|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **РОСЖЕЛДОР**  **федеральное государственное бюджетное образовательное учреждение высшего образования**  **«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ»**  **(СГУПС)**  Кафедра «Информационные технологии на транспорте»  **Лабораторная работа №3**  по дисциплине «Технология программирования»  Игра «Space Invaders»   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Проверил: | |  | Выполнил: | | | | старший преподаватель | |  | студент гр. | | БИСТ-311 | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ К.В. Спешилов  *Подпись* | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ К.Е. Коноваленко  *подпись* | | | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *дата проверки* | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *дата сдачи на проверку* | | | |  |  | | | |  |  | | |   Краткая рецензия:   |  | | --- | |  | |  | |  | |  | |  |  |  |  |  | | --- | --- | --- | |  |  |  | | *запись о допуске к защите* |  |  | |  |  |  | | *оценка по результатам защиты* |  | *подписи преподавателей и дата защиты* |   Новосибирск  2020 |

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc64987817)

[1 Правила игры 4](#_Toc64987818)

[2 Классы и причины их выделения 5](#_Toc64987819)

[3 Диаграмма классов 6](#_Toc64987820)

[4 Тесты 7](#_Toc64987821)

[4.1 Тесты Game 7](#_Toc64987822)

[4.2 Тесты Player 8](#_Toc64987823)

[4.2 Тесты Unit 8](#_Toc64987824)

[5 Метрики кода 10](#_Toc64987825)

[ЗАКЛЮЧЕНИЕ 11](#_Toc64987826)

# **ВВЕДЕНИЕ**

Целью данной лабораторной работы является разработка графического приложения. Тема проекта – игра «Жизнь». Требуется создать графическое приложение для игры, используя платформу WPF. Классы игры нужно выделить в библиотеку. Кроме того, требуется провести тестирование полученной библиотеки классов. Для достижения цели требуется выполнить следующие задачи:

- Создать графический интерфейс приложения;

- выделить классы в библиотеку, при необходимости переписать их;

- образовать зависимости между приложениями и библиотекой;

- Создать комплект тестов;

- Вычислить метрики кода.

Также требуется разработать диаграмму классов, используя правильные виды связей.

# **1 Правила игры**

Space Invaders – игра на результат, время игры не ограниченно. Необходимо отстреливать волны инопланетных захватчиков. Инопланетяне умирают с одного выстрела, за каждое убийство игрок получает очки. Игрок может перемещаться в лево и право, также проходить сквозь экран по кругу и проигрывает тогда, когда инопланетяне доходят до нижнего края экрана. Инопланетяне появляются в верху экрана с некоторой периодичностью, с каждой волной все быстрее. Целью игры является набрать как можно больше очков и оказаться на первом месте в таблице лидеров.

# **2 Классы и причины их выделения**

1)Класс «GameFileManager»

Причина создания – упаковка родственных операций, изоляция сложности. Предназначен для работы с файлами.

2)Класс «WpfInteraction»

Причины создания – изоляция сложности. Операции, связанные с пользовательским взаимодействием, были выделены в отдельный класс, чтобы отделить их от основной логики игры и тем самым понизить сложность.

3)Класс «WpfVisualizer»

Причина создания – изоляция сложности. Отрисовка поля была выделена в отдельный класс с целью отделения обслуживающей операции от основной логики игры.

4)Класс «MainWindow» Занимается взаимодействием с пользователем. Причина создания – изоляция сложности.

# **3 Диаграмма классов**

Диаграмма классов представлена на рисунке 1.

