How to use Kivy in your Python projects

Installation

Open your command line and enter the following*:

***Note: macOS users follow a different procedure from Windows users**

INSTALL ON MACOS

Download the kivy.app and copy it to Applications.

Run the `makesymlinks` in the window that opens when you open the dmg.

INSTALL ON WINDOWS

```
python -m pip install --upgrade pip wheel setuptools
```

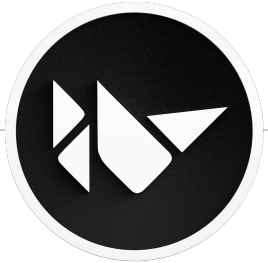
```
python -m pip install docutils pygments pypiwin32 kivy.deps.sdl2 kivy.deps.angle
```

```
python -m pip install kivy.deps.gstreamer
```

```
python -m pip install kivy
```

For a more in-depth tutorial on how to install Kivy, visit https://kivy.org/doc/stable/installation/installation.html

More modules and dependencies are available to install through pip on the website.



Hello!

I'm an NUI Framework

I can help you create a simple, cross-platform Natural User Interface for your program.

What does Kivy do?

Simplify

Kivy simplifies the process of creating a UI for your program. It has a default look and large array of interactive elements to get you started.

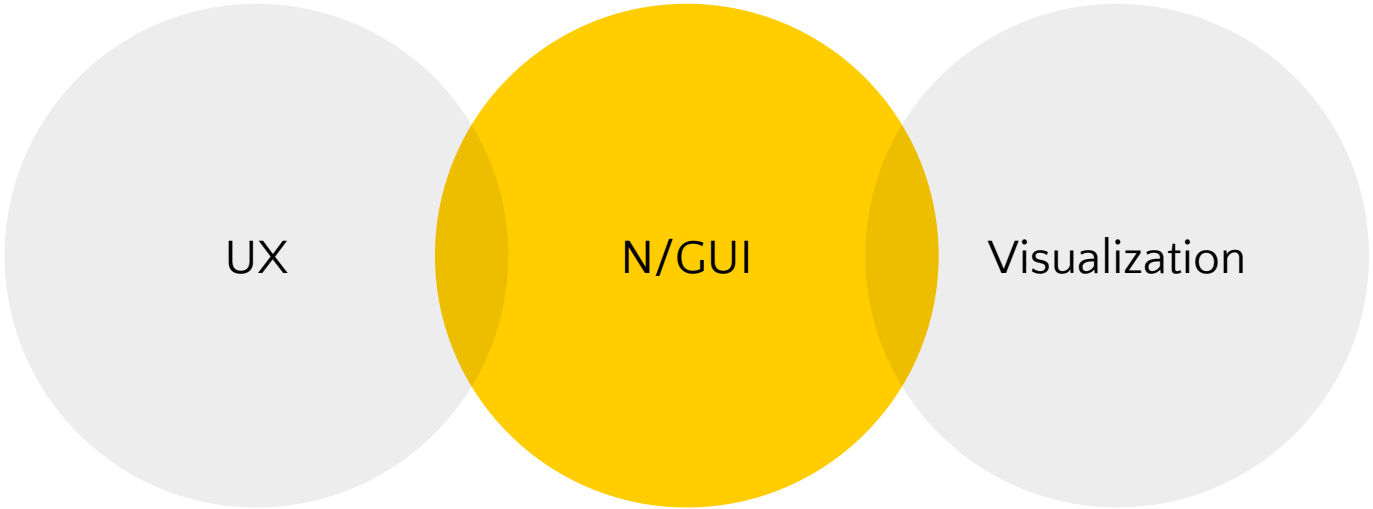
Organize

Kivy organizes your user interface in an easily legible and editable way. You can change the interaction between elements easily with one line of code.

Visualize

Kivy has many components that will help users manipulate and visualize 3D, video, and audio files.

What does Kivy do?



Writing in Kv

An easy way to describe user interfaces

Why we use Kv over Python

Kv

Clean, focused on layout and interactions.

Responsive to display size and platform.

Easy to read and edit.

Python

Complicated, focused on logic and function.

