

Description of Maya Python script: makecity.py

By: Suzanne Berger

Date: April 11, 2019

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:

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

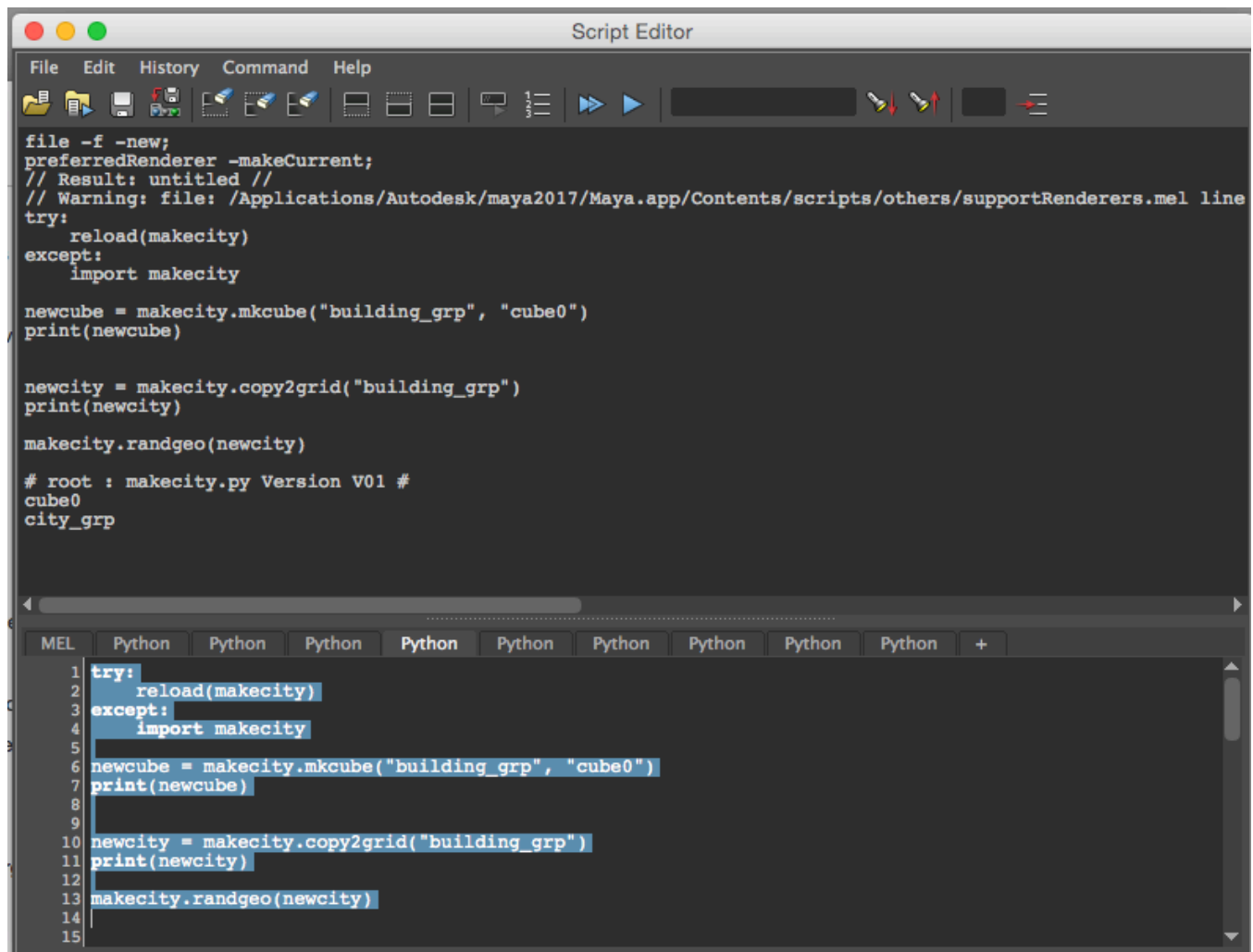
You can also create a shelf button with these commands by selecting these commands with your mouse then middle-mouse-button click/drag to the Custom shelf. Using the reload function is convenient while developing your script because it will then be unnecessary to restart Maya every time you change your Python script.

