Snakes on a Droid

A brief introduction to building
Android apps in Python with Kivy

A brief introduction to building Android apps in Python with Kivy

#ValleyDevFest

bit.ly/kivy-devfest

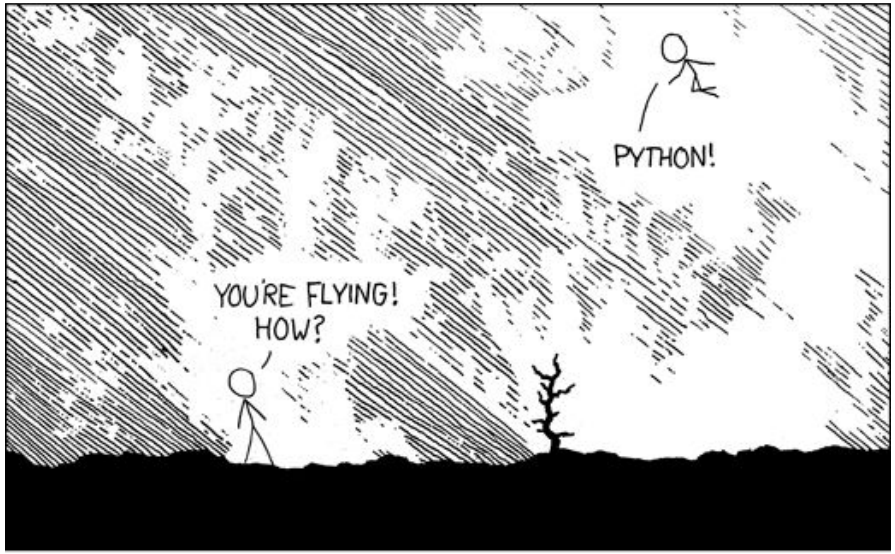
github.com/dmpayton/snakes-on-a-droid

Hi, I'm Derek.


I ❤ Python

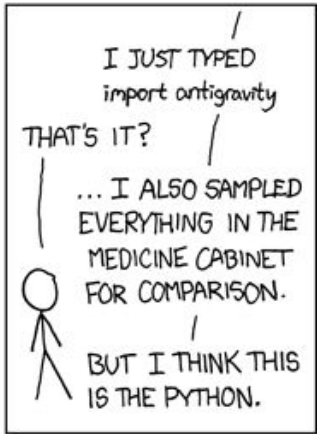

Wait, no...


I ❤ Python
conf.py
router.py
screens.py
store.py
utils.py
kv
main.py
bin
data
libs
garden
garden.router
garden.simpletable
utils
.gitignore
README.rst
activated-chords.json
buildozer.spec
requirements.txt
clearcolor=(1,1,1,1),
duration=.1
)
return screens.MenuScreen()
@route('/chords')
def chord_list(self):
return screens.ChordListScreen()
@route('/chords/<path:chord>')
def chord_detail(self, chord):
return screens.ChordDetailScreen(chord=chord.strip('/'))
@route('/practice')
def practice(self):
return screens.PracticeScreen()
@route('/scales')


PYTHON!
YOU'RE FLYING! HOW?


I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"


I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?

