2019

02 Getting Started with VPython

W.A. Morgan
Dickinson College

L.Q. English
Dickinson College

Getting Started with VPython

2.1 "Hello."

VPython is a language that is pretty easy to use, once you get used to it. Python is becoming widely used by scientists around the world, and VPython is a visual implementation of it. It will help you visualize the physics you will be learning.[1]

To learn the basics of programming in VPython, the first thing we will do is one of the simplest – having the program print the phrase "Hello, world!" – a traditional first program.

To do so, we will be in the Glowscript-VPython programming environment – an environment that is easy to use and understand. Here is what you do:

1. Launch a web browser (try Google Chrome), and go to the site www.glowscript.org . The webpage will look like this:

2. Sign in with a Google account.

3. Follow the link where it says "your programs are **here**."

4. Click on the "Create New Program" link.

5. In the dialog box that appears, type the title "MakingShapes", then click the "Create" button.

[1] See also D. Schroeder, "Physics Simulations in Python" (2018).

Figure 2.1: The Glowscript webpage.

6. You will now see an editing space that is blank except for the line "GlowScript 2.8 VPython" (possibly with a different version number). Click in the white space below that line and type the following, verbatim:

```
print("Hello, world!")
```

7. Then click the "Run this program" link at the top of the page. Your editing space should then vanish, replaced by a new text area with the words "Hello, world!"

8. Click on "Edit this program" and replace "Hello, world!" with something else. Run the program. You should get similar results.

If the program didn't work, check the spelling and capitalization and punctuation. Make sure that you typed the line precisely as shown above.

GlowScript-VPython is a programming environment that stores your programs in the cloud. The interface is very spare – however, there should be no difficulty just starting it up, making new programs, saving them, and managing them.

If you click "Edit this program", you can see the program you just typed. The first line, which is the same for all GlowScript programs, tells the system the language (VPython) and version (2.8) that is to be used. You must have this line, and you should always just leave it alone.

The next line tells the computer to call the print function and passes to it a string of characters. Don't worry that you don't know what the italicized words mean – we'll explain them here:

- The *function* is a pre-defined task that does something – in this case, print to the screen. Just as you don't need to know how a square-root key on a calculator works, a function does its job on the information *passed* to it. This information is called the *argument* or the *parameter* (they are slightly different, but there is not much of a difference). Here the argument is the *string*.
- The *string* is text that is either single- or double-quoted.
- To tell the program what to do it and when, the function is *called*, which is done in the act of stating the function (here, "print").

2.2 Box – Your First Shape

Now, let's do something a bit more exciting.

On a new line below your print command, type the following command:

```
box()
```

Here you're calling a function called `box`, and passing it no arguments at all (but notice that you still need the parentheses). Run the program again, and you should see a black rectangular area (called a canvas) containing a light gray square (the box), such as one shown in Figure 2.2. Again, this function is doing a lot of work "under the hood", but you do not need to concern yourself about how.

The box that you see on the screen is actually three-dimensional on the screen. You are viewing it closely from one side, so it appears as