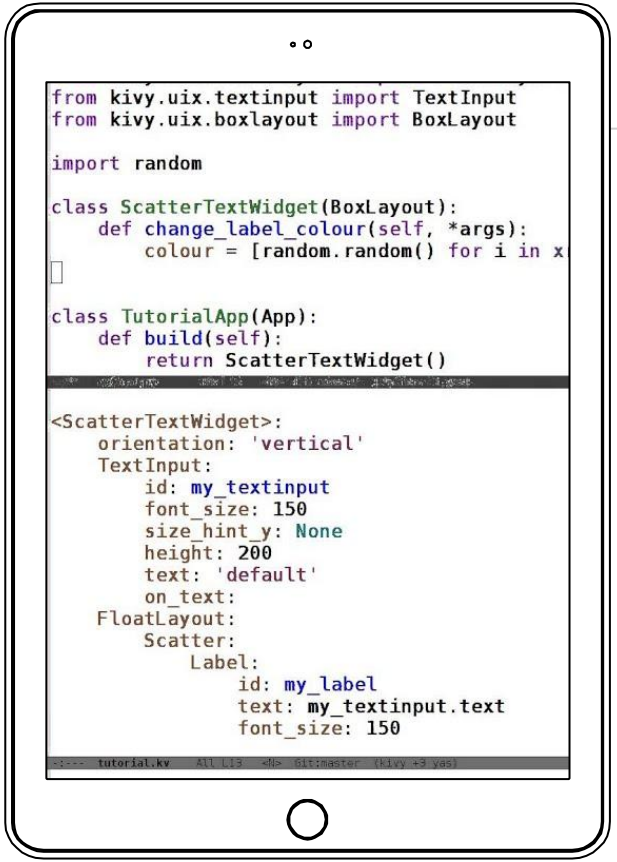
Rigid in terms of display size and compatibility not guaranteed.

Messy to edit or debug.

The Kv syntax

Call Kv code

Root Widget

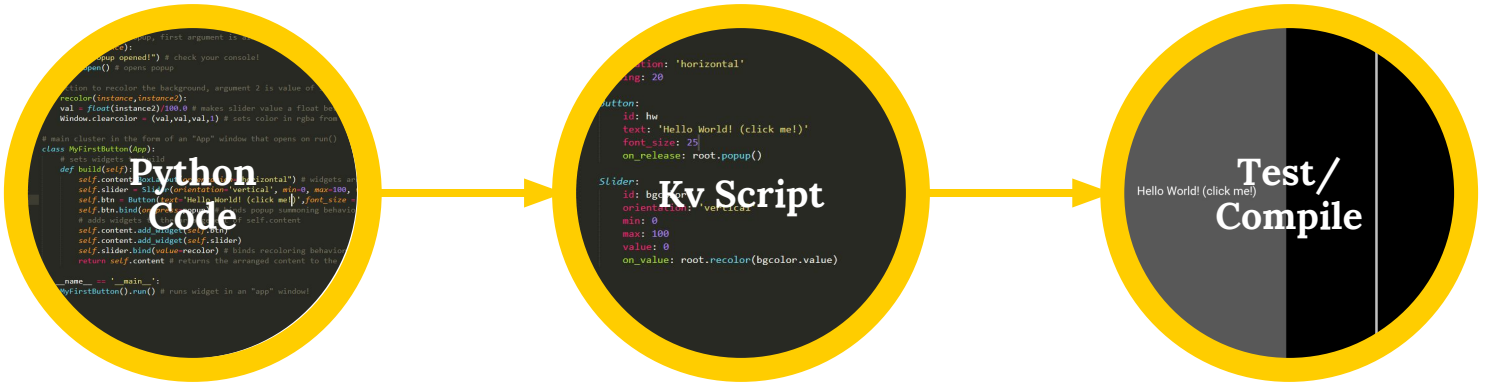


```
from kivy.uix.textinput import TextInput
from kivy.uix.boxlayout import BoxLayout

import random

class ScatterTextWidget(BoxLayout):
    def change_label_colour(self, *args):
        colour = [random.random() for i in x
[]

class TutorialApp(App):
    def build(self):
        return ScatterTextWidget()
```

