A tutorial.

1 Introduction

Things is an API for animating vector graphics in Python. It's written in Python and I really would like someone to fix that. It should be in super-quick C or something. Come on, gimme a hand!

This document purports to teach you how to use Things. It may do so, but I expect to make many mistakes and do other foolish things. Your best bet, as always, is to hit the source. Check out the demos (all have 'thingum' in their names) and check the actual Things code too. It's horrible to behold, but it's open.

2 Where stuff is

Please find the "tutorial" folder and work inside it. If you have not installed Things via setup.py then simply make a soft-link to "Things" *in* the tutorial folder (`cd tutorial; ln -s ../Things`). After this, scripts in here can do "from Things import ...".

This tutorial begins with an Inkscape SVG file. Open the file called "tutorial.svg". See Fig 1 on page 3.

2.0.1 The Origin

The Inkscape "page" or "canvas" is the lower-right quadrant of the total Things canvas. *This makes the top-left corner the middle of your Things canvas*. I have created a layer (see Fig 2 on the next page) called "setup" and draw a grid so that you can see a cross-shape. The middle of that cross is the *origin* (0,0) – which is the top-left of the Inkscape page.

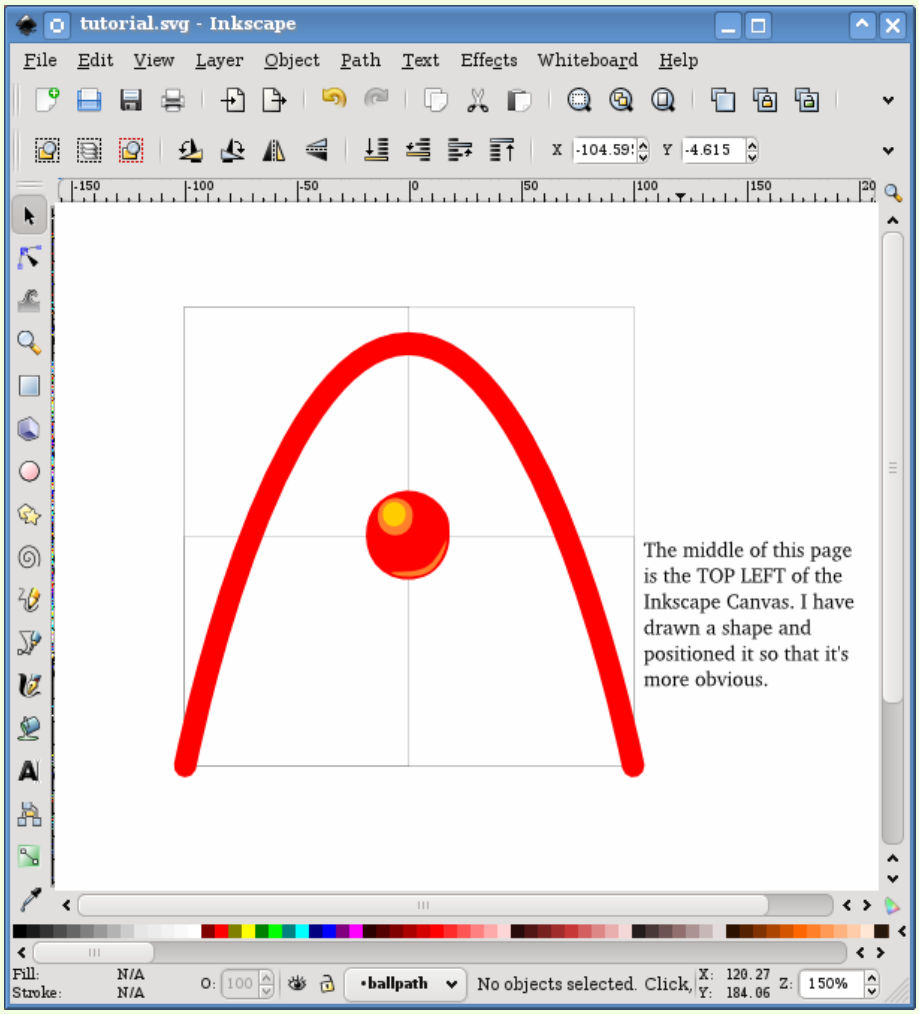

Fig. 1: tutorial.svg file

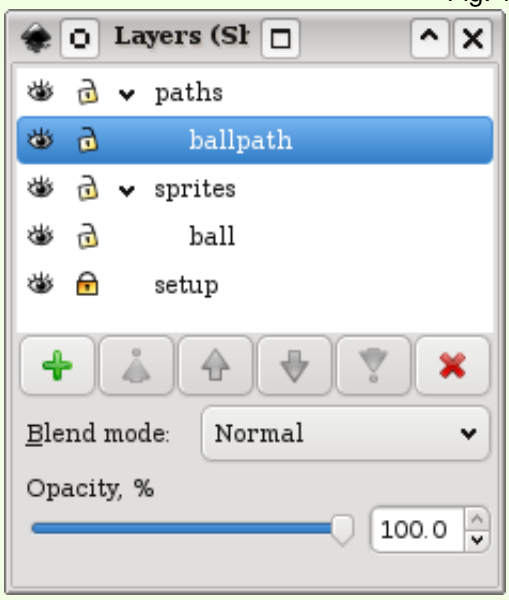

Fig. 2: Layers dialogue

2.0.2 Inkscape Settings

To make life *much* easier, open the global Inkscape preferences dialog and go to the section "Transforms". Make sure you choose "*Preserved*" from the choices in "Store transformation".

The trick is always to start your drawing where you want it to appear in Things. Usually, this is near the origin. Once you have done this, any change thereafter will be via a "transform" tag in the SVG file.

3 Layering in the SVG

If you look at Fig 2 on the preceding page you will see it's divided up into several layers. In this case "paths" and "sprites". The red ball (which looks like a nose) is in the sprites layer (on a sub-layer, the name of which is not important). The actual ball has been given an id. Right click it and choose > Object Properties. The id is "ball". The curved line is in a layer called "paths" and it has an id of "swoopy".

The two main layers (sprites and paths) are mandatory names. The sub-layer names are not important (they help you arrange things) but you *must* use them to contain you art. You can have many sub-layers, as many as you need to get your drawing done; just be sure to keep them all under a main layer.[1]

