más detalle en la documentación adicional. No obstante, este manual hace referencia al editor de scripts integrado en el software simCNC.

III. Arranque y descripción del editor de script Python

Para ejecutar el editor de script Python, haga clic en la opción "Macro" (Macro) en la barra superior de la ventana de simCNC, a continuación seleccione " Show Script Editor " en la lista desplegable.



1. „Run” (Start) - ejecuta el script.
2. „Stop” (Stop) - detiene el script.
3. „Stop Trajectory Planer” (Detener el planificador de trayectoria): equivalente al botón "STOP" en la pantalla principal de simCNC.
4. „Clear Console” (Borrar consola): elimina el contenido de la ventana de la consola de Python (consulte el punto 6).
5. Editar el script de Python: aquí podemos crear o editar el script.
6. Consola de Python: el lugar donde se muestra información sobre scripts (errores y excepciones), así como sus propios mensajes (el ejemplo se muestra a continuación).
7. Nombre del script.
8. Información sobre si el script se ha modificado desde la última vez que se guardó.
9. Información sobre el estado del script.



IV. Ctrl + Space – sugerencias en el editor de script Python

El editor de script Python está equipado con un mecanismo de sugerencias para ejecutarlo en la ventana del editor, simplemente presionando las teclas "Ctrl" + "Space" simultáneamente. Esta combinación se puede utilizar en cualquier momento.

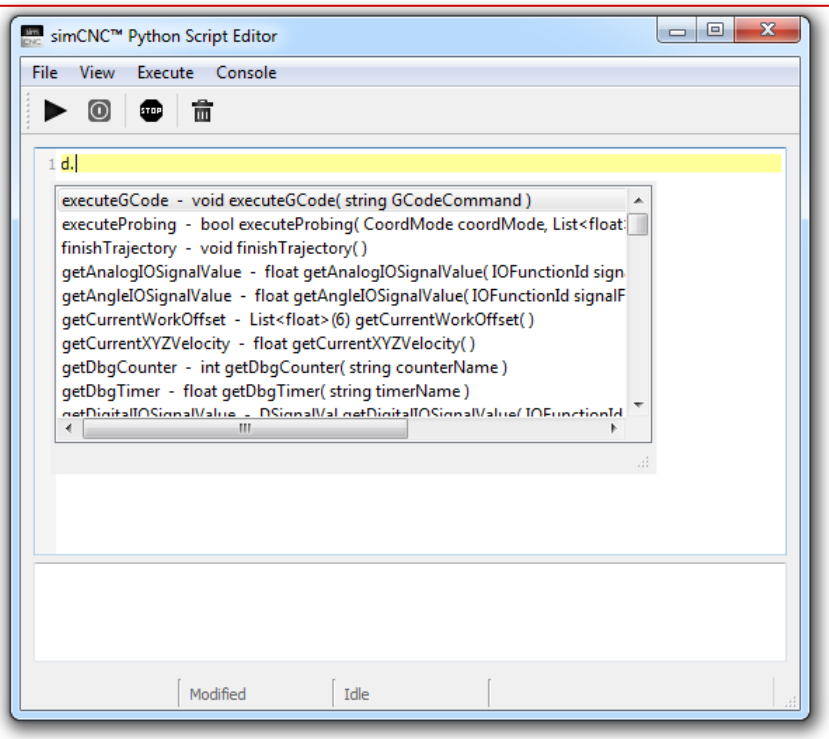
1) Cuando queremos ver la lista de funciones contenidas en la biblioteca "___ DEVICE.pyc".

En esta situación, escriba "d". y presiona las combinaciones de teclas "Ctrl" + "Space". Después de hacer esto, aparecerá una ventana que muestra la lista de funciones en la biblioteca "___ DEVICE.pyc"

Para seleccionar una de las funciones, debemos resaltarla con los cursores del teclado y confirmar la selección pulsando la tecla "Enter" o simplemente haciendo clic en la función seleccionada con el botón izquierdo del ratón.

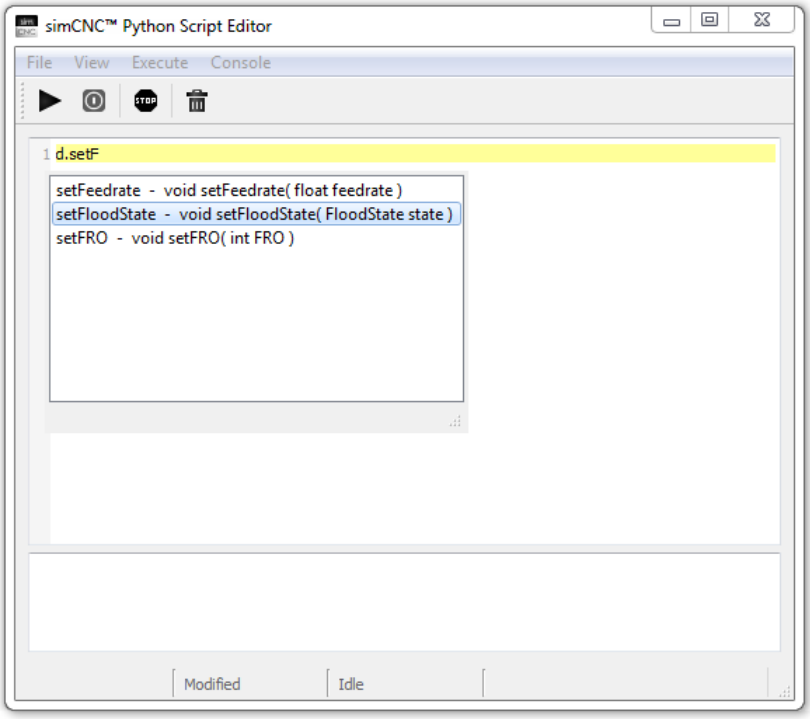
¡ADVERTENCIA!

La lista no incluye funciones que requieren la descarga de un módulo. Estamos hablando de las funciones que dan acceso directo a puertos digitales y analógicos.



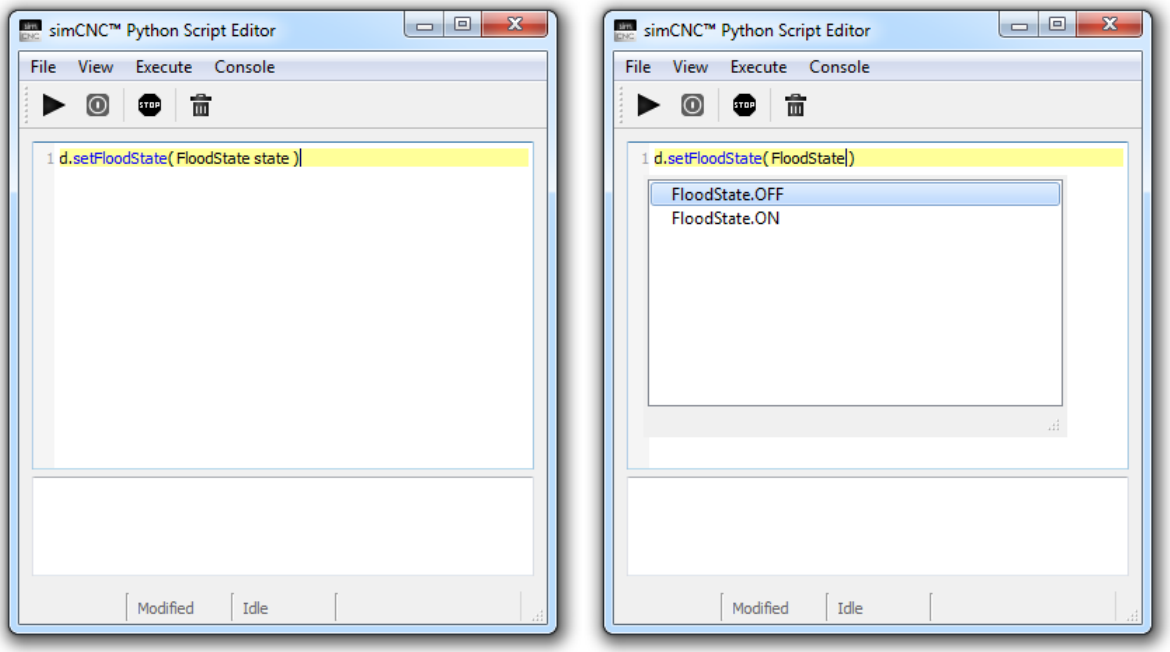
2) Cuando conocemos el nombre incompleto de la función:

En esta situación, también escribimos "d". y pulsamos la combinación de teclas "Ctrl" + "Space", pero esta vez no buscamos en la lista de funciones disponibles, sino que sólo escribimos el comienzo del nombre de la función que conocemos, y una ventana de sugerencias nos mostrará los finales coincidentes.



3) Cuando encontramos la función que necesitábamos, pero desconocemos el valor del argumento.

En esta situación, simplemente elimine el nombre del argumento y presione la combinación de teclas "Ctrl" + "Space" (a menos que se haya presionado antes), y la ventana de sugerencia nos mostrará todas las variantes coincidentes del valor del argumento.



V. Puertos digitales y analógicos - acceso directo

1) Descarga del módulo.

Para obtener acceso a los puertos digitales o dispositivos analógicos de CS-LAB, debe primero descargar el módulo.

Module d.getModule(ModuleType type, int ID) – Devuelve el módulo al dispositivo.

ARGUMENTOS DE FUNCIÓN:

ModuleType type – Tipo de dispositivo

DISTINGUIMOS:

- ModuleType.IP – Controladores CSMIO/IP-S, CSMIO/IP-A y CSMIO/IP-M
- ModuleType.MPG – Módulo de manipulación de eje manual CSMIO-MPG
- ModuleType.ENC – Módulo de roscado CSMIO-ENC
- ModuleType.IO – Módulo adicional de E/S CSMIO-IO
- ModuleType.DRV – Unidades SimDrive

int ID – Número del dispositivo

DISTINGUIMOS:

CSMIO-IO (ID de 0 a 15)
SimDrive (ID de 0 a 5)

Si utiliza varios módulos adicionales de E/S CSMIO-IO o servodrives simDrive, debe proporcionar el número de dispositivo correcto para poder distinguirlos. Esta regla no es aplicable al controlador CSMIO/IP, al módulo CSMIO-MPG y al CSMIO-ENC. En su caso, el número de dispositivo siempre es 0, porque siempre aparecen en el sistema de control individualmente.

RESULTADO DE LA FUNCIÓN:

Module - Módulo para dispositivos

EJEMPLOS:

```
mod_IP = d.getModule( ModuleType.IP, 0 )            # Descargue el módulo en el controlador CSMIO/IP #
mod_MPG = d.getModule( ModuleType.MPG, 0 )          # Descargue el módulo en CSMIO-MPG #
mod_ENC = d.getModule( ModuleType.ENC, 0 )          # Descargue el módulo en CSMIO-ENC #
mod_IO_0 = d.getModule( ModuleType.IO, 0 )          # Descargue el módulo en CSMIO-IO número 0#
mod_IO_15 = d.getModule( ModuleType.IO, 15 )        # Descargue el módulo en CSMIO-IO número 15#
mod_simDrive_0 = d.getModule( ModuleType.DRV, 0 )# Descargue el módulo en simDrive número 0 #
mod_simDrive_5 = d.getModule( ModuleType.DRV, 5 )# Descargue el módulo en simDrive número 5 #
```

¡ADVERTENCIA!

„mod_IP "," mod_MPG "," mod_ENC "," mod_IO_0 "," mod_IO_15 "," mod_simDrive_0 "y" mod_simDrive_5 "son los nombres de los módulos (identificadores) que se inventaron para los fines requeridos en las instrucciones. Los nombres de los módulos pueden crearse libremente, pero deben indicar claramente a qué dispositivos se aplican.

2) Puertos digitales

a) Lectura desde puertos digitales.

DIOPinVal getDigitalIO(IOPortDir direction, int digitalPin) – Devuelve el estado de un pin, un puerto de entrada digital o un puerto de salida (devuelve el estado de entrada o salida digital).

ARGUMENTOS DE FUNCIÓN:

IOPortDir direction – Especifica la dirección del puerto

DISTINGUIMOS:

IOPortDir.InputPort - Puerto de entrada
IOPortDir.OutputPort - Puerto de salida

int digitalPin – Número de pin (número de entrada o salida digital)

RESULTADO DE LA FUNCIÓN:

DIOPinVal – Estado del pin.