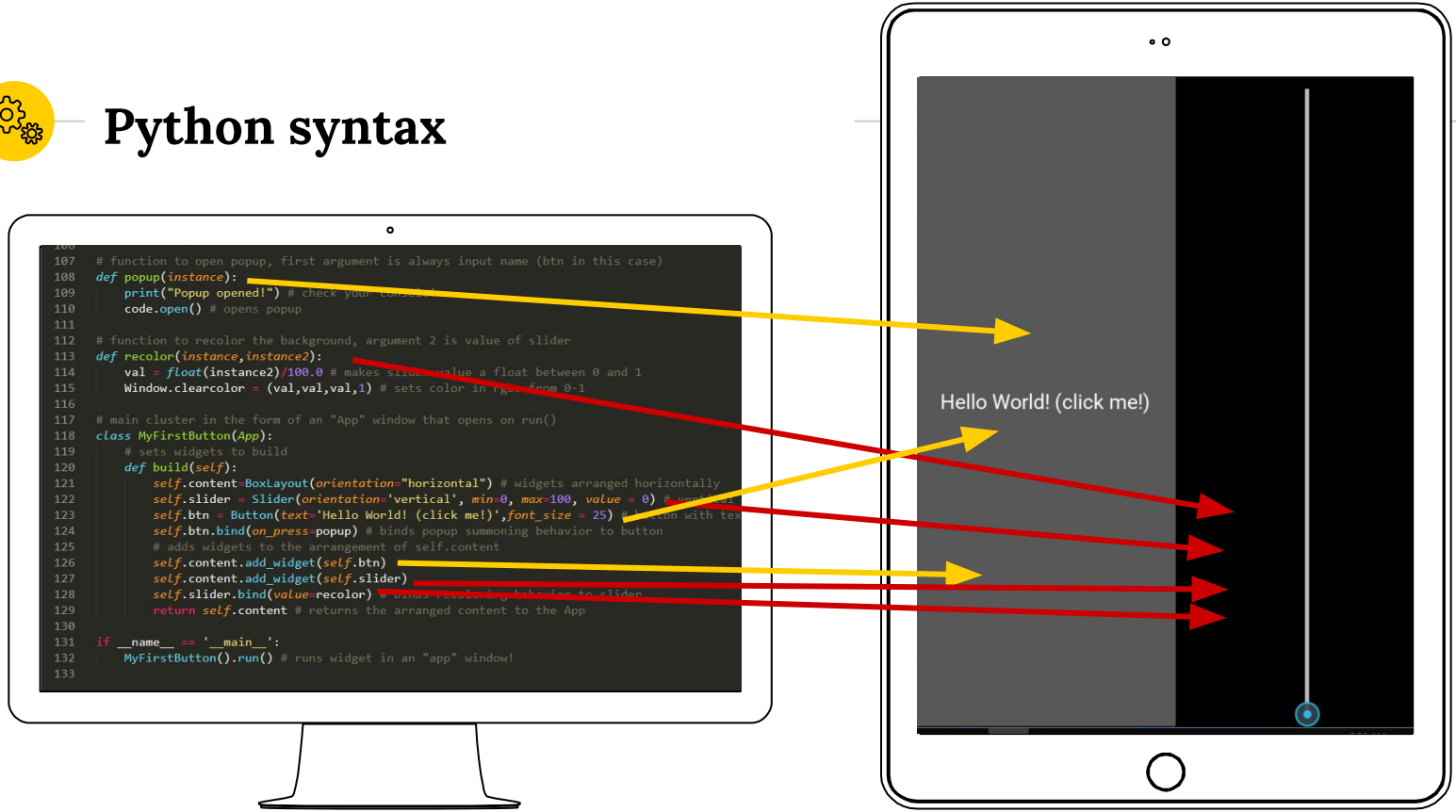
```
<ScatterTextWidget>:
    orientation: 'vertical'
    TextInput:
        id: my_textinput
        font_size: 150
        size_hint_y: None
        height: 200
        text: 'default'
        on_text:
    FloatLayout:
        Scatter:
            Label:
                id: my_label
                text: my_textinput.text
                font_size: 150
```