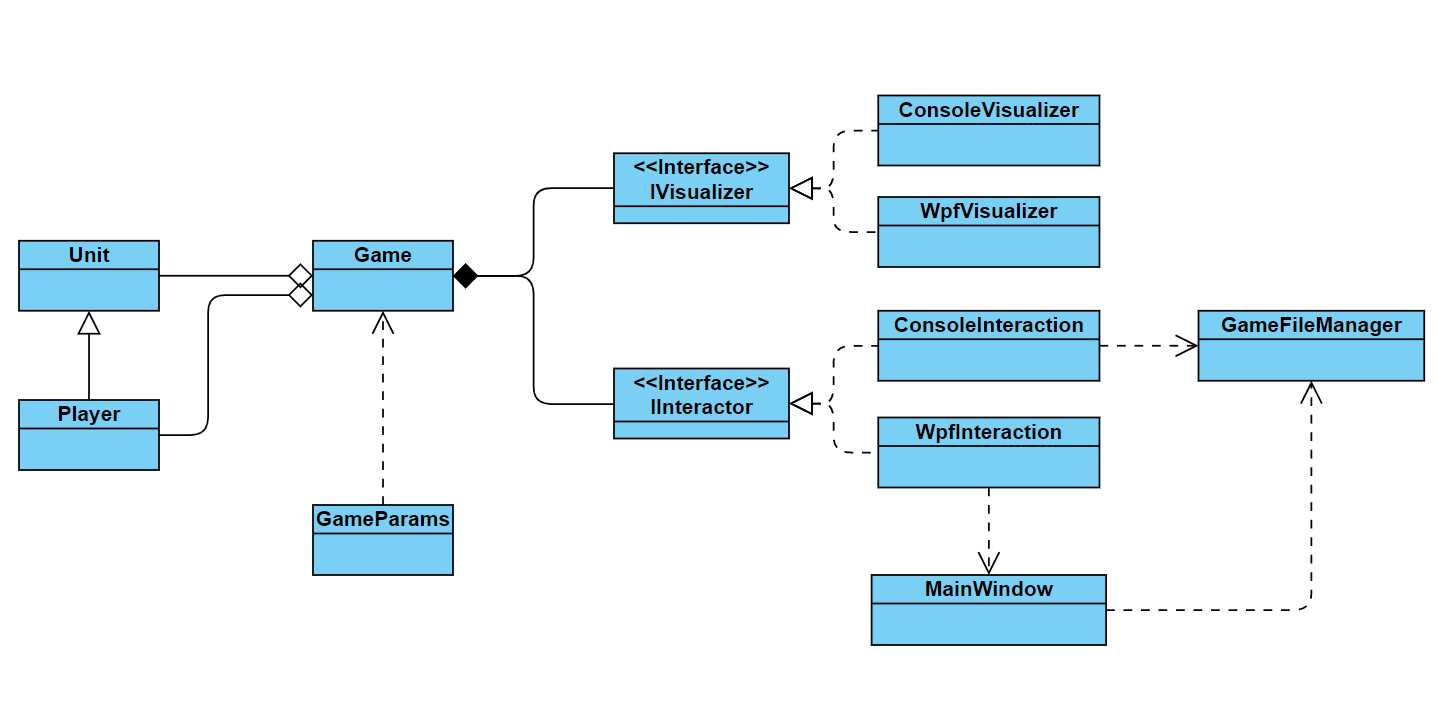
****

Рисунок 1 – Диаграмма классов

# **4 Тесты**

Модульные тесты были разработаны для классов Game, Unit и Cell, так как в них содержится логика игрового процесса. Для класса GameFileManager тесты не разрабатывались, так как он реализует работу с файлами.

## 4.1 Тесты Game

В данном классе происходит создание и передвижение врагов по полю, перемещение пуль. Для метода генерации разработана пара тестов:

1) EnemyNeedToGenTest – проверяет ситуацию, когда необходимо генерировать противников. Создаем поле с врагом и сохраняем на него ссылку. Проверим что после выполнения метода он находится на координате на один больше, и что на прошлой позиции находиться отличный от него объект.

2) и EnemyNotNeedToGenTest – когда модификация поля не требуется. Создаем поле с врагом и сохраняем на него ссылку. Проверим что после выполнения метода он находится на прежнем месте, и что на следующей позиции находиться отличный от него объект.

