МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЛЬВІВСЬКА ПОЛІТЕХНІКА



**АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ**

**КОМП’ЮТЕРНИХ СИСТЕМ**

Завдання 4: “ Документація для гри Rock-Paper-Scissors”

Виконав:

ст. гр. КІ-404

Мох М. П.

Приняв:

Федак П.Р.

Львів – 2024

ЗАВДАННЯ

1. Додайте коментарі doxygen для всіх публічних функцій, класів, властивостей, полів...

2. Створення документації на основі коментарів doxygen

**Варіант - 12**





**ТЕОРИТИЧНИЙ МАТЕРІАЛ**

**Doxygen documentation** — це автоматично згенерована документація для програмного коду, створена за допомогою інструмента **Doxygen**, що є потужним і гнучким інструментом для документування програмного забезпечення. Вона базується на спеціально форматованих коментарях у коді, які дозволяють описувати функції, класи, структури, змінні, параметри, а також інші елементи програмного забезпечення. Завдяки цьому Doxygen може значно полегшити підтримку та розуміння коду як для розробників, так і для користувачів API або бібліотек.

### **Особливості Doxygen documentation**

1. **Автоматичне створення документації**  
   Генерація технічної документації виконується автоматично на основі коментарів у вихідному коді. Це зменшує обсяг ручної роботи та забезпечує актуальність документації, оскільки вона завжди відповідає коду.
2. **Підтримка різних форматів**  
   Документація може бути згенерована у вигляді HTML-сторінок для перегляду в браузері, PDF-файлів для друку, LaTeX для наукових звітів, XML для подальшої обробки, RTF для інтеграції в текстові редактори та інших форматів. Це забезпечує гнучкість у виборі способу розповсюдження документації.
3. **Стандартизація стилю документації**  
   Використання єдиного формату коментування дозволяє забезпечити уніфікацію стилю документації в межах команди. Це особливо важливо для великих проєктів із багатьма розробниками.
4. **Графічні діаграми та залежності**  
   За допомогою інструментів, таких як **Graphviz**, Doxygen може генерувати графічні представлення залежностей між класами, функціями, модулями та іншими елементами проєкту. Це полегшує аналіз архітектури програмного забезпечення.
5. **Підтримка багатьох мов програмування**  
   Doxygen підтримує широкий спектр мов програмування, включаючи **C, C++, Python, Java, JavaScript, PHP, Fortran, IDL, VHDL** та інші. Це робить його універсальним інструментом для проєктів, які використовують кілька мов одночасно.
6. **Розширюваність**  
   Doxygen дозволяє додавати власні теги та налаштовувати конфігурацію для специфічних потреб проєкту, що робить його дуже адаптивним.

### **Навіщо використовувати Doxygen?**

1. **Спрощення підтримки та розвитку коду**  
   Зрозумілий опис функціоналу, параметрів, класів і модулів робить проєкт легшим для розуміння як для нових учасників команди, так і для майбутніх розробників.
2. **Документування API**  
   Doxygen ідеально підходить для створення документації API. Це важливо для бібліотек та інтерфейсів, які розповсюджуються серед сторонніх користувачів.
3. **Професійна презентація проєкту**  
   Згенерована документація виглядає акуратно й професійно, що може підвищити довіру до вашого програмного забезпечення серед користувачів та клієнтів.
4. **Прозорість у відкритих проєктах**  
   У спільних проєктах з відкритим кодом документація Doxygen допомагає іншим розробникам швидко розібратися з кодом і сприяти його покращенню.
5. **Ефективне навчання нових учасників команди**  
   Автоматично згенерована документація надає зрозумілу й структуровану інформацію, яка значно скорочує час навчання.