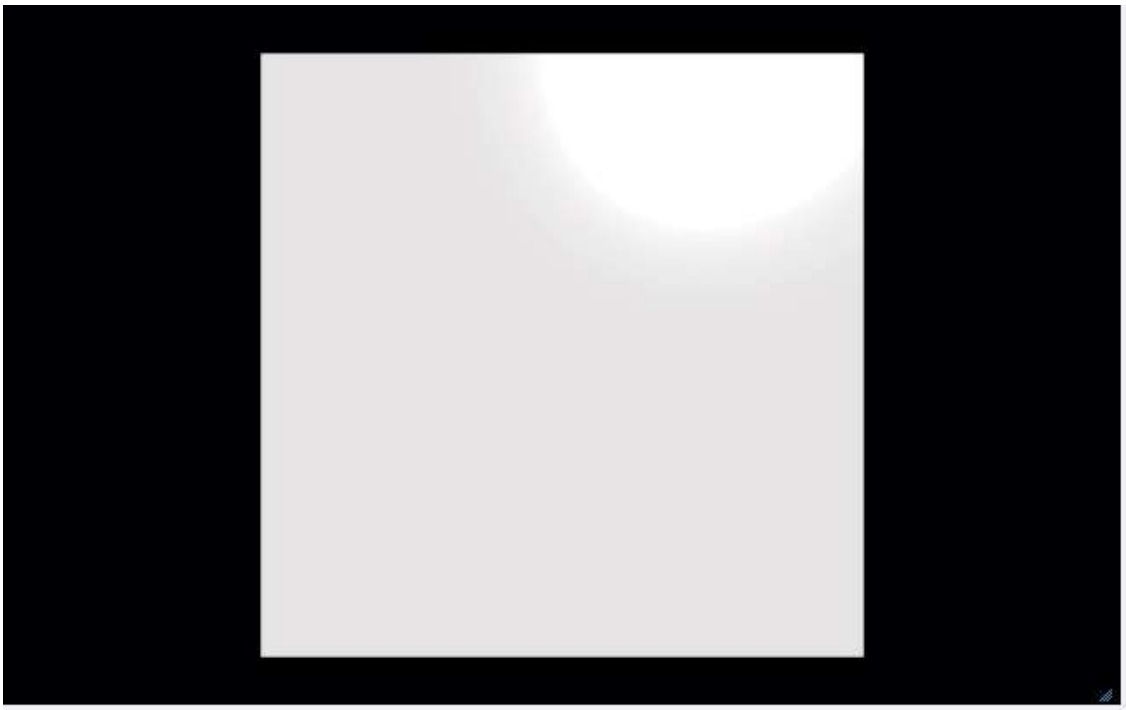

Figure 2.2: A Simple Box

a square. To change the perspective, you can do three things:

- **Rotate**: Right-click button on your mouse and drag the box one way or another, or, if you don't have a right mouse button (such as on a Macintosh), press the control key and drag using your mouse or trackpad.
- **Zoom**: Use the scroll wheel on your mouse, or, if there isn't one, press the alt or option key and drag using your mouse or trackpad.
- **Pan**: Hold down the shift key while you drag using the mouse or trackpad.

You can change the attributes of your box by passing some different parameters to the box function. Try typing this to replace the original box line:

```
box(pos=vector(1,0,0),size=vector(.5,.3,.2),
                color=color.green)
```

Here you're providing *three* parameters, separated by commas, and you're identifying them by their names, which means you can provide them in *any* order. The *pos* parameter specifies the position of the *center* of the box in Cartesian coordinates; the size parameter specifies its dimensions (so here, it is not a cube; and the color parameter is self-explanatory. In the first two cases, the parameter values are three-dimensional vectors, which you create using the vector function. We will be learning more about this powerful function soon. This function in turn takes three parameters, x, y, and z, which are usually given in that order. When first presented and before any rotation, the x direction points to the right; the y direction points up; and the z direction points directly outward, toward you (or "out of the screen").

You can learn more about other colors by clicking the **Help** link at the upper-right corner of the window. It might be beneficial to open the help page in a new browser tab. Then, from the second drop-down menu in the left sidebar, choose "Color/Opacity".

There you will find a list of pre-defined colors, and also see how you can use the vector function to create arbitrary colors.

Exercise: Create at least two more boxes, so you'll have a total of at least three, each with different positions, shapes, and colors. Keep their positions within the range -5 to 5 in each dimension, and keep their sizes small enough to leave plenty of room for more shapes with that range. Make sure to "Grab" (a MacOS program) a screenshot of what you did.

2.3 Another shape

Fortunately, VPython provides functions for creating many different shapes, but in this course you'll need just two others: spheres and cylinders. Try this instruction to create a sphere:

```
sphere(radius=0.25)
```

The `sphere` function can also accept the `pos` and `color` parameters, so use those now to change the defaults according to your taste. Don't forget to separate the parameters by commas!

2.4 Comments

Sometimes you want to have some comments to remind you what a certain line or area of a program is doing. All you need to do is put a # sign in front of the comment. Then VPython simply ignores everything following the # sign on that line. (It is very easy to "comment out" a line of code you want to **not** run, sometimes as a diagnostic. Then you can remove the # sign if you want the program to execute that command).

Exercise: Put a multi-line comment at the top o your program right after the line "GlowScript 2.8 VPython", to indicate the name of your program, your own name, and the day when you created it, and to give a one-sentence description of what it does. (From now on, please include a similar comment at the top of every program that you write.)