Sprites: These are "shapes" or "drawings" or just plain "Things". Like the ball in the tutorial, a sprite is any number of vector shapes you need to draw whatever you want.

1. You can have many sprites on each sub-layer.
2. All sprites need a unique id.
3. You can use many sub-layers below the "sprites" main layer.

Paths: These are simple lines, usually with a start and an end. They are used for making other Things follow along them. (Paths can also simply be drawn.)

1. You can have many paths on a single sub-layer.
2. Give each path a unique id.
3. You can use many sub-layers below the "paths" main layer.

Masks: These are similar to paths (similar as in identical...) their use is for clipping other Things and for providing hit-areas for things like buttons. Rules are the same as those for Paths.

Loops: These are like Sprites, only there are many frames in a loop. Loops are *groups* given sequential ids on the *same* sub-layer. The name of the *sub-layer* is how you identify what loop to draw – i.e. sub-layers names are important to Loops. The ids of each *group* (frame) are then sorted in order and drawn one after the other.

1. Keep all related groups, with sequential ids, on the *same* sub-layer.
2. Use ids like "Walk001", "Walk002" ... "Walk050" – to ensure correct sorting.

4 Inkscape tips

You will have to experiment for yourself, but there is one major tip: Draw stuff close to the origin.

1. Draw stuff close to the origin. Have I mentioned this?
2. Start with a rectangle (centered around the origin) and immediately *group* it (to itself). Then double-click it to go inside the group and draw your actual sprite or loop within that.
   This allows you to move the group around and keep the drawing properly aligned to the origin.

[1] In other words: It's a tree with main layers (sprite, paths, masks, loops) and branches (sub-layers) but that's it – no deeper; do not put sub-layers under other sub-layers.

3. If you use clones, be careful how you work with them. Draw them someplace off to the side and then never move them again. Also, it's always a good idea to make clones out of groups, so that you can add stuff to them later.
4. To have certain things excluded from your final animation, give them a *Label* of "hidden". (In the Object Properties; look below the id entry.) All items labelled hidden will not be drawn.
5. More to come as I find 'em....

5 Writing some code

5.1 Importing Things

Importing modules all depends on where your code is and where the modules are. For now, we make the working assumption that the folder "Things" is in the same place as your code.

