

Разработка игры «Крестики-нолики» на Python

Акентьев Данила Денисович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В данной статье представлена разработка классической игры "Крестики-нолики" с использованием языка программирования Python и графической библиотеки tkinter. Игровой процесс основан на стандартных правилах игры для двух участников, и включает проверку победных комбинаций и возможность перезапуска игры. Статья рассматривает аспекты создания графического интерфейса, взаимодействия с пользователем и использования событийных обработчиков.

Ключевые слова: Python, tkinter, игра, крестики-нолики, графический интерфейс, обработчики событий, разработка игр.

Developing a Tic-Tac-Toe game in Python

Akentev Danila Denisovich

Sholom-Aleichem Priamursky State University

Student

Abstract

This article presents the development of the classic Tic-Tac-Toe game using the Python programming language and the tkinter graphics library. The gameplay is based on the standard rules of the game for two participants, and includes checking winning combinations and the ability to restart the game. The article examines the aspects of creating a graphical interface, interacting with the user, and using event handlers.

Keywords: Python, tkinter, game, tic-tac-toe, graphical interface, event handlers, game development.

1 Введение

1.1 Актуальность

С развитием технологий программирования и появлением новых образовательных платформ, изучение программирования через создание простых игр стало популярным и эффективным методом обучения. Разработка игры "Крестики-нолики" на Python с использованием библиотеки tkinter позволяет на практике изучить основные принципы программирования и создания пользовательских интерфейсов. Такой подход может быть полезен как для начинающих разработчиков, так и для преподавателей, стремящихся использовать игры как средство обучения.

1.2 Обзор исследований

В статье Е.О. Новикова представлена разработка программы «Игра морской бой» на языке Python с использованием библиотеки Pygame. В данном исследовании автор описывает алгоритм работы игры, а также дает детальный анализ возможностей Pygame для создания более сложных игр. В частности, внимание уделяется методам, использующим графику и взаимодействие с пользователем, что позволяет создавать динамичные игры с графическим интерфейсом. Поскольку в статье рассматриваются возможности библиотеки для разработки игровых приложений, её выводы могут быть полезными при создании более сложных игр на Python, в том числе тех, которые включают случайные элементы или логику, аналогичную игре "Угадай число" [1].

Другим важным исследованием является работа Д.Ф. Ахметгалиевой и Г.Р. Галиаскаровой, в которой рассматривается создание игры "Змейка" на языке Python с использованием библиотеки tkinter. В статье акцентируется внимание на уникальном подходе к разработке игры, где вместо традиционной змейки использован персонаж Гомер Симпсон, поедающий пончики. Игра включает элементы геймплея, где с каждым съеденным пончиком персонаж удлиняется, а скорость игры увеличивается. Это решение подчеркивает возможности библиотеки tkinter для создания простых, но увлекательных игр с графическим интерфейсом. В работе подробно рассматриваются методы, использованные для реализации игрового процесса, что может быть полезно при создании других простых игр с элементами интерактивности [2].

Ранее вопросы разработки приложений с графическим пользовательским интерфейсом (GUI) с использованием Python и Tkinter привлекали внимание ряда исследователей. Так в своей статье А. С. Матвеева рассматривает применение паттерна «Команда» в создании модульных и структурированных приложений с использованием Tkinter и демонстрирует, как этот паттерн позволяет создавать более гибкие и расширяемые приложения, где каждая кнопка выполняет отдельную команду. В результате такая реализация значительно облегчает процесс разработки и последующей модификации программ, повышая их надежность и удобство использования [3].

Другим важным исследованием является работа А. М. Вохидова, Д.А.Вохидова, Р.Ф. Фармоновой и Д. Ш. Хафизовой. В этой статье рассматривается использование Tkinter и Python в области медицины, где описаны ключевые аспекты создания медицинских приложений с графическим интерфейсом. Исследователи акцентируют внимание на удобстве использования Tkinter для создания интуитивно понятных интерфейсов для медицинских работников и пациентов. В работе подробно представлены примеры практических приложений, использующих эту технологию для создания удобных и доступных инструментов [4].

В статье Д. А. Алигулович, З. И. Махмудова и Р. Ф. Сайфуллаев рассматривают преимущества и недостатки Python и Tkinter для разработки приложений с GUI. Подробно анализируют области применения Tkinter в

педагогической практике, акцентируя внимание на возможности создания кроссплатформенных решений, что делает их идеальными для использования в образовательных целях. В работе подчеркивается важность применения современных педагогических технологий с использованием этого инструмента [5].

1.3 Цель исследования

Целью данного исследования является разработка и анализ игры "Крестики-нолики" с использованием Python и tkinter.

2 Материалы и методы

Для разработки игры использован язык программирования Python и библиотека tkinter, предназначенная для создания графических пользовательских интерфейсов. Методология разработки включала проектирование игрового процесса, создание элементов управления (кнопок для поля игры), а также реализацию алгоритмов проверки победных комбинаций и перезапуска игры. В ходе работы использовались основы объектно-ориентированного программирования для создания структуры игры в виде класса и использования событийных обработчиков для реагирования на действия пользователя.