DISTINGUIMOS:

DIOPinVal.PinReset – Estado del pin bajo
DIOPinVal.PinSet – Estado del pin alto

EJEMPLOS:

Controlador CSMIO/IP-S, CSMIO/IP-A i CSMIO/IP-M

```
mod_IP = d.getModule( ModuleType.IP, 0 )

# Lectura del estado de la entrada digital núm. 8
if mod_IP.getDigitalIO( IOPortDir.InputPort, 8 ) == DIOPinVal.PinReset :
    print("CSMIO/IP digital input 8 = 0")
if mod_IP.getDigitalIO( IOPortDir.InputPort, 8 ) == DIOPinVal.PinSet :
    print("CSMIO/IP digital input 8 = 1")
```

```
# Lectura del estado de la salida digital núm. 4
if mod_IP.getDigitalIO( IOPortDir.OutputPort, 4 ) == DIOPinVal.PinReset :
    print("CSMIO/IP digital output 4 = 0")
if mod_IP.getDigitalIO( IOPortDir.OutputPort, 4 ) == DIOPinVal.PinSet :
    print("CSMIO/IP digital output 4 = 1")
```

CSMIO-MPG - Módulo de manipulación de eje manual

```
mod_MPG = d.getModule( ModuleType.MPG, 0 )

# Lectura del estado de la entrada digital núm. 3
if mod_MPG.getDigitalIO( IOPortDir.InputPort, 3 ) == DIOPinVal.PinReset :
    print("CSMIO-MPG digital input 3 = 0")
if mod_MPG.getDigitalIO( IOPortDir.InputPort, 3 ) == DIOPinVal.PinSet :
    print("CSMIO-MPG digital input 3 = 1")

# Lectura del estado de la salida digital núm. 0.
if mod_MPG.getDigitalIO( IOPortDir.OutputPort, 0 ) == DIOPinVal.PinReset :
    print("CSMIO-MPG digital output 0 = 0")
if mod_MPG.getDigitalIO( IOPortDir.OutputPort, 0 ) == DIOPinVal.PinSet :
    print("CSMIO-MPG digital output 0 = 1")
```

CSMIO-IO - Módulo adicional de E/S - número 0

```
mod_IO_0 = d.getModule( ModuleType.IO, 0 )

# Lectura del estado de la entrada digital núm. 7
if mod_IO _0.getDigitalIO( IOPortDir.InputPort, 7 ) == DIOPinVal.PinReset :
    print("CSMIO-IO 0, digital input 7 = 0")
if mod_IO_0.getDigitalIO( IOPortDir.InputPort, 7 ) == DIOPinVal.PinSet :
    print("CSMIO-IO 0, digital input 7 = 1")

# Lectura del estado de la salida digital núm. 2
if mod_IO_0.getDigitalIO( IOPortDir.OutputPort, 2 ) == DIOPinVal.PinReset :
    print("CSMIO-IO 0, digital output 2 = 0")
if mod_IO_0.getDigitalIO( IOPortDir.OutputPort, 2 ) == DIOPinVal.PinSet :
    print("CSMIO-IO 0, digital output 2 = 1")
```

CSMIO-IO - Módulo adicional de E/S - número 15

```
mod_IO_15 = d.getModule( ModuleType.IO, 15 )

# Lectura del estado de la entrada digital núm. 11
if mod_IO_15.getDigitalIO( IOPortDir.InputPort, 11 ) == DIOPinVal.PinReset :
    print("CSMIO-IO 15, digital input 11 = 0")
if mod_IO_15.getDigitalIO( IOPortDir.InputPort, 11 ) == DIOPinVal.PinSet :
    print("CSMIO-IO 15, digital input 11 = 1")
```

```
# Lectura del estado de la salida digital núm. 5
if mod_IO_15.getDigitalIO( IOPortDir.OutputPort, 5) == DIOPinVal.PinReset :
    print("CSMIO-IO 15, digital output 5 = 0")
if mod_IO_15.getDigitalIO( IOPortDir.OutputPort, 5) == DIOPinVal.PinSet :
    print("CSMIO-IO 15, digital output 5 = 1")
```

b) Guardar en puerto digital

void setDigitalIO(int digitalPin, DIOPinVal value) – Establece el estado del pin, puerto digital (establece el estado de la salida digital).

ARGUMENTOS DE FUNCIÓN:

int digitalPin – Número de pin (número de salida digital)
DIOPinVal value – Estado del pin

DISTINGUIMOS:

DIOPinVal.PinReset – Estado del pin bajo
DIOPinVal.PinSet – Estado del pin alto

EJEMPLO:

Controlador CSMIO/IP-S, CSMIO/IP-A i CSMIO/IP-M

```
import time
mod_IP = d.getModule( ModuleType.IP, 0 )

# Configuración de estado alta en la salida digital núm. 1
mod_IP.setDigitalIO( 1, DIOPinVal.PinSet )
time.sleep(1)

# Configuración de estado baja en la salida digital núm. 1
mod_IP.setDigitalIO( 1, DIOPinVal.PinReset )
```

CSMIO-MPG - Módulo de manipulación de eje manual

```
import time
mod_MPG = d.getModule( ModuleType.MPG, 0 )

# Configuración de estado alta en la salida digital núm. 0
mod_MPG.setDigitalIO( 0, DIOPinVal.PinSet )
time.sleep(1)

# Configuración de estado baja en la salida digital núm. 0
mod_MPG.setDigitalIO( 0, DIOPinVal.PinReset )
```

CSMIO-IO - Módulo adicional de E/S - número 0

```
import time
mod_IO_0 = d.getModule( ModuleType.IO, 0 )

# Configuración de estado alta en la salida digital núm. 5
mod_IO_0.setDigitalIO( 5, DIOPinVal.PinSet )
time.sleep(1)

# Configuración de estado baja en la salida digital núm. 5
mod_IO_0.setDigitalIO( 5, DIOPinVal.PinReset )
```

CSMIO-IO - Módulo adicional de E/S - número 15

```
import time
mod_IO_15 = d.getModule( ModuleType.IO, 15 )

# Configuración de estado alta en la salida digital núm. 3
mod_IO_15.setDigitalIO( 3, DIOPinVal.PinSet )
time.sleep(1)

# Configuración de estado baja en la salida digital núm. 3
mod_IO_15.setDigitalIO( 3, DIOPinVal.PinReset )
```

3) Puertos analógicos

a) Lectura desde puertos analógicos

float getAnalogIO(IOPortDir direction, int analogPin) – Devuelve el valor del voltaje del pin, el puerto de entrada o salida analógicos (devuelve el voltaje de la entrada o salida analógicos).

ARGUMENTOS DE FUNCIÓN:

IOPortDir direction – Especifica la dirección del puerto

DISTINGUIMOS:

IOPortDir.InputPort - Puerto de entrada
IOPortDir.OutputPort - Puerto de salida

int analogPin – Número de pin (número de entrada o salida analógica)

RESULTADO DE LA FUNCIÓN:

float – Valor de voltaje del pin

EJEMPLOS:

Controlador CSMIO/IP-S, CSMIO/IP-A i CSMIO/IP-M

```
mod_IP = d.getModule( ModuleType.IP, 0 )

# Lectura del valor del voltaje de la entrada analógica núm. 0
val = mod_IP.getAnalogIO( IOPortDir.InputPort, 0 )
print("CSMIO/IP analog input 0 = " + str(val) + "V")

# Lectura del valor del voltaje de la entrada analógica núm. 1
val = mod_IP.getAnalogIO( IOPortDir.InputPort, 1 )
print("CSMIO/IP analog input 1 = " + str(val) + "V")

# Lectura del valor del voltaje de la salida analógica núm. 0
val = mod_IP.getAnalogIO( IOPortDir.OutputPort, 0 )
print("CSMIO/IP analog output 0 = " + str(val) + "V")

# Lectura del valor del voltaje de la salida analógica núm. 1
val = mod_IP.getAnalogIO( IOPortDir.OutputPort, 1 )
print("CSMIO/IP analog output 1 = " + str(val) + "V")
```

CSMIO-MPG - Módulo de manipulación de eje manual

```
mod_MPG = d.getModule( ModuleType.MPG, 0 )

# Lectura del valor del voltaje de la entrada analógica núm. 0
val = mod_MPG.getAnalogIO( IOPortDir.InputPort, 0 )
print("CSMIO-MPG analog input 0 = " + str(val) + "V")

# Lectura del valor del voltaje de la entrada analógica núm. 1
val = mod_MPG.getAnalogIO( IOPortDir.InputPort, 1 )
print("CSMIO-MPG analog input 1 = " + str(val) + "V")
```

b) Guardar en puertos analógicos

void setAnalogIO(int analogPin, float value) – Establece el voltaje en el pin del puerto analógico.

ARGUMENTOS DE FUNCIÓN:

int analogPin – Número de pin (número de salida analógica)
float value – Valor de voltaje del pin

EJEMPLOS:

Controlador CSMIO/IP-S, CSMIO/IP-A i CSMIO/IP-M

```
mod_IP = d.getModule( ModuleType.IP, 0 )

# Ajuste del voltaje de 5V en la salida analógica número 0
val = 5
mod_IP.setAnalogIO( 0, val )

# Ajuste del voltaje de 2V en la salida analógica número 1
val = 2
mod_IP.setAnalogIO( 1, val )
```

¡ADVERTENCIA!

Controladores CSMIO/IP-S, CSMIO/IP-A, CSMIO/IP-M

El voltaje en los pines, los puertos de entrada y salida analógica puede variar de 0 V a 10 V con una resolución de 12 bits.

Módulo de manipulación de eje manual CSMIO-MPG

El voltaje en los pines, el puerto de entrada analógica, el módulo de manipulación de eje manual CSMIO-MPG, puede variar de 0V a 5V con una resolución de 12 bits.

4) Puertos del codificador - CSMIO-ENC

a) Lectura del ángulo del codificador

float getEncoderIOAngle(int channel = 0) - Devuelve el ángulo del codificador calculado a partir de la señal de índice.

ARGUMENTOS DE FUNCIÓN:

int channel – Canal del codificador

RESULTADO DE LA FUNCIÓN:

float – Ángulo del codificador

EJEMPLO:

```
mod_ENC = d.getModule( ModuleType.ENC, 0)

# Lectura del ángulo del codificador número 0
val = mod_ENC.getEncoderIOAngle( 0 )
print("CSMIO-ENC encoder chanel 0 angle = " + str(val))
```

b) Lectura de la posición del codificador

int getEncoderIOPostion(int channel = 0) - Devuelve la posición del codificador calculada a partir de la primera señal de índice.

ARGUMENTOS DE FUNCIÓN:

int channel – Canal del codificador

RESULTADO DE LA FUNCIÓN:

int – Posición del codificador

EJEMPLO:

```
mod_ENC = d.getModule( ModuleType.ENC, 0 )

# Lectura de la posición del codificador número 2
val = mod_ENC.getEncoderIOPosition( 2 )
print("CSMIO-ENC encoder chanel 0 postion = " + str(val))
```

c) Lectura de la velocidad del codificador

float getEncoderIORPM(int channel = 0) – Devuelve la velocidad de rotación del codificador (RPM).

ARGUMENTOS DE FUNCIÓN:

int channel – Canal del codificador

RESULTADO DE LA FUNCIÓN:

float – Velocidad de rotación del codificador

EJEMPLO:

```
mod_ENC = d.getModule( ModuleType.ENC, 0)

# Lectura de la velocidad de rotación del codificador número 4
val = mod_ENC.getEncoderIORPM( 4 )
print("CSMIO-ENC encoder chanel 0 rpm = " + str(val))
```

¡ADVERTENCIA!

Los canales 0, 2 y 4 del módulo de subprocesos son canales físicos, mientras que los canales 1, 3 y 5 son copias virtuales de canales físicos que se utilizarán en el futuro.

Hasta nuevo aviso, los canales físicos mencionados 0, 2 y 4 corresponden a los números de canal 0, 1 y 2 en la configuración simCNC.

VI. Leer y guardar coordenadas de eje

1) Lectura de coordenadas

List<float>(6) getPosition(CoordMode coordMode) - Devuelve el valor actual de la máquina o las coordenadas del software de todos los ejes.

ARGUMENTOS DE FUNCIÓN:

CoordMode coordMode - Especifica el tipo de coordenadas

DISTINGUIMOS:

CoordMode.Machine – Coordenadas de la máquina
CoordMode.Program – Coordenadas del programa

RESULTADO DE LA FUNCIÓN:

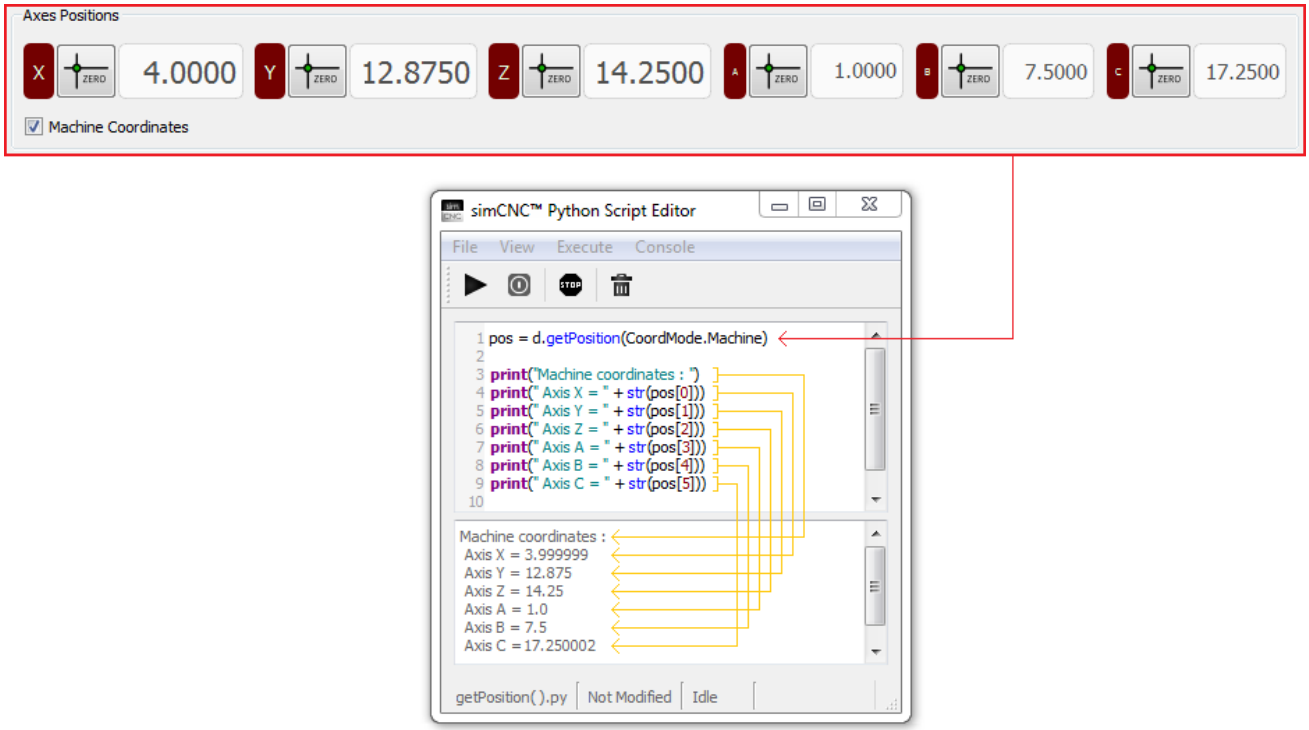
List<float>(6) – lista de 6 coordenadas (X, Y, Z, A, B, C) de la posición actual

EJEMPLOS:

Lectura de las coordenadas de la máquina de los ejes X, Y, Z, A, B y C

```
pos = d.getPosition( CoordMode.Machine )

print("Machine coordinates : ")
print(" Axis X = " + str(pos[0]))
print(" Axis Y = " + str(pos[1]))
print(" Axis Z = " + str(pos[2]))
print(" Axis A = " + str(pos[3]))
print(" Axis B = " + str(pos[4]))
print(" Axis C = " + str(pos[5]))
```

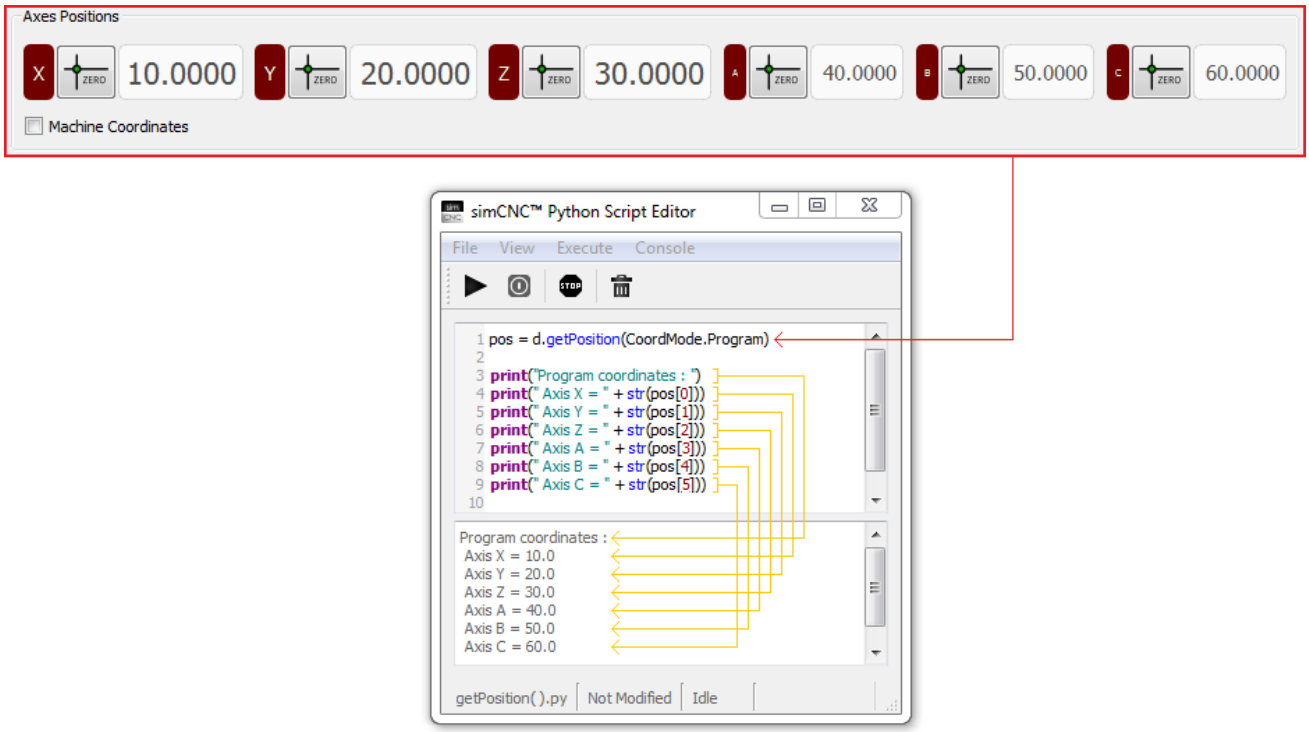
Al ejecutar el ejemplo anterior, la función „d.getPosition(CoordMode.Machine)" lee las coordenadas de la máquina de los 6 ejes y los coloca en una lista indexada de 6 elementos del 0 al 5. A continuación, utiliza la función „print( )" y los valores de coordenadas de máquina guardados en la lista se mostrarán en la consola de Python.



Lectura de coordenadas del programa en los ejes X, Y, Z, A, B y C

```
pos = d.getPosition( CoordMode.Program )

print("Program coordinates : ")
print(" Axis X = " + str(pos[0]))
print(" Axis Y = " + str(pos[1]))
print(" Axis Z = " + str(pos[2]))
print(" Axis A = " + str(pos[3]))
print(" Axis B = " + str(pos[4]))
print(" Axis C = " + str(pos[5]))
```

Al ejecutar el ejemplo anterior, la función „d.getPosition(CoordMode.Machine)" lee las coordenadas de la máquina de los 6 ejes y los coloca en una lista indexada de 6 elementos del 0 al 5. A continuación, utiliza la función „print( )" y los valores de coordenadas de máquina guardados en la lista se mostrarán en la consola de Python.



2) Registro de coordenadas

void setAxisProgPosition(Axis axis, float value) – Establece el valor de las coordenadas del programa del eje seleccionado.

ARGUMENTOS DE FUNCIÓN:

Axis axis – Eje en el que se guardará el nuevo valor de coordenada del programa

DISTINGUIMOS:

- Axis.X – Eje X
- Axis.Y – Eje Y
- Axis.Z – Eje Z
- Axis.A – Eje A
- Axis.B – Eje B
- Axis.C – Eje C

float value – Nuevo valor de las coordenadas del programa

EJEMPLOS:

Guardar las coordenadas del programa para los ejes X, Y, Z, A, B y C.

```
x = 10
y = 20
z = 30
a = 40
b = 50
c = 60

d.setAxisProgPosition( Axis.X, x )
d.setAxisProgPosition( Axis.Y, y )
d.setAxisProgPosition( Axis.Z, z )
d.setAxisProgPosition( Axis.A, a )
d.setAxisProgPosition( Axis.B, b )
d.setAxisProgPosition( Axis.C, c )
```

Lo mismo, solo que con el uso de una lista.

```
pos = (10, 20, 30, 40, 50, 60)

d.setAxisProgPosition(Axis.X, pos[0])
d.setAxisProgPosition(Axis.Y, pos[1])
d.setAxisProgPosition(Axis.Z, pos[2])
d.setAxisProgPosition(Axis.A, pos[3])
d.setAxisProgPosition(Axis.B, pos[4])
d.setAxisProgPosition(Axis.C, pos[5])
```

VII. Desplazamiento de los ejes de la máquina

void moveAxisIncremental(Axis axis, float distance, float velocity) - Mueve el eje seleccionado en una distancia establecida (movimiento incremental) a una velocidad determinada.

ARGUMENTOS:

Axis axis - Eje que efectuará el movimiento

DISTINGUIMOS:

Axis.X – Eje X
Axis.Y – Eje Y
Axis.Z – Eje Z
Axis.A – Eje A
Axis.B – Eje B
Axis.C – Eje C

float distance – Distancia del movimiento
float velocity – Velocidad del movimiento

EJEMPLOS:

```
d.moveAxisIncremental( Axis.X, 20, 1000 )
d.moveAxisIncremental( Axis.Y, 20, 1000 )
d.moveAxisIncremental( Axis.X, -20, 1000 )
d.moveAxisIncremental( Axis.Y, -20, 1000 )
```

Al ejecutar el ejemplo anterior, los ejes X e Y se moverán desde su posición actual, alrededor del perímetro del cuadrado, con un lado de 20 mm o pulgada a 1000 mm/min o pulgada/min.

void moveToPosition(CoordMode coordMode, List<float>[6] position, float velocity) - Se desplaza a la posición establecida, expresada en coordenadas de máquina o programa, con la velocidad resultante establecida.

ARGUMENTOS:

CoordMode coordMode - Especifica el tipo de coordenadas

DISTINGUIMOS:

CoordMode.Machine – Coordenadas de la máquina
CoordMode.Program – Coordenadas del programa

List<float>[6] position - Lista de 6 coordenadas (X, Y, Z, A, B, C) de la posición objetivo
float velocity - Establecer la velocidad resultante

EJEMPLOS:

```
X = 0
Y = 1
pos = d.getPosition(CoordMode.Program)

pos[X] = pos[X] + 20
d.moveToPosition( CoordMode.Program, pos, 1000 )
pos[Y] = pos[Y] + 20
d.moveToPosition( CoordMode.Program, pos, 1000 )
pos[X] = pos[X] - 20
d.moveToPosition( CoordMode.Program, pos, 1000 )
pos[Y] = pos[Y] - 20
d.moveToPosition( CoordMode.Program, pos, 1000 )
```

Al ejecutar el ejemplo anterior, los ejes X e Y se moverán utilizando las coordenadas del programa, moviéndose desde su posición actual, alrededor del perímetro de un cuadrado, con un lado de 20 mm o pulgadas, a una velocidad de 1000 mm/min o pulgadas/min.

```
X = 0
Y = 1
pos = d.getPosition(CoordMode.Machine)

pos[X] = pos[X] + 20
d.moveToPosition( CoordMode.Machine, pos, 1000 )
pos[Y] = pos[Y] + 20
d.moveToPosition( CoordMode.Machine, pos, 1000 )
pos[X] = pos[X] - 20
d.moveToPosition( CoordMode.Machine, pos, 1000 )
pos[Y] = pos[Y] - 20
d.moveToPosition( CoordMode.Machine, pos, 1000 )
```

Al ejecutar el ejemplo anterior, los ejes X e Y utilizarán las coordenadas de la máquina para moverse desde la posición actual, alrededor del perímetro de un cuadrado, con un lado de 20 mm o pulgadas, a una velocidad de 1000 mm/min o pulgadas/min.

Los dos ejemplos anteriores muestran cómo puede usar listas para modificar coordenadas.
Podrá ver este método de cálculo muy a menudo en los scripts que estarán disponibles en el sitio web de CS-LAB, por lo que merece la pena echarle un vistazo en este momento.

void parallelMoveToPosition(CoordMode coordMode, Axis axis, float position, float velocity) - Realiza el movimiento del eje independiente seleccionado a la posición establecida expresada coordenadas de máquina o software a la velocidad establecida. Esta función se ejecuta cuando la opción "Usar puntos externos" no está seleccionada.

ARGUMENTOS:

CoordMode coordMode - Especifica el tipo de coordenadas

DISTINGUIMOS:

CoordMode.Machine – Coordenadas de la máquina
CoordMode.Program – Coordenadas del programa

Axis axis – Un eje independiente que efectuará un movimiento

DISTINGUIMOS:

Axis.X – Eje X
Axis.Y – Eje Y
Axis.Z – Eje Z
Axis.A – Eje A
Axis.B – Eje B
Axis.C – Eje C

float position - Posición del eje objetivo
float velocity - Ajuste de velocidad

EJEMPLOS:

```
X = 0
pos = d.getPosition(CoordMode.Program)

d.parallelMoveToPosition( CoordMode.Program, Axis.X, 25, 100 )
d.parallelMoveToPosition( CoordMode.Program, Axis.X, pos[X], 20 )
```

Al ejecutar el ejemplo anterior, el eje X se moverá utilizando las coordenadas del programa a una posición de 25 mm o pulgadas, a 100 mm/min o pulgadas/min, y posteriormente regresará al punto de partida a 20 mm/min o pulgadas/min.

```
Z = 2
pos = d.getPosition(CoordMode.Program)

d.parallelMoveToPosition( CoordMode. Machine, Axis.Z, 78, 350 )
d.parallelMoveToPosition( CoordMode. Machine, Axis.Z, pos[Z], 500 )
```

Al ejecutar el ejemplo anterior, el eje Z se moverá utilizando las coordenadas de la máquina a 78 mm o pulgadas, a 350 mm/min o pulgadas/min, y posteriormente regresará al punto de partida a 500 mm/min o pulgadas/min.

float getCurrentXYZVelocity( ) - Devuelve la velocidad resultante actual de los ejes X, Y y Z.

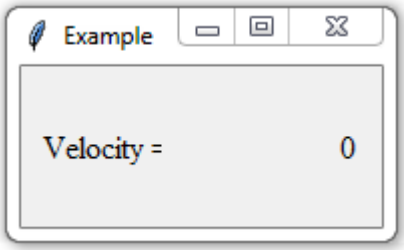
RESULTADO DE LA FUNCIÓN:

float - Velocidad resultante

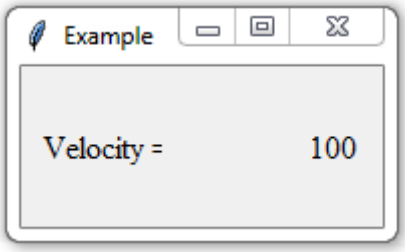
EJEMPLOS:

```
from tkinter import *

def show():
    velocity = d.getCurrentXYZVelocity( )
    L2["text"] = str(int(velocity))
    window.after(5, show)

window = Tk()
window.geometry("180x80")
window.title( "Example" )

L1 = Label(window, text="Velocity =", font= ("Times New Roman",12))
L1.place(x=10, y=30, height=20, width=65)

L2 = Label(window, font= ("Times New Roman",12), anchor = "e")
L2.place(x=70, y=30 ,height=20, width=100)

window.after(5, show)
mainloop()
```

Al ejecutar el ejemplo anterior, el script mostrará una ventana que muestra la velocidad resultante actual de los ejes X, Y y Z. Junto a una imagen que muestra la situación en la que ninguno de los ejes se está moviendo.



Para comprobar si el script funciona, seleccione la opción "Control desde el teclado" o utilice el atajo "Alt + j" y mueva el eje o los ejes con los cursores del teclado.



VIII. Sondeo

bool executeProbing(CoordMode coordMode, List<float>[6] position, int probeIndex, float velocity) - Realiza sondeos con la sonda seleccionada en una posición determinada, expresada en coordenadas de la máquina o programa, con una velocidad resultante determinada.

ARGUMENTOS:

CoordMode coordMode - Especifica el tipo de coordenadas

DISTINGUIMOS:

CoordMode.Machine - Coordenadas de la máquina
CoordMode.Program - Coordenadas del programa

List<float>[6] position - Lista de 6 coordenadas (X, Y, Z, A, B, C) de la posición objetivo
int probeIndex - Número de sonda (las configuraciones de la sonda se pueden encontrar en Settings/Modules/Special IO)
float velocity - Velocidad resultante

RESULTADO DE LA FUNCIÓN:

bool – Resultado de sondeo

DISTINGUIMOS:

True - En caso de que la sonda funcione
False - En caso de que el eje o los ejes hayan alcanzado la posición objetivo y la sonda no funcione

EJEMPLOS lo puede encontrar en la descripción de la próxima función

List<float>[6] getProbingPosition(CoordMode coordMode) - Devuelve la posición de respuesta de la sonda o la posición objetivo expresada en coordenadas de máquina o programa. La función devuelve la posición de destino sólo si la sonda no hubiera funcionado durante el proceso de sondeo.