Listing 1: Importing Things

```
1 from Things.ThingsApp import *
2 from Things.Thinglets import *
3 from Things.BoxOfTricks import *
```

1. ThingsApp: This is the main controller for Things. You always need it.
2. Thinglets: These are extra Things – see the API docs. This is optional.
3. BoxOfTricks: This is a mixture of objects and functions to help you draw things quickly.

5.2 Using a "Bag Of Stuff"

Your animations will require fonts, vectors and images (and later-on, sounds) to function – This is all known collectively as "Stuff". With Things, you can make "bags" to store stuff for later use. To open the tutorial.svg file into a bag, you would do this –

Listing 2: Declaring a Bag Of Stuff

```
1 bos = BagOfStuff()
2 bos.add("tutorial.svg","art")
```

The variable *bos* is now your bag. This bag hold the entire svg file – all the sprites, loops, paths and masks within it are reachable through the *bos* variable. You can also add more stuff to bos – that's the use of the "art" parameter; it's a key – choose another key for another resource and add it to bos.

Here are a few examples:

Listing 3: Bag Of Stuff examples

```
1 bos["art:ball"].draw( context )
2 bos["art:swoopy"].draw( context )
3 bos.add("picture.jpg","face")
4 bos["face"].height
5 bos["face"].draw( context, 20, 50 )
```

1. Fetches id "ball" and draws it to the current Cairo context (more on this later.)
2. Fetches id "swoopy" (the curved path) and draws it.
3. Adds a picture to the same bag. It is given a key of "face".
4. Get the height of that picture.
5. Draw the picture to point (20,50).

6 A quick animation

6.1 Run the demo

Look for "ball1.py" in the tutorial folder. Run it like this:

```
python ball1.py
```

6.2 Looking at the code

Listing 4: ball1.py

```
1  from Things.ThingsApp import *

3  class Ball(Thing):
4    def __init__(self ):
5      Thing.__init__(self)
6      self.keys( "#---------#----------#-------------#--------------#",
7          Props(x=-100,y=100),Props(),Props(x=100,y=100),
8          Props(),Props(x=-100,y=100)
9          )
10   def draw(self,ctx,fr):
11     bos["art:ball"].draw(ctx)

13 bos = BagOfStuff()
14 bos.add ( "tutorial.svg", "art")

16 app = AllThings ( 400, 400, speed = 30, title = "A␣Ball" )
17 app.add(Ball())
18 app.comeToLife ( )
```

Line 1: Here we import everything from ThingsApp.

Line 3: We make a Ball. You can skip this quickly.

Line 13 & 14: We make a Bag Of Stuff called "bos", then we add our svg file to it.

Line 16: We make an "app" – This will make the window (400 by 400), set the animation speed to 30 milliseconds and give the thing a title.

Line 17: We make an actual Ball() and add it to the app. The Ball has become a *child* of the app.

Line 18: We bring the whole thing to life!

Now, let's look at the Ball class:

Line 3: We define Ball as being a "Thing". Thing is one of the basic classes in the API. There are others, but we don't need them now.

Line 4: We always start with an __init__ for the class. It's standard Python, I hope.

Line 5: Tell the Thing (my super class) to initialize too. (We just overrode the __init__, so we have to call this one manually to get it to happen.)

Line 6: This is the magic line: It sets up a bunch of keys for the Ball. Each "#" sign is a *keyframe*. Each "-" sign is a *tween* frame. So you are seeing a progression from one place to another. For each *keyframe*, there is a Props object in the parameters (lines 7 and 8). Each Props is the "properties" of a keyframe and they define the x, y, scale, rotation and bunch of other stuff too.

