Description of Maya Python script: makecity.py

By: Suzanne Berger   Date: April 11, 2019   Updated: May 6, 2022

General Instructions for Maya Python Script Development

In order to run your own developed Python script from Maya, save your script in the *scripts* folder under your local home *maya* folder. Make sure that your Python script has a *.py* file extension. Then in the Maya script editor *Python* tab, enter the following commands:

```
import sys
if sys.version_info.major > 2:
    from importlib import reload

try:
    reload(yourscript)
except:
    import yourscript
```

You can also create a shelf button with these commands by selecting these commands with your mouse then middle-mouse-button click/drag to the Custom shelf. Using the reload function is convenient while developing your script because it will then be unnecessary to restart Maya every time you change your Python script. Note that if you are using Maya 2020 or earlier, the reload function is built-in and importing it from importlib is not necessary. However for Maya 2022 that is based on Python 3, the reload function must be accessed from importlib.

Here is link to complete instructions for developing Maya Python scripts:
https://help.autodesk.com/view/MAYAUL/2022/ENU/?guid=GUID-703B18A2-89E5-48A8-988A-1ED815D5566F

**Python Functions Optional Arguments**

Quote from this link:

Python allows function arguments to have default values; if the function is called without the argument, the argument gets its default value. Furthermore, arguments can be specified in any order by using named arguments.

Introduction

The functions contained in module, *makecity*, should be considered a first pass "hacky" prototype. The intent is to provide exercise examples for students learning Maya Python scripting. It should be noted that resulting Maya geometry would also be achieved using *nParticle Instancer*. The following describes the module's functions. The "ToDo" sections describe suggested additional features that students should do as homework assignments.