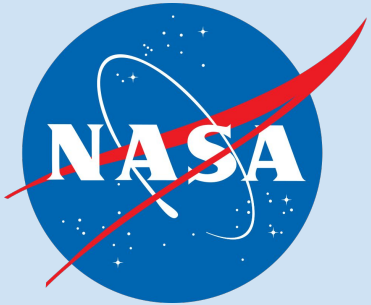

I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Batteries Included

Extensive Ecosystem

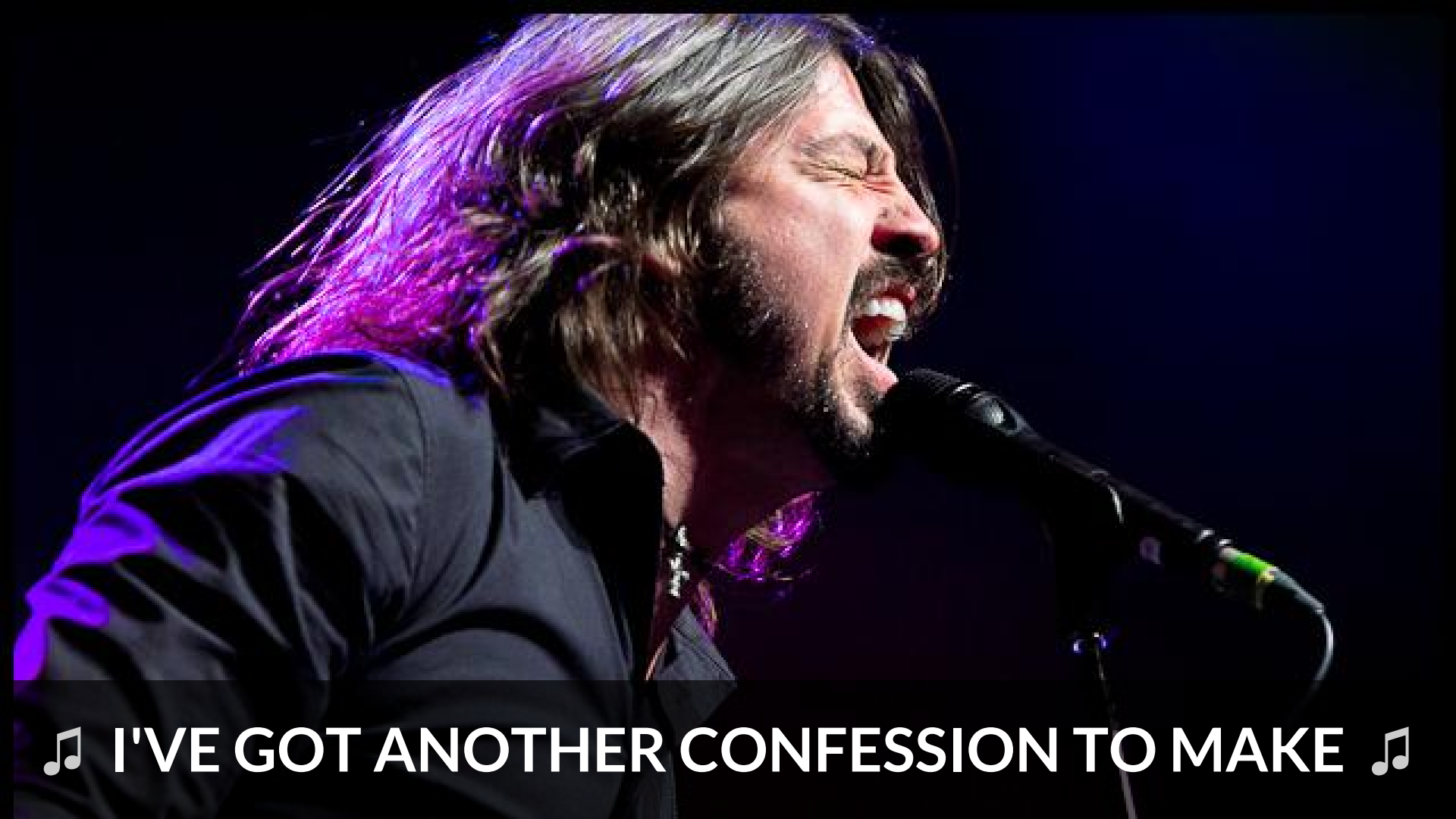
Amazing Community

Dependable



Google

Instagram


♫ I'VE GOT ANOTHER CONFESSION TO MAKE ♫


GIVES TALK ON ANDROID
NOT AN
ANDROID DEVELOPER

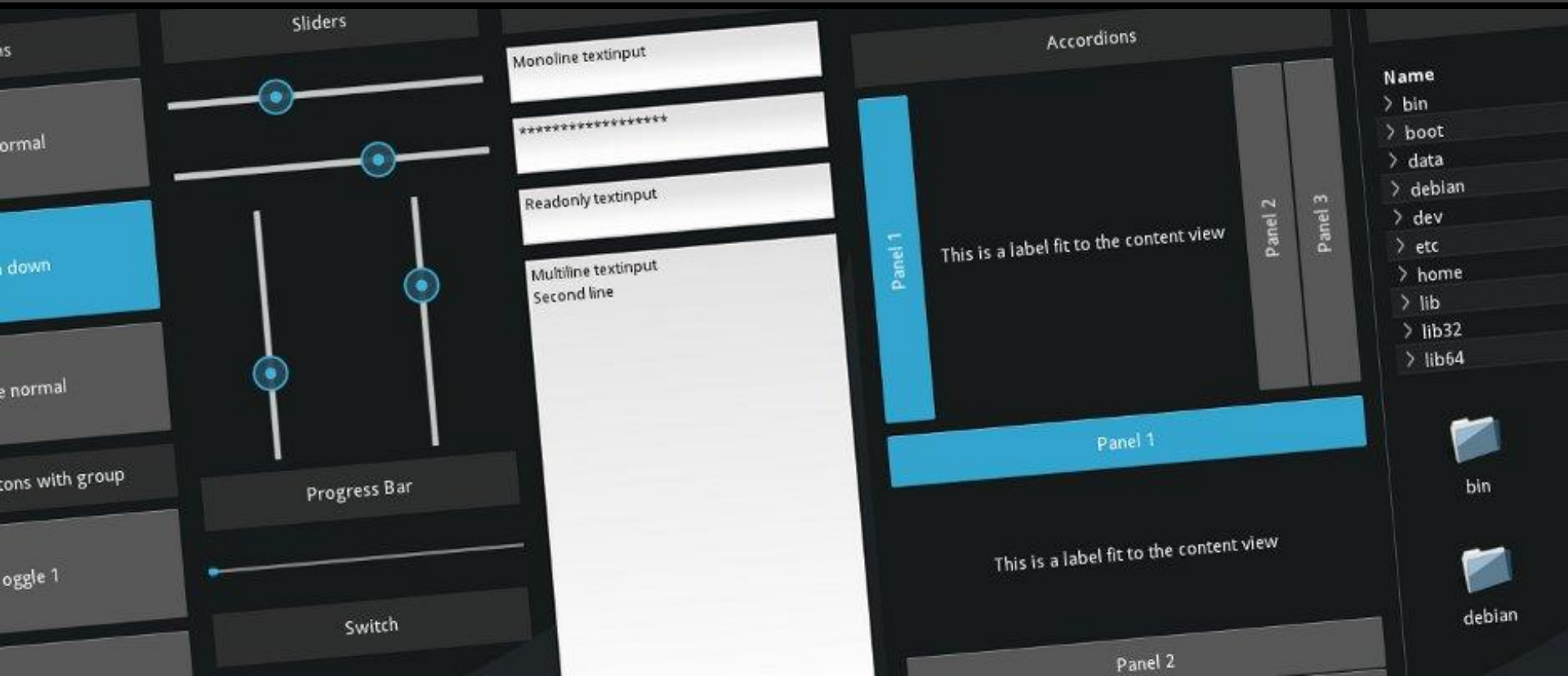

3:37 PM
kUSC.org
CLASSICAL | fm 91.5
Classical Music with Kimberlea Daggy
Antonin Dvorak: Symphony #9 "New World" in e Op 95

meh.

I **want** to build software that works on my phone…

…but in general, I prefer to code in **Python**

kivy


Sliders
Monoline textinput
Accordions
Name
bin
boot
data
debian
dev
etc
home
lib
lib32
lib64
Readonly textinput
Multiline textinput
Second line
Panel 1
This is a label fit to the content view
Panel 2
Panel 3
Progress Bar
Switch
Panel 1
This is a label fit to the content view
Panel 2
bin
debian

Kivy is...

Python library for creating multi-touch applications

Kivy is...

Open Source

MIT License

Kivy is...