ARGUMENTOS:

CoordMode coordMode - Especifica el tipo de coordenadas

DISTINGUIMOS:

CoordMode.Machine - Coordenadas de la máquina
CoordMode.Program - Coordenadas del programa

RESULTADO DE LA FUNCIÓN:

List<float>[6] - lista de 6 coordenadas (X, Y, Z, A, B, C) de la posición de acción de la sonda o la posición de destino de la misma.

EJEMPLOS para los anteriores:

```
Probing_vel = 10
Return_vel = 100
Distance = 20

Starting_position = d.getPosition( CoordMode.Program )
Maximum_position = Starting_position.copy()
Maximum_position[2] -= Distance

if ( d.executeProbing( CoordMode.Program, Maximum_position, 0, Probing_vel ) == False ):
    d.moveToPosition( CoordMode.Program, Starting_position, Return_vel )
    msg.err( "Probing Failed !!!" , "Inactive probe" )
else:
    d.moveToPosition( CoordMode.Program, Starting_position, Return_vel )
    Probe_position = d.getProbingPosition( CoordMode.Program )
    msg.info( "Position = " + str(Probe_position[2]),"Probing - Z axis" )
```

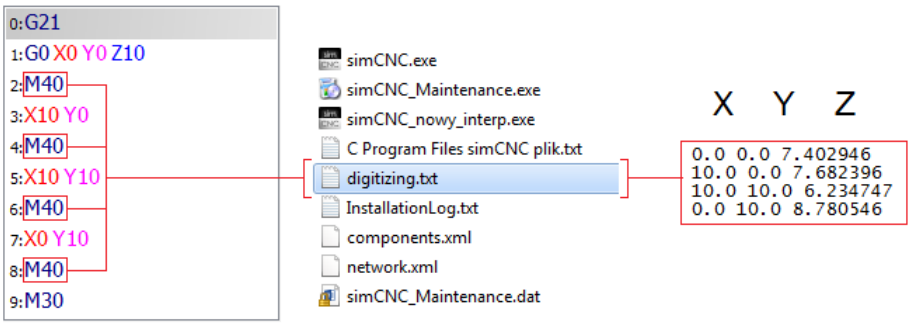
El script anterior muestra cómo realizar el sondeo utilizando el eje Z desde la posición actual a la distancia determinada por el parámetro „Distance”. Después de sondear, independientemente del resultado, el eje Z siempre vuelve a la posición inicial. El parámetro „Probing_vel” es responsable de la velocidad de sondeo, el parámetro „Return_vel” es responsable de la velocidad del retorno de la sonda a la posición inicial. El resultado del sondeo se presenta mediante una ventana emergente.

¡Una forma fácil de digitalizar!

El script presentado anteriormente, tras una pequeña modificación, puede actuar como una macro que guarda la cuadrícula de puntos de la superficie sondeada. El proceso de guardar la cuadrícula de puntos se usa con mayor frecuencia al copiar un bajorrelieve, o fresar o grabar en una superficie irregular, por ejemplo, una superficie de piedra.

```
import sys

File_path = "digitizing.txt"                    #C:\Program Files\simCNC\digitizing.txt

try:
    File = open(File_path, "a+")
except IOError:
    sys.exit("\n No access to file !!!")

Probing_vel = 100
Return_vel = 1000
Distance = 20
Starting_position = d.getPosition( CoordMode.Program )
Maximum_position = Starting_position.copy()
Maximum_position[2] -= Distance

if ( d.executeProbing( CoordMode.Program, Maximum_position, 0, Probing_vel ) == False ):
    d.moveToPosition( CoordMode.Program, Starting_position, Return_vel )
    Saved_text ="Probing Failed !!! \n"
else:
    d.moveToPosition( CoordMode.Program, Starting_position, Return_vel )
    Probe_position = d.getProbingPosition( CoordMode.Program )
    Saved_text = str(Probe_position[0]) + " " + str(Probe_position[1]) + " " + str(Probe_position[2]) + "\n"

try:
    File.write(Saved_text)
except IOError:
    File.close()
    sys.exit("\n File write error !!!")
```

Todo lo que tiene que hacer es guardar el script anterior, por ejemplo, con el nombre M40, y llamarlos desde un nivel gcod de vez en cuando. Cuando se llama al script, realizará un sondeo y posteriormente guardará las posiciones de actuación de la sonda junto con la posición de los ejes X e Y en el archivo C:\Program Files\simCNC\digitizing.txt. A continuación puede ver el efecto de la macro en el ejemplo de un simple gcode que forma un movimiento cuadrado de 10x10 alrededor del perímetro.



En caso de que la sonda no funcione en la distancia especificada por el parámetro „Distance" en el archivo „digitizing.txt" aparecerá la entrada „Probing Failed !!!".

```
0.0 0.0 6.716997
10.0 0.0 7.318946
Probing Failed !!!
0.0 10.0 8.143046
```

VIII. Revista de herramientas

1) Número de herramienta.

int getSelectedToolNumber( ) - Devuelve el número de la herramienta seleccionada desde el nivel de Gcode, línea MDI o script Python.

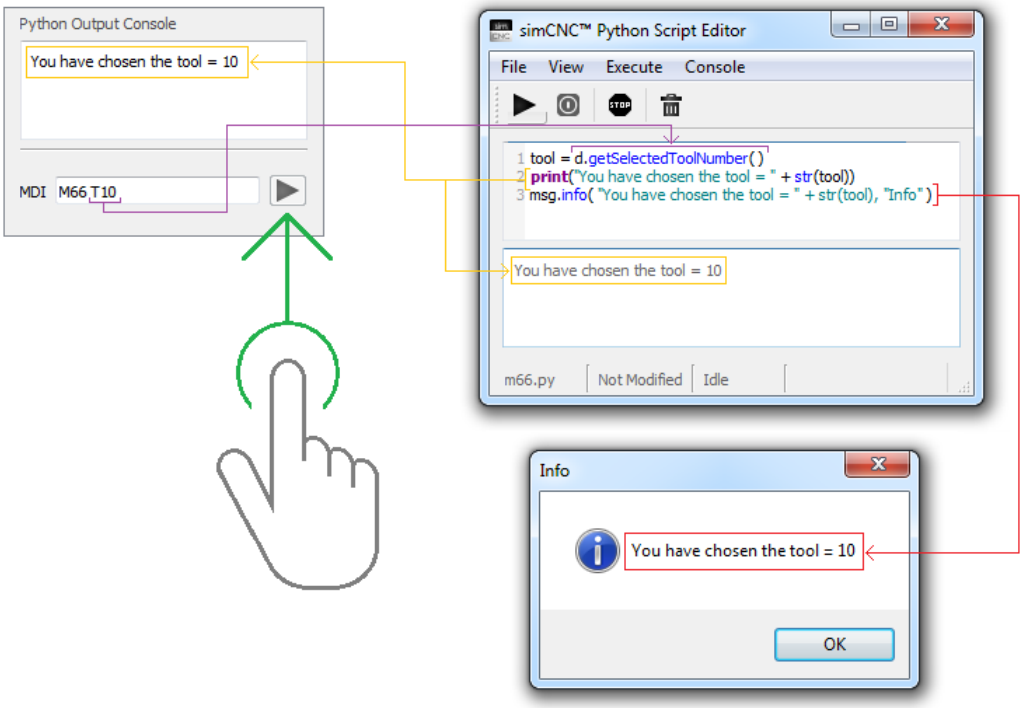
RESULTADO DE LA FUNCIÓN:

int - Número de herramienta

EJEMPLO:

```
tool = d.getSelectedToolNumber( )
print("You have chosen the tool = " + str(tool))
msg.info( "You have chosen the tool = " + str(tool), "Info" )
```

Guarde el script anterior como, por ejemplo, M66, y a continuación llámelo desde la línea Gcode o MDI agregando el comando "T" + cualquier número de herramienta (por ejemplo, M66 T10). Después de esta acción, debería aparecer una ventana que nos informa qué número de herramienta se ha seleccionado. Por ejemplo, usamos la línea MDI y la herramienta número 10. Recomendamos usar la macro M66 a propósito para mostrar que la función d.getSelectedToolNumber( ) funciona con cualquier macro, no solo M6.



void setSelectedToolNumber(int toolNumber) - Establece el número de la herramienta seleccionada, equivalente al comando "Txx", al que se llama desde la línea Gcode o MDI.

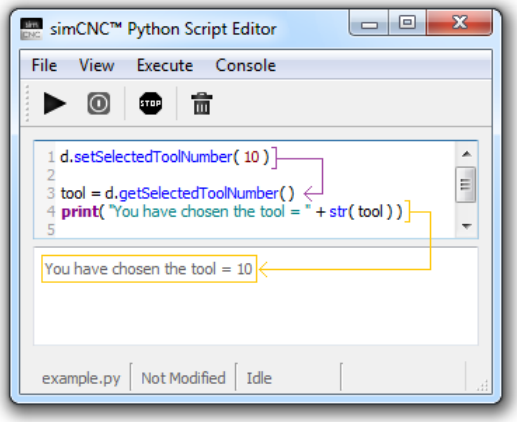
ARGUMENTOS:

int toolNumber - Número de herramienta

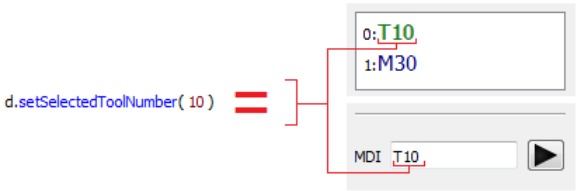
EJEMPLO:

```
d.setSelectedToolNumber( 10 )                    # Establece el número de la herramienta seleccionada

tool = d.getSelectedToolNumber( )                # Devuelve el número de la herramienta seleccionada.
print( "You have chosen the tool = " + str( tool ) )
```

Al ejecutar el ejemplo anterior, la función " d.setSelectedToolNumber( )" establece el número de la herramienta seleccionada en 10, y la función ya conocida " d.getSelectedToolNumber( )", devuelve el mismo número de herramienta.



Como puede ver fácilmente ahora, la función d.setSelectedToolNumber(10) realiza exactamente la misma tarea que el comando T10, que es llamado desde la línea Gcode o MDI.



void setSpindleToolNumber(int toolNumber) - Establece el número de herramienta en el eje.

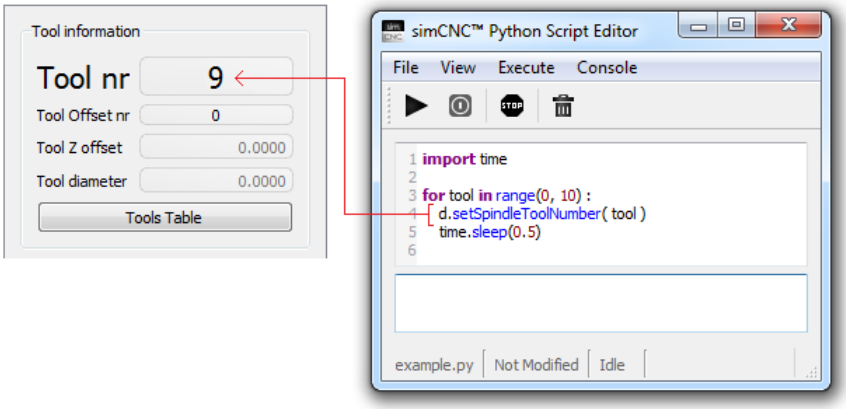
ARGUMENTOS:

int toolNumber - Número de herramienta

EJEMPLO:

```
import time

for tool in range(0, 10) :
  d.setSpindleToolNumber( tool )
  time.sleep(0.5)
```

Al ejecutar el ejemplo anterior, la función " d.setSpindleToolNumber(tool)" situada en el bucle "for" irá incrementando el número de herramienta en la pantalla simCNC cada 0.5 segundos hasta llegar a 9.



int getSpindleToolNumber( ) - Devuelve el número de la herramienta situada en el eje.

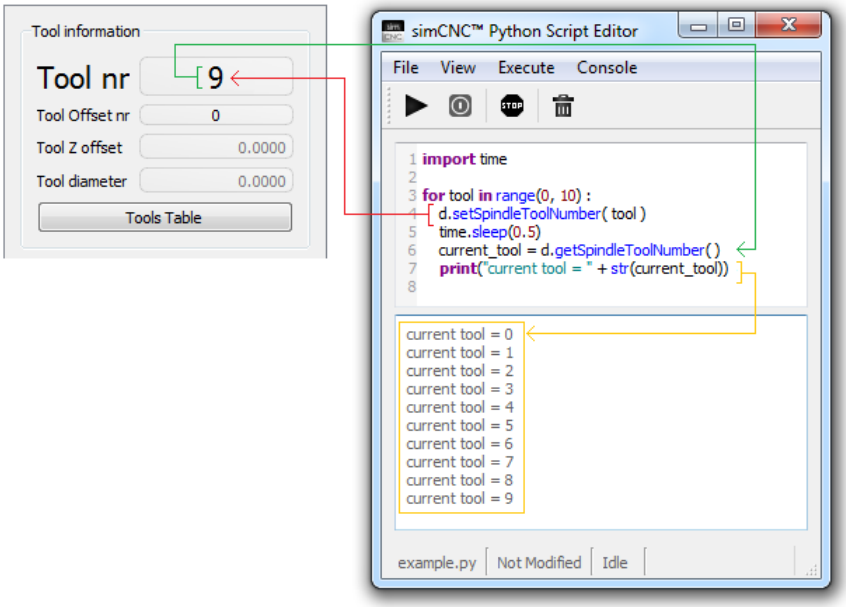
RESULTADO DE LA FUNCIÓN:

int - Número de herramienta

EJEMPLO:

```
import time

for tool in range(0, 10) :
  d.setSpindleToolNumber( tool )
  time.sleep(0.5)
  current_tool = d.getSpindleToolNumber( )
  print("current tool = " + str(current_tool))
```

El script anterior es una continuación del ejemplo anterior. Esta vez, se ha agregado una función al bucle "for" "current_tool = d.getSpindleToolNumber( )", que devuelve el número actual de la herramienta situada en el eje después de cada bucle de "for"; a continuación, el número de herramienta utilizado se presenta en la consola de Python con la función "imprimir".