Here is a screen grabs showing execution of *makecity* from Maya's script editor:
Note that with newer versions of Maya that use Python 3, the reload function must be accessed using importlib module like this:
from importlib import reload
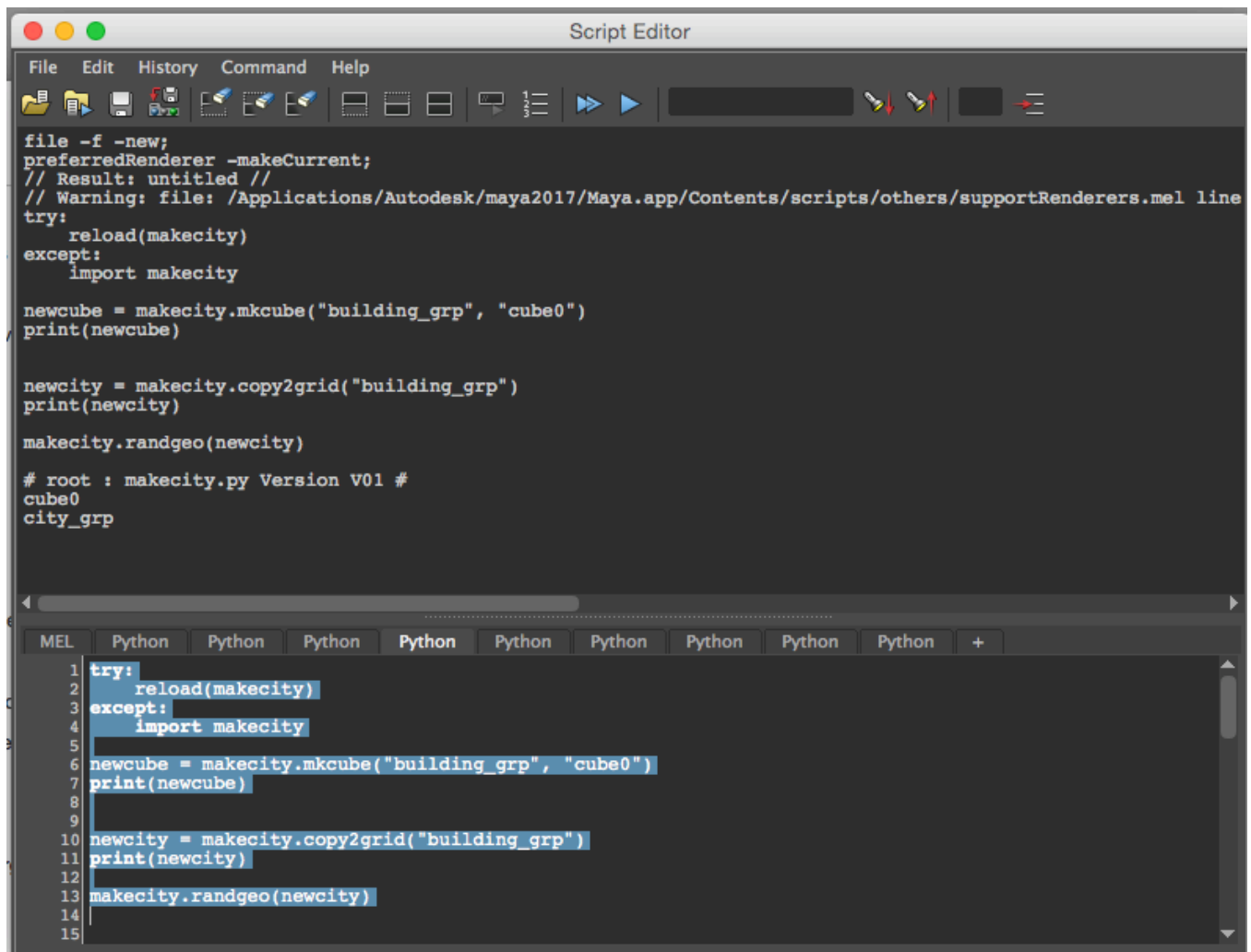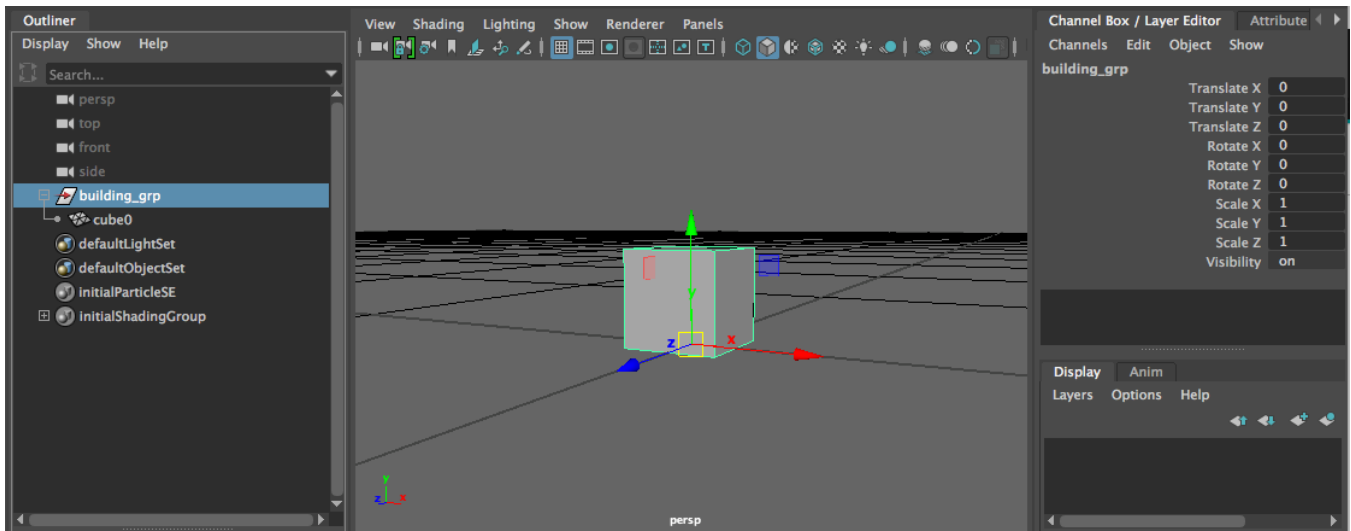
makecity.mkcube

Make polygon cube under group node at origin with dimension 1,1,1 and pivot at origin. This function is convenient for development of subsequent functions.

Args:
   grp_name: object name of parent group to contain cube
   cube_name: object name of cube node

Example:
   makecity.mkcube("building_grp", "cube0")

makecity.mkbldgs

ToDo: This function is not yet implemented.

Create simple geometry nodes and group under group named by input argument.

Args:

    grp_name: group node name to contain building geos

ToDo: Complete function

Create group node to contain various building models that get subsequently chosen at random in duplication functions: copy2grid and copy2vtx. Buildings can either be created as simple geometry or provided in selection list. Building model options should be provided as keyword arguments.

makecity.copy2grid

Duplicate geo over grid. Total number of copies = numrows * numcols.

