МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ

Лабораторна робота 4

Виконав:
студент групи КІ-401
Шаклеін О.Т.
Прийняв:
Шпіцер А.С

Львів 2024

Тема: Створення Doxygen документації

Порядок виконання лабораторної роботи:

Task 4. Create doxygen documentation:

1. Add doxygen comments for all public functions, classes, properties, fields...
2. Generate documentation based on doxygen comments
3. Required steps

Виконання роботи

Було виконано наступні кроки для створення документації:

1. **Додавання Doxygen-коментарів**: Вихідний код серверної та клієнтської частини гри було доповнено коментарями у форматі Doxygen. Для кожної публічної функції, методу та параметру було додано коментарі, що описують їх призначення, параметри та значення, які вони повертають. Наприклад, для функції get_winner() було додано опис, який пояснює, як визначається переможець між гравцем і сервером.

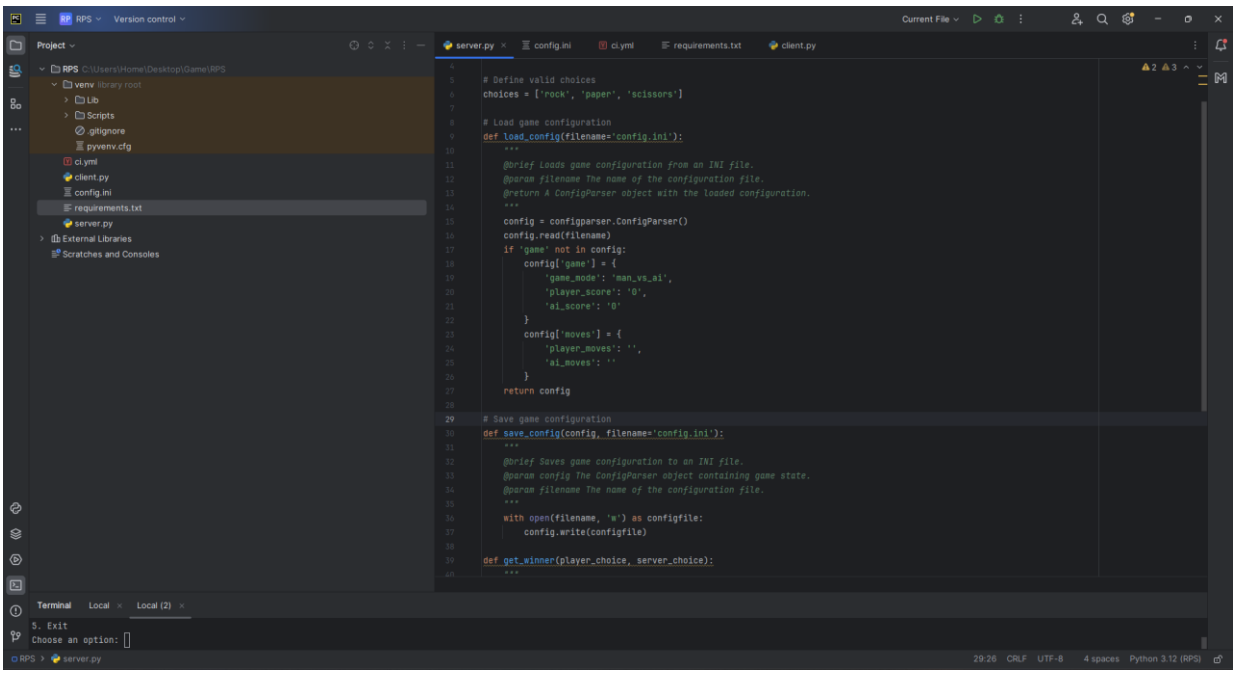

Рис. 1. Приклад коментарів у коді server.py з використанням тегів @brief, @param, @return.

