Prac07：

box\_layout\_demo.py with modifications

from kivy.app import App  
from kivy.lang import Builder  
  
  
class BoxLayoutDemo(App):  
 def build(self):  
 self.title = **"Box Layout Demo"** self.root = Builder.load\_file(**'box\_layout.kv'**)  
 return self.root  
  
 def handle\_greet(self):  
 print(**"test"**)  
 self.root.ids.output\_label.text = **"Hello "** + self.root.ids.input\_name.text  
  
BoxLayoutDemo().run()

box\_layout\_demo.kv with modifications

BoxLayout:  
 orientation: 'vertical'  
*# orientation: 'horizontal'* Button:  
 text: 'Clear'  
 TextInput:  
 id: input\_name  
 text: ''  
 Button:  
 text: 'Greet'  
 on\_press: app.handle\_greet()  
 Button:  
 text: 'three'  
 Label:  
 text: 'Enter your name'  
 id: output\_label  
 color: (1, 1, 0, 1)

squaring.py with modifications

*"""  
CP1404/CP5632 Practical  
Kivy GUI program to square a number  
Lindsay Ward, IT@JCU  
Started 13/10/2015  
"""*from kivy.app import App  
from kivy.lang import Builder  
from kivy.core.window import Window  
  
\_\_author\_\_ = **'Lindsay Ward'**class SquareNumberApp(App):  
 *""" SquareNumberApp is a Kivy App for squaring a number """* def build(self):  
 *""" build the Kivy app from the kv file """* Window.size = (200, 100)  
 self.title = **"Square Number"** self.root = Builder.load\_file(**'squaring.kv'**)  
 return self.root  
  
 def handle\_calculate(self, value):  
 *""" handle calculation (could be button press or other call), output result to label widget """* result = value \*\* 2  
 self.root.ids.output\_label.text = str(result)  
  
SquareNumberApp().run()

squaring.kv with modifications

BoxLayout:  
 orientation: 'vertical'

BoxLayout:  
 orientation: 'horizontal'  
 TextInput:  
 id: input\_number  
 text: '7'  
 font\_size: 48  
 multiline: False  
 Button:  
 text: 'Square'  
 *# the following line specifies the function in the app class to call when the button is pressed* on\_press: app.handle\_calculate(int(input\_number.text))  
 Label:  
 id: output\_label  
 font\_size: 48  
 color: (1, 0.76, 0.77, 1)  
 Label:  
 text: 'Enter a number and press "Square"'

convert\_miles\_km.py

from kivy.app import App  
from kivy.lang import Builder  
from kivy.app import StringProperty  
  
class MilesToKmApp(App):  
 result = StringProperty()  
  
 def build(self):  
 self.title = **"Convert Miles to Kilometers"** self.root = Builder.load\_file(**'convert\_miles\_km.kv'**)  
 return self.root  
  
 def handle\_convert(self, input):  
 mile = self.handle\_valid(input)  
 kilometer = mile \* 1.609  
 self.result = str(kilometer)  
  
 def handle\_increment(self, input, increment):  
 mile = self.handle\_valid(input) + increment  
 self.root.ids.input\_miles.text = str(mile)  
  
 def handle\_valid(input):  
 try:  
 return float(input)  
 except:  
 return 0.0  
  
MilesToKmApp().run()

convert\_miles\_km.kv

BoxLayout:  
 orientation: 'vertical'  
 BoxLayout:  
 orientation: 'horizontal'  
 TextInput:  
 id: input\_miles  
 font\_size: 48  
 on\_text: app.handle\_convert(self.text)  
 BoxLayout:  
 orientation: 'vertical'  
 size\_hint\_x: 0.2  
 Button:  
 text: 'Up'  
 on\_press: app.handle\_increment(1)  
 Button:  
 text: 'Down'  
 on\_press: app.handle\_increment(-1)  
 Label:  
 id: output\_label  
 font\_size: 48  
 color: (1, 1, 0, 1)  
 text: app.result

dynamic\_labels.py with Labels (not Buttons)

from kivy.app import App  
from kivy.lang import Builder  
from kivy.uix.label import Label  
from kivy.app import StringProperty  
  
class DynamicLabelsApp(App):  
 name\_text = StringProperty()  
  
 def \_\_init\_\_(self, \*\*kwargs):  
 super().\_\_init\_\_(\*\*kwargs)  
 self.names = [**'AA'**, **'BB'**, **'CC'**, **'DD'**]  
  
 def build(self):  
 self.title = **"Dynamic Labels"** self.root = Builder.load\_file(**'dynamic\_labels.kv'**)  
 return self.root  
  
 def create\_widgets(self):  
 for name in self.names:  
 temp\_label = Label(text=name, id=name)  
 temp\_label.bind(on\_release=self.press\_entry)  
 self.root.ids.entries\_box.add\_widget(temp\_label)  
  
 def press\_entry(self, instance):  
 name = instance.id  
 self.name\_text = str(name)  
  
 def clear\_all(self):  
 self.root.ids.entries\_box.clear\_widgets()  
  
  
DynamicLabelsApp().run()