Args:
　　bldg_grp: group node containing geo to be copied
　　numrows: number of rows in grid, default = 4
　　numcols: number of columns in grid, default = 4
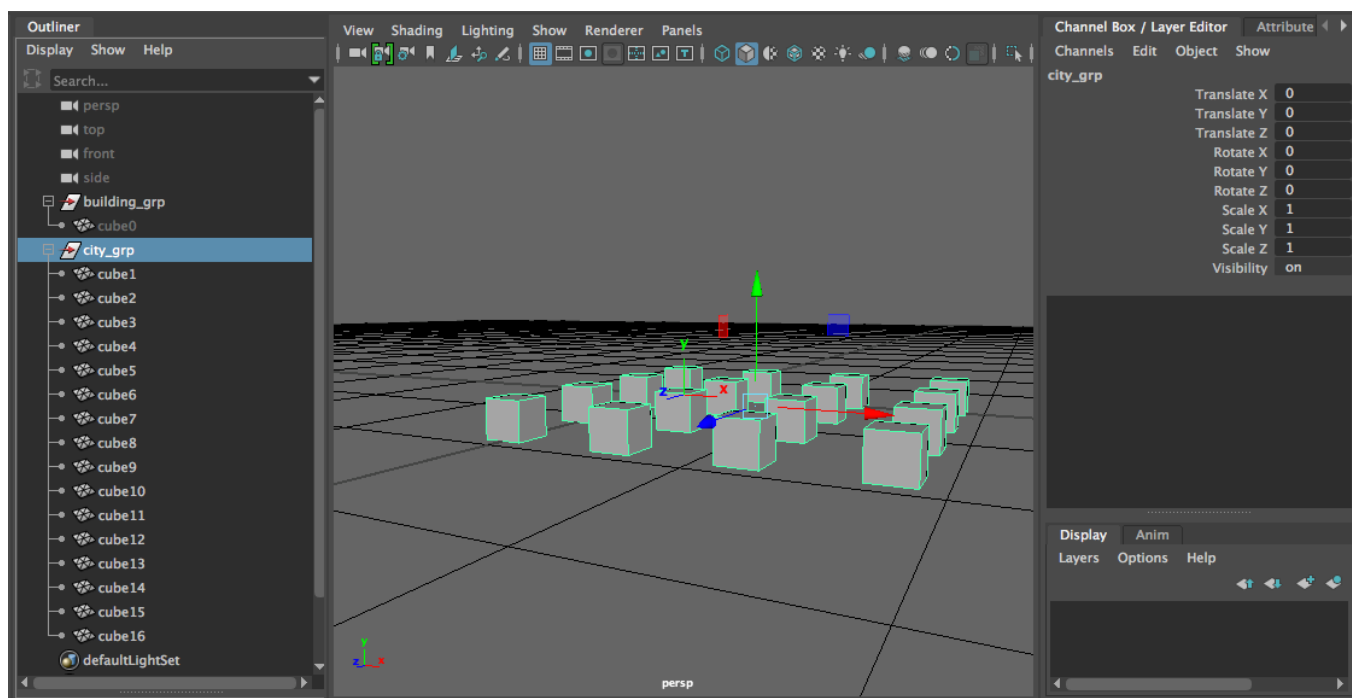　　stepx: x translate step for each geo copy, default = 3
　　stepz: z translate step for each geo copy, default = 3
　　hide: hide original geo after copies completed, default = True

Example:
　　newcity = makecity.copy2grid("building_grp")

ToDo: If more than 1 geo under group, then use *random.choice(bldg_list)* to get a random choice geo to be copied.


makecity.randgeo

Randomize scale on all geo under group.

Args:
    city_grp: group node containing geo to randomize. If None then get geo from selection.

