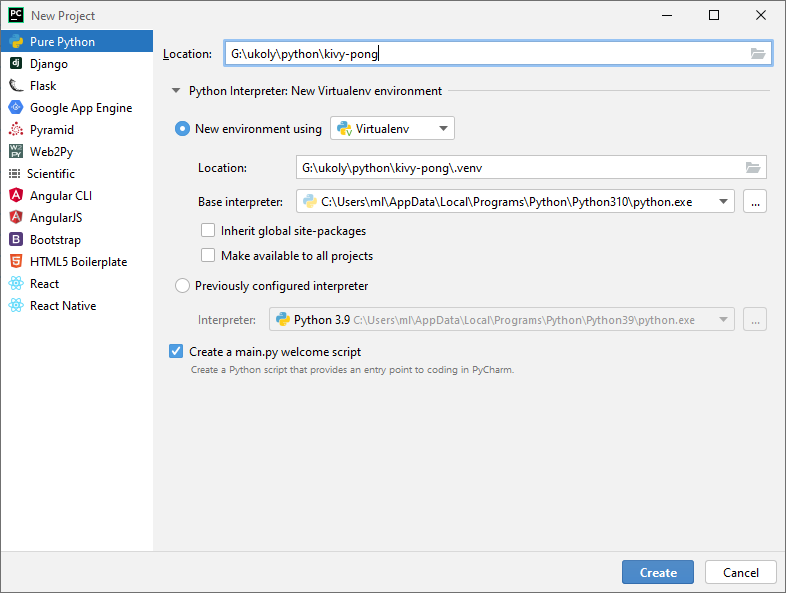
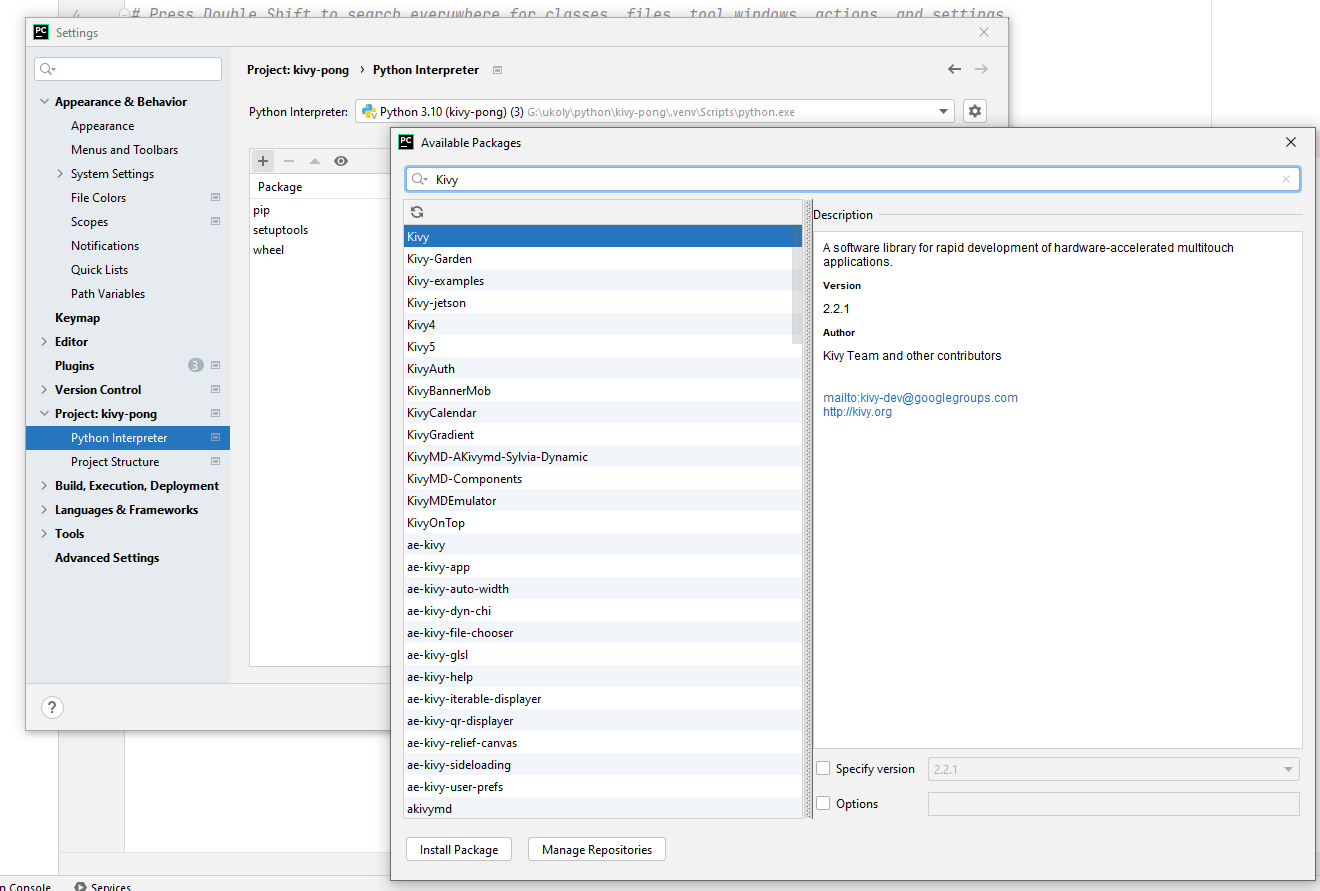
