Journal of Advanced Research in Computer Graphics and Multimedia Technology
Volume 5, Print Issue 1 - 2023, Pg. No. 22-26
Peer Reviewed Journal

Research Article

Snake Game Using Python: Educational Project

Aakansha Rani[1], Shivasheesh Kumar Gupta[2]

[1,2]Chandigarh University, Gharuan Mohali, india.

INFO

Corresponding Author:
Aakansha Rani, Chandigarh University, Gharuan Mohali, India.
E-mail Id:
19BCS2098@gmail.com

How to cite this article:
Rani A, Gupta SK. Snake Game Using Python: Educational Project. *J Adv Res Comp Graph Multim Tech.* 2023; 5(1): 22-26.

Date of Submission: 2023-03-25
Date of Acceptance: 2023-04-28

ABSTRACT

Python is a high level programming language that is frequently contrasted with others like Java, Java script, PERL. Python programming has a number of advantages, including open source, productivity, speed. High level programming languages like Python are quite popular and emphasise code readability in their architecture. Additionally, a syntax that enables programmers to express concepts in significantly less lines of code than in languages like C++ or Java. Its standard library is extensive and complete. Pygame, PyFltk, other GUI models are a few examples.

Keywords: Python, programming language, C++ , Java.

Introduction

Python is a popular general-purpose high-level programming language. Python has an advantage over other programming languages due to its wide range of applications. There are a tonne of GUI frameworks for Python, including Tkinker, many additional platform-specific solutions, binding to platform-specific technologies.

Python supports a variety of programming languages, including imperative, functional, object-oriented programming, it teaches a dynamic type system. It has a sizable, thorough library that contains a variety of modules. The language comes with elements that should make it possible to write concise programmes on both a small and large scale.

GUI vs. CLI

Python has a large number of modules, some of which are graphical and others of which are console based. Text is handled by CLI programmes using the print() and input() functions. These programmes allow users to type text using the keyboard while text is displayed on the screen.

The input() function in the CLI programme is used to input data from the keyboard. Since real-time game application code should run continuously without waiting for user input, action games cannot be created in CLI. However, Python also has a few modules that let you build programmes with a graphical user interface. A window with images and colour will be displayed by GUI programmes, which are suitable for creating games as well.

Applications of Python

Some of the fields where we can implement python GUI based application are:

- Games
- Scientific applications
- Web frame works and applications - Business and enterprise application - Image processing application.
- Graphic design applications
- Language development
- Prototyping

Games Using Python

Python is an extremely capable development environment for game applications. It offers a lot of auxiliary modules for creating game-related applications. Python has a variety of ports for every PC and gaming consoles, allowing you to create games on one platform and run them on another. Python is now platform independent as a result.

Literature Survey

In order to use the image processing library, this paper explains how to develop an extendable GUI (Graphical User Interface) using the Python module and Python's Tkinter.

The platform is referred to as GNU-Vu. It uses a variety of computer languages and is cross-platform for processing images. Python, C++, the Tk module are all used in the project. The GUI is created using Tk, the image processing algorithm is implemented using C++, the GUI and image processing libraries are combined using Python. Utilising Tkinter has the advantage of simplifying GUI programme development.[1]

Using the SimPy module, the author adds a new layer to discrete event simulation (DES). A Simulation Application can be built using the SimPy module as a foundation. Manufacturing in Python (ManPy) is the name of the new simulation engine. The endeavour is a component of the "Decision Support in Real-Time for Efficient Agile Manufacturing" (DREAM) project. The project's goal is to create a semantic simulation platform that is open-source and free.[2]

PyGame in the Python 3.4 environment was used to imitate the scenario. To transport packets from any source node to a chosen destination node, the programme simulates a cloud network virtualization platform. The simulations show that the packet chooses the best route between the source and destination nodes.[3]

The purpose of this work is to develop a Python programme that can automatically translate Python into traditional Chinese in order to assist those who wish to learn computer programming but are unable to do so due to a lack of understanding of or inability to understand English. This programme makes use of a Tk-Based GUI programme. The project was primarily developed with those nations in mind, where people have numerous challenges and English is not the official language. Any Python programme can be translated, edited, viewed in Turtle Graphics using the GUI programme.[4]

The paper discusses how to connect a web server to a plant parameter so that we may constantly track the status of the processing plant from various locations by multiple supervisors. And the Internet of Things and Python are used for this. In order to control the temperature parameter at any desired value, the author here also accounts for it. Data will be delivered to the supervisor's personal computer so that he can use the Graphical User Interface to examine and keep track of the status of plant parameters.[5]

The Python-based VISION/HPC system. visual programming interface with drag and drop. It simplifies difficult programming operations like dragging and connecting GUI flow chart icons. Numpy, an acronym for numerical array and linear algebra, SciPy, matplotli, the Python imaging library, IPython, an acronym for interactive parallel computing, are all included in this open-source Python package that runs on Windows.[6]