Here is link to complete instructions for developing Maya Python scripts:
<http://help.autodesk.com/view/MAYAUL/2018/ENU/?guid=GUID-703B18A2-89E5-48A8-988A-1ED815D5566F>

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 grab showing execution of *makecity* from Maya’s script editor:



The screenshot shows the Maya Script Editor window. The title bar reads "Script Editor". The menu bar includes "File", "Edit", "History", "Command", and "Help". The toolbar contains icons for file operations, editing, and execution. The main text area displays the following Python code:

```
file -f -new;
preferredRenderer -makeCurrent;
// Result: untitled //
// Warning: file: /Applications/Autodesk/maya2017/Maya.app/Contents/scripts/others/supportRenderers.mel line
try:
    reload(makecity)
except:
    import makecity

newcube = makecity.mkcube("building_grp", "cube0")
print(newcube)

newcity = makecity.copy2grid("building_grp")
print(newcity)

makecity.randgeo(newcity)

# root : makecity.py Version V01 #
cube0
city_grp
```

Below the main text area is a tabbed interface with tabs for "MEL" and multiple "Python" scripts. The first "Python" tab is active, showing the same code as the main text area, with line numbers 1 through 15 on the left margin.

makecity.mkcube

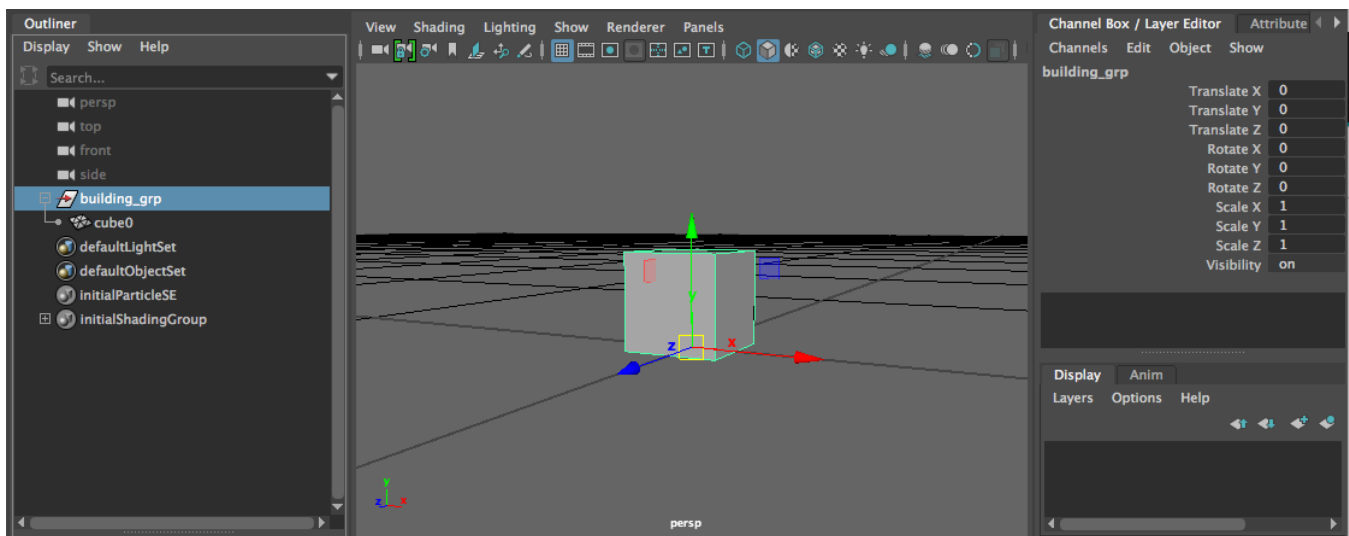
Make polygon cube under group node at origin with dimension 1,1,1 and pivot at origin. This function is convenient for development of subsequent functions.

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.copy2grid

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

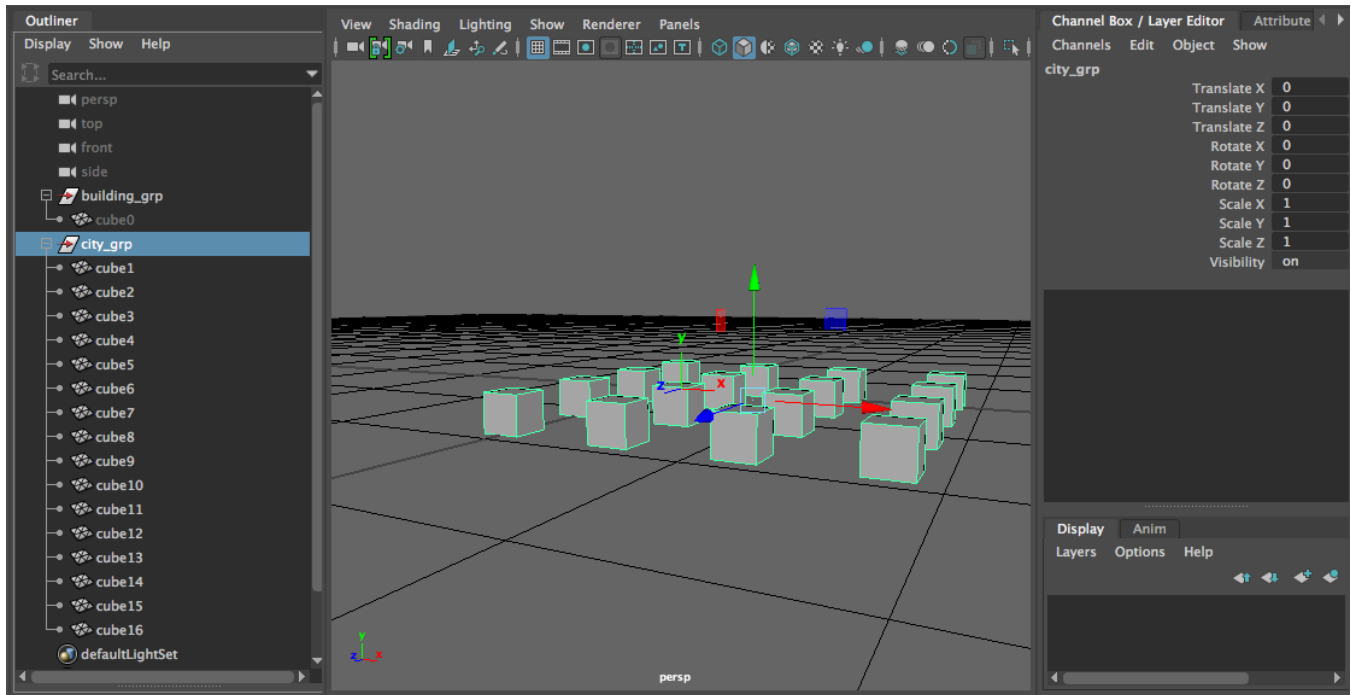
Args:

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 to be copied.



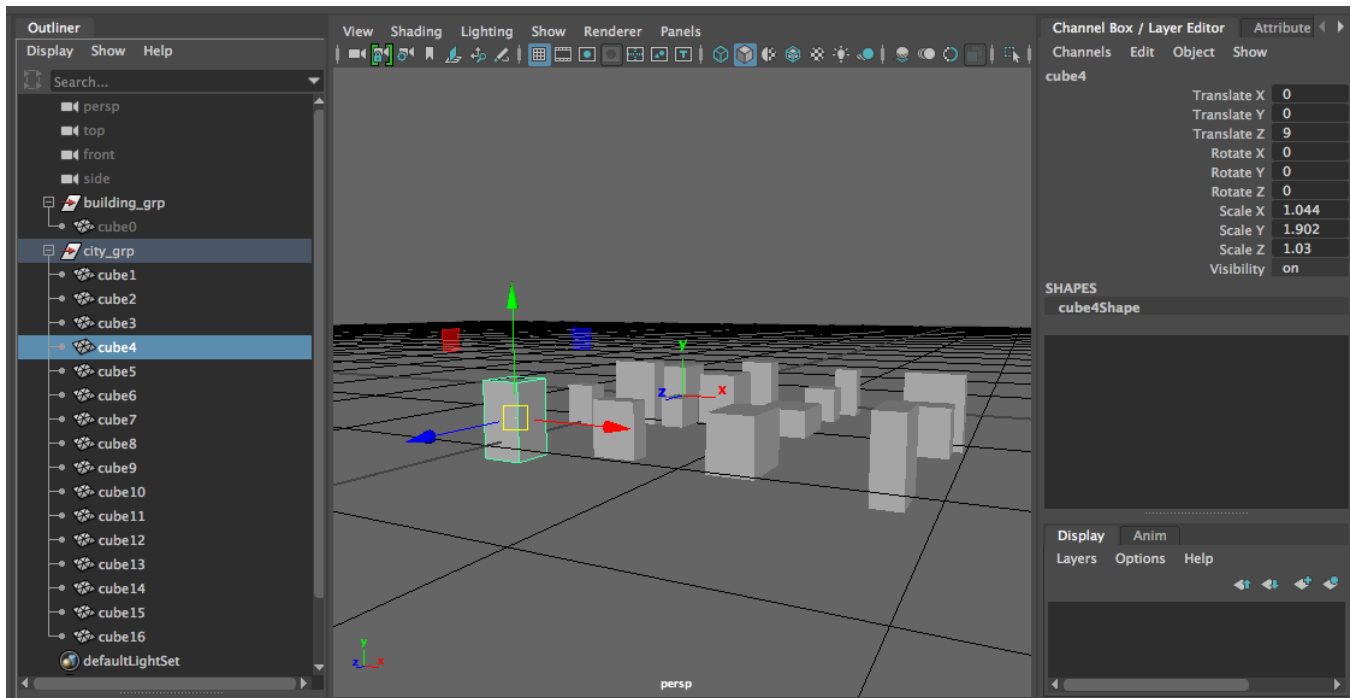
makecity.randgeo

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 to randomize rotation as well. Add optional arguments for both scale and rotate ranges of randomization.



makecity.copy2vtx

ToDo: !!! This function is not implemented yet !!!

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", "nurbsSphere1")
```

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. Proceed to add changes to the code. Change and save versions in an iterative manner and test along the way.