-:--- tutorial.kv All L13 <N> Bitmaster (kivy +3 yas)

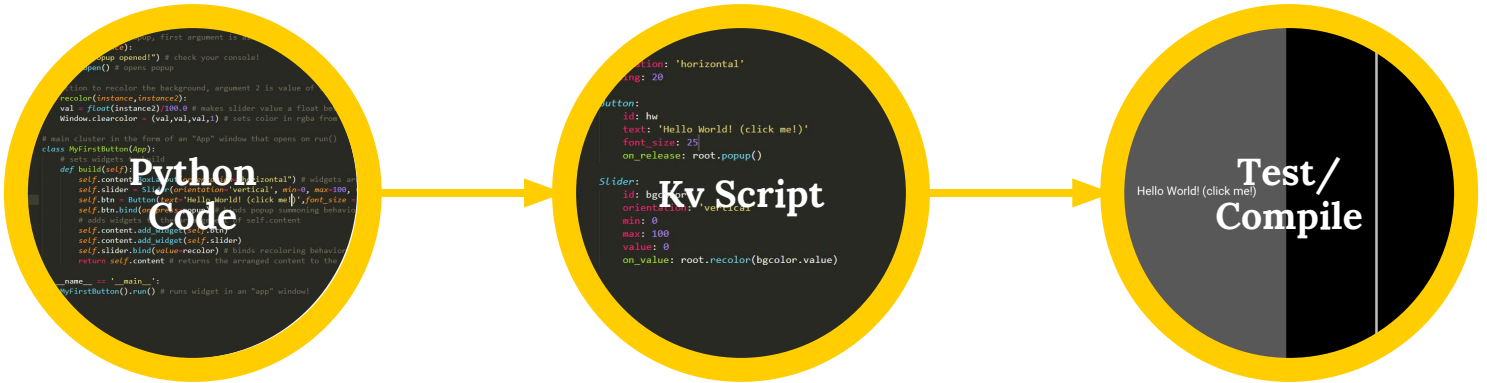
The Kv Workflow



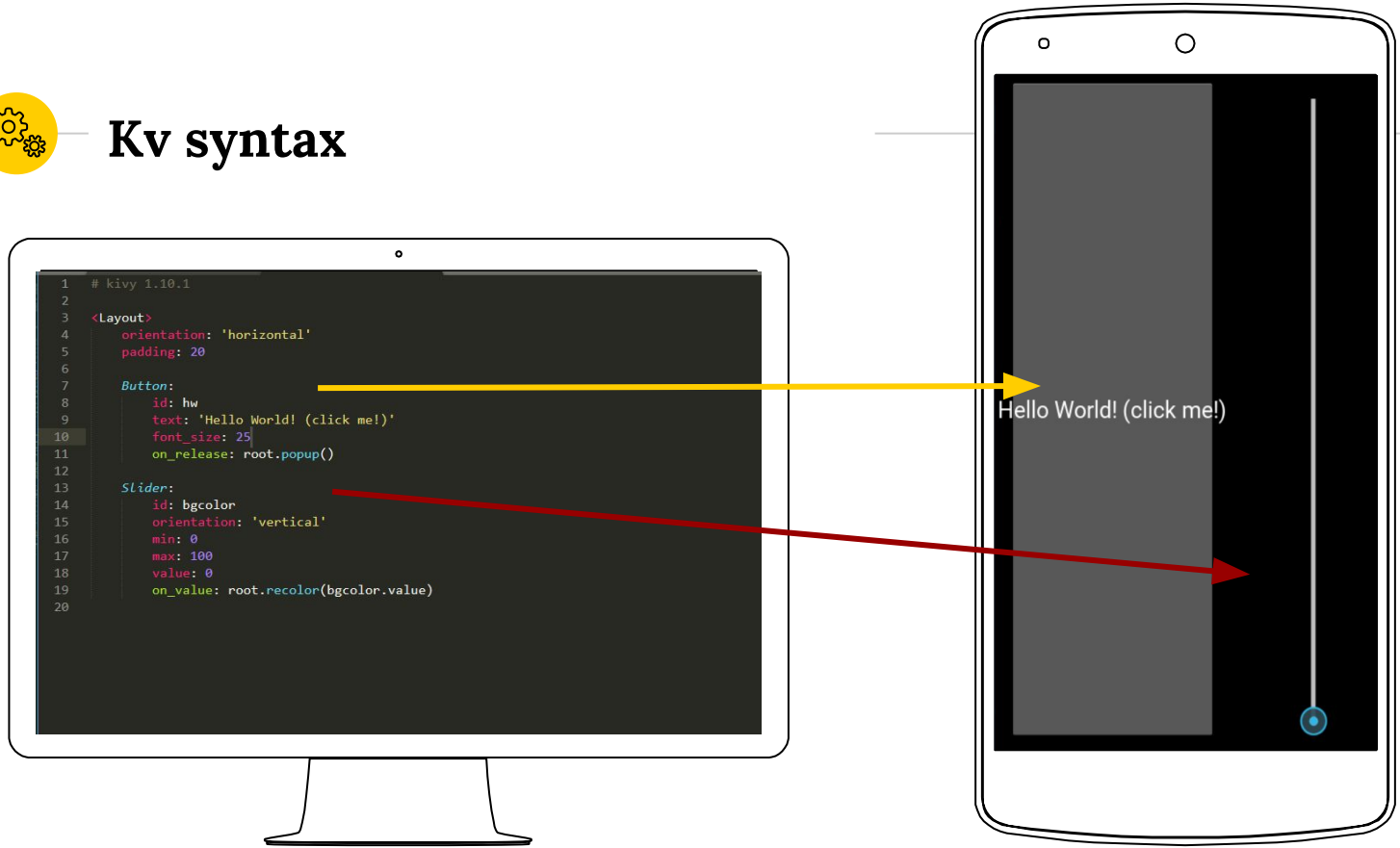
Python syntax



The Kv Workflow



Kv syntax


1 # kivy 1.10.1
2
3 <Layout>
4 orientation: 'horizontal'
5 padding: 20
6
7 Button:
8 id: hw
9 text: 'Hello World! (click me!)'
10 font_size: 25
11 on_release: root.popup()
12
13 Slider:
14 id: bgcolor
15 orientation: 'vertical'
16 min: 0
17 max: 100
18 value: 0
19 on_value: root.recolor(bgcolor.value)
20
Hello World! (click me!)

```
    TabbedPanel:
        id: tp
        do_default_tab: False

        TabbedPanelItem:
            id: tab_fl
            text: 'FloatLayout'
            on_release: app.showcase_floatlayout(fl)
            FloatLayout:
                CFloatLayout:
                    id: fl
        TabbedPanelItem:
            text: 'BoxLayout'
            on_release: app.showcase_boxlayout(box)
            FloatLayout
                CBoxLayout:
                    id: box
        TabbedPanelItem:
            text: 'GridLayout'
            on_release: app.showcase_gridlayout(grid)
            FloatLayout
                CGridLayout:
                    id: grid
                    rows: 3
        TabbedPanelItem:
            on_release: app.showcase_stacklayout(stack)
            text: 'StackLayout'
            FloatLayout
                CStackLayout:
                    id: stack
        TabbedPanelItem:
            text: 'AnchorLayout'
            on_release: app.showcase_anchorlayout(anchor)
            FloatLayout
                CAnchorLayout:
                    id: anchor
                    BoxLayout:
                        orientation: 'vertical'
                        size_hint: .4, .5
                        Button
                        Button
                            text: 'anchor_x: {}'.format(anchor.anchor_x)
```

