International Journal of Management Science and Information Technology (IJMSIT)
Vol. 2, No. 2, July – December 2022.
E-ISSN: 2774-5694, P-ISSN:2776-7388
DOI: https://doi.org/10.35870/ijmsit.v2i2.699

USING THE PYTHON LIBRARY TO CREATE SIMPLE GAME ANIMATIONS

Adi Sadli
STIKES Tanawali Persada Takalar
Email: adisadli@gmail.com

Article Info

Article history:

Received June 30, 2022
Revised July 1, 2022
Accepted August 8, 2022

Keywords:

Game developer
Game engine
Library
Python
Pygame
Sprite

ABSTRACT

The development of the game world which has recently been growing rapidly, has aroused the interest of some young people, to become a Maker Game (Game Developer), why this job is in great demand because it looks very cool. Maybe it has crossed our minds when playing games on PC or Mobile how they can make this game.

As for Python, it is not only used for programming using Machine Learning models, but also for making games, even the well-known game engines that are commonly used to create complex 2D/3D games such as Unity use Python as a programming language other than C# or C++.

Python is the most popular programming language or it would not hurt to say that it is the next generation programming language. In every emerging field in computer science, Python is actively present. Python has an extensive library for various fields such as Artificial intelligence (Pytorch, TensorFlow), Machine Learning (Numpy, Pandas, Matplotlib) and Game development (Pygame,Pyglet).

In this opportunity, we will know a little about creating a simple game using the Pygame library (Python library). Pygame is a Python module that is designed for game application development. Pygame is one of the most used libraries/modules. This advantage of pygame allows us to create a game with complete features. Pygame is portable, can run on almost all operating systems. To note, Pygame has been downloaded more than a million times, and millions of visits to the official website. Pygame is released by the GNU General Public License (GPL) so that we can make a game application that is free, open source, shareware, freeware and commercial games.

Learning python, of course, must be done gradually and focus on areas of learning that are of interest because of the large number of python modules or libraries with their functions and uses. To become a game developer, we must know the basics of making games, one of which is how to create animations on a game object.

Making this game animation only revolves around how to make an object consisting of sprites or several images into animation and use the up, down, left and right arrow keys to move the object. The objects in the game consist of objects commonly called player and opposite.

Corresponding Author:

Adi Sadli
STIKES Tanawali Persada Takalar, Indonesia
Email: adisadli@gmail.com

1. INTRODUCTION

Python is a multipurpose interpretive programming language. Unlike other languages that are difficult to read and understand, the Python programming language is now almost used for all fields ranging from research, artificial intelligence, games, web-based applications, or used to create search engines like Google [1]. In general, the python programming language can be used in software development , research, mathematics, system scripting or web-based applications.

In 1991 the Python programming language was first introduced by Guido van Rossum. Then by the Python Software Foundation it was taken over and developed until now [20]. Almost all operating systems have supported the Python programming language, including the Linux operating system, where almost all distributions have included Python in it [2]. According to StackOverflow (https://stackoverflow.com) it calls python the fastest growing programming language.

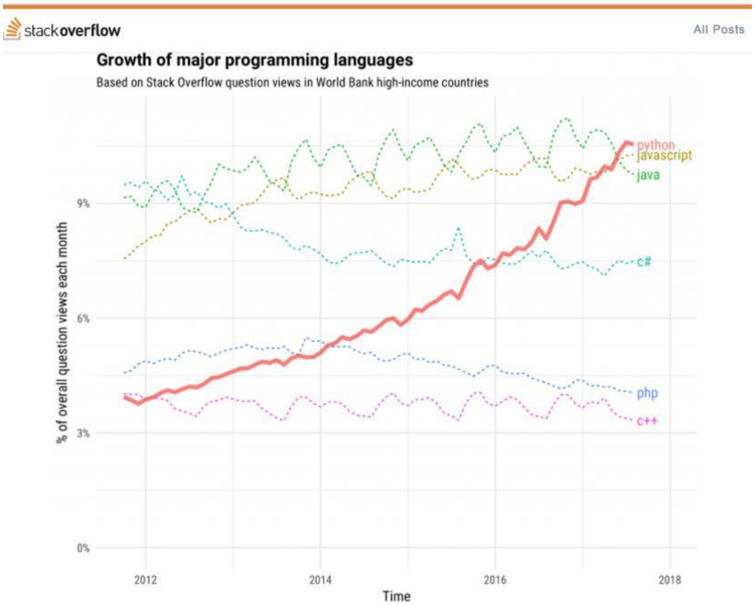


Figure 1. Python Programming Language Development Statistics

Python including language programming level tall, as well as language programming which C++ uses, C, Java, Perl nor Pascal.

