Kivy: a sweet new app development framework

Andy Wilson

January 11, 2012

hey who let you in here

my experience with mobile/desktop apps:

hey who let you in here

my experience with mobile/desktop apps:

- Kivy user since Saturday

hey who let you in here

my experience with mobile/desktop apps:

- Kivy user since Saturday
- ETS (TraitsUI, Chaco) for numpy/scipy-type apps

hey who let you in here

my experience with mobile/desktop apps:

- Kivy user since Saturday
- ETS (TraitsUI, Chaco) for numpy/scipy-type apps
- Occasionally fiddle with Qt and friends: PyQT4, pyside

hey who let you in here

my experience with mobile/desktop apps:

- Kivy user since Saturday
- ETS (TraitsUI, Chaco) for numpy/scipy-type apps
- Occasionally fiddle with Qt and friends: PyQT4, pyside
- I tried to do something with pygame once but got bored

the skinny

- Fast

the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0

the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0
  - important bits written in C/Cython

the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0
  - important bits written in C/Cython
- Simple: beautiful API

the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0
  - important bits written in C/Cython
- Simple: beautiful API
- Multitouchtouchtouch

the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0
  - important bits written in C/Cython
- Simple: beautiful API
- Multitouchtouchtouch
- Crossplatform: Linux, Windows, OSX, Android

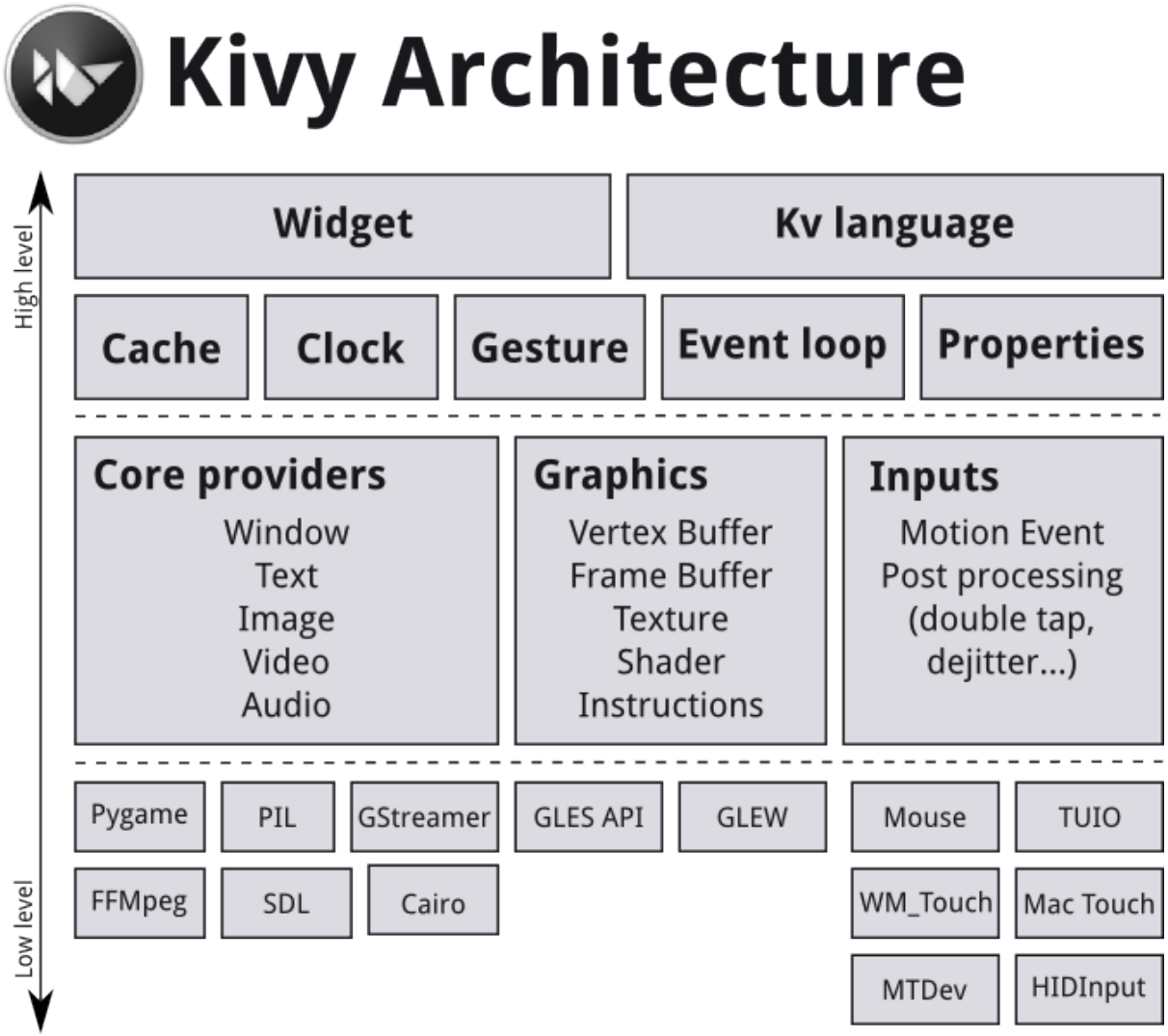
the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0
  - important bits written in C/Cython
- Simple: beautiful API
- Multitouchtouchtouch
- Crossplatform: Linux, Windows, OSX, Android , ...iOS?

the skinny

- Fast
  - graphics system implemented OpenGL ES 2.0
  - important bits written in C/Cython
- Simple: beautiful API
- Multitouchtouchtouch
- Crossplatform: Linux, Windows, OSX, Android , ...iOS?
- License: LGPL3

howsisworkthen?



source: http://kivy.org/docs/guide/architecture.html

widgets

Widgets are...

widgets

Widgets are...