ToDo: Add optional arguments for range values to randomize scale. Use hard-coded args in random.uniform call as default values for optional random scale arguments. Add optional arguments to randomize rotation or do not randomize rotation at all.
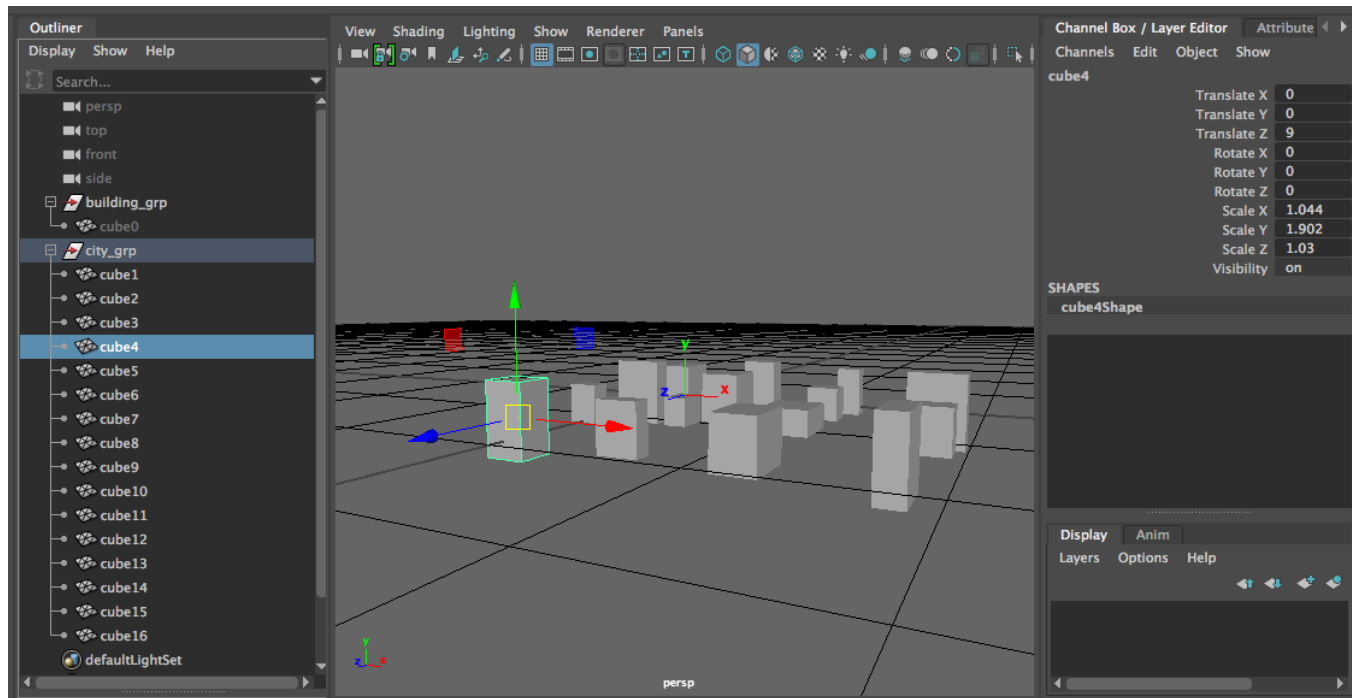
makecity.copy2vtx
ToDo: This function is not yet implemented.

Copy geometry objects under input bldg_grp argument to all surface points on input obj arguments.

Args:
    bldg_grp: group node containing geo to be copied
    obj: poly or nurbs surface containing position points for copy

Example:
    new_city = makecity.copy2vtx("building_grp", "nurbsPlane1")

Final Details

ToDo: In order to make random behavior repeatable when running script over and over, place a call to the random.seed(int) function at the top of the makecity.py script under the globals section. Example: random.seed(25). Note that because this call will have global scope for entire module, there should be no indentation.

How To Edit This Script

To edit/modify *makecity.py* do the following:

1. Go to github link: https://github.com/skoshiwosh/democode
2. Click on makecity.py listed under democode
3. Open the standard Python editor, *idle*,
4. From the main menu click File->New File. Note that the *idle* app should already be pasted to your desktop. If not ask, Markus for help.
5. In github, click on <Raw> in top right header of *makecity.py* display. The web page will change to raw text that you can then copy/paste into *idle* editor
6. Save the new file to *makecity.py* in the *scripts* folder under your local home *maya* folder.
7. Before adding changes, test script with no changes to make sure copy was correct. (i.e no unexpected hard-tabs in copy that would break script)
8. Proceed to add changes to the code. Change and save versions in an iterative manner and test along the way.

Look at figures on the next page. The first one is a screen grab of an example scene created using modified makecity.py script. The second figure shows python commands for creating a polygonal shape by modifying a cube. The action was first done manually and the echoed mel commands were then translated to Maya's Python commands in the maya.cmds module.