Python: A "Toy" Language

David Beazley (@dabeaz)
Is this about toy problems?

```python
def fibonacci(n):
    if n <= 2:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```
No, let's talk about kids...

(oh no...)
An amazing science project

Advanced Biology
An amazing science project

Advanced Biology
... and Chemistry
An amazing science project

Advanced Biology

... and Chemistry

Machine Learning
An amazing science project

Advanced Biology
... and Chemistry

Machine Learning

A natural hacking instinct
You Get to Build Things

Diabolical block towers
You Get to Build Things

A doghouse (for toy dog)
You Get to Build Things

Or maybe a birdhouse...
(for real birds)
Hacker Pro-Tip

Kids are the ultimate excuse for buying "tools"

- Drill press
- Telescope
- Magnifying glass
- Catapult
- Oscilloscope
- Soldering Iron
- Laser
- Welding torch
Hacker Pro-Tip

Kids are the ultimate excuse for buying "tools"

- Drill press
- Telescope
- Magnifying glass
- Catapult
- Oscilloscope
- Soldering Iron
- Laser
- Welding torch

You know, for kids. The big one. Yourself.
Question:

how do you get your kid to want to be a diabolical pirate scientist BDFL?
Question:

how do you get your kid to want to be a diabolical pirate scientist BDFL? ... in space.
A thought:
maybe I could make toys
A thought: maybe I could make toys with Python?
A thought:
maybe I could make toys
with Python?

Python + Kids + Building Stuff == Fun
A thought:
maybe I could make toys with Python?

Python + Kids + Building Stuff == Fun
(also good way to avoid working on book)
Yes, this book
Seriously though...

David Reid @dreid

Retweeted 1151 times
Expand  Reply  Retweet  Favorited  More

Writing Python programs to make toys... well, yeah. Duh!
Making Things

There's a bit of "maker" movement going on
- 3D Printing
- Hacker spaces
- Arduino, Raspberry Pi
- DIY

It's been on my mind
- "Oh, that might be cool"

However, I've never been that actively involved
Issue

Drawing and clicking "print" is a big "meh."

3D printing seems just a bit too magical

If machine breaks, could I figure out how to fix it?

More enjoyment from figuring out how to do something than actually doing it
DIY CNC Milling

So, shortly after last PyCon, this caught my eye...

- ShapeOko Project
- 1-man project (Edward Ford)
- Chicago area
- Super cheap $
CNC Milling

In a nutshell: Computer controlled whirling knives

- Scary speed
- Lot's of noise
- Flying chips
- Thrilling Danger!
CNC Milling

In a nutshell: Computer controlled whirling knives

- Scary speed
- Lot's of noise
- Flying chips
- Thrilling Danger!

It sounded perfect! You know, for kids...
DIY != Industrial

Related searches: cnc milling machine, haas cnc mill, cnc router, cnc lathe
Kit Arrival
Just to be clear...

... I have never done anything with CNC ever.

(I'm a software geek, what could possibly go wrong?)
Wheel Assembly:
- Insert a bearing into one side. These are force fit tolerances so you might have to push hard.
- Press the bearing until it's seated against the inside lip of the v-wheel.
- Slide an M5 bolt through the bearing (outside in) - you'll use this to align the precision washer and the other bearing.
- Slide the precision washer over the bolt and let it slide all the way down until it hits the bearing you just inserted.
- Slide the other bearing over the bolt and let it slide down until it hits the rim of the v-wheel.
- Press the second bearing into the v-wheel just as you did the first.
- When the assembly is complete, you can remove the M5 bolt.

Note: do not forget the precision washer!
Smooth Idler Assembly:
1. Insert a bearing into the open side. These are force fit tolerances, so you might have to push hard.
2. Press the bearing until it's seated against the inside lip of the idler drum.
3. Slide an M6 bolt through the bearing (outside in) - you'll use this to align the precision washer and the other bearing.
4. Slide the precision washer over the bolt and let it slide all the way down until it is touching the bearing you just inserted.
5. Slide the other bearing over the bolt and let it slide down until it hits the rim of the drum.
6. Press the second bearing into the drum just as you did the first until it is tight against the precision washer.
7. When the assembly is complete, you can remove the M6 bolt.

Note: do not forget the precision washer!
Assembly

Assembly Instructions:

1. Take note of the top hole orientation.
2. When assembling the plate (view shown), the cross mount holes (circled) need to be in the top right of the plate. If you assemble these backwards, you'll have to disassemble the plate and flip the wheels around. It's not the end of the world, but it's tedious and frustrating if you make that mistake.
Y-Axis Motor Mount Plate

This is the other side of the gantry. Pay attention to the hole orientation at the top of the plate. Take note of a couple of things:
- These holes need to be on the left (opposite of the previous sub-assembly). The image shows the correct orientation.
- Each bolt that goes through the plate gets a washer on BOTH sides.
- The smooth idlers get a total of 4 washers per bolt.
  - One on the backside of the plate
  - One on the front side of the plate
  - One on the end of the 1/4" spacer
  - One between the idler and the nut that keeps it all together.
Assembly

X-axis Motor Mount Plate

This is more of a pre-assembly than a fully contained sub-assembly. You're going to assemble this as shown in the drawing. Make note of a couple of things:

- The hole orientation at the top of the plate doesn't matter. I have mine to the left, but they can be to the right (as shown).
- The bottom set of bolts are the long M5 bolts (55mm)
- When you tighten the motor into place, do so in an X pattern. You want the face of the motor to be equally pressing against the plate.
- Don't tighten down the v-wheel bolts. Just loosely thread on the nuts to keep everything together.
- When you are done with the assembly, set it aside, you'll be bolting this to the next assembly bulk.
Assembly

Z-Axis Mount Plate

This is the second part of your pre-assembly.

Assemble this as shown in the drawing. Make note of a couple of things:
- Pay attention to the number of washers on each bolt; it's critical.
- Pay attention to the orientation of the plate. The larger holes in the center (denoted with an arrow) need to be at the bottom (as shown).
- Tighten the non-eccentric washers.
- Just snug the dinm nut, we'll want to align it with the threaded rod later, then tighten it down.

<table>
<thead>
<tr>
<th>Part</th>
<th>Description</th>
<th>Quantity</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>M4</td>
<td>Hex Head Screw</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>M3</td>
<td>Hex Head Screw</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>M6</td>
<td>Nut</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

M3-Phillips Pan Head Screws

Z-Axis Mount Plate Assembly

SHAPEOKO
Notes:
- Clean ends of extrusion of any debris.
- Snug bolts on Gantry - Y Idle Mount Plate.
- Slide Gantry - XZ Plate onto Makerslide Extrusion
- Snug Bolts on Gantry - Y Motor Mount Plate
- Work your way from each side in a X pattern.
- Align top of extrusion with top of plate.
- Keep square
<table>
<thead>
<tr>
<th>ITEM</th>
<th>QTY</th>
<th>PART NUMBER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>SM-GA01</td>
<td>Shapeoko Gantry Assembly</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>SM-FS01</td>
<td>Shapeoko - Frame Assembly</td>
</tr>
</tbody>
</table>

**Insertion Nut**

**Belt Anchor**

**M5 x 10mm SHCS**

**M5 Flat Washer**

**Shapeoko v1.1**
Assembly
Assembly
Assembly

Curse Words Uttered:

CNC Mill - 0
Stroller - 137

(Stroller cost more)
Hello World

ShapeOko
Practical First Job

bash-3.2s python3
Python 3.2.2 (default, Mar 18 2012, 20:35:38)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more in formation.
>>> import gil
>>> gil.remove_gil()
Press any key to continue
Lincoln Logs
Wood Train & Fire Truck
Ladder: For Fire Truck
Detachable Wing
Flying Wooden Train
Flying Wooden Train with Hidden Pirate Treasure Chest
Flying Wooden Train with Hidden Pirate Treasure Chest
"Kid, if this doesn't stop, I'm going to start making you write a requirements doc."
Flying Wooden Train with Hidden Pirate Treasure Chest

"Kid, if this doesn't stop, I'm going to start making you write a requirements doc--in Docbook XML"
Or maybe...
... wait for it
A miniature bike shed!
Random Consequence...

"Daddy, can you make something?"
Random Consequence...

"Daddy, can you make something?"

Actual meaning: Install a new iPad app.

(I digress)
Electronics/Tech

Stepper Motors (3)

Arduino w/ Grblshield

Laptop (USB)
Software

It's just serial ports... use pyserial