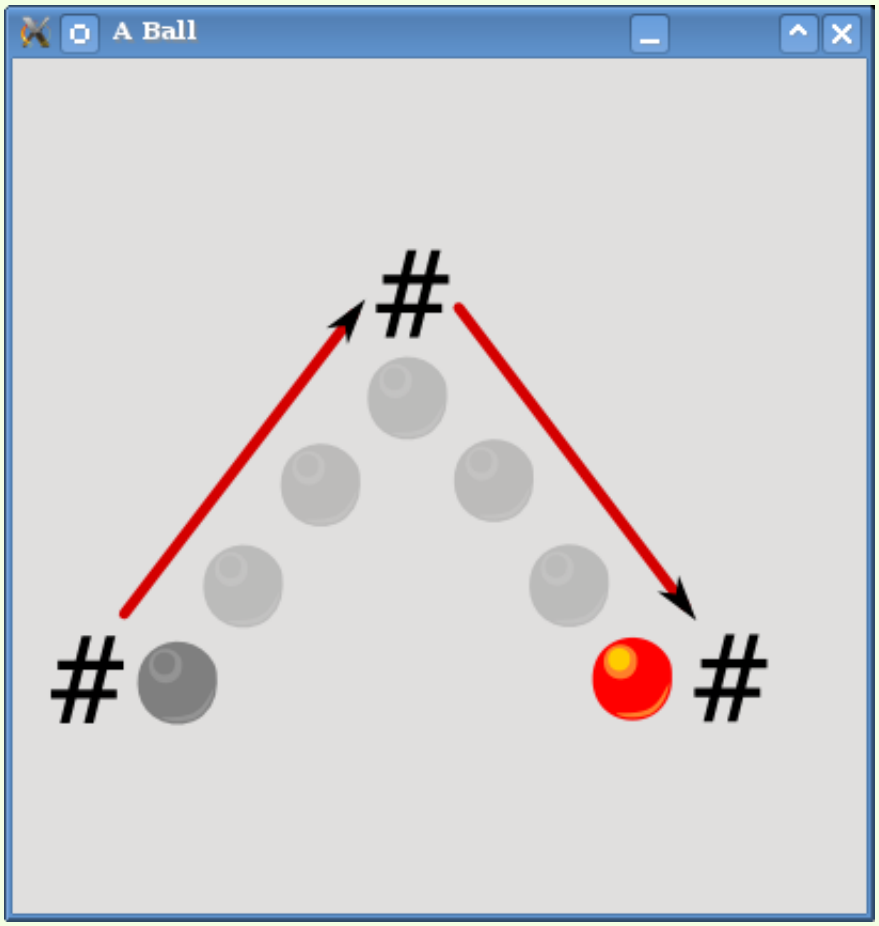

Fig. 3: Ball 1 animation

Line 7 & 8: The props are setup so that the animation starts from (x:-100,y:100), goes across to (0,0) [Props()] and then on to (100,100). After that is returns to where it started, at the same stops.

Line 10: This is a method that is called by the API when it's time to draw the Thing in question. The parameters are a *context*[2] and a frame number. The context is the most useful. The frame number can help you in other ways – like changing what is drawn according to the frame number.

Line 11: Here we employ the draw() method of the BagOfStuff. We pass the ctx variable along to it. This will actually draw the ball we originally drew in the SVG file.

And that's it. Quite easy I think.

7 Evolving the animation

Listing 5: ball2.py

```
1 from Things.ThingsApp import *
2 class JiggleBall(Thing):
3   def __init__(self ):
4     Thing.__init__(self)
```

[2] *Context: usually shortened to 'ctx'. This is a Cairo concept – it's a reference to a toolkit of tricks for drawing. Things will provide this context for all the methods that do drawing; hence you see it in the draw() method.*

```
5       self.keys ("#-----------------#-----------------#",
6           Props(sy=0.9),Props(rot=6.28),Props(sy=0.9)
7           )
8     def draw(self,ctx,fr):
9       bos["art:ball"].draw(ctx)

11  class Ball(Thing):
12    def __init__(self ):
13      Thing.__init__(self)
14      self.keys( "#"+"-"*40+"#"+"-"*40+"#"+"-"*40+"#"+"-"*40+"#",
15        Props(x=-100,y=100),Props(y=-100),Props(x=100,y=100),
16        Props(y=-100),Props(x=-100,y=100)
17        )
18      self.add( JiggleBall() )

20  bos = BagOfStuff()
21  bos.add ( "tutorial.svg", "art")

23  app = AllThings ( 400, 400, speed = 30, title = "A Ball" )
24  app.add( Ball() )

26  app.comeToLife ( )
```

Line 1: Here we import everything from ThingsApp.

Line 2: We define a JiggleBall thing. We'll get back to this soon.