3 Результаты и обсуждения

Для создания графического интерфейса используется стандартная библиотека tkinter, которая предоставляет удобные инструменты для создания окон, кнопок, меток и обработки событий. Модуль messagebox из этой библиотеки используется для вывода окон с информацией о победе или ничьей (рис.1).

```
import tkinter as tk
from tkinter import messagebox
```

Рисунок 1. Импорт библиотек

Весь функционал игры в классе Game, что позволяет организовать код в структуру, легко поддающуюся расширению и модификации. Конструктор класса (__init__) принимает параметр master, который является главным окном приложения. Далее устанавливается заголовок окна игры с помощью метода title (рис.2).

```
class Game: 1 usage
    def __init__(self, master):
        self.master = master
        self.master.title("Крестики-нолики")
```

Рисунок 2. Код

В следующей части кода создается игровое поле, представленное списком из 9 элементов, каждый из которых изначально пуст (содержит пробел " "). Это поле будет хранить состояние каждой ячейки в игре: "X", "O" или пусто. Переменная `current_player` отвечает за текущего игрока, который делает ход. Изначально первым игроком является "X" (рис.3).

```
self.board = [" " for _ in range(9)] # игровое поле
self.current_player = "X" # первый игрок
```

Рисунок 3. Код

Далее создаются 9 кнопок, которые представляют собой клетки поля игры (рис.4). Для каждой кнопки:

- Устанавливается текст по умолчанию " ", который позже будет изменяться на символ игрока.
- Настроены шрифт и размер кнопки.
- Каждый элемент в списке `buttons` получает привязку к функции `make_move(i)` с уникальным индексом `i` для каждой кнопки. Этот индекс используется для обработки ходов.

```
self.buttons = [tk.Button(master, text=" ", font=("Arial", 24), width=5, height=2,
                          command=lambda i=i: self.make_move(i)) for i in range(9)]
```

Рисунок 4. Код

В дальнейшей части происходит размещение кнопок в окне с использованием метода `grid` (рис.5). Он позволяет расположить кнопки в виде сетки 3x3 (по 3 кнопки в каждой строке), где:

- Индекс строки определяется через `i // 3` (целочисленное деление на 3).
- Индекс столбца вычисляется как `i % 3` (остаток от деления на 3).

```
for i, button in enumerate(self.buttons):
    button.grid(row=i // 3, column=i % 3)
```

Рисунок 5. Код

Метод `make_move` вызывается при каждом клике на кнопку (рис.5). Он выполняет несколько ключевых действий:

- Проверяет, пустая ли клетка, с помощью условия `if self.board[index] == " "`. Если клетка пуста, текущий игрок может сделать ход.
- Обновляет состояние игрового поля, записывая символ текущего игрока в соответствующую ячейку.
- Обновляет текст кнопки, показывая символ игрока на экране.
- Вызывает метод `check_winner`, чтобы проверить, не победил ли один из игроков.

- Если победитель найден, появляется сообщение с информацией о победе, и игра сбрасывается с помощью метода `reset_game`.
- Если игровое поле полностью заполнилось (и нет победителя), появляется сообщение о ничьей.
- Если игра не завершена, меняется текущий игрок с "X" на "O" и наоборот.

```
def make_move(self, index): 1 usage
    if self.board[index] == " ":
        self.board[index] = self.current_player
        self.buttons[index].config(text=self.current_player)
        if self.check_winner():
            messagebox.showinfo(title="Победа!", message=f"Игрок {self.current_player} выиграл!")
            self.reset_game()
        elif " " not in self.board:
            messagebox.showinfo(title="Ничья!", message="Игра закончилась вничью!")
            self.reset_game()
        else:
            self.current_player = "O" if self.current_player == "X" else "X"
```

Рисунок 6. Код

Метод `check_winner` проверяет все возможные выигрышные комбинации на поле (рис.7). Он использует список `winning_combinations`, который содержит индексы клеток для каждой возможной комбинации выигрыша:

- Горизонтальные (например, (0, 1, 2)).
- Вертикальные (например, (0, 3, 6)).
- Диагонали (например, (0, 4, 8)).

Для каждой комбинации проверяется, что все три клетки содержат одинаковые символы (либо "X", либо "O"), и они не пустые.

Если хотя бы одна из комбинаций удовлетворяет этому условию, метод возвращает `True`, что означает победу одного из игроков. В противном случае возвращается `False`.

```
def check_winner(self): 1 usage
    winning_combinations = [
        (0, 1, 2), (3, 4, 5), (6, 7, 8), # горизонтали
        (0, 3, 6), (1, 4, 7), (2, 5, 8), # вертикали
        (0, 4, 8), (2, 4, 6)             # диагонали
    ]
    for combo in winning_combinations:
        if self.board[combo[0]] == self.board[combo[1]] == self.board[combo[2]] != " ":
            return True
    return False
```

Рисунок 7. Код

Метод `reset_game` используется для сброса игры после победы или ничьей (рис.8). Он выполняет следующие действия:

- Очищает игровое поле, снова заполняя его 9 пустыми ячейками.
- Сбрасывает текущего игрока на "X", чтобы начать новый матч.