This section introduces the "Geant4Py" Geant4-Python interface, which offers a collection of Python modules for using Geant4 on Python. Additionally, it demonstrates a variety of Geant4Py-based applications, including compile-free scripts, online histogramming analysis with ROOT, Web applications, GUI applications, tools for physics validation, as well as uses for educational and medical simulation. It also discusses runtime performance, which can be adjusted between execution speed and interactivity for each use-case.[7]

We use the PyDas module to integrate the various components of the SNS DAS system. It makes it possible to quickly and easily automate user-defined neutron scatter studies. It offers both IPython command line scripting and the wxPython GUI for doing common experiments. The data are presented and basic analysis is done using Numpy and matplotlib. Along with information on PyDas architectures and implementations, the publication provides statistics on data acquisition systems.[8] The paper explains how the enhanced educational tool known as aura, which was created using Python and PyQt Python bindings, was designed, programmed, applied. The article introduces a novel idea: the use of a single tool that can connect the syntaxes of diverse programming languages and algorithm architectures. Since students can relate the programming languages, it also vastly improves their capacity for comprehension and memory. Students who are just starting out with the languages are not likely to benefit from the software. The application helps students learn about understanding algorithms, which helps them learn several languages utilising an interactive GUI on a single platform. The paper shows how to use Python and PyQt bindings to create a fully functional application with rich features that implements innovative algorithm building techniques. This application includes a browser, a combination of programming platforms, a code generator, a real-time code sharing hub that is built into a single interface. This also includes research on how Python functions as a building tool. Python requires a lot fewer lines of code than feature-rich coding languages while maintaining the stability, interoperability, strength of the programme.[9]

Propesed System

In this paper, we suggest building a traditional snake game with Python's Pygame GUI module and Turtle to evaluate both programmes' functionality.

The turtle module gives us visuals that are both object- and procedure-oriented. Tkinter is used by Turtle for its base graphics. To do that, we need a copy of Python that has Tk support loaded on it. The turtle module offers commands that allow users to manage a turtle's movement forward and backward, select the type of pen it is holding, more.

We can make designs out of the tracks the turtle leaves by adjusting its position, orientation, the pen it is holding.

The "simple direct media layer" (SDL) is the foundation of Pygame, unlike the other Python game creation package. SDL is a cross-platform library that offers direct 3D and openGL low level access to the graphics hardware, keyboard, mouse, joystick events. The majority of platforms, including Windows, Linux, Mac OS X, Android, are supported by SDL. SDL is developed in C, with active development occurring in C++, C#, Python, Java.

A significant trend in game creation over the past few years has prompted us to switch to high level programming languages. The game engine, which helps the game run as quickly as possible, the game logic, which is in charge of instructing the game engine to perform an action, are often the two main components of a game. A great C engine for creating 2D games is Pygame and SDL.

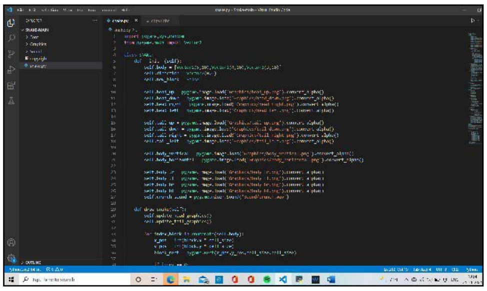
Implementation and Analysis

Using the Pygame module, we have constructed the well-known snake game in Python. Playing the game is really simple. It will be played by a single player, his task will be to direct the snake as it moves continuously inside a field of square cells. The snake has the ability to travel in any one of the four directions—left, right, up, down—at any one moment. Using the four arrow keys on the keyboard, the player can modify the snake's direction. The snake will die and the game will be ended if it collides with any of the sides or with itself. Additionally, a background sound will play as it expires.

A single food item will be randomly positioned in the frame. The food will be consumed each time the snake hits it with its head, lengthening the snake by one cell. A fresh piece of food will be manufactured in a randomly selected cell in the field that was empty before the meal was consumed. Also, a background music will play everytime the snake consumes the meal once more. A built-in module in PyGame allows you to add sound to any game.

How to Program?