Line 11: We define a Ball. This Ball is almost the same as that in ball1.py

Line 14: Here we define keys for the Ball. We are using the Python multiply string notation to specify many tweens: "-" * 40 means forty "-" signs. So we have 5 keyframes with 40 tween frames between them – this gives a nice smooth animation.

Line 15-16: We have also changed the top keyframe's y position to -100. The ball climbs higher on the canvas now.

Line 18: This is new. Here we make a JiggleBall and add it to our self (we are a Ball at this point). What has happened is that we have placed a Thing inside a Thing. In future, Ball() will also imply JiggleBall(). Note that we haven't provides a draw() method in Ball. We could, but it's not necessary; the draw() method of JiggleBall will come to the rescue.

Line 20-26: These are the same as you have seen. They fill a bag, start the app and then bring it to life.

Line 2-4: These are a standard beginning for a Thing.

Line 5: The JiggleBall is going to have it's own animation. These are its keys. There are 3 keyframes with some tweens.

Line 6: The corresponding 3 properties that are going to be tweened are defined here. We are going to start from a scale-y of 0.9 head towards scale-y of 1 *and* rotation 6.28 and then back to scale-y 0.9. This has the effect of squashing the ball vertically (just a little) and making it rotate once around. In my book, this gives the ball a 'jiggle'.

Line 8-9: Here we draw() the actual ball from the SVG file. Simple really.

8 Following a path

Up to now, the ball has moved like a robot – in straight lines – but we can do better than that! Let's use that curved path that is drawn in the SVG file and make the ball follow along. We will also introduce some more cool stuff to get the ball to 'squash' when it hits the ground.

Listing 6: ball3.py

```
1  from Things.ThingsApp import *

3  class JiggleBall(Thing):
4    def __init__(self ):
5      Thing.__init__(self)
6      self.keys  ( "#------------#------------#",
7          Props(sy=0.9),Props(rot=6.28),Props(sy=0.9)
8          )
9    def draw(self,ctx,fr):
10     bos["art:ball"].draw(ctx)

12 class Ball(Thing):
13   def __init__(self ):
14     Thing.__init__(self)
15     self.keys  ( "#--------#---------#",
16         Props(sy=0.5),Props(),Props(sy=0.5)
17         )
18     self.stops ( "         ^" )
19     self.labels( "         ^","squishdown")

21     self.add( JiggleBall() )

23   def squish(self):
24     self.goPlay("squishdown")

26 class Throw(FollowThing):
27   def __init__(self ):
28     FollowThing.__init__(self,"throw path")

30     self.keys  (90, self.path)

32     # Only use self.lifespan *after*
33     # a call to keys!
34     ls = self.lifespan - 10

36     self.funcs( " " * ls + "^",self.goSquish)

38     self.loops=False

40     self.ball=Ball()
41     self.add( self.ball )

43   def goSquish(self):
44     self.ball.squish()

46   def path(self,ctx):
47     bos["art:swoopy"].draw(ctx)
```

```
49 bos = BagOfStuff()
50 bos.add ( "tutorial.svg", "art")

52 app = AllThings ( 400, 400, speed = 90, title = "Throwing a Ball" )
53 app.add( Throw() )

55 app.comeToLife ( )
```

Line 3: The same JiggleBall.

Line 12: The Ball, but it's changed a bit.

Line 15 & 16: Sets keys and tweens to go from squished, to round, to squished again.

Line 18: Introduces a new idea: a stop frame. The command lets you point to the frames in the keys() method by using "^" characters. You can see that we have put a *stop* into the frame of the second keyframe.

Line 19: Another new idea: a label frame. We name that frame "squishdown" – also by placing a "^" where we need it.

Line 21: Here we make the JiggleBall and add it to the Ball.

Line 23 & 24: This squish() method will be called from some other place. It will cause me (the Ball) to go and play from the frame labelled "squishdown". This will squash the ball and return it.

Line 26: Here we define a "Throw" object that is a FollowThing. This is a special Thing that causes it's children to follow a given path.

Line 27-28: The usual init and then super init. Very important lines, even if they are boring

Line 30: The keys method for FollowThing is slightly different: supply a number of keys and a function ref.

number of keys would, in an ideal world, be the life-span of this animation. It is not. There is a bug in the FollowThing code and you will have to experiment with this. (You may get *more* keys than you expect. *Please* help me fix this code!)