2) Longitud de la herramienta

void setToolLength(int toolNumber, float toolLength) - Establece el valor de desplazamiento de la longitud de la herramienta.

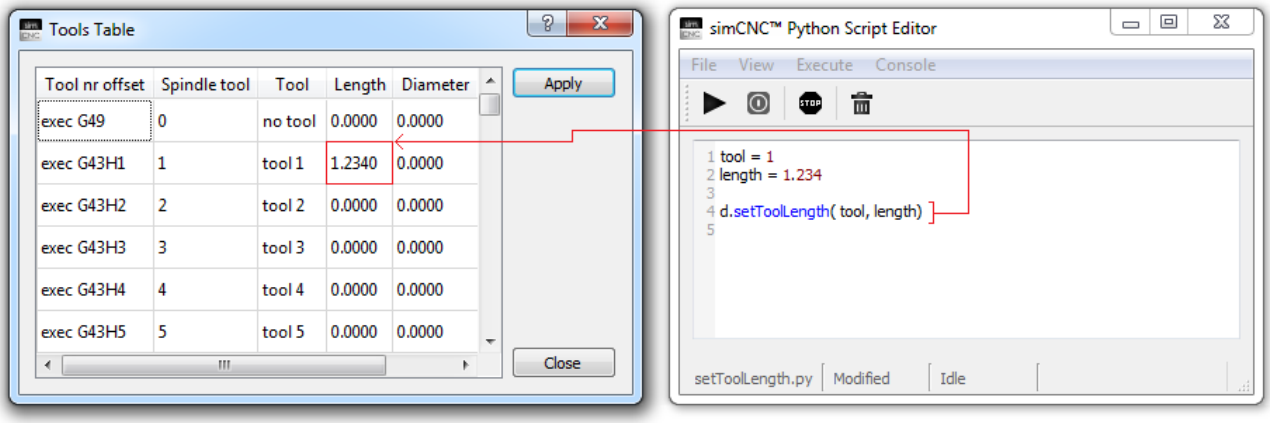
ARGUMENTOS:

int toolNumber - Número de herramienta
float toolLength - Valor de desplazamiento de longitud de herramienta

EJEMPLO:

```
tool = 1
length = 1.234

d.setToolLength( tool, length )
```

Al ejecutar el ejemplo anterior, el valor de longitud de la herramienta número 1 se establecerá en 1.234.



float getToolLength(int toolNumber) - Devuelve el valor de desplazamiento de la longitud de la herramienta.

ARGUMENTOS:

int toolNumber - Número de herramienta

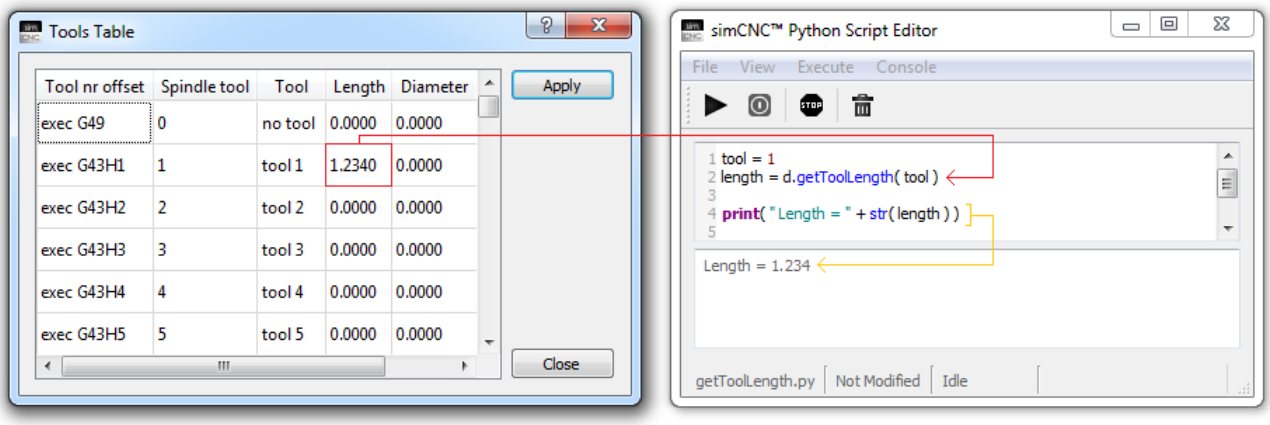
RESULTADO DE LA FUNCIÓN:

float - Valor de desplazamiento de longitud de la herramienta

EJEMPLO:

```
tool = 1
length = d.getToolLength( tool )

print( " Length = " + str( length ) )
```

Al ejecutar el ejemplo anterior en la consola de Python, se mostrará el valor de desplazamiento de longitud de la herramienta número 1.



void setToolOffsetNumber(int toolNumber) - Establece el número de desplazamiento de la longitud de la herramienta.

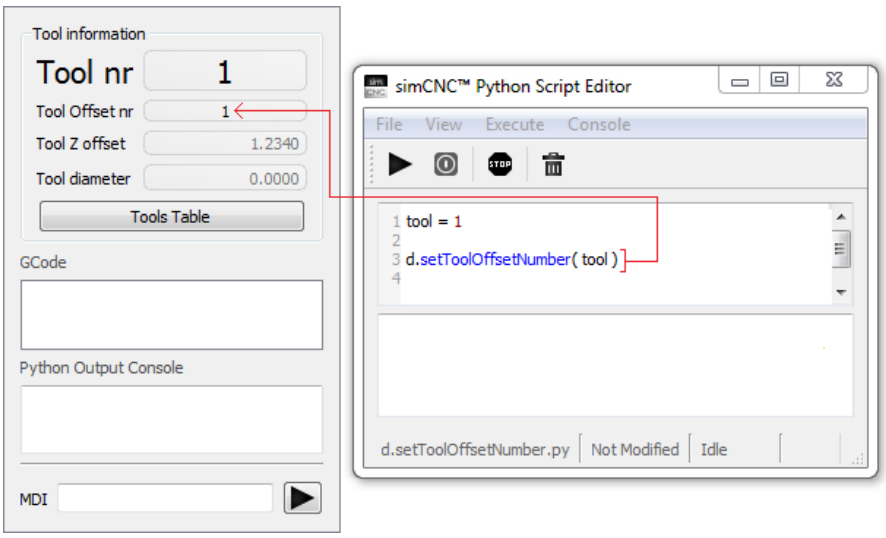
ARGUMENTOS:

int toolNumber - Número de herramienta

EJEMPLO:

```
tool = 1

d.setToolOffsetNumber( tool )
```

Al ejecutar el ejemplo anterior, el número de desplazamiento de la longitud de la herramienta se establecerá en 1.



int getToolOffsetNumber( ) - Devuelve el número de desplazamiento de longitud de la herramienta actualmente en uso.

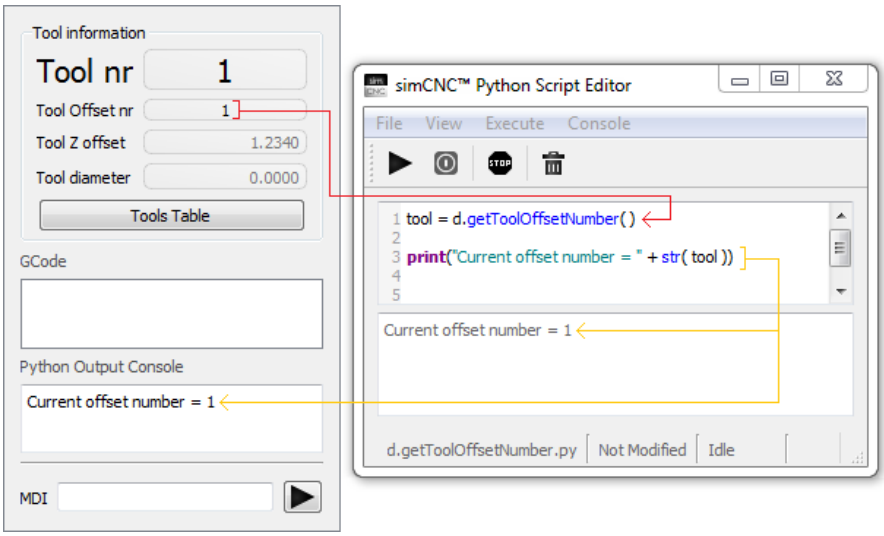
ARGUMENTOS:

int - Número de herramienta

EJEMPLO:

```
tool = d.getToolOffsetNumber( )

print("Current offset number = " + str( tool ))
```

Al ejecutar el ejemplo anterior, la consola de Python mostrará el número de desplazamiento de la longitud de la herramienta actualmente en uso.



3) Diámetro de la herramienta

¡ADVERTENCIA!

SimCNC no utiliza actualmente el valor del diámetro de la herramienta, lo que significa que simCNC no admite actualmente la función de compensación del diámetro de la herramienta.

void setToolDiameter(int toolNumber, float toolDiameter) - Establece el valor del diámetro de la herramienta.

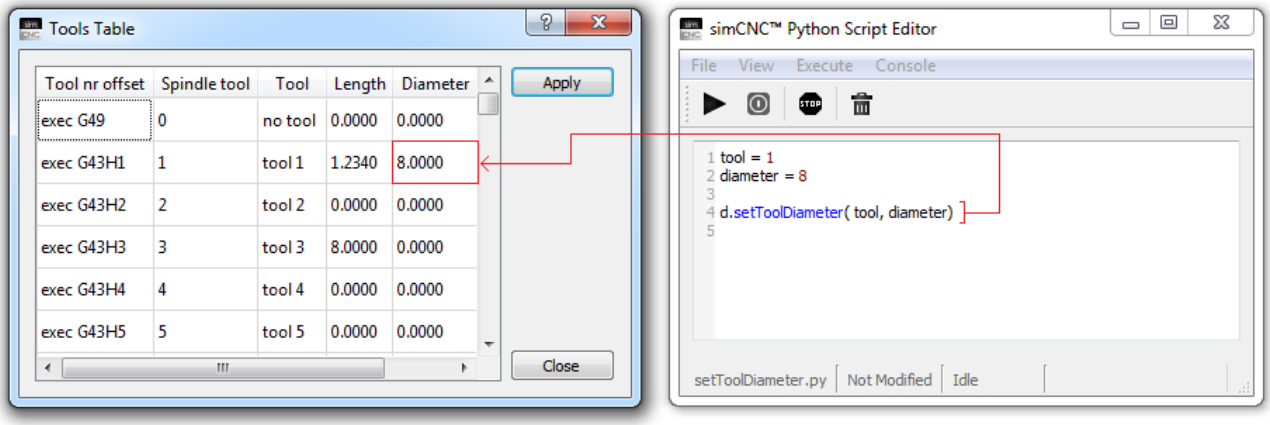
ARGUMENTOS:

int toolNumber - Número de herramienta
float toolDiameter - Diámetro de la herramienta

EJEMPLO:

```
tool = 1
diameter = 8

d.setToolDiameter( tool, diameter )
```

Al ejecutar el ejemplo anterior, el diámetro de herramienta número 1 se establecerá en 8.



float getToolDiameter(int toolNumber) - Devuelve el valor del diámetro de la herramienta de las herramientas.

ARGUMENTOS:

int toolNumber - Número de herramienta

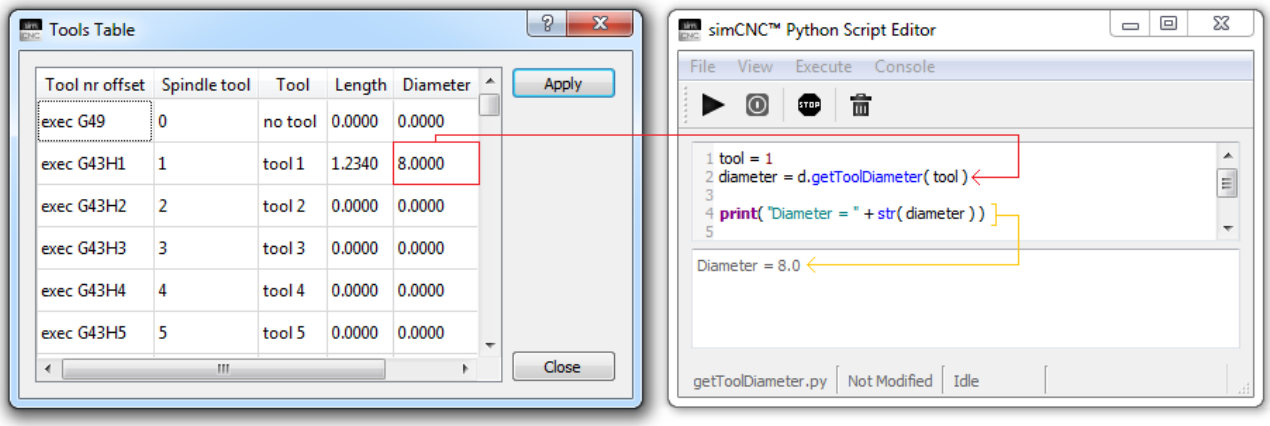
RESULTADO DE LA FUNCIÓN:

float - Diámetro de la herramienta

EJEMPLO:

```
tool = 1
diameter = d.getToolDiameter( tool )

print( "Diameter = " + str( diameter ) )
```

Al ejecutar el ejemplo anterior, la consola de Python mostrará el valor del diámetro de la herramienta número 1.



IX. Husillo, refrigerante y niebla.

1) Husillo

void setSpindleSpeed(float spindleSpeed) - Establece la velocidad objetivo del husillo (RPM).

