VPython Reference Sheet

Download VPython: You can download VPython from http://vpython.org and install it on your own computer.

On-line help: There is an on-line reference manual for VPython. In the text editor (IDLE), on the Help menu choose "Visual". The first link, "Reference manual", gives detailed information on spheres, arrows, etc. In the text editor (IDLE), on the Help menu choose "Python Docs" to obtain detailed information on the Python programming language upon which VPython is based. We will use only a small subset of Python's extensive capabilities.

1. Creating a program file

(a) Bring up IDLE for Python by double-clicking on the snake icon. You will get an empty edit window.

(b) Enter the following two lines of code at the beginning of your program.

```
from visual import *
from __future__ import division
```

(***Note: this should be typed*** "**underscore underscore future underscore underscore**". This is a total of four underscores; two before "future" and two after "future." ***There are no spaces between the underscores and "future"***. This is an important line of code; it instructs Python to consider a fraction like 1/2 to be a floating point number 0.5, instead of taking the integer part of the result, which would be zero in this case. In future versions of Python this will be the default behavior, but it isn't yet, so we need to invoke it.)

(c) From the file menu choose Save As. Save on your hard drive, making sure to add ".py" to the end of your filename.

2. Creating objects in VPython

`sphere:`

A sphere has attributes `pos`, `radius`, and `color`. The `pos` attribute specifies the location of the center of the sphere.

```
ball = sphere(pos=vector(3, 5, -11), radius=0.15, color=color.magenta)
```

The preceding statement creates a sphere named `ball` whose attributes can be referred to with the dot syntax:

```
ball.pos, ball.radius, ball.color
```

`arrow:`

The basic attributes of an arrow are `pos`, `axis`, and `color`. The `pos` attribute specifies the location of the tail of the arrow. The `axis` attribute specifies a vector that points from the tail to the tip of the arrow.

```
arrow(pos=vector(-2, -4, 5.5), axis=vector(-31.5, 25, -3.7), color=color.yellow)
```

3. Scalar constants or variables in VPython

At the beginning of a program, you can create named constants. For example,

```
g = 9.8
G = 6.7e-11
oofpez = 9e9       ##One Over Four Pi Epsilon Zero
qproton = 1.6e-19  ## charge on a proton
s = 1e-8           ## a constant distance
```

In the rest of the program you can use these names in equations, just as you would on paper.

4. Vector variables in VPython

Creating a vector:

```
velocity = vector(0, -1.8e4, 0)
```

Components of a vector may be referred to by adding "`.x`", "`.y`", or "`.z`" to the name of a vector:

`velocity.x` is the x-component of the vector "velocity" defined above

`baseball.pos.z` is the z-component of the position vector of a sphere named "`baseball`"

5. Common mathematical expressions

To square a variable or number in VPython, you type `**2`; a^2 would be `a**2`; and $(r_x)^2$ would be `r.x**2`

To take a square root of a number or an expression, use `sqrt( )`; $\sqrt{3}$ would be written `sqrt(3)`.

π is a constant named "`pi`", which is already defined by VPython.

Sines and cosines are `sin()` and `cos(),` for example `sin(pi/2)`. Trig functions use radians, not degrees!

To add a quantity to a variable, type:

```
myvariable = myvariable + 3
```

This means: get the current value of `myvariable`, add 3 to it, and replace the previous value of `myvariable` with the result.

6. Simple loops in VPython

```
deltat = 0.5
t = 0
while t < 10:     ## statements to be done inside loop are indented
   t = t + deltat
   print t
print 'End of loop'
```

In this code, the variable t is given the initial value of zero before the loop begins. The while statement instructs VPython to execute the indented statements over and over, until the value of t becomes equal to or greater than 10. At that point, the indented statements will no longer be executed. The next unindented statement is executed, which in this case prints 'End of loop'.