Path function – here we pass self.path which you can find on line 46. It's a function that draws the path that will be followed.

Line 34: We use the self.lifespan property after a call to keys (because it's only set at that point) to work out how long the animation is. In this case we make a variable (ls) set to 10 less than the length.

Line 36: We are setting function frames on the FollowThing. Here we say: At 'ls' number of spaces followed by "^" (here), please run the self.goSquish function. So, at 10 (or 9) frames from the end of the animation it will call that function.

Line 38: Every Thing (and they are all Timeline objects) will loop around by default. Here we set the loop property to False to prevent that. It is done so you can see the animation pass just once. If you remove the line, the ball will start over on the left and keep looping.

Line 40 & 41: We make a Ball and add it to self (the FollowThing). The ball will now follow the path! Notice that we made the ball by way of a reference variable: self.ball. This is so we can refer to the ball from other places like the goSquish function.

Line 43 & 44: goSquish is defined. We tell self.ball to run squish(). Go look at line 23.

Line 46 & 47: We use our bag to pull the path id "swoopy" out and draw it. Simple.

Line 49-55: The usual stuff to get Things going. Note I have set the speed to 90; this will slow the animation down quite a lot so you can see the stages.

Wow, that was certainly more advanced. If you run ball3.py you will see the ball squish up, travel along an arc and squish down. There an odd bit at the end as it stays there, jiggling.

9 Bouncing back and forth

How would we make that ball bounce to and fro? Well, I am sure there are many ways to do it. We could control the ball's path with sin and cos based on the frame number in a draw() method. We could have two FollowThings and play one after the other. Perhaps there are more tricks. For the next example, I will use the same FollowThing but simply reflect it along the x axis – to make the ball travel back and forth.

Listing 7: ball4.py

```
1  from Things.ThingsApp import *

3  class JiggleBall(Thing):
4    def __init__(self ):
5      Thing.__init__(self)
6      self.keys  ( "#----------------#-----------------#",
7          Props(sy=0.9),Props(rot=6.28),Props(sy=0.9)
8          )
9    def draw(self,ctx,fr):
10     bos["art:ball"].draw(ctx)

12 class Ball(Thing):
13   def __init__(self ):
14     Thing.__init__(self)
15     self.keys  ( "#--------#---------#", Props(sy=0.5),Props(),Props(
           sy=0.5))
16     self.stops ( "         ^" )
17     self.labels( "         ^","squishdown")

19     self.add( JiggleBall() )
20   def squish(self):
21     self.goPlay("squishdown")

23 class Throw(FollowThing):
24   def __init__(self ):
25     FollowThing.__init__(self,"throw path")
26     self.keys  (90, self.path, startAtFirstNode=False)
27     ls = self.lifespan - 10
28     self.funcs( " " * ls + "^        ^", self.goSquish, self.goFlip)
29     self.ball=Ball()
30     self.add( self.ball )
31   def goSquish(self):
32     self.ball.squish()
33   def goFlip(self):
34     self.parentThing.flip()

36   def path(self,ctx):
37     bos["art:swoopy"].draw(ctx)
```

```
39 class TwoWay(Thing):
40   def __init__(self ):
41     Thing.__init__(self, "two way flipper")
42     self.keys  ( "##", Props(),Props(sx=-1))
43     self.stops ( "^" )
44     self.add( Throw() )
45     self.f=False
46   def flip(self):
47     self.f = not(self.f)
48     if self.f:
49       self.goStop(2)
50     else:
51       self.goStop(1)

53 bos = BagOfStuff()
54 bos.add ( "tutorial.svg", "art")

56 app = AllThings ( 400, 400, speed = 30, title = "A Ball" )
57 app.add(TwoWay())

59 app.comeToLife ( )
```

Note JiggleBall and Ball are pretty much the same.

Line 57: Let's start at the end (a good Python tip in general). Here we see that TwoWay is being instanced and added to app. Let's go look at that.

Line 39: TwoWay is defined, etc.

Line 42: It gets two keyframes. One is standard, the other has sx = -1. (sx is size-x). When on keyframe 2, everything will be flipped about and thus be going in the opposite direction.

Line 43: This sets a stop – on the first keyframe. We want the animation to stay here until we say otherwise.