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

rows: None
cols: 3

Cross-platform

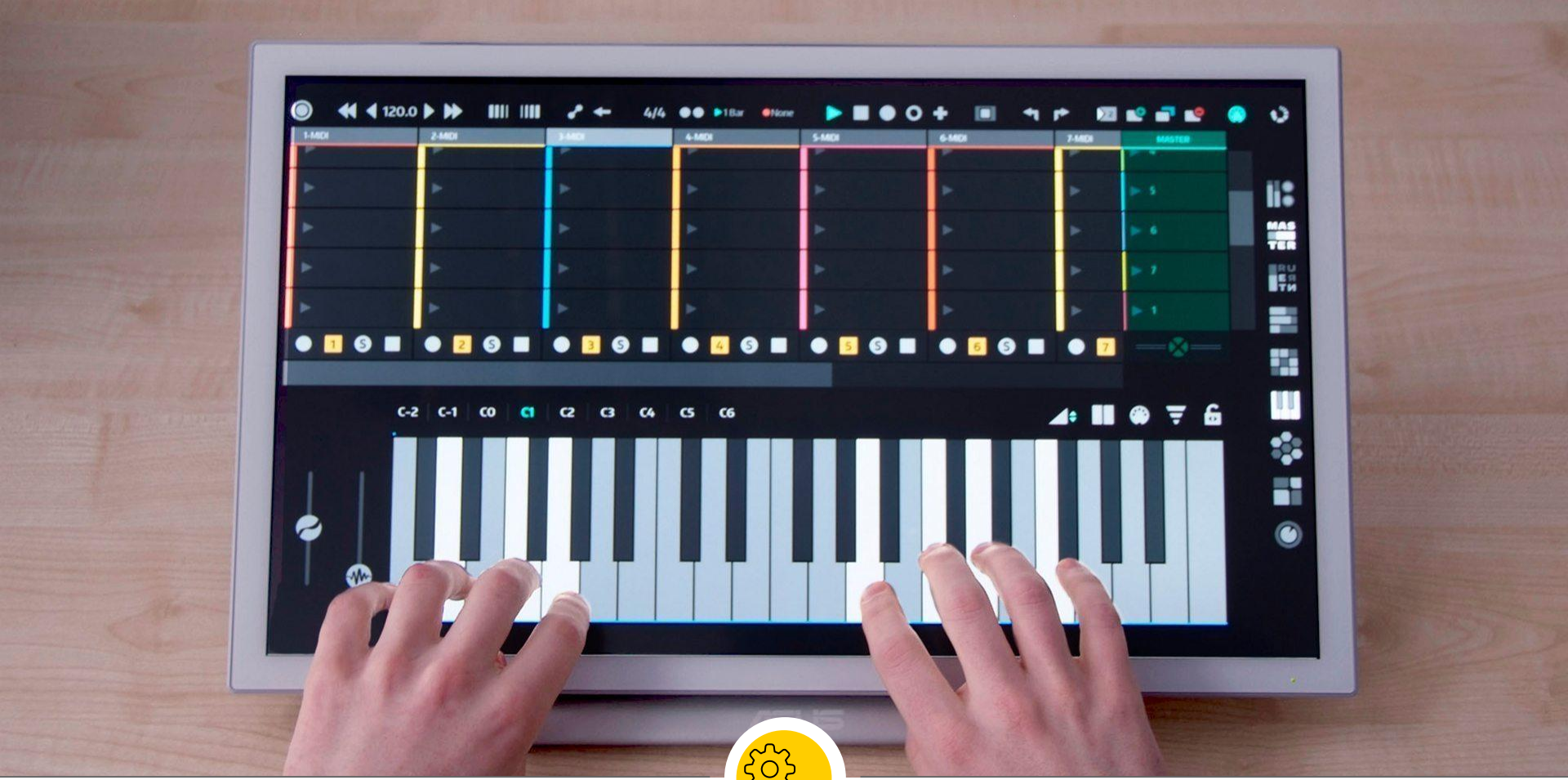
Kivy's biggest advantage is its versatility on mainstream platforms.

Applications

- The Mission Pinball Framework
- Gravity Ride on Google Play (Android)
- Yeco music controller (macOS and Windows)
- The noBOOTH Photo Station
- Various RPi Projects

Yeco Music Controller

A touch control suite for Ableton Live.


120.0
4/4
1-MIDI
2-MIDI
3-MIDI
4-MIDI
5-MIDI
6-MIDI
7-MIDI
MASTER
C-2
C-1
C0
C1
C2
C3
C4
C5
C6

Thanks!

Any questions ?

Other materials:

- @suddenlykevin on GitHub
- https://www.kivy.org/

Credits

https://kivy.org/#gallery

http://www.yeco.io/

https://kivy.org/doc/stable/