Лабораторная работа №11
Работа с событиями виджетов фреймворка

Цель работы: сформировать умения работать с событиями виджетов фреймворка Kivy.

Задание:
1. Изучить теоретические сведения.
2. Выполнить задания в соответствии с вариантом.

Теоретические сведения

В Kivy реализовано два способа реагирования на события:
– явное связывание визуального элемента с заданной функцией;
– неявное связывание визуального элемента с заданной функцией.

Для явного связывания визуального элемента с заданной функцией создадим новый файл Button_Otklik1.py и внесем в него следующий код (модуль создан в коде на языке Python):

```
# модуль Button_Otklik1.py
from kivy. app import App
from kivy. uix. button import Button

class MainApp (App):
…… def build (self):
…… … … button = Button (text=«Кнопка»,
…… … … …… … … size_hint= (.5,.5),
…… … … …… … … pos_hint= {’center_x’:.5, ’center_y’:.5})
…… … … button.bind(on_press=self.press_button)
…… … … return button

…… def press_button (self, instance):
…… … … print («Вы нажали на кнопку!»)

MainApp().run ()
```

Здесь в базовом классе реализовали две функции:
– в первой (def build) создали кнопку, поместили ее в центре окна приложения и связали событие нажатие кнопки (on_press) с функцией – press_button;
– во второй функции (def press_button) прописали действия, которые необходимо выполнить при касании кнопки (в качестве такого действия задан вывод в терминальное окно сообщения ««Вы нажали на кнопку!»).

Реализуем тот же пример с использованием языка KV. Для этого создадим файл с именем Button_Otklik11.py и напишем в нем следующий код:

```
# модуль Button_Otklik11.py
from kivy. app import App
from kivy.lang import Builder

KV = «»»
Button:
…… text: «Кнопка»
…… size_hint:.5,.5
…… pos_hint: {'center_x':.5, 'center_y':.5}
…… on_press: app.press_button (root)
«»»

class MainApp (App):
…… def build (self):
…… …… return Builder. load_string (KV)

…… def press_button (self, instance):
…… …… print («Вы нажали на кнопку!»)

MainApp().run ()
```

Здесь в строковой переменной KV обрабатывается событие нажатия кнопки (on_press). При возникновении данного события выполняется обращение к функции приложения press_button, которая находится в корневом модуле (root).

На языке Kivy достаточно просто организована обработка событий:

«событие: функция обработки события»

Для неявного связывания визуального элемента с заданной функцией создадим новый файл Button_Otklik2.py и внесем в него следующий код:

```
# модуль Button_Otklik2.py
from kivy. app import App
from kivy. uix. button import Button

class Basic_Class1 (App):
…… def build (self):
…… … … button = Button (text=«Кнопка»,
…… … … … … ….. … size_hint= (.5,.5),
…… … … … … ….. … pos_hint= {'center_x':.5, 'center_y':.5})
…… return button

…… def press_button (self):
…… … … print («Вы нажали на кнопку!»)

My_App = Basic_Class1 () # приложение на основе базового
класса
My_App.run () # запуск приложения
```

В данном коде создана кнопка button на основе базового класса Button, но эта кнопка не имеет связи с функцией обработки события ее нажатия, хотя сама функция press_button присутствует. С первого взгляда данный код может показаться странным, так как кнопка

button не связана с функций реакции на событие нажатия кнопки. Такую связку можно реализовать на уровне языка KV. Вспомним, что при запуске головного модуля Kivy автоматически ищет файл с таким же названием, что и у базового класса (в данном случае файл – basic_class1.kv), и выполняет запрограммированные там действия. Создадим файл с именем basic_class1.kv и внесем в него следующий программный код:

```
# файл basic_class1.kv
<Button>:
…… on_press: app.press_button ()
```

Таким образом, связь отклика на нажатия кнопки перенесли из основного модуля, в связанный модуль на языке KV.

Задания

Задание 1

Создать exe-приложение, которое будет запрашивать у пользователя ввод даты, анализировать сайт ggpk.by и выводить сообщение есть ли расписание на следующий день (лабораторная работа №6 «Разработка программ, передающих и получающих данные по протоколу HTTP»).

Задание 2

Создать exe-приложение (Лабораторная работа №5 «Использование возможностей объектно-ориентированного программирования в Python»)

В1-В6: Напишите игру по следующему описанию. Есть класс «НЛО» и класс «ЛюдиХ». У НЛО один объект, имеющий здоровье 1000hp. Создается от 1 до 5 людей Х (случайное число), имеющих здоровье 100 hp каждый. В случайном порядке НЛО и ЛюдиХ бьют друг друга. Тот, кто бьет, здоровья не теряет. У того, кого бьют, оно уменьшается на N очков от одного удара (N случайное число). После каждого удара надо выводить сообщение, кто кого атаковал, и сколько у противников осталось здоровья. Как только у НЛО заканчивается ресурс здоровья или погибают все ЛюдиХ, программа завершается сообщением о том, кто одержал победу.

В7-В13: Напишите игру по следующему описанию. В игре несколько уровней. Цель игры - победить драконов. На первом уровне – один дракон, на втором уровне – два, на третьем – три и т.д. Номер уровня вводит пользователь. У каждого дракона устанавливается здоровье в 100 очков. У игрока – 500 очков. Пользователь вводит число – силу удара. На это число уменьшается здоровье дракона. Когда здоровье уменьшится до 0, дракон погибает, наступает битва со следующим (если в уровне несколько драконов). Дракон наносит удар – случайное число в диапазоне 10-90. Игра завершена, когда все драконы уровня убиты или погиб игрок.

Контрольные вопросы:

1. Приведите примеры событий. Какое основное событие у кнопки?
2. Как реализовано реагирование на события в Kivy?
3.Запишите, как будет выглядеть обработка события «нажатие кнопки» непосредственно в коде на KV.