Line 44: We add a Throw() instance. That's the FollowThing.

Line 45-51: We have an instance variable 'f' to act as a toggle. We have a function called flip. Within that, depending on the toggle, we command our timeline to go and stop on either frame 1 or 2.

Line 23: Here's the Throw() class. It's much the same as the last time, but we have an extra func on:

Line 28: Note how we have two "^" pointers – they represent the last section of the animation along the path. They set two functions to be run: goSquish and goFlip.

Line 34: goFlip() has the job of telling my parent (I am a Throw, and I was added in TwoWay, so TwoWay is my parentThing) to run the flip() function. Here is where the ball changes direction – right at the end of its journey along the path.

10 Final polish

You have seen how things can gradually become more complex; how they can contain other things and talk up and down the chains created. The whole idea is to think in terms of containers that move. Let's add some bling to the demo for the final installation of this tutorial.

Fig. 4: ball5 running

Listing 8: ball5.py

```
from Things.ThingsApp import *
from Things.BoxOfTricks import *

class JiggleBall(Thing):
  def __init__(self ):
    Thing.__init__(self)
    self.keys  ( "#----------------#--------------#",
        Props(sy=0.9),Props(rot=6.28),Props(sy=0.9)
        )
  def draw(self,ctx,fr):
    bos["art:ball"].draw(ctx)

class Ball(Thing):
  def __init__(self ):
    Thing.__init__(self)
    self.keys (  "#------------------#------------------#",
      Props(sy=0.5),Props(),Props(sy=0.5)
      )
    self.stops ( "                   ^" )
    self.labels( "                   ^","squishdown")

    self.add( JiggleBall() )
  def squish(self):
    self.goPlay("squishdown")
```

```
26 class Shadow(Thing):
27   def __init__(self ):
28     Thing.__init__(self)
29     self.keys  ( ".#------------------------#-----------#",
30         Props(a=0,sz=2),Props(),Props(a=0,sz=2)
31         )
32     self.stops ( "^                           ^" )
33     self.funcs ( "                                      ^",
34       self.flip
35       )
36     self.labels( " ^                          ^",
37       "darken","lighten"
38       )
39     self.fd = fuzzydot()
40   def draw(self,ctx,fr):
41     self.fd.draw(ctx)
42   def flip(self):
43     self.parentThing.flip()

45 class Throw(FollowThing):
46   def __init__(self ):
47     FollowThing.__init__(self,"throw path")
48     self.keys  (90, self.path, startAtFirstNode=False)
49     ls = self.lifespan - 36
50     self.funcs( "^"+" "*ls+"^                    ^           ^",
51       (self.tellshadow,"up"),(self.tellshadow,"down"),
52       self.tellsquish,self.tellflip
53       )
54     self.ball=Ball()
55     self.add( self.ball )

57   def tellshadow(self, dir):
58     if dir=="up":
59       SHADOW.shadow.goPlay("lighten")
60     else:
61       SHADOW.shadow.goPlay("darken")
62   def tellsquish(self):
63     self.ball.squish()
64   def tellflip(self):
65     self.parentThing.flip()

67   def path(self,ctx):
68     bos["art:swoopy"].draw(ctx)

70 class TwoWay(Thing):
71   def __init__(self ):
72     Thing.__init__(self)
73     self.keys  ( "##", Props(),Props(sx=-1))
74     self.stops ( "^" )
75     self.add( Throw() )
76     self.f=False
77   def flip(self):
78     self.f = not(self.f)
```

```
79      if self.f:
80        self.goStop(2)
81      else:
82        self.goStop(1)

84  class TwoShadow(Thing):
85    def __init__(self ):
86      Thing.__init__(self)
87      self.keys  ( "##", Props(),Props(sx=-1))
88      self.stops ( "^" )
89      self.shadow = Shadow()
90      self.add( self.shadow, globalProps=Props(y=102,x=100,sy=0.2) )
91      self.f = False
92    def flip(self):
93      self.f = not(self.f)
94      if self.f:
95        self.goStop(2)
96      else:
97        self.goStop(1)

99  class Background(DrawThing):
100   c=hexfloat("#5d6c53")
101   b=hexfloat("#a7ac93")
102   def draw(self,ctx,fr):
103     ctx.set_source_rgb(*Background.b)
104     ctx.paint()
105     ctx.set_source_rgb(*Background.c)
106     ctx.rectangle(-200,70,400,330)
107     ctx.fill()


110 bos = BagOfStuff()
111 bos.add ( "tutorial.svg", "art")

113 app = AllThings ( 400, 400, speed = 20, title = "A Ball" )
114 app.add( Background() )
115 SHADOW=TwoShadow()
116 app.add( SHADOW )
117 app.add(TwoWay())

119 app.panZoom(True)
120 app.comeToLife ( )
```

Note JiggleBall and Ball are pretty much the same.