The simplest way to organise a game's code is to separate it into six pieces [Figure 1].



```
import pygame,sys,random
from pygame.math import Vector2

class SNAKE:
    def __init__(self):
        self.body = [Vector2(5,10),Vector2(4,10),Vector2(3,10)]
        self.direction = Vector2(0,0)
        self.new_block = False

        self.head_up = pygame.image.load('Graphics/head_up.png').convert_alpha()
        self.head_down = pygame.image.load('Graphics/head_down.png').convert_alpha()
        self.head_right = pygame.image.load('Graphics/head_right.png').convert_alpha()
        self.head_left = pygame.image.load('Graphics/head_left.png').convert_alpha()

        self.tail_up = pygame.image.load('Graphics/tail_up.png').convert_alpha()
        self.tail_down = pygame.image.load('Graphics/tail_down.png').convert_alpha()
        self.tail_right = pygame.image.load('Graphics/tail_right.png').convert_alpha()
        self.tail_left = pygame.image.load('Graphics/tail_left.png').convert_alpha()

        self.body_vertical = pygame.image.load('Graphics/body_vertical.png').convert_alpha()
        self.body_horizontal = pygame.image.load('Graphics/body_horizontal.png').convert_alpha()

        self.body_tr = pygame.image.load('Graphics/body_tr.png').convert_alpha()
        self.body_tl = pygame.image.load('Graphics/body_tl.png').convert_alpha()
        self.body_br = pygame.image.load('Graphics/body_br.png').convert_alpha()
        self.body_bl = pygame.image.load('Graphics/body_bl.png').convert_alpha()
        self.crunch_sound = pygame.mixer.Sound('Sound/crunch.wav')

    def draw_snake(self):
        self.update_head_graphics()
        self.update_tail_graphics()

        for index,block in enumerate(self.body):
            x_pos = int(block.x * cell_size)
            y_pos = int(block.y * cell_size)
            block_rect = pygame.Rect(x_pos,y_pos,cell_size,cell_size)
```

Figure 1

Importing the local Python game names and the Python game module is necessary when loading a module.

Resource handling classes: These classes deal with the most fundamental resources that are used the most. Examples include loading graphics, playing audio, connecting and disconnecting from the network, loading game save files, loading any other resources that may be available.

The classes that you will utilise for your game objects are defined by the game object classes.

Any other game functionalities that may be required, such as keyboard movement, snake movement, food generation, etc. Any code that you include in the main game logic but which makes it more difficult for players to understand should always remain in its own routines.

Initialise the game: The pygame objects, the background, the game objects, any additional code you want to include in your programme must all be initialised.

The main loop is where we'll handle any input that comes in via the arrow buttons or keyboard. Additionally, we will add code to update the game's components and, eventually, its screen.

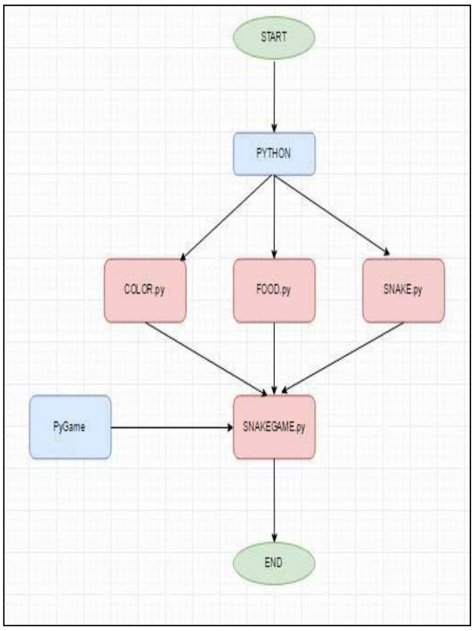


Figure 2.Flow chart of Snake game

Analysis

Silliness is ingrained: Pygame is designed to make using software enjoyable. Every 3.1415 seconds, something silly is added.

Core functionalities are implemented using C and assembly code. Python uses C because it is 10–20 times faster than Python code for essential functions. - Simple and easy to use: Pygame features more than 20 submodules that make it simple to programme games. Since practically every function in Python is already built-in, anyone can create games with Pygame.

Pygame includes modules such as does not need a GUI to perform all of its functions: Pygame can be used without a monitor if we wish to process images, receive joystick input, or play sounds.

- Minimal coding required: Pygame does not demand that we write hundreds of thousands of lines of code for features that we might not even use or may not use all that frequently. Pygame's main goal is to make programmes simple while allowing for easy addition of extras like GUI libraries and other effects.

Our investigation has led us to the conclusion that while pygame is not the finest game library, it is a wonderful choice for a beginning because it is simple to use and comprehend.