### **Основи написання коментарів для Doxygen**

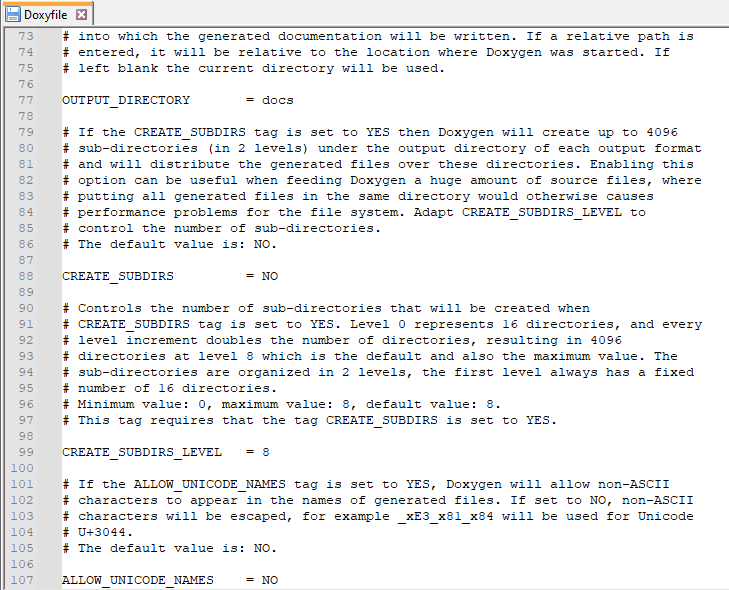
Документація створюється на основі спеціальних коментарів, які розробник додає у вихідний код. Для цього використовуються багаторядкові коментарі /\*\* \*/ або однорядкові коментарі ///.

**Приклади ключових тегів:**

* @brief — Короткий опис функції чи елемента.
* @param — Опис параметрів функції.
* @return — Опис значення, яке повертає функція.
* @author — Ім'я автора коду.
* @version — Інформація про версію проєкту чи модуля.
* @date — Дата створення чи оновлення коду.
* @warning — Попередження про можливі ризики використання.
* @see — Посилання на інші пов’язані елементи.
* @example — Надання прикладу використання.

**ВИКОНАННЯ РОБОТИ**

1. Створення doxygen коментарів у коді, щоб вподальшому можна було згенерувати doxygen documantaion.
2. Створення Doxyfile для генерування документації, для клієнта та сервера.



1. У cmd за допомогою команди doxygen Doxyfile, успішно згенерували doxygen documantaion.

