МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Донецкий национальный технический университет»

Факультет ИСП

Кафедра ПИ им Л.П.Фельдмана

Лабораторная работа № 4

на тему: «Создание самодокументирующегося кода»

по курсу: «Профессиональная практика программной инженерии»

Проверил:

асс. каф. ПИ им. Л.П.Фельдмана Филипишин Д.А.

Выполнил:

ст. гр. ПИ-21б

Костюков В.К.

Донецк-2024

Цель работы – научиться добавлять в программный код специальным образом оформление докблок-комментарии, для последующей автоматической генерации API reference, а также познакомиться с форматом оформления документации DocBook.

Вариант 26. Genshin (или аналог (персонаж от 3го лица). Обязательно учесть связь с Steam);

1. Регистрация на GitHub (рис. 1). Аккаунт: [Shipelka](https://github.com/Shipelka/TPP-Game-PPPI-). Репозиторий: [TPP-Game-PPPI-](https://github.com/Shipelka/TPP-Game-PPPI-).

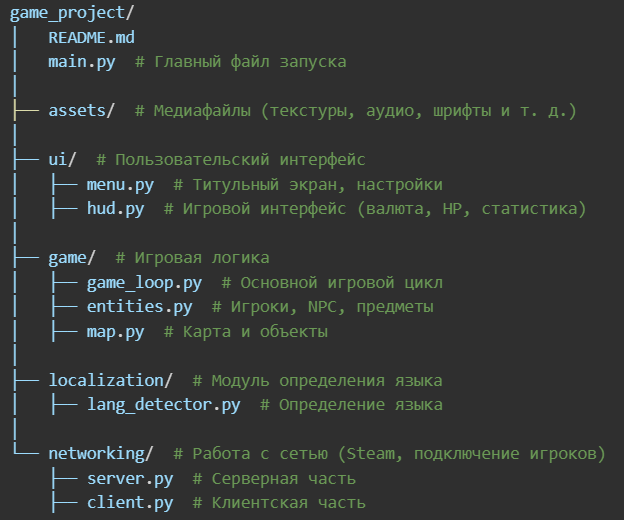


Рисунок 1 – Структура проекта

class Player:  
 *"""  
 Класс, представляющий игрока.  
  
 Атрибуты:  
 name (str): Имя игрока.  
 hp (int): Количество здоровья игрока (по умолчанию 100).  
 shield (int): Количество щита игрока (по умолчанию 50).  
 currency (int): Количество валюты игрока (по умолчанию 0).  
 kills (int): Количество убийств игрока (по умолчанию 0).  
 deaths (int): Количество смертей игрока (по умолчанию 0).  
 assists (int): Количество помощи игрока (по умолчанию 0).  
  
 Методы:  
 take\_damage(amount): Игрок принимает урон. Урон сначала поглощается щитом, если он есть, затем уходит на здоровье.  
 die(): Игрок умирает и выводится сообщение о смерти.  
 """* def \_\_init\_\_(self, name):  
 *"""  
 Инициализация игрока.  
  
 Аргументы:  
 name (str): Имя игрока.  
 """* self.name = name  
 self.hp = 100  
 self.shield = 50  
 self.currency = 0  
 self.kills = 0  
 self.deaths = 0  
 self.assists = 0  
  
 def take\_damage(self, amount):  
 *"""  
 Игрок принимает урон. Если у него есть щит, то он поглощает часть урона.  
  
 Аргументы:  
 amount (int): Количество урона, которое игрок должен получить.  
 """* if self.shield > 0:  
 absorbed = min(amount, self.shield)  
 self.shield -= absorbed  
 amount -= absorbed  
 self.hp -= amount  
 if self.hp <= 0:  
 self.die()  
  
 def die(self):  
 *"""  
 Метод, вызываемый, когда здоровье игрока падает до нуля. Выводится сообщение о смерти.  
 """* print(f"{self.name} has died.")  
  
  
class NPC:  
 *"""  
 Класс, представляющий неигрового персонажа (NPC).  
  
 Атрибуты:  
 type (str): Тип NPC.  
 hp (int): Количество здоровья NPC.  
  
 Методы:  
 take\_damage(amount): NPC принимает урон. Когда его здоровье падает до нуля, NPC считается побежденным.  
 """* def \_\_init\_\_(self, type, hp):  
 *"""  
 Инициализация NPC.  
  
 Аргументы:  
 type (str): Тип NPC.  
 hp (int): Количество здоровья NPC.  
 """* self.type = type  
 self.hp = hp  
  
 def take\_damage(self, amount):  
 *"""  
 NPC принимает урон. Если его здоровье падает до нуля, NPC побежден.  
  
 Аргументы:  
 amount (int): Количество урона, которое NPC должен получить.  
 """* self.hp -= amount  
 if self.hp <= 0:  
 print(f"{self.type} defeated.")