As with high-level programming languages, there are also low-level programming languages and also commonly called machine languages, namely assembly programming languages, as it is well known that computers can only carry out commands with low-level languages, for that in their implementation, high-level programming languages must do several things. process to be changed to a low-level programming language, even so it is not a significant weakness for high-level programming languages [3],[19].

There is 2 type application for process language level tall to language level low, namely: compiler and interpreter

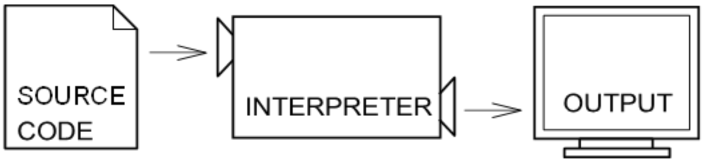


Figure 2: Interpreter

Python as Interpreter, interpreter here means that the computer directly executes its instructions through reading the programming language that has been written or scripting without having to compile it first into machine language [18]. Unlike visual programming such as Android or Java, the interpreter reads a list of programs written in a high-level language and executes them line by line .

1.1. Data Types and Operators

Like other programming languages, Python also has several unique data types. Some of the data types that exist in Python and are generally written as follows:

a. Boolean, to determine the value True (True) and False (False) or can also be written as 0 and 1
b. String, a collection of data consisting of sentences. Usually marked ' and ', or flanked by “and“, or between “”” and “”” to form paragraphs.
c. Object, a data set containing attributes and methods 1
d. And others

The operators in python are almost the same as operators in other programming languages such as arithmetic operators (Arithmetic Operators) or mathematical calculations, assignments (Assignment Operators), binary (Bitwiser Operators), logic (Logical Operators), membership (Membership Operators), comparison (Comparison Operator) and identity (Identity Operator) [6].

1.2. Library

Libraries in Python are additional program code that is used for certain needs. Python has more than 140,000 libraries that have been developed through opensource projects [7]. Depending on the need, well-known ones like Numpy for data scientists dealing with numbers, Pandas which functions to process data such as join, distinct, group by, aggregation techniques, and other techniques such as SQL, and Pygame to create games [8].

1.3. Scripting

Script in python is a file with a line of code in the python language that has a .py extension. This file will be read and executed by the python interpreter line by line [9]. Python scripts can be written using any text editor. The easiest example of this text editor is using Notepad (Windows), or Nano (Linux). However, to make programming easier, it is usually recommended to write python scripts with a Code Editor such as Pycharm, VS Code, Sublime Text, or Atom.

Python Script Example

```
print("Hello world")
name = "Adi"
age = 18
print("My name", name, "my age", age, "years")
```

1.4. Object Oriented Programming (OOP)

Object Oriented Programming (OOP) or object-oriented programming is a programming technique in which everything in the program is modeled like objects that exist in the real world. have characteristics or attributes as well as actions or behavior (behavior) [10]. For Python itself, initially using OOP but still using the old programming paradigm, namely structured programming. Therefore, Python is also known as a hybrid.

Example of class creation syntax in Python.

```
classClassName:
'''class docstring'''
class_body
```

A Class may have a documentation string or a docstring or not because this is optional. And this documentation string can be accessed using the ClassName format [11]. The class_body contains a statement in the form of all the attributes, functions, and data of the class.