```python
import serial
ser = serial.Serial(
    '/dev/tty.usbmodem641', 9600)

def command(cmd):
    ser.send(cmd.encode('ascii')+b'\n')
    resp = ser.readline()
    if resp != b'Ok\n':
        raise RuntimeError(resp)
```

Simple command/response protocol
GCode

Movement controlled by simple commands

G1 Z10
G1 X0 Y0
G1 Z-2
G1 X50 Y10
G1 X20 Y40
G1 X0 Y0
G1 Z0

It's a lot like plotting/turtle graphics
Whirling Knives

You're in the physical world

Plotting with 25000 RPM end mill
Physics

- No instantaneous motion
- Hardware limitations
- Material properties

(0,0) → (20,40) → (50,10) → (0,0)
Geometry Restrictions

OK

UNDERCUTS

FEATURES ON BACK SIDE

(cut can't exactly into vertices)
It's Inexact

There are real hardware "errors"

- Wobble in rotary tools (runout)
- Misalignment/centering issues
Materials Science

Wood

Machinable Wax

HPDE Plastic

Acrylic
A Science Experiment?

Feed rates, rotational speed, cut depth, etc.
CAM Software

Maybe there's a reason why there's a whole industry of expensive "Computer Aided Manufacturing" Software.

Of course, there's a free Python one too...

(I have not used it)
CNC is "simple" enough to write scripts

```
part = [  
    (0, 0),  
    (0, 44.175),  
    (23.175, 44.175),  
    (23.175, 37.175),  
    (60, 37.175),  
    (60, 44.175),  
    (83.175, 44.175),  
    (83.175, 0),  
    (0, 0)  
]  
```
DIY CAM

Direct streaming of GCode

```python
command('F1000')
for z in range(1,10):
    command('G1 Z-%s' % z)
for x, y in part:
    command('G1 X%s Y%s' % (x,y))
command('G1 Z0')
```

You'll see those whirling knives moving around!
Programming Errors

They take physical form!
Real Dangers

Rotating Knives!

- Drill through table
- Jamming
- Shattered End Mill
- Harmonic vibration
- Destruction

A simple sign error can get interesting
Real Dangers

Rotating Knives!

- Drill through table
- Jamming
- Shattered End Mill
- Harmonic vibration
- Destruction

A simple sign error can get interesting

TDD?
Is there anything it can't do?

Why not CAM?

**Solution?**

**IPython Notebook?**
Project: Marble Track

Plot mathematical functions into curvy marble track

\[ f(x) = 30 \times \sin(0.15 \times x) \]
Project: Marble Track

Track Cross-section

Top View

Track Path

Normal vector

f'(x)
Demo : IPython Notebook

Marble Track Planner

Change the parameters below to define basic track properties. Units in mm.

In [2]:
WIDTH = 29.4125 # Overall width of the track
RAIL_TO_RAIL = 9.4125 # Width inner-to-inner rail
RAIL_WIDTH = 3.225 # Width of rail itself

Parameters for the length of track and resolution. Units in mm.

In [3]:
LENGTH = 600 # Workpiece length
DX = 1 # X increment in calculations

Tool Parameters. Units in mm.

In [4]:
CUT_DIAMETER = 3.175 # Diameter of the end-mill
CUT_RADIUS = CUT_DIAMETER / 2
Demo: IPython Notebook

IP[y]: Notebook

Track Path Function

Define a function for the track path. \(x\) has units of mm.