Рисунок 2 – Пример файл с ДокБлоками entities.py

**Sphinx**: Это одна из наиболее популярных библиотек для генерации документации. Она поддерживает различные форматы вывода, включая HTML, PDF, ePub и другие. Документация Sphinx может быть написана на языке разметки reStructuredText.

Структура полученных файлов представлена на рисунке 3.

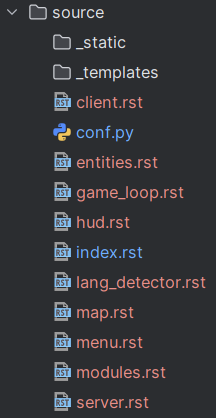


Рисунок 3 – Структура папки source



Рисунок 4 – Главная страница полученного сайта с документацией

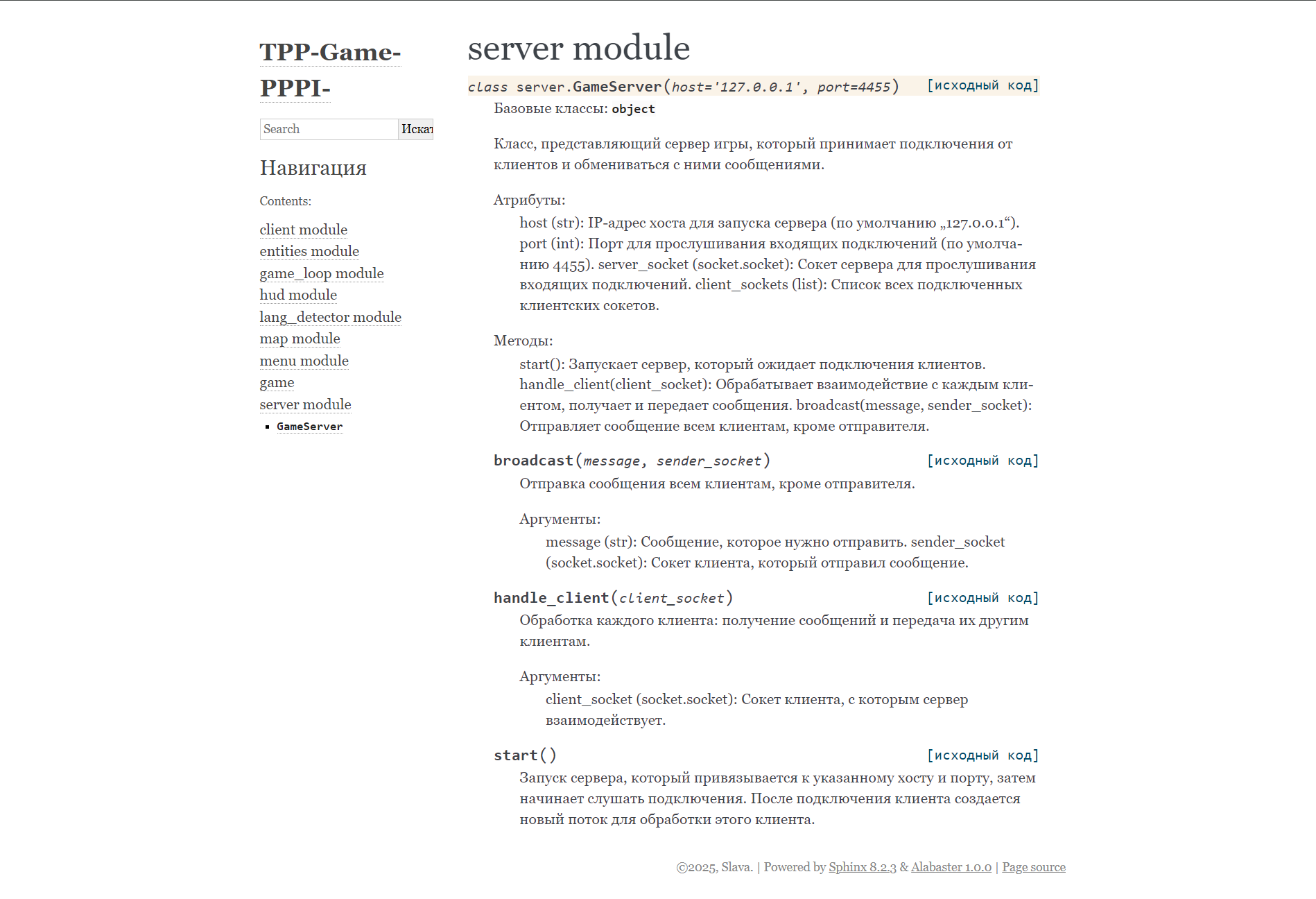


Рисунок 5 – Пример документации файла server.py