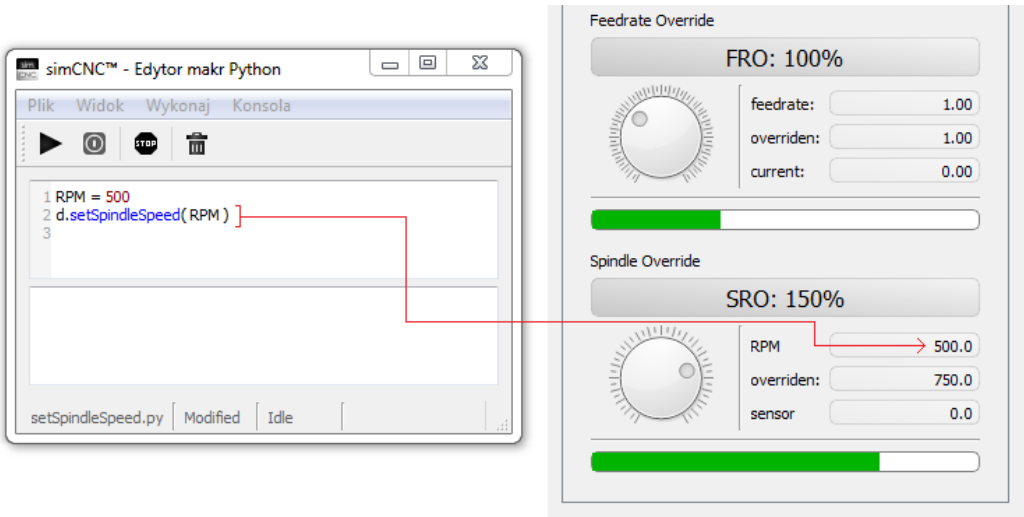
ARGUMENTY:

float spindleSpeed – ajuste de la velocidad del husillo (RPM)

EJEMPLO:

```
RPM = 500
d.setSpindleSpeed( RPM )
```

Al ejecutar el ejemplo anterior, la velocidad determinada del husillo se establecerá en 500 RPM.



float getSpindleSpeed( ) - Devuelve el punto de ajuste de velocidad del husillo (RPM).

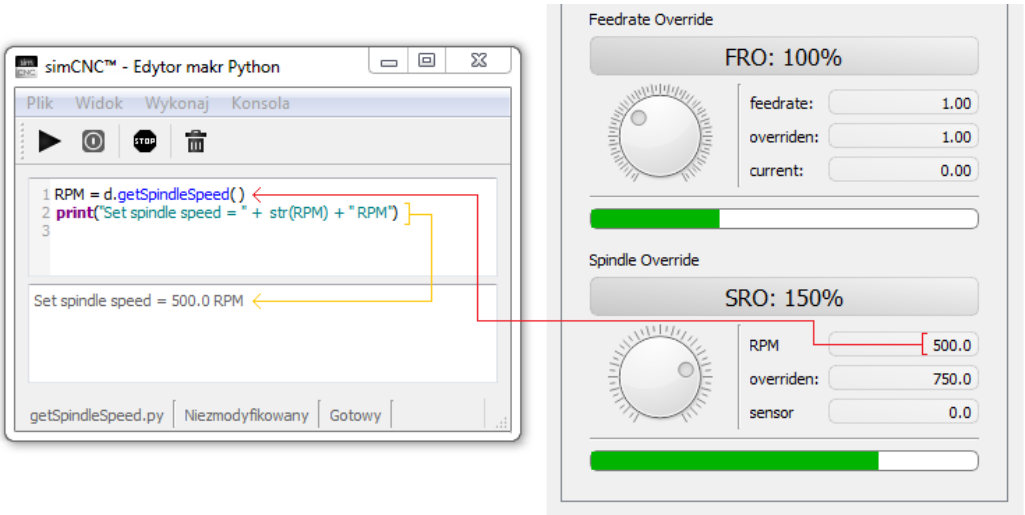
RESULTADO DE LA FUNCIÓN:

float – ajuste de la velocidad del husillo (RPM)

EJEMPLO:

```
RPM = d.getSpindleSpeed( )
print("Set spindle speed = " + str(RPM))
```

Al ejecutar el ejemplo anterior, la consola de Python mostrará la velocidad establecida del husillo.



void setSpindleState(SpindleState spindleState) - Establece el estado del husillo

ARGUMENTOS:

SpindleState spindleState - estado del husillo

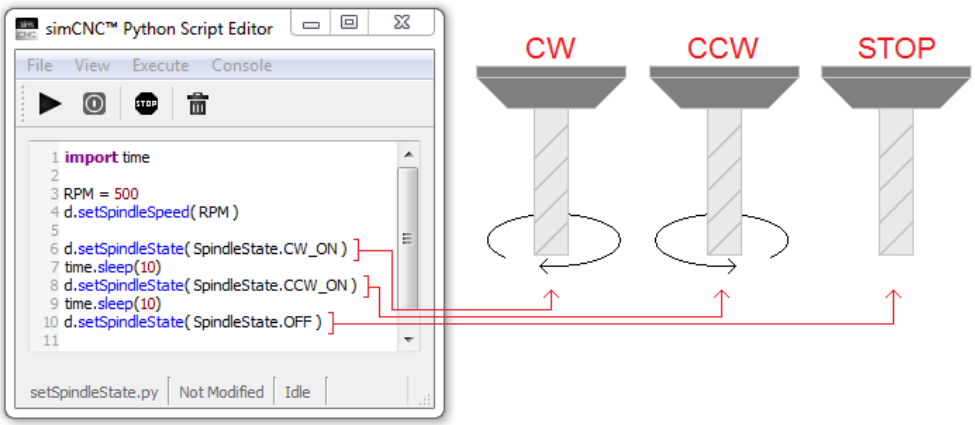
DISTINGUIMOS:

SpindleState.CW_ON - Rotación del husillo en sentido de las agujas del reloj
SpindleState.CCW_ON - Rotación del husillo en sentido contrario a las agujas del reloj
SpindleState.OFF - Velocidad del husillo desactivada

EJEMPLO:

```
import time

RPM = 500
d.setSpindleSpeed( RPM )

d.setSpindleState( SpindleState.CW_ON )
time.sleep(10)
d.setSpindleState( SpindleState.CCW_ON )
time.sleep(10)
d.setSpindleState( SpindleState.OFF )
```

Al ejecutar el ejemplo anterior, el husillo girará en el sentido de las agujas del reloj durante 10 segundos, luego lo hará en sentido contrario y se apagará después de 10 segundos.
Tenga en cuenta que la función setSpindleState suspende la ejecución del script durante la rampa de aceleración y desaceleración del husillo.



SpindleState getSpindleState( ) - **Devuelve el estado actual del husillo.**

RESULTADO DE LA FUNCIÓN:

SpindleState - estado del husillo

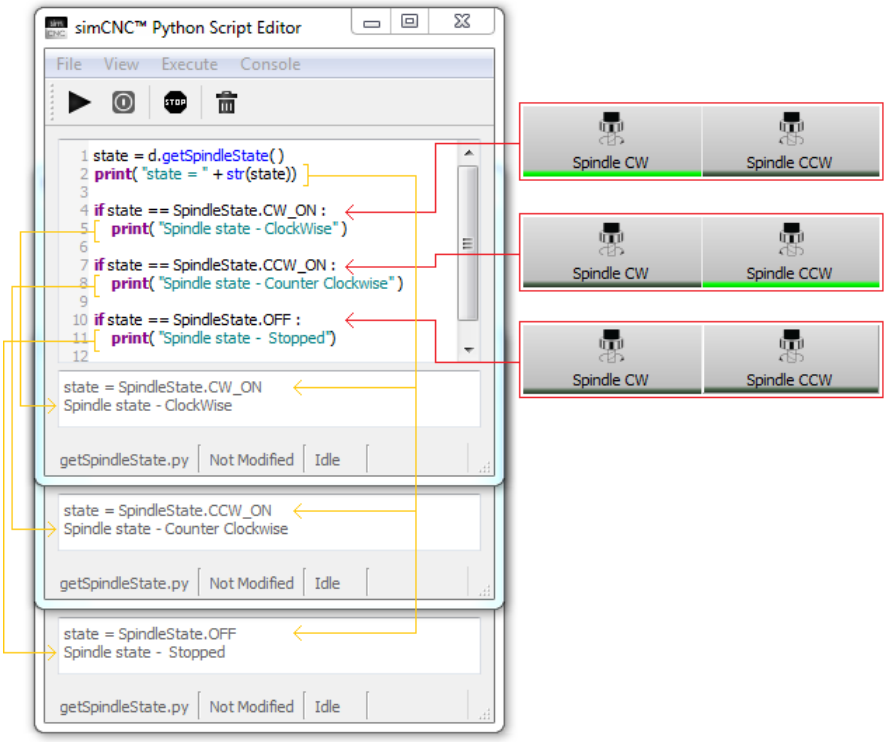
DISTINGUIMOS:

SpindleState.CW_ON	- Rotación del husillo en sentido de las agujas del reloj
SpindleState.CCW_ON	- Rotación del husillo en sentido contrario a las agujas del reloj
SpindleState.OFF	- Velocidad del husillo desactivada

EJEMPLO:

```
state = d.getSpindleState( )
print( "state = " + str(state))

if state == SpindleState.CW_ON :
  print( "Spindle state - ClockWise" )

if state == SpindleState.CCW_ON :
  print( "Spindle state - Counter Clockwise" )

if state == SpindleState.OFF :
  print( "Spindle state - Stopped")
```

Al ejecutar el ejemplo anterior, el estado actual del eje se mostrará en la primera línea de la consola Python como resultado de la función d.getSpindleState (). Se mostrará un segundo mensaje en la segunda línea que describe el estado actual del husillo con mayor claridad.
El siguiente boceto muestra cómo funciona el script para todos los estados del husillo.



void waitForSpindleSetSpeed(int timeout_sec) - Espera a que el husillo supere la velocidad de rotación especificada durante un tiempo determinado. La velocidad esperada del husillo está determinada por la velocidad establecida y el parámetro „ Spindle Ready Level” („Umbral de espera”).

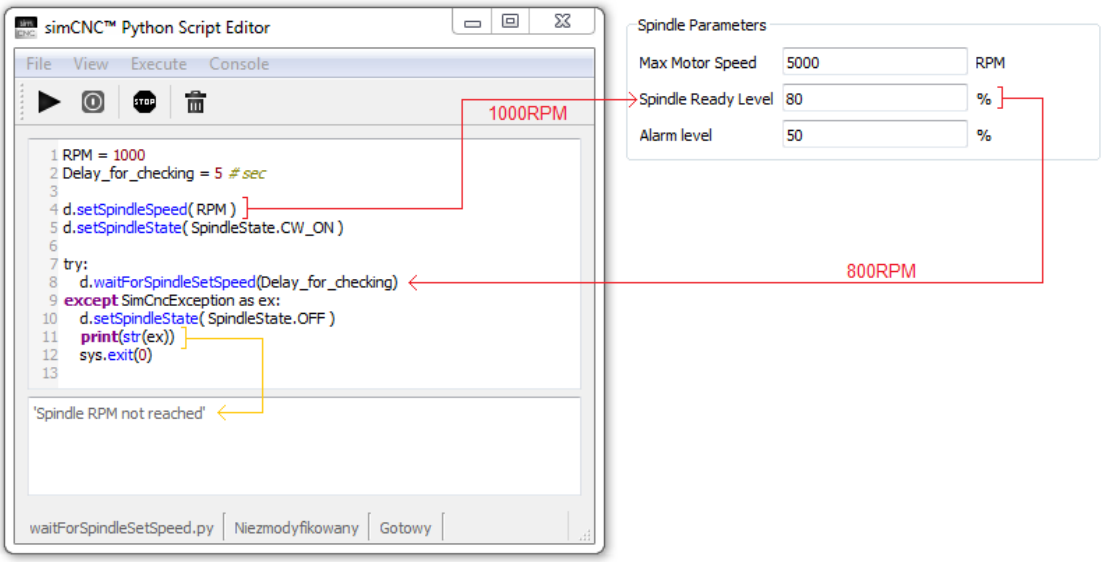
ARGUMENTOS:

int timeout_sec - tiempo de espera (en segundos)

EJEMPLO:

```
RPM = 500
Delay_for_checking = 5 # sec

d.setSpindleSpeed( RPM )
d.setSpindleState( SpindleState.CW_ON )

try:
  d.waitForSpindleSetSpeed(Delay_for_checking)
except SimCncException as ex:
  print(str(ex))
  sys.exit(0)
```

Al ejecutar el ejemplo anterior, el husillo girará en el sentido de las agujas del reloj (línea 5 - ver la foto a continuación) con una velocidad establecida de 1000 RPM (línea 4). A continuación, la función "waitForSpindleSetSpeed" suspenderá la ejecución del script hasta que el husillo alcance el 80% de la velocidad del husillo establecida o 800 RPM (80% de 1000 RPM = 800 RPM). Si el husillo no alcanza las 800 RPM en 5 segundos, la función "waitForSpindleSetSpeed" generará una excepción "SimCncException" (línea 9). Esto detendrá el husillo (línea 10), mostrará el contenido de la excepción en la consola de Python (línea 11) y finalizará el script a petición (línea 12) para evitar que se ejecute nuevamente. Para que funcione el ejemplo anterior, se requiere tener el módulo CSMIO-ENC ya que proporciona información sobre la velocidad actual del husillo.



El ejemplo anterior, tras de una pequeña modificación, puede usarse como una macro M3, que suspenderá la ejecución de gcodu mientras el husillo se encuentre acelerando a la velocidad esperada, o detendrá la máquina en caso de problemas para alcanzar esa velocidad de rotación.

```
Delay_for_checking = 5 # sec

d.setSpindleState( SpindleState.CW_ON )

try:
  d.waitForSpindleSetSpeed( Delay_for_checking )
except SimCncException as ex:
  d.setSpindleState( SpindleState.OFF )
  print(str(ex))
  d.stopTrajectory( )
  sys.exit(0)
```

2) Refrigerante

void setFloodState(FloodState state) - Establece el estado del líquido refrigerante.