Simple class example:

```
Book class:
'''Basic class of all books''
book_number= 0

def __init__(self, title, year_issue):
self.title = title
self.publish_year = published_year
Books.book_number += 1

def show_number(self):
print("Total Books:", Books.book_number)

def show_detail(self):
print("Title:", self.title)
print("Year of publication:", self.year_published)
print() [4]
```

1.5. IDE (Integrated Development Environment)

IDE (Integrated Development Environment) is a computer program that is commonly used by developers to build applications or websites. Inside the IDE consists of from a variety of different tools according to their functions, IDE usually also contains a text editor for writing code, tools for compilation and a debugger that shows errors in program code. For pyrhon itself, there are several IDEs that can be used, one of which is pycharm. The existence of this IDE is an important factor in its competition because it includes productive tools such as quick fixes [12]. There are three versions of the IDE available, the first for the Apache licensed Community, the Education version, and the Professional version. The first two versions are open source and can be used for free, while the Professional version is a paid version. The features of each version may vary depending on which version is used but for the community version of the IDE this might be very interesting because it has different features such as syntax highlighting, auto-completion and live code verification [13]. The paid version itself also has more complete and advanced features such as database management, frameworks that may be better than the community version such as Django, Flask, Google App, Engine, Pyramid, and web2py.

Advantages:

a. Actively supportive user community.
b. Syntax correction and Code verification
c. Ease of editing and debugging Python code

Deficiency:

a. Application loading is slow when run
b. Requires some adjustments before the existing project can be used [5].

2. RESEARCH METHOD

The research model carried out is a Research and Development model where this research method goes through a series of processes to develop a product or improve an existing product [14].

The product in question can be in the form of learning modules, books, computer applications and games, big data and others.

3 methods are often used in conducting research and development:

a. Descriptive Method : to collect initial data about a product.
b. Evaluative Method : conduct a series of trials on the product.
c. Experimental Method : testing the durability and the extent of the usability of the product that has been made.

In this case, the purpose of this research is to introduce the python programming language as a tool that can also be used to create games in addition to the other uses mentioned above [17].

3. RESEARCH RESULTS AND DISCUSSION

3.1 Hardware Specifications

a. Microsoft Windows 10, 8 64-bit
b. Minimum 2 GB RAM, Recommendation 8 GB RAM
c. Minimum 2.5 GB HD, SSD is better
d. Python version 2.7, Python 3.5 or later

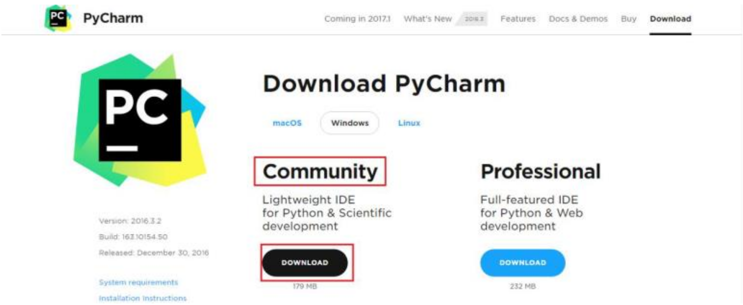
3.2 Software used

PyCharm (Python IDE), Microsoft Windows 10, 64 bit, can also be on linux.

3.3 Results

3.3.1 Installing PyCharm (Python IDE)

Download PyChram on the official website www.jetbrains.com/pycharm/download/#section=windows



Download PyCharm

Community

Lightweight IDE for Python & Scientific development

179 MB

Professional

Full-featured IDE for Python & Web development

232 MB

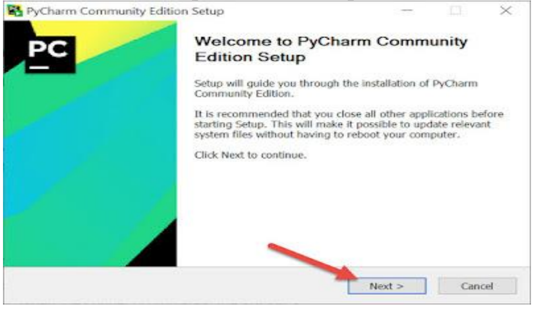
Version: 2016.3.2

Build: 163.10154.50

Released: December 30, 2016

Figure 3. Download Pycharm on the official website

Here, the community version is free, but for the professional version, we need to buy a license. Run the PyCharm Installer, then click Next to continue the installation as shown in the example image 4 below. Then follow the procedure until the installation process is complete.