```python
In [5]: def f(x):
    if x < 300:
        return 0.2*x*sin(0.03*x)
    else:
        return 0.2*(600-x)*sin(0.03*x)
```

Compute set of \(x\)-coordinates and plot the basic path.

```python
In [6]: xpts = arange(0,LENGTH, DX)
ypts = [f(x) for x in xpts]
```

```python
In [52]: plot(xpts, ypts)
```

Out[52]: [matplotlib.lines.Line2D at 0x36dcb0]
Demo: IPython Notebook

### Cutting Region

Compute and plot the cutting region for the piece.

```python
In [53]: def derivative(f, x, dx=0.0001):
    
    ...,
    
    Centered derivative approximation
    ...
    
    return (f(x+dx) - f(x-dx))/(2*dx)
```

```python
In [54]: import math
def normal(x, d):
    
    Given an x coordinate and a normal direction d:
    ...
    y = f(x)
dx = derivative(f, x)
if dx == 0:
    nx = x
    ny = y + d
else:
    m = -1.0/dx
    theta = math.atan(m)
cx = d*cos(theta)
cy = d*sin(theta)
x = x + cx
ny = y + cy
return nx, ny
```

```python
In [57]: p = plot(xpts, ypts, upper_xpts, upper_ypts, lower_xpts, lower_ypts)
```

![Plot of cutting region](image)
Demo: IPython Notebook

### Tool-path computation

```
In [58]: def make_path(xpts, start_dist, end_dist, delta):
   ...:     path = []
   ...:     d = start_dist
   ...:     for n in range(int((end_dist - start_dist) / delta + 1)):
   ...:         if d > end_dist:
   ...:             d = end_dist
   ...:         pts = [normal(x, d) for x in xpts]
   ...:         if n % 2 == 0:
   ...:             pts.reverse()
   ...:         path.extend(pts)
   ...:         d += delta
   ...:     return path

Define tool paths for regions outside the rails

In [59]: top_path = make_path(xpts, (RAIL_TO_RAIL + RAIL_WIDTH / 2), (WIDTH / 2), CUT_RADIUS)

In [64]: plot(*all_paths)
```

"""Out[64]: [<matplotlib.lines.Line2D at 0x2bb52f0>, <matplotlib.lines.Line2D at 0x2bb53f0>, <matplotlib.lines.Line2D at 0x2bb5730>]
"""

In [64]: plot(*all_paths)
Milling
Milling
Milling
Milling
Milling
Rough Cut
Challenges

• Exceptions are common
• There's a "panic" button
• Often stop/resume
• Still working on software for it...
User Testing
Integration Testing
Variation

Ripple Track
Future Direction?

Resin Casting

(Photos: Guerilla Guide to CNC)
Future Direction?

Printed Circuit Boards?
Is There a Point?

- CNC is a long-solved problem
- CAD/CAM is long-solved
- Why bother?
Is There a Point?

- CNC is a long-solved problem
- CAD/CAM is long-solved
- Why bother?

Answer: It's every bit as fun as my first computer...
Is There a Point?

- CNC is a long-solved problem
- CAD/CAM is long-solved
- Why bother?

Answer: It's every bit as fun as my first computer...

(and in 1978, programming was a solved problem).
Is There a Point?

- CNC is a long-solved problem
- CAD/CAM is long-solved
- Why bother?

Answer: It's every bit as fun as my first computer...

(and in 1978, programming was a solved problem).

And nothing shouts out “fun” like a JCL script.
It's Supposed To Be Fun

Why did you start using Python?

... it's okay to admit it. It was probably because it was fun.

(if it's not fun, you're not doing it right)
Bigger Point?

"Every child is an artist. The problem is how to remain an artist when we grow up."

- Pablo Picasso

s/artist/hacker/
Final Comments

- You can do this!
- ShapeOko (http://www.shapeoko.com)
- Guerrilla Guide to CNC (http://lcamtuf.coredump.cx/gcnc/)
- Thanks!
- Follow at @dabeaz