Line 26: We start a Shadow (which is a Thing).

Line 29: We set the keys so that:

1. There is a blank frame (“.”) first.
2. A tween from alpha 0, size 2 down to normal, then back.

Line 32: Stops are set on the blank frame and on the middle keyframe.

Line 33: A func is set on the last keyframe; it will run self.flip(). From this you can infer that the shadow will also be doing a size-x minus flip trick – it will happen in synch with the ball's flip.

Line 36: Here we set a couple of labels so that we can control the flow later.

Line 39 & 41: We define a fuzzydot into the ref self.fd. This fuzzydot comes from the *BoxOfTricks* module. It's a simple circle drawn with a gradient that alpha's to 0 on the edges. Line 41 uses the self.fd ref to call the draw() method of the fuzzydot; as usual, we pass the context along.

Line 43: This is where we ask our parentThing to run the flip() routine. [3]

Line 45: Our Throw Thing has picked up a few tricks in:

Line 50: You can see there are more functions at play. Note the format to use when you want to pass an argument to one: (myfunc, "myarg", "otherargs"). The brackets are essential when you have arguments. The various funcs in this line fire-off actions in the shadow and the ball.

Line 59 & 61: Depending on the argument (dir, for direction) the global SHADOW reference (defined on line 115) is used to fetch the shadow instance and thence to tell it to go and play from a certain label. "lighten" is the shadow-out to alpha 0, "darken" is the shadow-in to alpha 1.

Line 70: This is the TwoWay Thing; same as before. It holds (contains) the Throw object, so when it get flipped, so does whatever is in it – thus the Throw object also flips.

Line 84: This does the same thing for the Shadow object.

Line 89 & 90: We make a local ref. and then add it. Note how we use the *globalProps* argument here: this is set to a Props() object that places the entire Shadow animation at a certain position.

Line 99-107: We use another kind of Thing here – a DrawThing. This is a very simple Thing that does not require an __init__. All you need is a draw() method and you are off. This one draws two rectangles – one for the ground and one for the sky.
Note how *class-level* variables are being used. c and b are fetched via the class: Background.b
Another thing to notice here is the argument to the set_source_rgb() command. We use * to turn Background.b into a list, which the command expects. Have a look at the *hexfloat* function in BoxOfTricks.py, you will see it returns a tuple of colour values in a style that python-cairo can use.

Line 114: We add the Background to the app.

Line 115: Here we make the global instance SHADOW – it's a TwoShadow object. I know, I know... bad naming.

Line 116: We add the SHADOW

Line 117: We instance and add a TwoWay.

Notice how we added many Things to the app. This is quite normal.

Line 119: panZoom is set to True. Now you can use the mouse-wheel to zoom in and out. I wanted this to allow pan-zooming (like Inkscape does) but I can't get the maths right (hint: I need help here too.)

There you go. The ball now bounces, squishes and is followed by a shadow that grows, shrinks and fades. It's all happening against a background and in only a few lines of Python code! Nice.

Written by Donn Ingle
Oudebosch
May 2009

[3] You may wonder why I put self.flip in the funcs and not simply self.parentThing.blah – well, it's because in the __init__ routine, which is where you do stuff like keys() and funcs(), there *is no* parentThing reference yet; it's None. It's best to be indirect and jump to a local function – thence to the parentThing (or wherever.)

Index

11 Licence

GNU Free Documentation License

Version 1.3, 3 November 2008

<http://fsf.org/>

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "**Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document

may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "**publisher**" means any person or entity that distributes copies of the Document to the public.

A section "**Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "**Acknowledgements**", "**Dedications**", "**Endorsements**", or "**History**".) To "**Preserve the Title**" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the

publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with … Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.