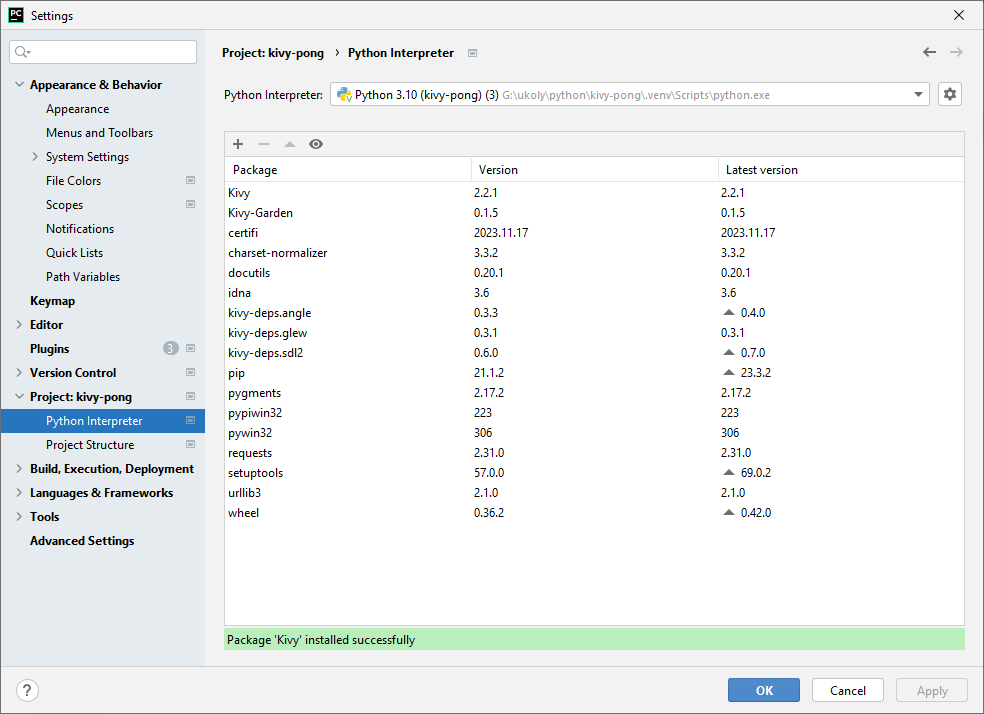
# Hra Pong s využitím frameworku Kivy