- anything you draw

widgets

Widgets are...

- anything you draw
- anything you touch

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)
- trees

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)
- trees
  - there's always at least one root widget

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)
- trees
  - there's always at least one root widget
  - parent.add_widget(child) - attach a widget

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)
- trees
  - there's always at least one root widget
  - parent.add_widget(child) - attach a widget
  - parent.children - a list of widgets

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)
- trees
  - there's always at least one root widget
  - parent.add_widget(child) - attach a widget
  - parent.children - a list of widgets
  - parent.remove_widget(child) - remove a widget

widgets

Widgets are...

- anything you draw
- anything you touch
- anything you want to remember (data)
- trees
  - there's always at least one root widget
  - parent.add_widget(child) - attach a widget
  - parent.children - a list of widgets
  - parent.remove_widget(child) - remove a widget
  - parent.clear_widgets() - remove all widgets

hello world

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

class HelloWidget(Widget):
    def __init__(self, **kwargs):
        super(HelloWidget, self).__init__(**kwargs)
        with self.canvas:
            Label(text='hello world!')

class WorldApp(App):
    def build(self):
        return HelloWidget()

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

properties

Widgets can have properties. Properties are values you can bind callback functions to.

properties

Widgets can have properties. Properties are values you can bind callback functions to.
Examples:

- StringProperty

properties

Widgets can have properties. Properties are values you can bind callback functions to.
Examples:

- StringProperty
- ListProperty

properties

Widgets can have properties. Properties are values you can bind callback functions to.
Examples:

- StringProperty
- ListProperty
- BooleanProperty

properties

Widgets can have properties. Properties are values you can bind callback functions to.
Examples:

- StringProperty
- ListProperty
- BooleanProperty
- ObjectProperty

property example

```
from kivy.properties import StringProperty
from kivy.uix.widget import Widget

def some_callback(instance, value):
    print "callback fired: property changed to " + value

class Foo(Widget):
    bar = StringProperty('starting value')

    def __init__(self, **kwargs):
        super(Foo, self).__init__(**kwargs)
        self.bind(bar=some_callback)

if __name__ == '__main__':
    f = Foo()
    f.bar = 'changeit'
    # callback fired: property changed to changeit
```

clock

There's also a clock.

clock

There's also a clock.
It does what you think it does.

clock

There's also a clock.
It does what you think it does.

- clock.schedule_once(some_function, number_of_seconds)

clock

There's also a clock.
It does what you think it does.

- clock.schedule_once(some_function, number_of_seconds)
- clock.schedule_interval(some_function, number_of_seconds)

clock

There's also a clock.
It does what you think it does.

- clock.schedule_once(some_function, number_of_seconds)
- clock.schedule_interval(some_function, number_of_seconds)

input... more input

Input events come with all widgets; just override these methods

input... more input

Input events come with all widgets; just override these methods

- on_touch_down() - someone touchs/clicks this widget

input... more input

Input events come with all widgets; just override these methods

- on_touch_down() - someone touchs/clicks this widget
- on_touch_move() - someone drags this widget

input... more input

Input events come with all widgets; just override these methods

- on_touch_down() - someone touchs/clicks this widget
- on_touch_move() - someone drags this widget
- on_touch_up() - someone stops touching/clicking this widget

input... more input

Input events come with all widgets; just override these methods

- on_touch_down() - someone touchs/clicks this widget
- on_touch_move() - someone drags this widget
- on_touch_up() - someone stops touching/clicking this widget

widgets can choose to intercept a touch or pass it along to their underwidgets

graphics

Every widget has a canvas, and it works like this:

```
    class Foo(Widget):
        def __init__(self, **kwargs):
            super(Foo, self).__init__(**kwargs)
            with self.canvas:
                # set color
                Color(1., 0., 0.)
                # draw stuff with primitives
                Rectangle(pos=(20, 30), size=(100, 100))
            # nevermind; erase what we just did
            self.canvas.clear()
```

graphics

Canvas instructions get translated to OpenGL instructions.
It's very snappy.

graphics

Canvas instructions get translated to OpenGL instructions.
It's very snappy.
But when building something non-trivial, you end up having to keep track of canvas state yourself (when to clear, when to draw).

kivy language (.kv)

Enter the kivy language.

```
<Foo>:
    canvas:
        Color:
            rgb: 1., .0, .0
        Rectangle:
            pos: (20, 30)
            size: (100, 100)
```

Now widgets know how to draw themselves.

stop

demo time

other nicities

- cache

other nicities

- cache
- vectors!

other nicities

- cache
- vectors!
- gesture recognition

[CITATION NEEDED]

homepage: `http://kivy.org`
on github: `http://github.com/kivy/kivy`

two (wo)men enter one (wo)man leaves

`http://kivy.org/#contest`

- make a game with Kivy
- put it on github
- prizes: android tablets, github subscriptions, t-shirts
- registration deadline is Jan 25th - contest ends Jan 31st