Тесты для метода обработки снарядов:

1) ProjectileCollideExitTest – проверяет поведение при попадании во врага. Снаряд должен уничтожится. Снаряд и противник должны уничтожится. Создаем поле с противником и снарядом, расположенным за ход от попадания. После выполнения метода список снарядов должен быть пуст.

2) ProjectileExitTest – проверяет поведение при достижении снарядом конца экрана. Снаряд должен уничтожится. Создаем снаряд на последней клетке поля. После выполнения метода список снарядов должен быть пуст.

3) ProjectileCollideDestoyTest - проверяет поведение при смертельном попадании во врага. Снаряд и противник должны уничтожится. Создаем поле с противником и снарядом, расположенным за ход от попадания. После выполнения метода список снарядов должен быть пуст, а содержимое поля на месте противника null.

## 4.2 Тесты Player

В данном классе обрабатывается передвижение игрока по полю.

1. MovePositiveOKTest – проверяет перемещение вправо. Создаем игрока, проверяем что после выполнения метода с параметром 1 его координаты по х совпадают с ожидаемыми.
2. MoveNegativeOKTest – проверяет перемещение влево. Создаем игрока, проверяем что после выполнения метода с параметром -1 его координаты по х совпадают с ожидаемыми.
3. MoveZeroTest - Создаем игрока, проверяем что после выполнения метода с параметром 0 его координаты по х не изменились.
4. MoveBiggerThenTest - Создаем игрока, проверяем что после выполнения метода с параметром большим 1 его координаты по х не изменились.
5. MoveBiggerThenNTest - Создаем игрока, проверяем что после выполнения метода с параметром меньшим -1 его координаты по х не изменились.

## 4.2 Тесты Unit

В данном классе обрабатывается поведение игровых объектов. Тесты для метода столкновения:

1. CollisionTest – проверяет изменение здоровья при нанесении урона. Создаваем объект с n количеством здоровья, вызываем метод на k положительного урона. Проверяем что здоровье юнита стало равно n-k.
2. CollisionHealTest – проверяет изменение здоровья при нанесении отрицательного урона(лечение). Создаваем объект с n количеством здоровья, вызываем метод на k отрицательного урона. Проверяем что здоровье юнита стало равно n+k.
3. EmptyfyByZeroTest – проверяет что при получении летального урона существо меняет свой тип на пустой. Создаем юнит с 1 здоровьем, вызываем метод с параметром 1. Проверяем что тип сменился на Empty
4. EmptyfyByZeroTest – проверяет что при получении летального урона существо меняет свой тип на пустой. Создаем юнит с 1 здоровьем, вызываем метод с параметром большим 1. Проверяем что тип сменился на Empty
5. EmptyfyMoreThenZeroTest - проверяет что при получении не летального урона существо не меняет свой тип на пустой. Создаем юнит с 3 здоровьем, вызываем метод с параметром 1. Проверяем что тип остался прежним.
6. CollisionNotRevive – проверяет что мертвое существо не получит урона или лечения если уже мертво. Создаем существо с типом “Empty”, вызываем метод. Здоровье и тип не должны измениться.

# **5 Метрики кода**

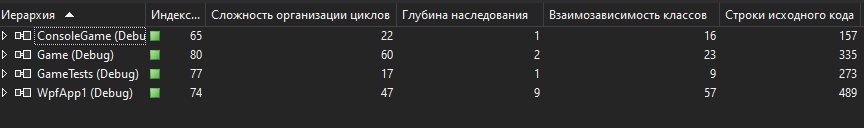


Рисунок 2 – Метрики

# **ЗАКЛЮЧЕНИЕ**

В ходе лабораторной работы классы игры были выделены в библиотеку классов. Были выделены и обоснованы новые классы. Было разработано графическое приложение. В проект консольного и WPF приложения была добавлена ссылка на библиотеку классов. Была обновлена диаграмма классов. Был создан комплект модульных тестов для классов Game, Player и Unit, тесты были описаны.