Отчет по выполнению тестового задания

Разработчик LOD (C#)

Инициатор разработки: Сироткин Артём
Должность: Студент 2 курса СКБ231

Контакты: ВКонтакте, @TRUGGYONEPRO

Репозиторий Git: ссылка.

Вступление:

В данном файле описан метод решения тестового задания на должность разработчика LOD (C#).

Задача 1

1. Реализовать класс Shape, который имеет метод GetArea. Реализовать наследуемые от Shape классы Circle и Rectangle с методом GetArea, который считает площадь фигуры: круга для Circle, прямоугольника для Rectangle, соответственно.

    - Создаем класс Shape, стоит отметить, что он является абстрактным, т.к. создаваемый класс не подразумевает конкретного воплощения: фигурой может быть квадрат, треугольник, круг и т.д.
    - В теле Shape реализуем абстрактный метод GetArea, который определяет интерфейс для всех наследуемых фигур.
    - Далее реализуем класс Circle, который наследует базовый класс Shape.
    - В теле класса Circle – переменную, отвечающую за радиус окружности, и конструктор, принимающий значение радиуса; переопределяем метод GetArea для окружности и возвращаем формулу площади круга.
    - Соответственно и для класса Rectangle.

    - Далее приводим пример использования классов.

2. Создать интерфейс IMovable с методом Move, реализовать классы Car и Bicycle с использование полученного интерфейса. Объяснить, зачем нужны интерфейсы, чем они могут быть полезны.
    - Создать интерфейс IMovable, в теле которого находится метод Move.
    - Реализуем два класса, Car и Bicycle, с использованием созданного ранее интерфейса, в теле которых описываем метод Move().
    - Далее приводим пример использования классов.

Зачем нужны интерфейсы?

-

Чем интерфейсы могут быть полезны?

-

3.
4. Реализовать класс BankAccount, который будет содержать информацию о балансе счета. Класс должен иметь методы для внесения и снять денег со счета, иметь проверку на то, достаточно ли денег на счету для снятия.
    - Создаем основной класс BankAccount, который содержит:
        - Приватное поле balance для хранения счета клиента
        - Конструктор, который принимает начальный balance и устанавливает его.
        - Метод CashIn предназначен для пополнения счета, получает внесенную сумму, проверяя ее, и добавляет ее к текущему балансу.
        - Метод CashOut: предназначен для снятия средств со счета, получает запрашиваемую сумму, проверяя ее, и происходит вычет средств с баланса клиента.
        - Метод ShowBalance возвращает текущий баланс клиента.
    - Далее приводим пример использования классов.

Задача 2

1. Реализовать упрощенную систему LOD на Unity: необходимо изменять модель в зависимости от удаленности от нее игрока. Дополнить систему заменой моделей, которые находятся за спиной игрока, объяснить полученный код. Для возможности проверки системы сделать статичными модели, которые были изменены после того, как остались за спиной игрока.

Примечание – упрощенная система LOD реализована в виде скрипта, включающий сразу указанное дополнение в задаче

- Создания для тестов небольшой «комнаты» при помощи ProBuilder.
- Добавление «игрока» и скриптов, реализовывающих его передвижение и вид как от 3-ого, так и от 1-ого лица.
- Создание в Blender 3D простого тестового объекта – «бочка» с различными уровнями детализациями.
- Написание скрипта – упрощенной системы LOD:
    - Создание переменных отвечающих за ссылки на три вида полигональных моделей объекта.
    - Создание ссылки на игрока-камеру.
    - Задание переменных на уровни дистанций для переключения моделей.
    - Создание логической переменной, отвечающей за нахождение объекта в поле зрения игрока (за спиной игрока).
    - Реализация метода UpdateLOD(), отвечающего за работу LOD.
    - Написание метода SetActiveModel(), реализующий вкл-выкл полигональных моделей объекта.
- Тестирование.

Описание кода: Мы должны применить скрипт на объект, на который мы хотим использовать систему LOD. Передаем ссылки на камеру/игрока, три полигональных модели, которые являются дочерними