Welcome to PyCharm Community Edition Setup

Setup will guide you through the installation of PyCharm Community Edition.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 4. Pycharm Installation Process

3.3.2 Starting Design

The design of this game is only to make game object animations or what is commonly called a player, this player uses animated sprites of birds. Figure 5. Sprite is a collection of images with certain movements that can be in the form of any character or movement arranged in such a way that they can produce animated images [15]. In Sprites there are 2 main components, namely:

a. Frame The

frame referred to here is usually defined as a worksheet (worksheet) or more simply, the number of images in an animation or video. Making sprites need to pay attention to the number of frames. Because the number of sprite frames will affect the smooth movement of the animation.

b. Duration

Duration is the length of time the sprite animates, the unit of duration is usually seconds or seconds. The amount of duration will affect the movement of the animation, for example, an animation with a number of frames of 30 with a duration per second (1 second) will be different from an animation with a number of frames of 30 and a duration of 2 seconds. Sprite animation with a duration of 3 seconds will look slower than the duration of 2 seconds.

In addition to player, there is also what is called opposite, this term is usually used as the opposite of the player object in game play [16]. Opposite here is also made using animated sprites.

For the player object, the keyboard is controlled using up, down, left and right arrows. The movement starts from the left to the right of the screen. The opposite object, the movement is from the right to the left of the layer and will appear randomly.

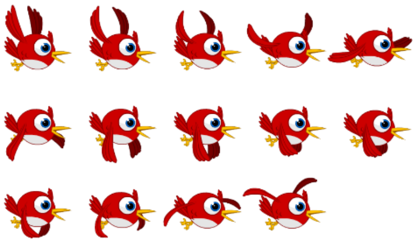


Figure 5. Sprites Animated Player

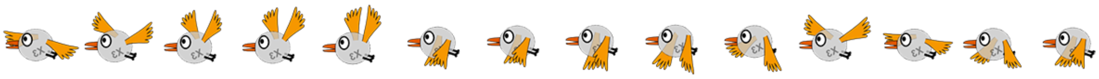


Figure 6. Opposite Animated Sprites

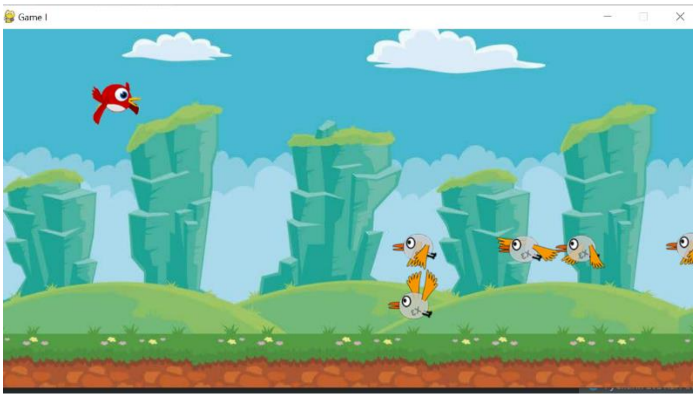


Figure 7. The Final Result of Making Simple Game Animations

Each of the animated sprites will be cut into a single file with a .png extension using an image editor application, such as Adobe Photoshop so that the results are like this:

player_1.png *player_2.png*

Figure 8. Animated sprites when they have been cut into one image object

For the player there are about 14 image objects that will be animated, and for the opposite there are 14 image objects that will also be animated.

3.3.3 Coding

To make coding, you must first open the python IDE that was installed previously, namely pycharm, after that create a new project.

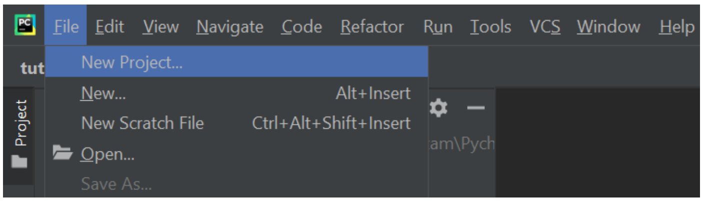
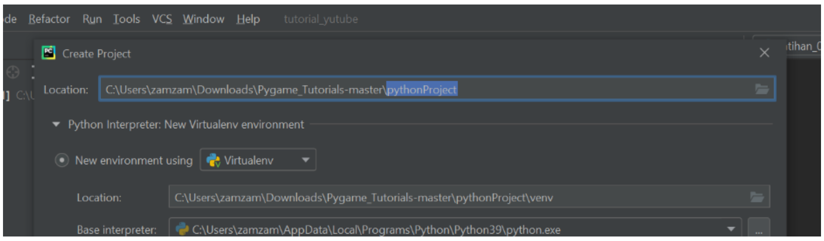