ARGUMENTOS:

FloodState state - Estado del líquido refrigerante

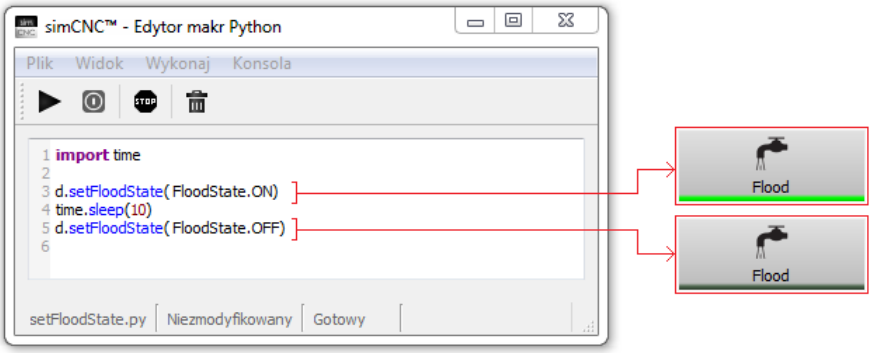
DISTINGUIMOS:

FloodState.ON - Encender el refrigerante
FloodState.OFF - Apagar el refrigerante

EJEMPLO:

```
import time

d.setFloodState( FloodState.ON )
time.sleep(10)
d.setFloodState( FloodState.OFF )
```

Al ejecutar el ejemplo anterior, el refrigerante se encenderá durante 10 segundos.



FloodState getFloodState( ) - Devuelve el estado del líquido refrigerante.

RESULTADO DE LA FUNCIÓN:

FloodState - Estado del líquido refrigerante

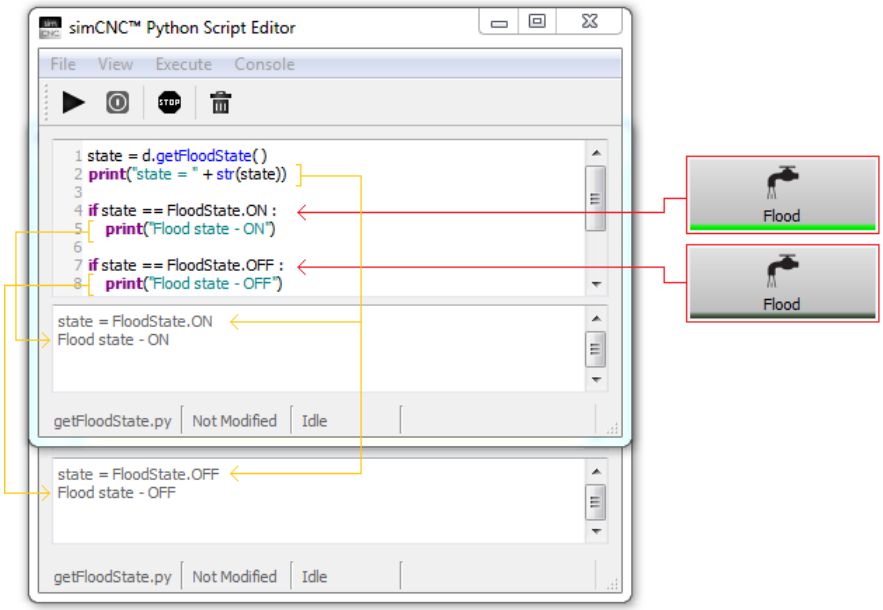
DISTINGUIMOS:

FloodState.ON - Refrigerante encendido
FloodState.OFF - Refrigerante apagado

EJEMPLO:

```
state = d.getFloodState( )
print("state = " + str(state))

if state == FloodState.ON :
    print("Flood state - ON")

if state == FloodState.OFF :
    print("Flood state - OFF")
```

Al ejecutar el ejemplo anterior, el estado actual del refrigerante se mostrará en la primera línea de la consola de Python como resultado de la función d.getFloodState (). En la segunda línea aparecerá un segundo mensaje que describe el estado actual del refrigerante con mayor claridad.
El siguiente boceto muestra cómo funciona el script para ambos estados del refrigerante.



3) Niebla

void setMistState(MistState state) - Establece el estado de niebla de refrigeración.

ARGUMENTOS:

MistState state - Estado de la niebla de refrigeración

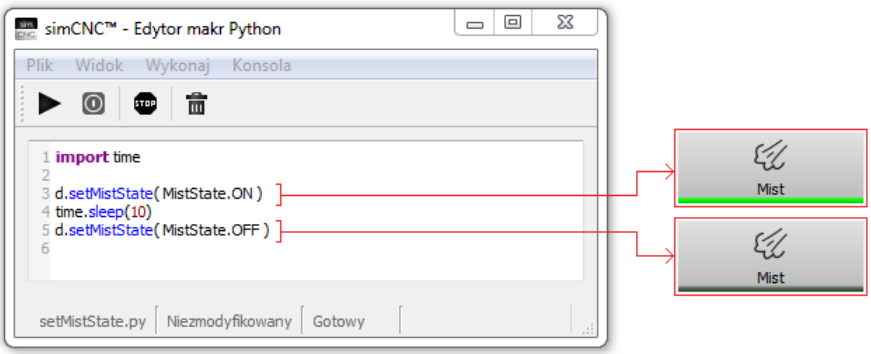
DISTINGUIMOS:

MistState.ON - Encender la niebla
MistState.OFF - Apagar la niebla

EJEMPLO:

```
import time

d. ( MistState.ON )
time.sleep(10)
d.setMistState( MistState.OFF )
```

Al ejecutar el ejemplo anterior, el refrigerante se encenderá durante 10 segundos.



MistState getMistState( ) - Devuelve el estado de la niebla de refrigerante.

RESULTADO DE LA FUNCIÓN:

MistState – Estado de la niebla de refrigeración

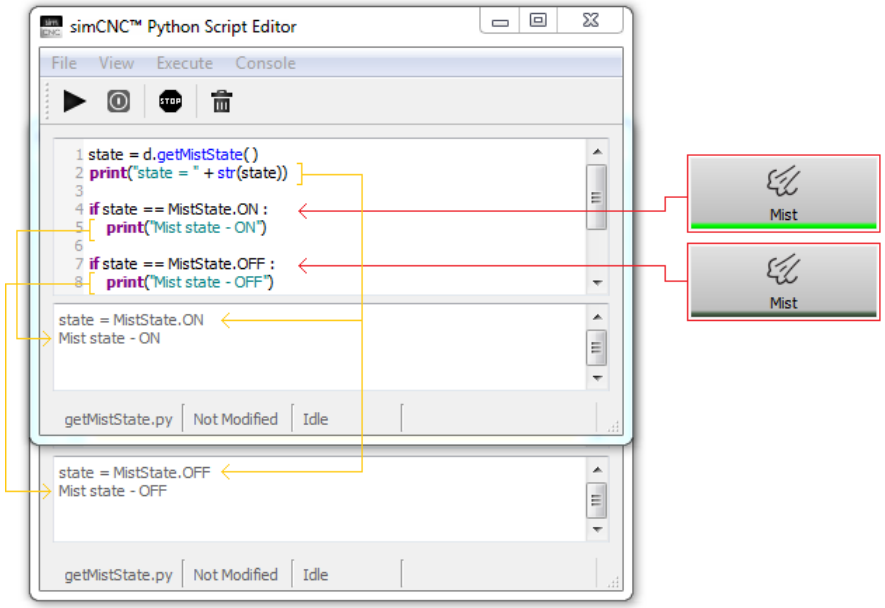
DISTINGUIMOS:

MistState.ON - Niebla activada
MistState.OFF - Niebla desactivada

EJEMPLO:

```
state = d.getMistState( )
print("state = " + str(state))

if state == MistState.ON :
    print("Mist state - ON")

if state == MistState.OFF :
    print("Mist state - OFF")
```

Al ejecutar el ejemplo anterior, el estado de niebla actual se mostrará en la primera línea de la consola de Python, como resultado de la función getMistState (). En la segunda línea aparecerá un segundo mensaje que describe el estado actual de la niebla con mayor claridad.
El siguiente boceto muestra cómo funciona el script para ambos estados de niebla.



X. Desplazamiento de trabajo - bases materiales.

List<float>[6] getCurrentWorkOffset( ) – Devuelve los valores de coordenadas del desplazamiento de trabajo actual.

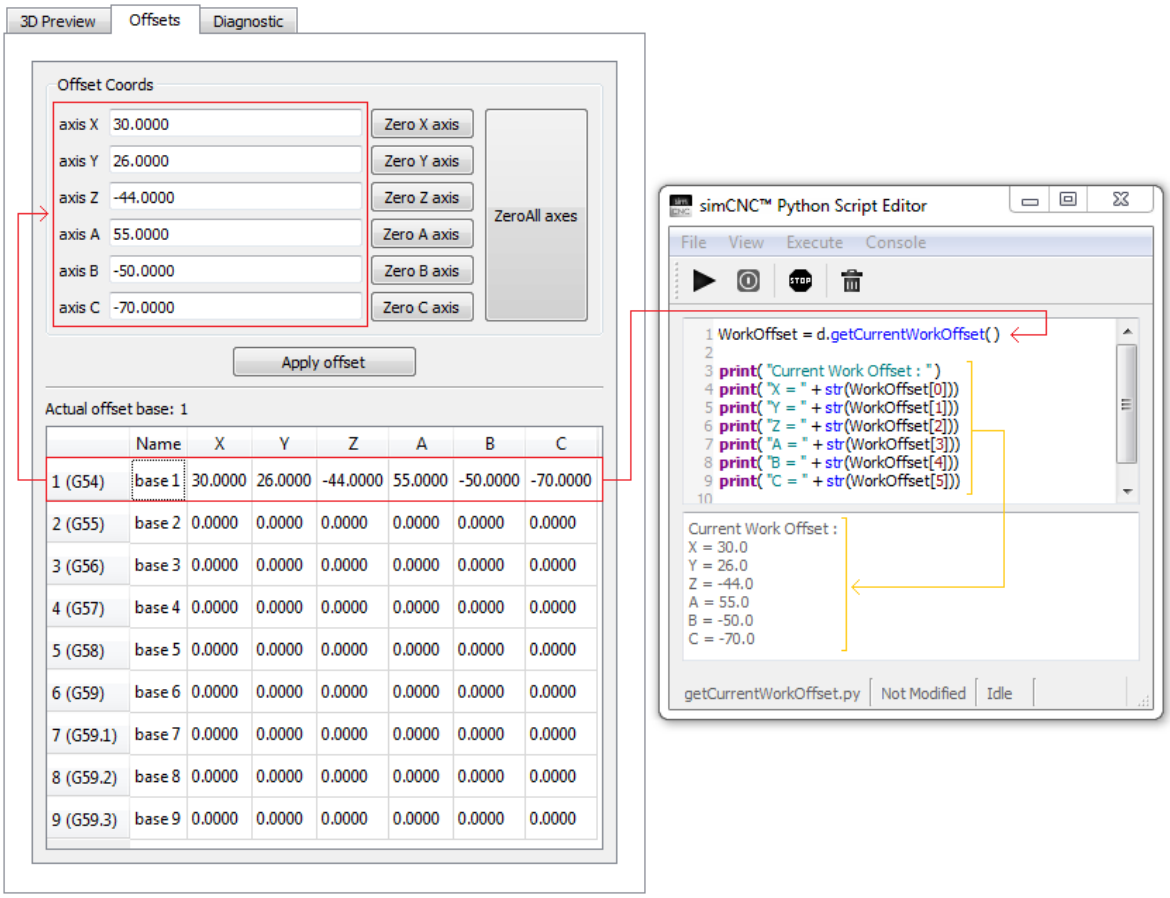
RESULTADO DE LA FUNCIÓN:

List<float>[6] - lista de 6 coordenadas (X, Y, Z, A, B, C) que definen el desplazamiento de trabajo

EJEMPLO:

```
CurrentWorkOffset = d.getCurrentWorkOffset( )

print( "Current Work Offset : " )
print( "X = " + str(CurrentWorkOffset[0]))
print( "Y = " + str(CurrentWorkOffset[1]))
print( "Z = " + str(CurrentWorkOffset[2]))
print( "A = " + str(CurrentWorkOffset[3]))
print( "B = " + str(CurrentWorkOffset[4]))
print( "C = " + str(CurrentWorkOffset[5]))
```

Al ejecutar el ejemplo anterior, el valor de las coordenadas del desplazamiento de trabajo actual se leerá y se mostrará en la consola de Python.



List<float>[6] getWorkOffset(int workOffsetIndex) - Devuelve los valores de las coordenadas del desplazamiento de trabajo en el índice especificado.