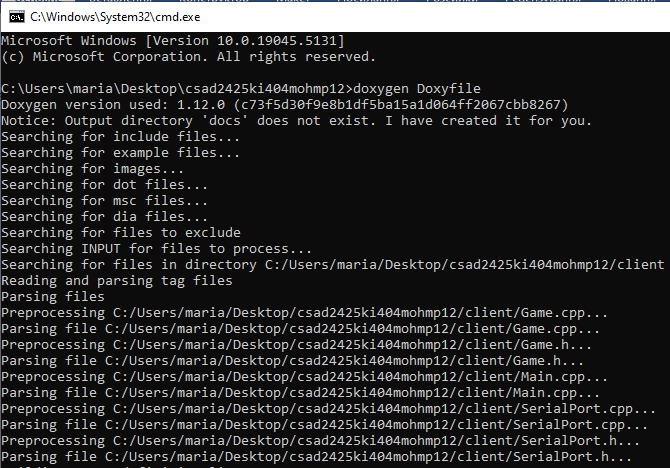


Рис 1. Використання команди doxygen Doxyfile

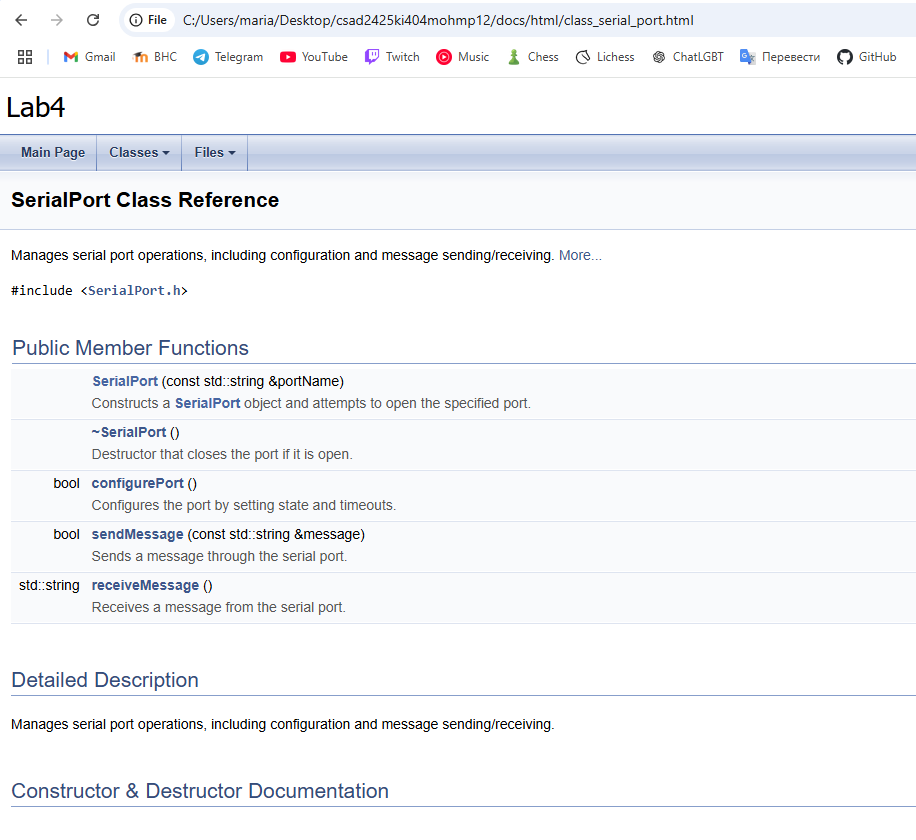


Рис 2. Документація класу SerialPort

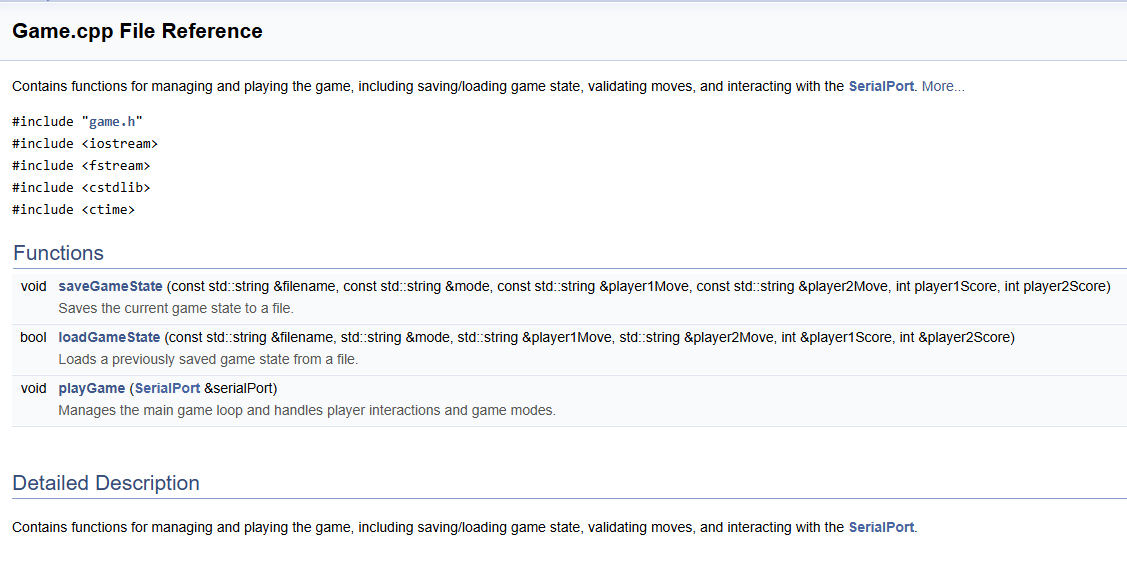


Рис 3. Документація Game.cpp

**ВИСНОВОК**

У ході виконання лабораторної роботи було успішно реалізовано всі поставлені завдання. До всіх публічних функцій, класів, властивостей і полів були додані Doxygen-коментарі, які забезпечили чітку, структуровану та деталізовану документацію кожного елемента коду. Це значно спрощує аналіз та подальшу підтримку проєкту, а також підвищує його якість і зрозумілість.

На основі доданих коментарів була створена документація за допомогою інструмента Doxygen. Вона містить технічний опис реалізації, детальну інформацію про функціонал, а також візуалізацію залежностей між елементами проєкту, що сприяє кращому розумінню архітектури розробленого коду.

СПИСОК ЛІТЕРАТУРИ

1. Van Heesch, Dimitri. Doxygen: Documentation Generator Tool. Офіційна документація Doxygen: <https://www.doxygen.nl>
2. Graphviz. Graph Visualization Software. Офіційний сайт Graphviz: <https://graphviz.org>.
3. Stroustrup, Bjarne. The C++ Programming Language. 4th Edition, Addison-Wesley, 2013. Інформація про мову C++: <https://www.stroustrup.com>.