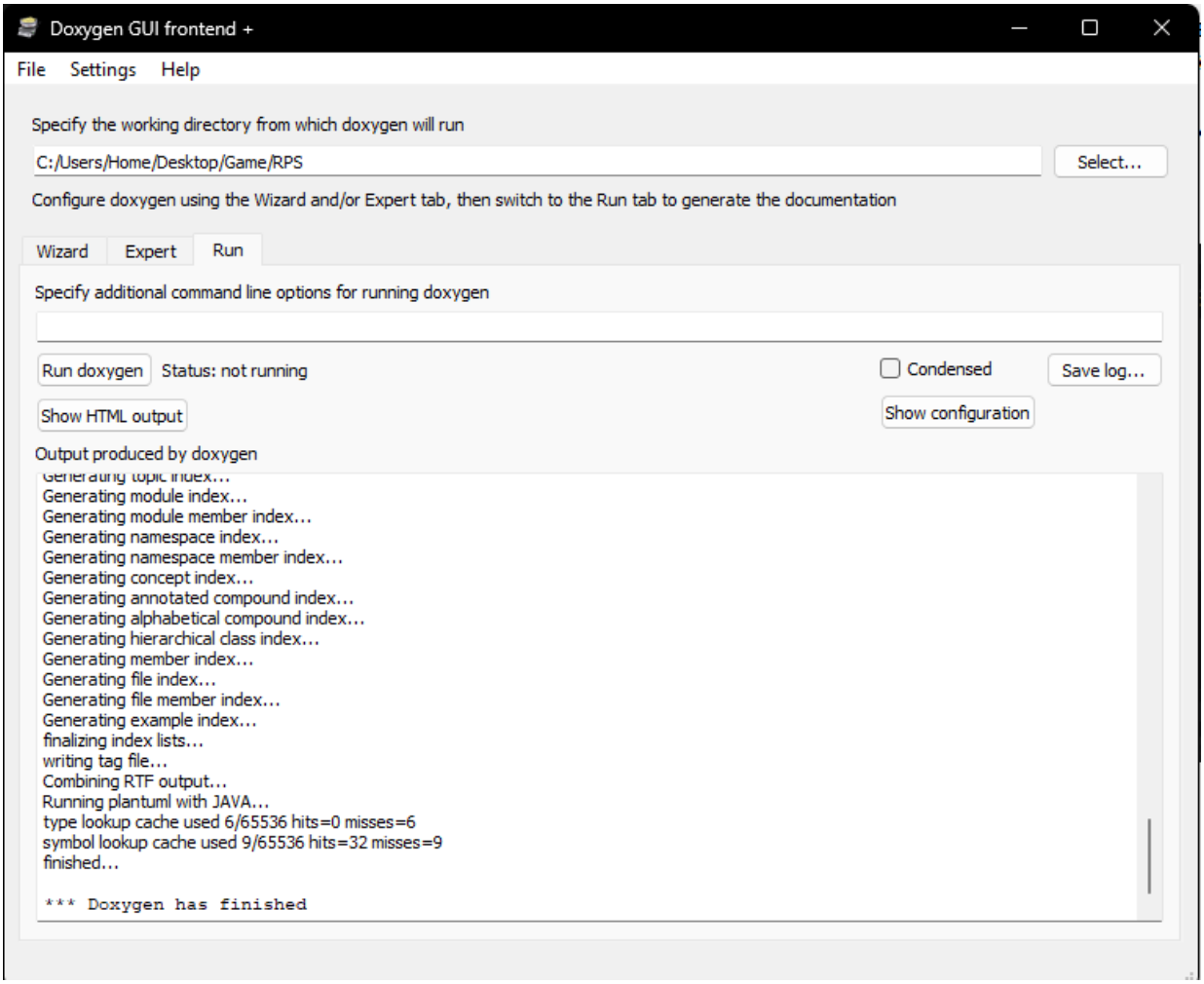

Рис. 2. Налаштування Doxygen

Генерація документації: Після налаштування файлу конфігурації було запущено команду doxygen Doxyfile, яка згенерувала HTML-документацію у папці docs. У результаті, було створено структуровану документацію з описами всіх функцій та класів, що містяться у проекті.

Рис. 3. Згенерована HTML-документація

Висновки

Виконання роботи дозволило створити повноцінну технічну документацію для гри "Камінь-Ножиці-Папір". Додавання коментарів у форматі Doxygen значно спростило процес генерації документації та забезпечило детальний опис функціональності проекту. Згенерована HTML-документація може бути використана для ознайомлення з проектом іншими розробниками та для його подальшого розвитку.

Додатки

server.py

```
import serial
import random
import configparser

# Define valid choices
choices = ['rock', 'paper', 'scissors']

# Load game configuration
def load_config(filename='config.ini'):
    """
    @brief Loads game configuration from an INI file.
    @param filename The name of the configuration file.
    @return A ConfigParser object with the loaded configuration.
    """
```

```
    config = configparser.ConfigParser()
    config.read(filename)
    if 'game' not in config:
        config['game'] = {
            'game_mode': 'man_vs_ai',
            'player_score': '0',
            'ai_score': '0'
        }
        config['moves'] = {
            'player_moves': '',
            'ai_moves': ''
        }
    return config

# Save game configuration
def save_config(config, filename='config.ini'):
    """
    @brief Saves game configuration to an INI file.
    @param config The ConfigParser object containing game state.
    @param filename The name of the configuration file.
    """
    with open(filename, 'w') as configfile:
        config.write(configfile)

def get_winner(player_choice, server_choice):
    """
    @brief Determines the winner between player and server choices.
    @param player_choice The choice made by the player.
    @param server_choice The choice made by the server.
    @return A string representing the result: "player", "server", or "draw".
    """
    if player_choice == server_choice:
        return "draw"
    elif (player_choice == "rock" and server_choice == "scissors") or \
         (player_choice == "scissors" and server_choice == "paper") or \
         (player_choice == "paper" and server_choice == "rock"):
        return "player"
    else:
        return "server"
```

client.py

```
import serial

def display_menu():
    """
    @brief Displays the game menu options to the player.
    """
    print("Rock-Paper-Scissors Game Menu:")
    print("1. New Game")
    print("2. Save Game")
    print("3. Load Game")
    print("4. Play Move")
    print("5. Exit")

def run_client():
    """
    @brief Runs the client program for the Rock-Paper-Scissors game.
    Establishes a serial connection and interacts with the user to play the game.
    """
    ser = serial.Serial('COM12', 9600, timeout=1)
    print("Rock-Paper-Scissors client is running on COM12...")

    try:
        while True:
```

```
        display_menu()
        choice = input("Choose an option: ").strip()

        if choice == "1":
            ser.write("new\n".encode())
            print("New game command sent.")
        elif choice == "2":
            ser.write("save\n".encode())
            print("Save game command sent.")
        elif choice == "3":
            ser.write("load\n".encode())
            print("Load game command sent.")
        elif choice == "4":
            player_choice = input("Enter your choice (rock, paper, or scissors): ").strip().lower()
            ser.write(f"play {player_choice}\n".encode())
            print(f"Sent move: {player_choice}")
        elif choice == "5":
            print("Exiting game.")
            break

        # Read the response from the server
        response = ser.readline().decode().strip()
        print(f"Server response: {response}")

except KeyboardInterrupt:
    print("Client shutting down.")
finally:
    ser.close()
```