File Edit View Navigate Code Refactor Run Tools VCS Window Help
tut
Project
New Project...
New... Alt+Insert
New Scratch File Ctrl+Alt+Shift+Insert
Open...
Save As...
am\Pych

Figure 9. Creating a new project in pycharm



Location: C:\Users\zamzam\Downloads\Pygame_Tutorials-master\pythonProject

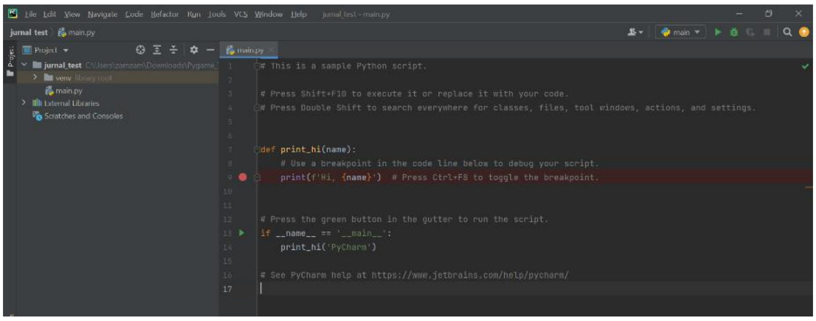
Python Interpreter: New Virtualenv environment

● New environment using Virtualenv

Location: C:\Users\zamzam\Downloads\Pygame_Tutorials-master\pythonProject\venv

Base interpreter: C:\Users\zamzam\AppData\Local\Programs\Python\Python39\python.exe

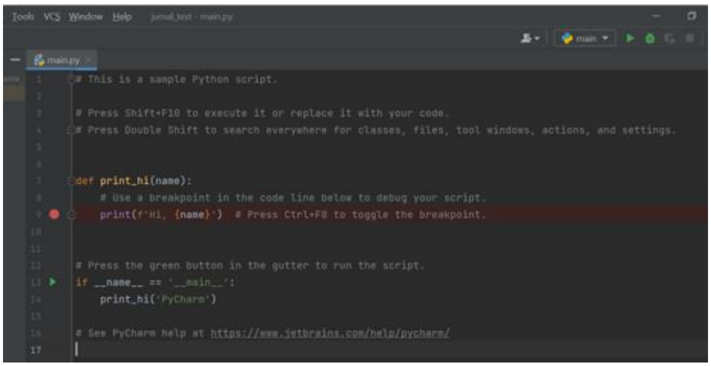
Figure 10. New project form , specify the name of the project and the location of the folder


File Edit View Navigate Code Refactor Run Tools VCS Window Help jurnal_test - main.py
jurnal test › main.py
Project
jurnal_test C:\Users\zamzam\Downloads\Pygame_
venv library root
main.py
External Libraries
Scratches and Consoles
main.py
main

```
# This is a sample Python script.

# Press Shift+F10 to execute it or replace it with your code.
# Press Double Shift to search everywhere for classes, files, tool windows, actions, and settings.


def print_hi(name):
    # Use a breakpoint in the code line below to debug your script.
    print(f'Hi, {name}')  # Press Ctrl+F8 to toggle the breakpoint.


# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    print_hi('PyCharm')

# See PyCharm help at https://www.jetbrains.com/help/pycharm/
```