Screen Shots

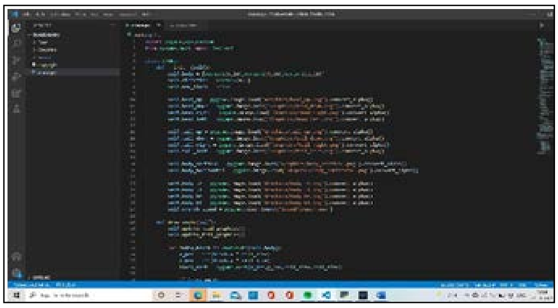


Figure 3

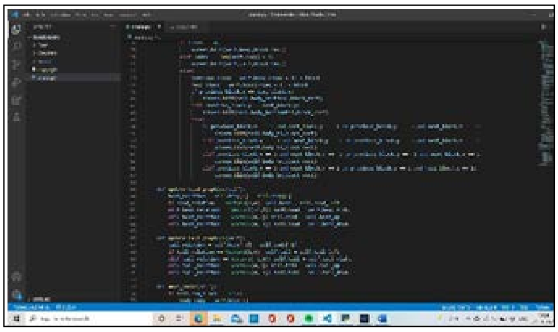


Figure 4

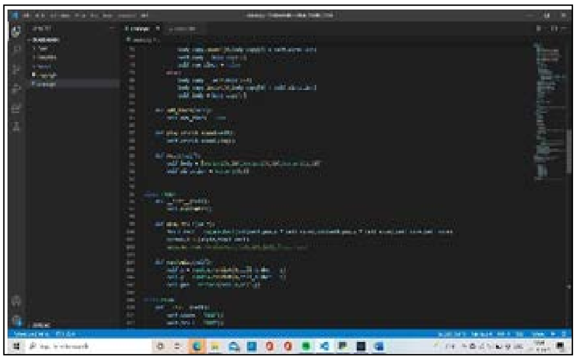


Figure 5

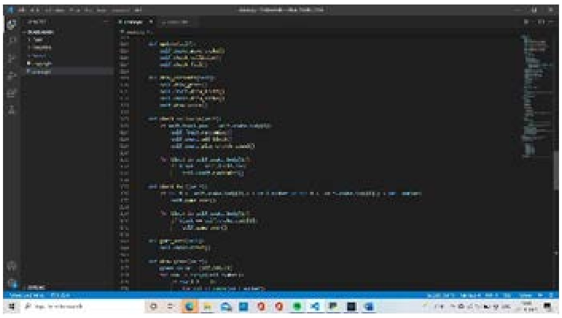


Figure 6

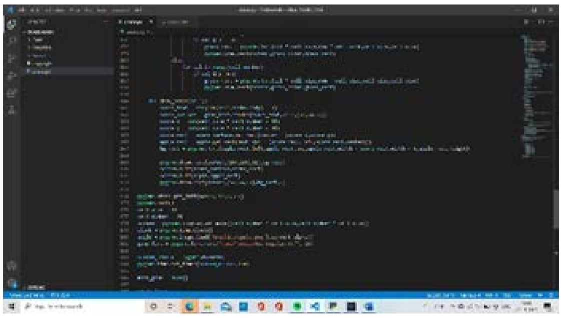


Figure 7

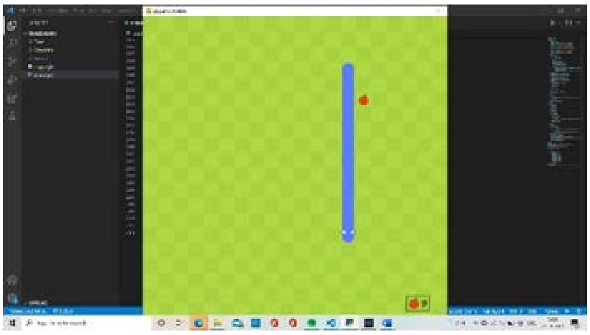


Figure 8

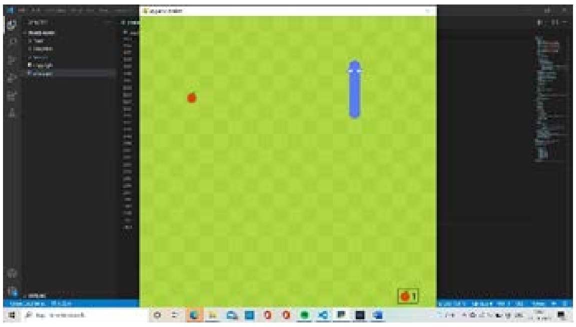


Figure 9

Conclusion

With the help of the Python Pygame module, we were succeeded in producing a miniature version of the old-fashioned snake game. The user-controlled version of the age-old snake game has been created and tested satisfactorily. Using Pygame, a Python GUI tool, we studied several project management approaches to create tiny projects. Being able to see and interact with the code you have written is incredibly thrilling and satisfying in today's new age of technical advancement when creating games. Pygame is not an ideal tool for creating a high-level game, but it is unquestionably an excellent module for creating games of a smaller size.

References

Media Attributes

1. Python documentation. http://www.python.org/doc.
2. Pygame documentation. https://www.pygame.org/docs/
3. Game Samples. 4.https://www.reddit.com/r beginnerprojects/comments/4et6xh/made_a_tictctoe_game with_graphical_interface_and/s
4. www.geeksforgeek.org
5. www.eduíeka.co