Cross-platform

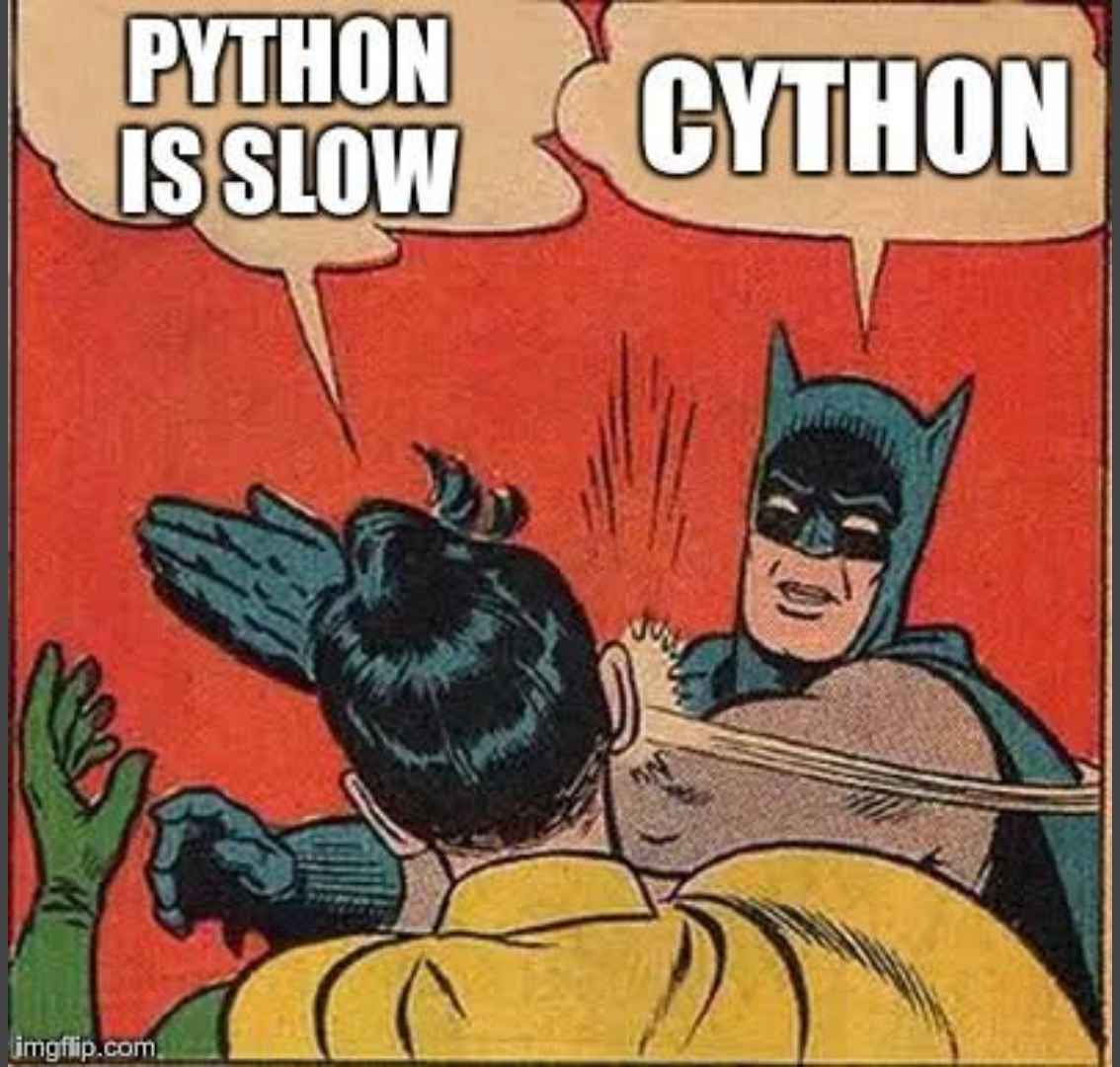
Kivy is...

Fast.

Kivy is...

 Production ready 

You said Kivy is fast, but...


PYTHON IS SLOW
CYTHON
imgflip.com

cython

The missing link between the **simplicity of Python** and the **speed of C**


HELLO
WORLD

Installing Kivy

```
$ pip install Cython
$ pip install kivy
```

01. hello, world

main.py

```
from kivy.app import App
from kivy.uix.label import Label

class DemoApp(App):
    def build(self):
        return Label(text='hello, world', font_size=60)

if __name__ == '__main__':
    TestApp().run()
```

01. hello, world

main.py

```
from kivy.app import App
from kivy.uix.label import Label

class DemoApp(App):
    def build(self):
        return Label(text='hello, world', font_size=60)

if __name__ == '__main__':
    TestApp().run()
```

01. hello, world

main.py

```
from kivy.app import App
from kivy.uix.label import Label

class DemoApp(App):
    def build(self):
        return Label(text='hello, world', font_size=60)

if __name__ == '__main__':
    TestApp().run()
```

01. hello, world

main.py

```
from kivy.app import App
from kivy.uix.label import Label

class DemoApp(App):
    def build(self):
        return Label(text='hello, world', font_size=60)

if __name__ == '__main__':
    TestApp().run()
```

02. multi-touch

main.py

```
from kivy.app import App
from kivy.uix.label import Label
from kivy.uix.scatterlayout import ScatterLayout

class DemoApp(App):
    def build(self):
        scatter = ScatterLayout()
        label = Label(text='hello, world', font_size=60)
        scatter.add_widget(label)
        return scatter

if __name__ == '__main__':
    TestApp().run()
```