Figure 11. Initial view after successfully creating a new project

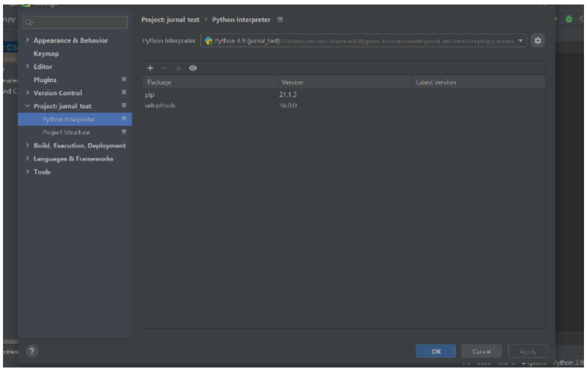
After creating and naming a new project, then create a file to write coding and run it later or we can also use the main.py file which by default will open when you first create a project.



```
# This is a sample Python script.

# Press Shift+F10 to execute it or replace it with your code.
# Press Double Shift to search everywhere for classes, files, tool windows, actions, and settings.


def print_hi(name):
    # Use a breakpoint in the code line below to debug your script.
    print(f'Hi, {name}')  # Press Ctrl+F8 to toggle the breakpoint.


# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    print_hi('PyCharm')

# See PyCharm help at https://www.jetbrains.com/help/pycharm/
```

Figure 12. Main.py . file

main.py file can be empty first by deleting its contents. Once empty, the file is ready to be filled with the coding. Next, import the pygame library, by clicking on the File-Settings menu, a display like this appears:



Python Interpreter: Python 3.9 (jurnal_test)

Package	Version	Latest version
pip	21.1.3	
setuptools	56.0.0	

Figure 13. How to import the pygame library

Select the (+) sign, then find the package by typing pygame, then click Install Package. To install this package we must use an internet connection.

Create an image folder to place the animated sprite images that have been split into .png image files earlier. The trick is to click File-New-Directory, write the name of the directory, for example an image.

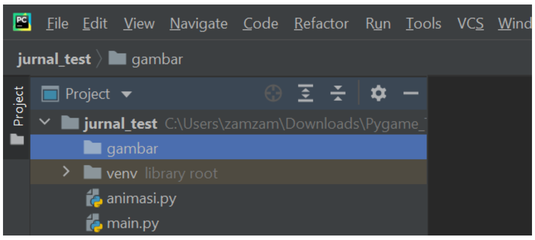

File Edit View Navigate Code Refactor Run Tools VCS Wind
jurnal_test 〉 gambar
Project
jurnal_test C:\Users\zamzam\Downloads\Pygame_
gambar
venv library root
animasi.py
main.py

