Overview

Python is a programming language in which scripts for Lexocad are written.

You do not need a separate Python installation: Python is implemented in Lexocad and runs out of the box.

If you are new to Python, please read the official Python tutorial here: https://docs.python.org/3.6/tutorial/

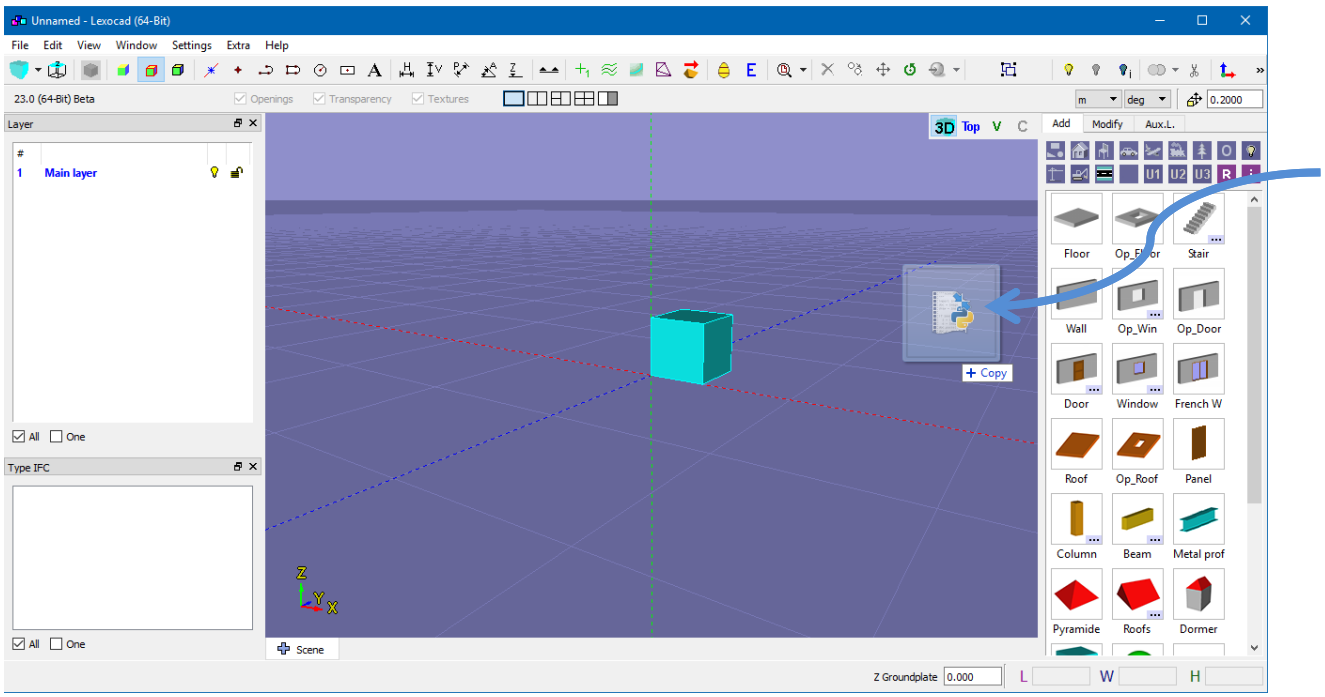
Quick Start

Below is a very simple script. The easiest way to test and get it working:

- Go: *Extra > Python > Python Console*. Copy the script below and paste it inside the interactive console, eventually press the [Enter] key.

```
# Create Box example

# Import Lexocad libraries
import OpenLxApp as lx

# Get the active document
doc = lx.Application.getInstance().getActiveDocument()

# Create a geometry in the document
block = lx.Block.createIn(doc)
block.setXLength(3)
block.setYLength(4)
block.setZLength(5)

# Create an element in the document and assign the geometry to it
elem = lx.Element.createIn(doc)
elem.setGeometry(block)

# Recompute the document
doc.recompute()
```

- Saved scripts (text files with ".py" extension) are loaded into Lexocad by drag'n'drop from Windows File Explorer and other programs.



- Frequently used scripts can be saved inside the folder "Lexocad.x64\Python\ Scripts" and quickly accessed in *Extra > Python > Python Scripts*.
- To see console output (error messages and print statements), go: *Extra > Python > Python Console*.

A closer look at the script

Let us examine the example a bit closer.

```
# Import Lexocad libraries
import OpenLxApp as lx
```

The import statement loads existing Python code from another file - a module. The Python binding to Lexocad is defined by modules. By importing a module you gain access to Lexocad's functions through the API (Application Programming Interface).

```
# Get the active document
doc = lx.Application.getInstance().getActiveDocument()
```

To actually display something in the 3D scene we need to store the “active document” in a variable: everything that we create later will be assigned to the document.

```
# Create a geometry in the document
block = lx.Block.createIn(doc)
block.setXLength(3)
block.setYLength(4)
block.setZLength(5)
```

3D elements have an underlying geometry. Here we create geometry of type “Box” and we assign it to the above document.

```
# Create an element in the document and assign the geometry to it
elem = lx.Element.createIn(doc)
elem.setGeometry(block)
```

Here we create the 3D element and we assign it to the above document. The above geometry is then assigned to the element itself.

```
# Recompute the document
doc.recompute()
```

Once we are done adding a new element, we have to tell Lexocad to rebuild the 3D scene: the method “`recompute()`” applied to the document will do that.

Modules

For a list of the available modules and their classes go to http://www.openlexocad.com/api-documentation/.

Editors

If you want to use a special editor to create your Python scripts, take a look at the following ones:

- Eclipse https://www.eclipse.org/
- Notepad++ https://notepad-plus-plus.org/
- PyCharm https://www.jetbrains.com/pycharm/
- Python Tools for Visual Studio https://github.com/Microsoft/PTVS

Python license

https://docs.python.org/3.6/license.html