02. multi-touch

main.py

```
from kivy.app import App
from kivy.uix.label import Label
from kivy.uix.scatterlayout import ScatterLayout

class DemoApp(App):
    def build(self):
        scatter = ScatterLayout()
        label = Label(text='hello, world', font_size=60)
        scatter.add_widget(label)
        return scatter

if __name__ == '__main__':
    TestApp().run()
```

02. multi-touch

main.py

```
from kivy.app import App
from kivy.uix.label import Label
from kivy.uix.scatterlayout import ScatterLayout

class DemoApp(App):
    def build(self):
        scatter = ScatterLayout()
        label = Label(text='hello, world', font_size=60)
        scatter.add_widget(label)
        return scatter

if __name__ == '__main__':
    TestApp().run()
```

03. events

main.py

```
from kivy.app import App
from kivy.uix.button import Button
from kivy.uix.floatlayout import FloatLayout
from kivy.uix.popup import Popup

...
```

03. events

main.py

```
...

class DemoApp(App):
    def build(self):
        layout = FloatLayout()
        open_button = Button(
            text='click me!',
            size_hint=(.5, .5),
            pos_hint={'center_x': .5, 'center_y': .5}
        )
        layout.add_widget(open_button)

        ...
```

03. events

main.py

```
...

popup = Popup(
    title='hello, world',
    auto_dismiss=False,
    size_hint=(.3, .3)
)
close_button = Button(text='close me!')
popup.add_widget(close_button)

...
```

03. events

main.py

```
...

open_button.bind(on_release=popup.open)
close_button.bind(on_release=popup.dismiss)

return layout

...
```

Kv Design Language

04. Kv lang

main.py | demo.kv

```
from kivy.app import App

class DemoApp(App):
    pass

if __name__ == '__main__':
    DemoApp().run()
```

04. Kv lang

main.py | demo.kv

```
FloatLayout:
    id: layout

    Button:
        id: open_button

    Popup:
        id: popup

        Button:
            id: close_button
```

04. Kv lang

main.py | demo.kv

```
FloatLayout:
    id: layout

    Button:
        id: open_button
        text: 'click me!'
        size_hint: (.5, .5)
        pos_hint: {'center_x': .5, 'center_y': .5}
        on_release: root.ids['popup'].open()

    Popup:
        id: popup
        ...

        Button:
            id: close_button
            ...
```

04. Kv lang

main.py | demo.kv

```
FloatLayout:
    ...

    Button:
        ...

    Popup:
        id: popup
        title: 'hello, world'
        auto_dismiss: True
        size_hint: (.3, .3)
        on_parent: if self.parent == layout: layout.remove_widget(self)

        Button:
            id: close_button
            text: 'close me!'
            on_release: root.ids['popup'].dismiss()
```


DEMO

We haven't built an **Android** app...

We've built a **Kivy** app, and we need to **package** it for Android.

python-for-android

Packages Python apps
for Android

python-for-android

Don't use it (directly).

buildozer

Tool for creating application packages

buildozer

Build for Android or iOS using a common spec file.

buildozer

```
$ pip install buildozer
```

buildozer

```
$ buildozer init
```

buildozer.spec

```
[app]

# (str) Title of your application
title = My Application

# (str) Package name
package.name = myapp

# (str) Package domain (needed for android/ios packaging)
package.domain = org.test

# (str) Source code where the main.py live
source.dir = .

# (list) Source files to include (let empty to include all the files)
source.include_exts = py,png,jpg,kv,atlas

...
```

buildozer.spec

```
...

#
# Android specific
#

# (bool) Indicate if the application should be fullscreen or not
fullscreen = 1

# (list) Permissions
android.permissions = INTERNET

# (int) Android API to use
android.api = 19

# (int) Minimum API required
android.minapi = 9

...
```

buildozer

```
$ buildozer android_new debug
```

buildozer

```
$ buildozer android_new debug /
  deploy run
```

How do you do actual **Androidy** things?

pyjnius

Python module to access Java classes through the JNI

pyjnius

```
from time import sleep
from jnius import autoclass

Hardware = autoclass('org.test.android.Hardware')
Hardware.accelerometerEnable(True)

for x in xrange(20):
    print(Hardware.accelerometerReading())
    sleep(.1)
```