Figure 14. Image directory

The complete code is like this:

```
import pygame

import random

#defenisikan variabel screen

SIZE = WIDTH, HEIGHT = 900, 460

#warna background screen

BACKGROUND_COLOR = pygame.Color('white')

#Frames per second

FPS = 15

screen = pygame.display.set_mode(SIZE)

pygame.display.set_caption("Game I")


#class object

class Player(pygame.sprite.Sprite):

    def __init__(self):

        super(Player, self).__init__()

        self.images = []

        self.images.append(pygame.image.load('gambar/player_01.png'))

        self.images.append(pygame.image.load('gambar/player_02.png'))

        self.images.append(pygame.image.load('gambar/player_03.png'))

        self.images.append(pygame.image.load('gambar/player_04.png'))

        self.images.append(pygame.image.load('gambar/player_05.png'))

        self.images.append(pygame.image.load('gambar/player_07.png'))

        self.images.append(pygame.image.load('gambar/player_08.png'))

        self.images.append(pygame.image.load('gambar/player_09.png'))

        self.images.append(pygame.image.load('gambar/player_10.png'))

        self.images.append(pygame.image.load('gambar/player_11.png'))

        self.images.append(pygame.image.load('gambar/player_12.png'))

        self.images.append(pygame.image.load('gambar/player_13.png'))
        self.images.append(pygame.image.load('gambar/player_14.png'))
        self.index = 0
        self.img = self.images[self.index]
        self.rect = self.img.get_rect()
        self.rect.topleft = (0,50)
        self.speed = 20

    def move(self):
        self.rect.left += 0
    def moveUp(self):
        if self.rect.top <= 0:
            self.rect.top = 0
        else:
            self.rect.top -= self.speed
    def moveDown(self):
        if self.rect.top >= (HEIGHT) - self.rect.height:
            self.rect.top = (HEIGHT) - self.rect.height
        else:
            self.rect.top += self.speed
    def moveLeft(self):
        if self.rect.left <= 0:
            self.rect.left = 0
        else:
            self.rect.left -= self.speed
    def moveRight(self):
        if self.rect.left >= WIDTH - self.rect.width:
            self.rect.left = WIDTH - self.rect.width
        else:
            self.rect.left += self.speed
    def update(self):
        self.index += 1
        if self.index >= len(self.images):
            self.index = 1
        self.image = self.images[self.index]

class Opposite(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.images = []
    def update(self):
        self.index += 1
        if self.index >= len(self.images):
            self.index = 1
        self.image = self.images[self.index]


class Opposite(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.images = []
        self.images.append(pygame.image.load('gambar/opposite_01.png'))
        self.images.append(pygame.image.load('gambar/opposite_02.png'))
        self.images.append(pygame.image.load('gambar/opposite_03.png'))
        self.images.append(pygame.image.load('gambar/opposite_04.png'))
        self.images.append(pygame.image.load('gambar/opposite_05.png'))
        self.images.append(pygame.image.load('gambar/opposite_06.png'))
        self.images.append(pygame.image.load('gambar/opposite_07.png'))
        self.images.append(pygame.image.load('gambar/opposite_08.png'))
        self.images.append(pygame.image.load('gambar/opposite_09.png'))
        self.images.append(pygame.image.load('gambar/opposite_10.png'))
        self.images.append(pygame.image.load('gambar/opposite_11.png'))
        self.images.append(pygame.image.load('gambar/opposite_12.png'))
        self.images.append(pygame.image.load('gambar/opposite_13.png'))
        self.images.append(pygame.image.load('gambar/opposite_14.png'))
        self.index = 0
        self.image = self.images[self.index]
        self.rect = self.image.get_rect()
        self.rect.left = WIDTH + self.rect.width
        self.birds_height = [HEIGHT * 0.32, HEIGHT * 0.75, HEIGHT * 0.60, H
30]
        self.rect.centery = self.birds_height[random.randrange(0, 3)]
        self.speed = random.randrange(2, 8)

        self.rect.centery = self.birds_height[random.randrange(0, 3)]
        self.speed = random.randrange(2, 8)


    def update(self):
        self.index += 1
        if self.index >= len(self.images):
            self.index = 0
        self.image = self.images[random.randrange(0, 14)]
        self.rect.left -= self.speed
        if self.rect.right < 0:
            self.rect.left = WIDTH + self.rect.width
            self.birds_height = [HEIGHT * 0.32, HEIGHT * 0.75, HEIGHT * 0.6
HEIGHT - 30]
            self.rect.centery = self.birds_height[random.randrange(0, 3)]


def main():
    pygame.init()
    bck = pygame.image.load("gambar/background.jpg").convert()
    x = 0
```

```
def main():
    pygame.init()
    bck = pygame.image.load("gambar/background.jpg").convert()
    x = 0
    my_player = Player()
    my_playergroup = pygame.sprite.Group()
    my_birdgroup.add(my_bird)
    for i in range(5):
        m = Opposite()

      my_playergroup.add(m)
  clock = pygame.time.Clock()
  while True:
      for event in pygame.event.get():
          if event.type == pygame.QUIT:
              pygame.quit()
              quit()
      keys = pygame.key.get_pressed()
      if keys[pygame.K_LEFT] :
          print("Kiri")
          my_player.moveLeft()

      if keys[pygame.K_RIGHT]:
          print("Kanan")
          my_player.moveRight()
          x -= 5
          if keys[pygame.K_UP]:
              print("Atas")
              my_player.moveUp()
          if keys[pygame.K_DOWN]:
              print("Bawah")
              my_player.moveDown()

          rel_x = x % bck.get_rect().width
          screen.blit(bck, (rel_x - bck.get_rect().width, 0))
          if rel_x < WIDTH:
              screen.blit(bck, (rel_x, 0))
          x -= 0
          my_playergroup.update()
          my_playergroup.draw(screen)
          pygame.display.update()
          clock.tick(10)

  if __name__ == '__main__':
      main()
```

4. CONCLUSION

This writing is only intended for those who want to learn python in a different perspective, namely making basic game animations apart from the many functions of the python programming language . For writing code the author tries to be as simple as possible so that programmers who use python can easily understand the context. The purpose of this application is as an example of how to make simple animations using the pygame library in the python programming language. As for the future, this game application will be developed using a simulation of neural network algorithms or artificial intelligence.