1. Založení projektu a instalace Kivy







**Základní kostra projektu:**

import kivy

from kivy.app import App

from kivy.uix.widget import Widget

# Hlavní třída pro Pong hru

class PongGame(Widget):

pass

# Třída pro spuštění hry

class PongApp(App):

def build(self):

return PongGame()

# Spuštění hry

if \_\_name\_\_ == '\_\_main\_\_':

PongApp().run()

1. **Vytvoření grafického rozhraní - soubor pong.kv**

<Ball>:  
 size: 25, 25  
 canvas:  
 Ellipse:  
 pos: self.pos  
 size: self.size  
  
<Paddle>:  
 size: 25, 100  
 canvas:  
 Rectangle:  
 pos: self.pos  
 size: self.size  
  
<PongGame>:  
 ball: pong\_ball  
 player1: player\_left  
 player2: player\_right  
 canvas:  
 Color:  
 rgba: 0, 1, 0, .5  
 Rectangle:  
 pos: self.pos  
 size: self.size  
  
 Color:  
 rgba: 1, 1, 1, 1  
 Rectangle:  
 pos: self.center\_x - 5, 0  
 size: 10, self.height  
  
 Label:  
 font\_size: 70  
 center\_x: root.width / 4  
 top: root.top - 50  
*# text: str(root.player1.score)* text: "0"  
  
 Label:  
 font\_size: 70  
 center\_x: root.width \* 3 / 4  
 top: root.top - 50  
*# text: str(root.player2.score)* text: "0"  
  
 Ball:  
 id: pong\_ball  
 center: self.parent.center  
  
 Paddle:  
 id: player\_left  
 x: 0  
 center\_y: root.center\_y  
  
 Paddle:  
 id: player\_right  
 x: root.width - self.width  
 center\_y: root.center\_y

*# Třída pro míček*class Ball(Widget):  
 pass

class Paddle(Widget):  
 pass

1. **Pohyb míčku**

*# Třída pro míček*class Ball(Widget):  
 pass  
 *# Rychlost míčku ve dvou osách* velocity\_x = NumericProperty(0)  
 velocity\_y = NumericProperty(0)  
 velocity = ReferenceListProperty(velocity\_x, velocity\_y)  
  
 *# Metoda pro aktualizaci pozice míčku* def move(self):  
 self.pos = Vector(\*self.velocity) + self.pos

*# Hlavní třída pro Pong hru*class PongGame(Widget):  
 ball = ObjectProperty(None)  
def serve\_ball(self, vel=(6, 0)):  
 self.ball.center = self.center  
 self.ball.velocity = vel  
 *#* def update(self, dt):  
 self.ball.move()

*# Třída pro spuštění hry*class PongApp(App):  
 def build(self):  
 game = PongGame()  
 game.serve\_ball()  
 Clock.schedule\_interval(game.update, 1.0 / 60.0)  
 return game

1. **Pálky, odraz míčku**

*# Třída pro pálku*class Paddle(Widget):  
 score = NumericProperty(0) *# Skóre hráče  
  
 # Metoda pro pohyb pálky nahoru* def move\_up(self):  
 self.y += 10 *# Pohyb nahoru o 10 pixelů  
  
 # Metoda pro pohyb pálky dolů* def move\_down(self):  
 self.y -= 10 *# Pohyb dolů o 10 pixelů* def bounce\_ball(self, ball):  
 if self.collide\_widget(ball):  
 vx, vy = ball.velocity  
 offset = (ball.center\_y - self.center\_y) / (self.height / 2)  
 bounced = Vector(-1 \* vx, vy)  
 vel = bounced \* 1  
 ball.velocity = vel.x, vel.y + offset

class PongGame(Widget):  
 ball = ObjectProperty(None)  
 player1 = ObjectProperty(None)  
 player2 = ObjectProperty(None)

def serve\_ball(self, vel=(6, 0)):  
 self.ball.center = self.center  
 self.ball.velocity = vel  
 *#* def update(self, dt):  
 self.ball.move()  
 *# bounce off paddles* self.player1.bounce\_ball(self.ball)  
 self.player2.bounce\_ball(self.ball)

1. **Pohyb pálky, změna skóre**

*# Hlavní třída pro Pong hru*class PongGame(Widget):  
 ball = ObjectProperty(None)  
 player1 = ObjectProperty(None)  
 player2 = ObjectProperty(None)  
def serve\_ball(self, vel=(6, 0)):  
 self.ball.center = self.center  
 self.ball.velocity = vel  
 *#* def update(self, dt):  
 self.ball.move()  
 *# bounce off paddles* self.player1.bounce\_ball(self.ball)  
 self.player2.bounce\_ball(self.ball)

*# went off to a side to score point?* if self.ball.x < self.x:  
 self.player2.score += 1  
 self.serve\_ball(vel=(6, 0))  
 if self.ball.right > self.width:  
 self.player1.score += 1  
 self.serve\_ball(vel=(-6, 0))  
 *#* def on\_touch\_move(self, touch):  
 if touch.x < self.width / 3:  
 self.player1.center\_y = touch.y  
 if touch.x > self.width - self.width / 3:  
 self.player2.center\_y = touch.y