But, what about the **dream**, man?

plyer

Platform-independent API for common hardware features

plyer

pyjnius on Android

pyobjus on iOS

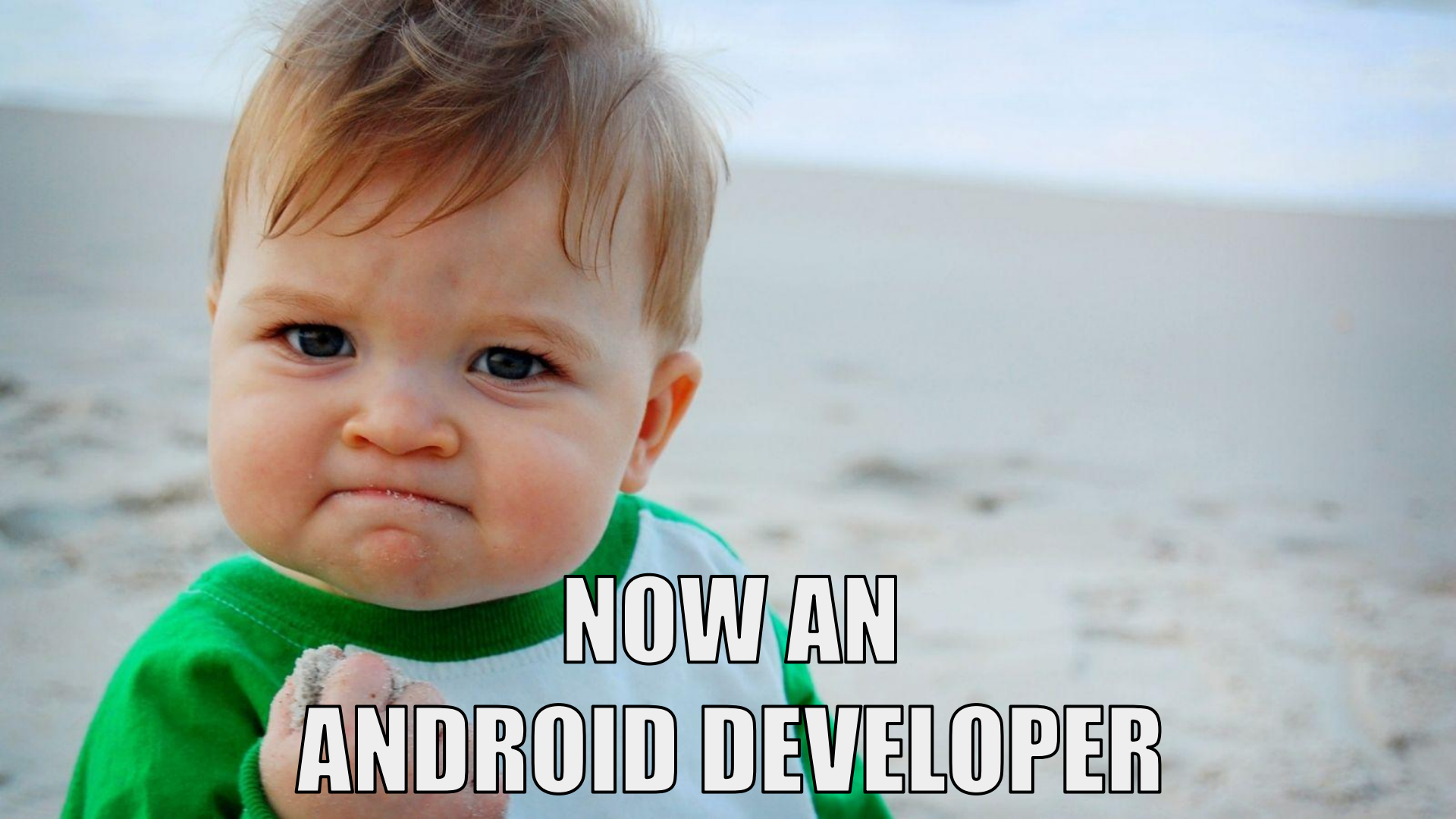
plyer

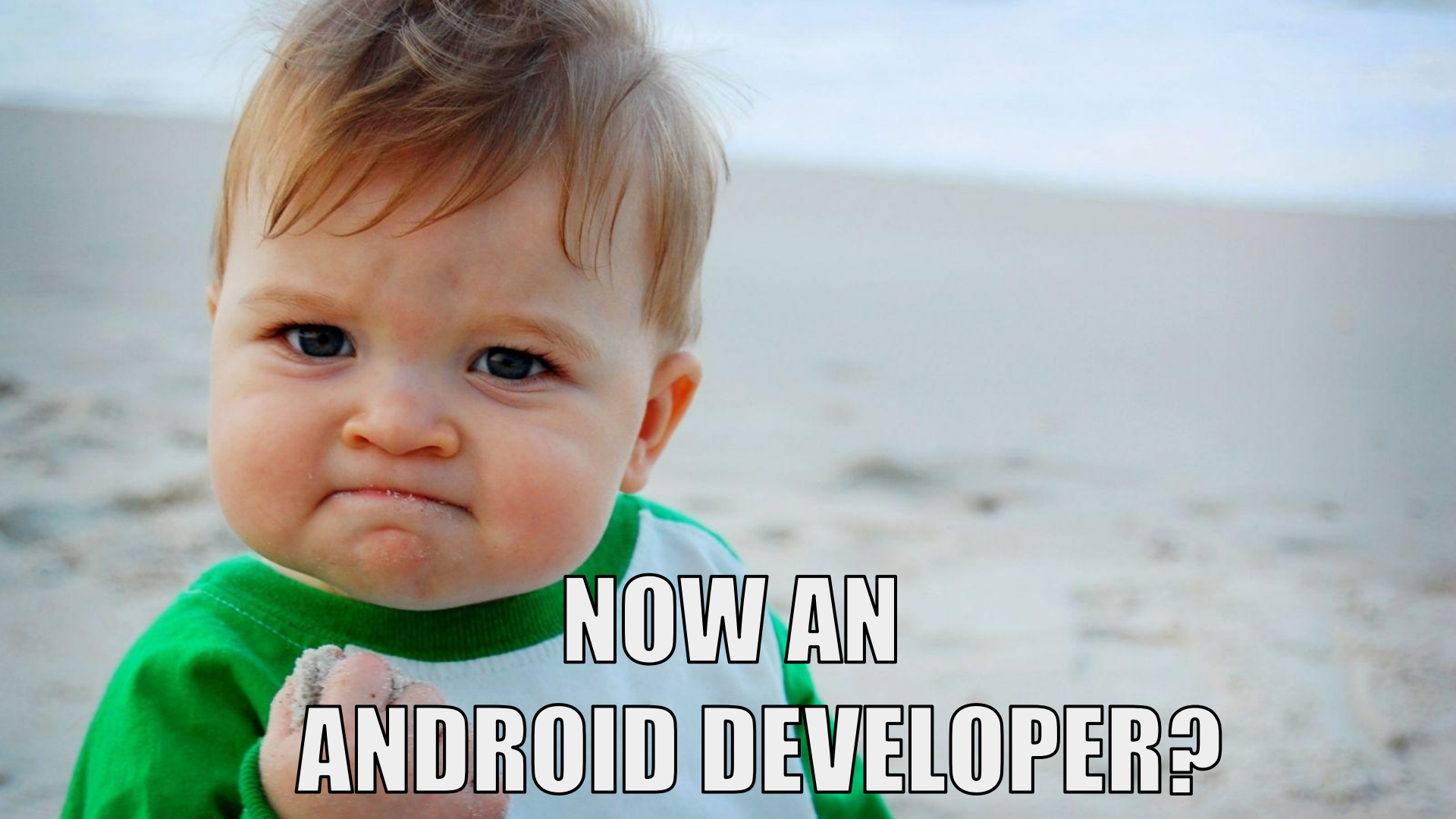
```
from time import sleep
from plyer import accelerometer

accelerometer.enable()

for x in xrange(20):
    print(accelerometer.acceleration)
    sleep(.1)
```

I ❤ Guitar


Chordwise


GIVES TALK ON ANDROID
NOT AN
ANDROID DEVELOPER


NOW AN
ANDROID DEVELOPER


NOW AN
ANDROID DEVELOPER?

kivy.org

Shameless Plug

The **Fresno Python User Group** meets every **4th Tuesday** of the month, right here at **Bitwise**.

FresnoPython.com

Thanks!

Hit me up

derek.payton@gmail.com

twitter.com/dmpayton github.com/dmpayton