- Очищает текст всех кнопок, чтобы они снова отображали символы пустоты.

```
def reset_game(self): 2 usages
    self.board = [" " for _ in range(9)]
    self.current_player = "X"
    for button in self.buttons:
        button.config(text=" ")
```

Рисунок 8. Код

В следующей части программы создается основное окно с помощью `tk.Tk()`, в котором будет отображаться интерфейс игры (рис.9). Затем создается объект класса `Game`, который инициализирует игру. В конце запускается главный цикл приложения через метод `mainloop()`, который обеспечивает работу интерфейса и отслеживание событий (кликов по кнопкам).

```
if __name__ == "__main__":
    root = tk.Tk()
    game = Game(root)
    root.mainloop()
```

Рисунок 9. Код

Внешний вид игры выглядит следующим образом (рис.10).

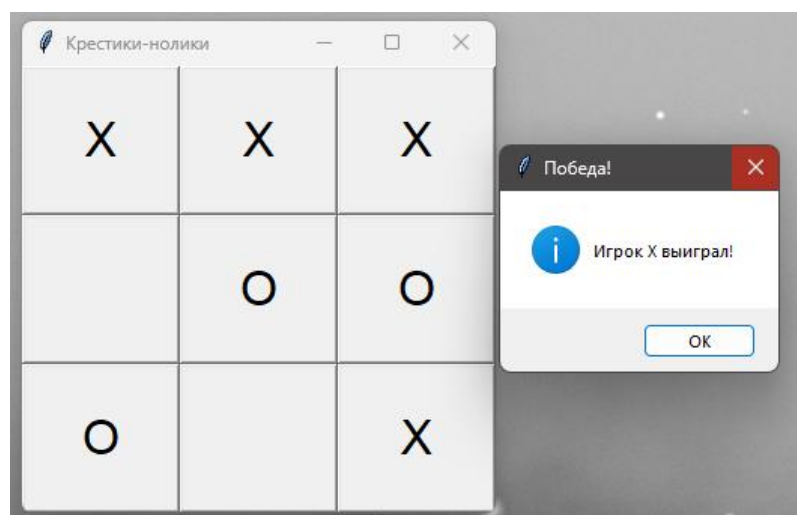


Рисунок 10. Внешний вид

4 Выводы

В ходе работы над проектом была успешно реализована игра "Крестики-нолики" с использованием языка программирования Python и библиотеки `tkinter` для создания графического интерфейса. Разработка игры продемонстрировала основные принципы программирования, включая

обработку событий, работу с графическими интерфейсами и применение объектно-ориентированных подходов.

Реализованный проект охватывает ключевые аспекты разработки простых игровых приложений:

Создание и управление графическими элементами интерфейса (кнопками) с помощью tkinter.

Разработка логики игры, включая проверку победных комбинаций и определение ничьей.

Реализация механизма смены игроков и перезапуска игры по завершению партии.

Данный проект является полезным учебным примером для начинающих программистов, поскольку помогает понять, как работать с графическими библиотеками в Python, а также научиться проектировать и разрабатывать простые игры. Он может служить основой для более сложных проектов, включая добавление новых функций (например, игрового ИИ или многопользовательского режима). Также такой подход к разработке может быть использован в образовательных целях для обучения основам программирования и создания пользовательских интерфейсов.

Библиографический список

1. Новиков Е.О. Разработка программы «Игра морской бой» на языке Python // Прогрессивные технологии и экономика в машиностроении: сборник трудов XIV Всероссийской научно-практической конференции для студентов и учащейся молодежи. 2023. С. 207-210
2. Ахметгалеева Д.Ф., Галиаскарова Г.Р. Создание игры-змейка на языке программирования Python «Гомер ест пончики» // Colloquium-journal. 2019. № 17-2(41). С. 24-27
3. Матвеева А. С. Реализация паттерна поведения «Команда» в графическом интерфейсе пользователя на Python // Постулат. 2025. №1.
4. Вохидов А.М., Вохидов Д.А., Фармонова Р.Ф., Хафизова Д.Ш. Разработка графическим пользовательским интерфейсом программ в пакете tkinter с использованием современных педагогических технологий в области медицины // Роль информационных технологий в управлении ресурсами персонала здравоохранения 2022. Т. 34. С. 181-184
5. Aligulovich D.A., Mahmudova Z.I., Sayfullayev R.F. Tibbiyot yo'nalishida zamonaviy pedagogik texnologiyalarini qo'llab tkinter paketida gui dasturlarini tuzish // Евразийский журнал математической теории и компьютерных наук. 2022. №2(12). С. 31-35.