1. **Odraz míčku od horního/dolního mantinelu**

def update(self, dt):  
 self.ball.move()  
 *# bounce off paddles* self.player1.bounce\_ball(self.ball)  
 self.player2.bounce\_ball(self.ball)  
 *# bounce ball off bottom or top* if (self.ball.y < self.y) or (self.ball.top > self.top):  
 self.ball.velocity\_y \*= -1

1. **Ovládání pálek pomocí klávesnice, omezení pohybu pálek**

*# Hlavní třída pro Pong hru*class PongGame(Widget):  
 ball = ObjectProperty(None)  
 player1 = ObjectProperty(None)  
 player2 = ObjectProperty(None)  
 *#* def \_\_init\_\_(self, \*\*kwargs):  
 super(PongGame, self).\_\_init\_\_(\*\*kwargs)  
 self.\_keyboard = Window.request\_keyboard(self.\_keyboard\_closed, self)  
 self.\_keyboard.bind(on\_key\_down=self.\_on\_keyboard\_down)  
  
 def serve\_ball(self, vel=(6, 0)):  
 self.ball.center = self.center  
 self.ball.velocity = vel  
 *#* def update(self, dt):  
 self.ball.move()  
 *# bounce off paddles* self.player1.bounce\_ball(self.ball)  
 self.player2.bounce\_ball(self.ball)  
 *# Omezení pohybu pálky* self.player1.y = min(self.height - self.player1.height, max(0, self.player1.y))  
 self.player2.y = min(self.height - self.player2.height, max(0, self.player2.y))  
  
 *# bounce ball off bottom or top* if (self.ball.y < self.y) or (self.ball.top > self.top):  
 self.ball.velocity\_y \*= -1  
 *#  
 # went off to a side to score point?* if self.ball.x < self.x:  
 self.player2.score += 1  
 self.serve\_ball(vel=(6, 0))  
 if self.ball.right > self.width:  
 self.player1.score += 1  
 self.serve\_ball(vel=(-6, 0))  
 *#* def on\_touch\_move(self, touch):  
 if touch.x < self.width / 3:  
 self.player1.center\_y = touch.y  
 if touch.x > self.width - self.width / 3:  
 self.player2.center\_y = touch.y  
 *#* def \_keyboard\_closed(self):  
 self.\_keyboard.unbind(on\_key\_down=self.\_on\_keyboard\_down)  
 self.\_keyboard = None  
  
 def \_on\_keyboard\_down(self, keyboard, keycode, text, modifiers):  
 if keycode[1] == 'w':  
 self.player1.move\_up()  
 elif keycode[1] == 's':  
 self.player1.move\_down()  
 *# Přidat klávesy pro druhého hráče* elif keycode[1] == 'up':  
 self.player2.move\_up()  
 elif keycode[1] == 'down':  
 self.player2.move\_down()

**Window.request\_keyboard**: Tato metoda je volána pro získání instance klávesnice z aktuálního okna. Metoda **request\_keyboard** se používá k získání odkazu na klávesnici, aby mohl váš program reagovat na stisky kláves.

**self.\_keyboard\_closed**: Toto je odkaz na metodu nebo funkci, která bude volána, když se klávesnice "uzavře" nebo "zruší", například když uživatel klikne mimo textové pole nebo když aplikace ztratí fokus. Je to druh "callbacku", který zajišťuje správné uvolnění zdrojů nebo ukončení naslouchání klávesnici, když už není potřeba.

**self**: Tento argument specifikuje, kdo je "requester" klávesnice, tedy objekt, který o klávesnici požádal. V tomto případě to je instance třídy, ve které je tento kód umístěn.

**self.\_keyboard.bind(on\_key\_down=self.\_on\_keyboard\_down)**:

**self.\_keyboard.bind(...)**: Tato metoda se používá k navázání události klávesnice na konkrétní funkci nebo metodu.

**on\_key\_down=self.\_on\_keyboard\_down**: Tento argument říká, že kdykoliv je stisknuta klávesa (událost **on\_key\_down**), měla by být volána metoda **self.\_on\_keyboard\_down**. **self.\_on\_keyboard\_down** je metoda, kterou definujete v rámci své třídy, a která obsluhuje logiku toho, co se stane, když je stisknuta určitá klávesa.

Celkově tento kód říká: "Chci naslouchat klávesnici. Když je klávesa stisknuta, zavolej metodu **\_on\_keyboard\_down**. Když klávesnice přestane být používána, zavolej metodu **\_keyboard\_closed**." Toto je základní pro zajištění interakce s klávesnicí ve vaší aplikaci Kivy.