ARGUMENTOS:

int workOffsetIndex - índice (número) del desplazamiento de trabajo

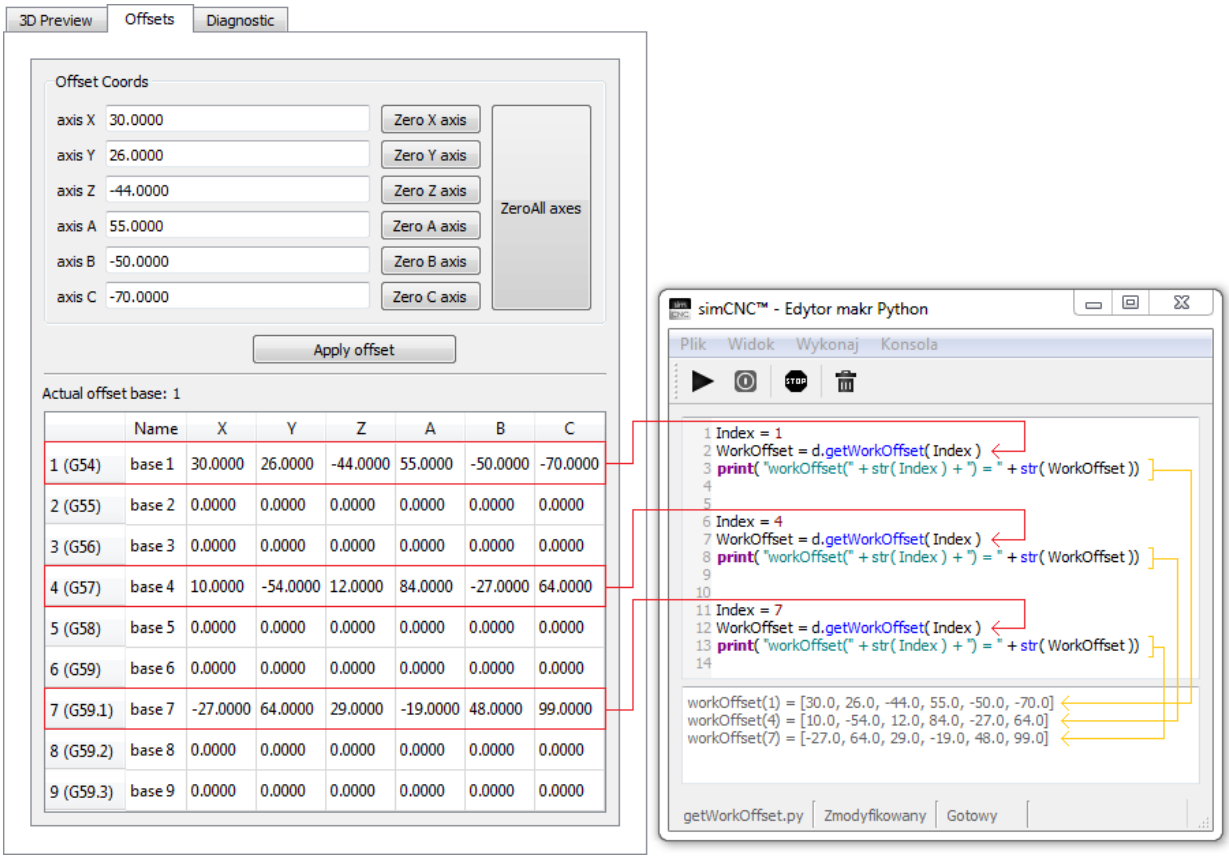
RESULTADO DE LA FUNCIÓN:

List<float>[6] - lista de 6 coordenadas (X, Y, Z, A, B, C) del desplazamiento de trabajo

EJEMPLO:

```
Index = 1
WorkOffset = d.getWorkOffset( Index )
print( "workOffset(" + str( Index ) + ") = " + str( WorkOffset ))

Index = 4
WorkOffset = d.getWorkOffset( Index )
print( "workOffset(" + str( Index ) + ") = " + str( WorkOffset ))

Index = 7
WorkOffset = d.getWorkOffset( Index )
print( "workOffset(" + str( Index ) + ") = " + str( WorkOffset ))
```

Al ejecutar el ejemplo anterior, los valores de coordenadas del desplazamiento de trabajo número 1 (G54), 4 (G57) y 7 (G59.1) se leerán y se mostrarán en la consola de Python.



¡ADVERTENCIA!

Actualmente, simCNC tiene la capacidad de guardar 9 desplazamientos de trabajo (9 bases de datos de materiales).
Como resultado, los desplazamientos de trabajo pueden llevar un índice de 1 (G54) a 9 (G59.3).
Para una mayor claridad: 1 = G54 | 2 = G55 | 3 = G56 | 4 = G57 | 5 = G58 | 6 = G59 | 7 = G59.1 | 8 = G59.2 | 9 = G59.3

int getWorkOffsetNumber( ) - Devuelve el índice del desplazamiento de trabajo actual.

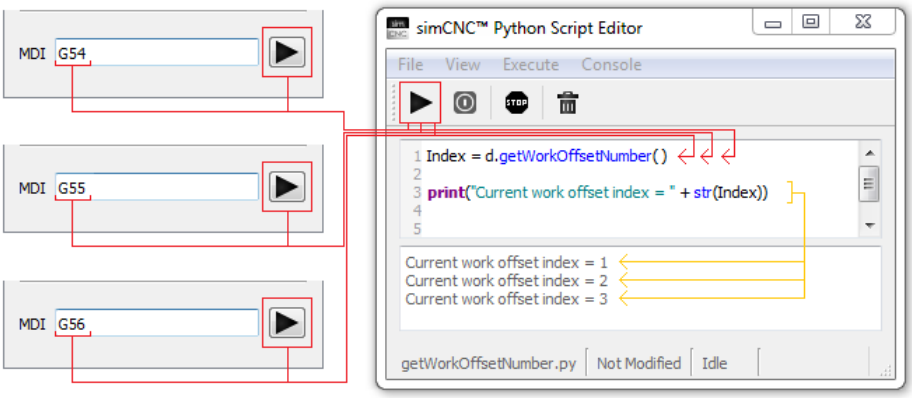
RESULTADO DE LA FUNCIÓN:

int - índice (número) del desplazamiento de trabajo

EJEMPLO:

```
Index = d.getWorkOffsetNumber( )

print("Current work offset index = " + str(Index))
```

Antes de ejecutar el ejemplo anterior, escriba el comando "G54" en la línea MDI y ejecútelo presionando el botón en el lado derecho de la línea MDI. A continuación ejecute el script de ejemplo, y cuando lo haga, la consola de Python mostrará el índice del desplazamiento de trabajo utilizado actualmente. Repita los mismos pasos con los comandos "G55" y "G56" y obtendrá el mismo efecto que en la foto de abajo.



setWorkOffset(int workOffsetIndex, List<float>[6] workOffsetValues) - Cambia los valores de coordenadas del desplazamiento de trabajo en el índice especificado.

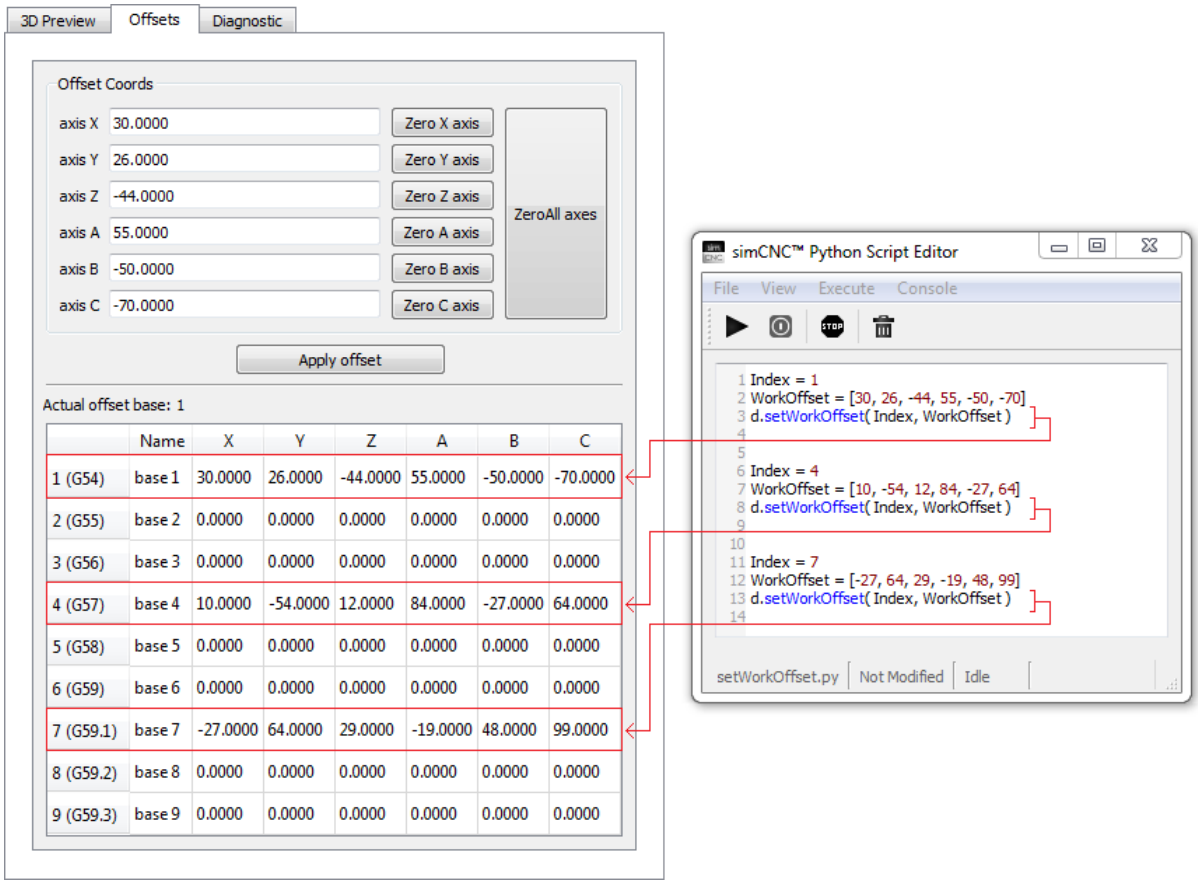
ARGUMENTOS:

int workOffsetIndex - índice (número) del desplazamiento de trabajo
List<float>[6] workOffsetValues - nuevos valores de 6 coordenadas (X, Y, Z, A, B, C) del desplazamiento de trabajo

EJEMPLO:

```
Index = 1
WorkOffset = [30, 26, -44, 55, -50, -70]
d.setWorkOffset( Index, WorkOffset )

Index = 4
WorkOffset = [10, -54, 12, 84, -27, 64]
d.setWorkOffset( Index, WorkOffset )

Index = 7
WorkOffset = [-27, 64, 29, -19, 48, 99]
d.setWorkOffset( Index, WorkOffset )
```

Al ejecutar el ejemplo anterior, los desplazamientos de trabajo con el índice 1 (G54), 4 (57) y 7 (G59.1) se establecerán en los nuevos valores de coordenadas [30, 26, -44, 55, -50, -70], [10, -54, 12, 84, -27, 64] y [-27, 64, 29, -19, 48, 99].



setCurrentWorkOffset(List<float>[6] workOffsetValues) - Cambia los valores de coordenadas del desplazamiento de trabajo actual.

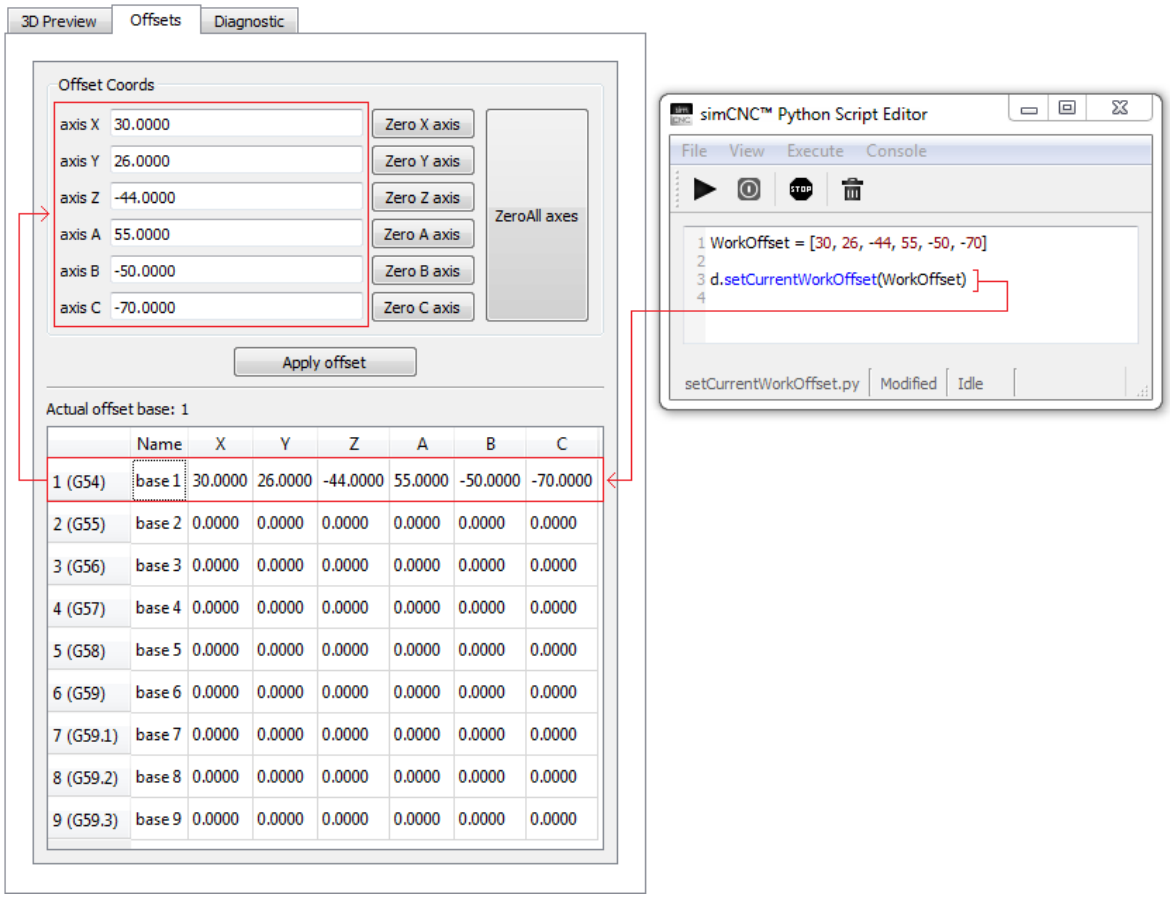
ARGUMENTOS:

List<float>[6] workOffsetValues - nuevos valores de 6 coordenadas (X, Y, Z, A, B, C) del desplazamiento de trabajo actual

EJEMPLO:

```
WorkOffset = [30, 26, -44, 55, -50, -70]

d.setCurrentWorkOffset( WorkOffset )
```

Al ejecutar el ejemplo anterior, el desplazamiento de trabajo con el índice 1 (G54) se establecerá en el nuevo valor de coordenadas [30, 26, -44, 55, -50, -70].



setWorkOffsetNumber(int workOffsetIndex) - Cambia el índice del desplazamiento de trabajo actual.

ARGUMENTOS:

int workOffsetIndex – índice (número) del desplazamiento de trabajo

EJEMPLO:

```
Index = 4

d.setWorkOffsetNumber( Index )
```

Al ejecutar el ejemplo anterior, el desplazamiento de trabajo utilizado actualmente se cambiará a un desplazamiento de trabajo del índice 4 (G57).