REFERENCES

[1] Ridwan Fadjar , 2013. Learn Basic Python Programming , West Java, Indonesia

[2] Belajarpython , Python . viewed August 12, 2021, https:// learningp ython.com/tutorial/apa-itu-python

[3] Hendri, 2003, Quickly Proficient in Python , IlmuKomputer.com.

[4] Pythonindo, 2019, Object Oriented Programming in Python. viewed August 18, 2021, https://www.pythonindo.com/pemrograman-orientasi-objek-di-python/

[5] WebApp, 2021, List of 10 Best Python IDEs and Code Editors of the Year . viewed August 10, 2021 , https://appkey.id/pembuatan-application/mobile-programming/python-ide

[6] Youtube, 2021, Learn Python by Building Five Games - Full Course . view: 01 August 2021, https://appkey.id/pembuatan-application/mobile-programming/python-ide

[7] Askpython, 2021, Easy Games in Python, viewed 05 July 2021, https://askpython.com/python/examples/easy-games-in-python

[8] Pygame, 2021, GettingStarted , viewed August 10, 2021, https://pyga me.org/

[9] Codepolitan, 2021, Learn How to Make 2D Games with Python and Arcade, viewed June 14, 2021, https://www.codepolitan.com/learning-how-to-make-game-2d-with-python-dan-arcade-5adff78d1db9b/

[10] Abdul Kadir, 2019, Easy Steps to OpenCV & Python Programming, Jakarta, Elex Media Komputindo

[11] Romzi, Muhammad & Kurniawan, Budi 2020, 'Python Programming Learning With Algorithmic Logic Approach', Journal of Informatics Engineering Mahakarya , vol. 3, no. 2, hh. 37-44.

[12] Ramsari, Nopi & Ramadhan, Gilang 2018, 'The Naila's Survival 2d Side Scrolling Game Based on Android', FIKI Journal of Information and Communication Technology, vol. 8, no. 2.

[13] Kevin & Amelia Ananda 2017, 'Macromedia Flash-based English Game Application using the waterfall method', Indonesia: Simki-Techsain , Vol 01 No 07.

[14] Martono & Teguh, Kurniawan 2015, 'Game Development Using Game Engine Game Maker', Indonesia: Journal of Computer Systems (JSISKOM), vol 5 No 1.

[15] Mahardhika & Putra, Alfian. 2015, 'Developing 2D Adventure Android Game MR Potato Using Unity' . Indonesia: Go Infotech Scientific Journal, Volume 21 No.2.

[16] Mestadi, Walid et al, 2018, 'An Assessment of Serious Games Technology: Toward an Architecture for Serious Games Design' . Morocco: Hindawi, International Journal of Computer Game Technology, Volume 2018, Articel ID 9834565.

[17] Adi, Munadi 2014, 'Development of an Educational Game Knowing the Heroes of the Indonesian Nation's Heroes'. Indonesia: Scientific and Computer Journal (KOMPUTA), ISSN 2089-9033.

[18] Putri, Bellia Dwi Cahya 2016, 'Design of Flash-Based Gobak Sodor Multiplayer Game Applications in Local Networks. Journal of Technology and Computer Systems (JTSiskom), Vol 4 No.2.

[19] Soleh, Rachmat 2018.' User Experience Analysis of Multiplayer Online Battle Arena (Moba) Game Using Game Experience Questionnaire (GEQ) on Game DOTA 2', Journal of Information Technology and Computer Science Development, Vol 2 No.9 ISSN:3067-3076.

[20] Utami, Reny, Purnama Bambang Eka & Sukadi, 'Creating a Multiplayer Dakon Traditional Game using Java', Indonesian Journal on Computer Science (IJCSS), ISSN: 1979-9390.