|  |  |
| --- | --- |
|  | МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ТАТАРСТАН **Государственное автономное профессиональное образовательное учреждение** **«ЗЕЛЕНОДОЛЬСКИЙ МЕХАНИЧЕСКИЙ КОЛЛЕДЖ»** (ГАПОУ «ЗМК») |

**09.02.03 «Программирование в компьютерных системах»**

# **МДК 03.01. ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

**Отчет о практических работах**

**Исполнитель**: Смирнов Кирилл Витальевич

**Группа**: 207

**Преподаватель:** Алемасов Евгений Павлович

**Дата сдачи** 05.12.2023 **Оценка** \_\_\_\_\_\_\_\_\_\_\_\_

**Подпись преподавателя**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ЗЕЛЕНОДОЛЬСК – 2023**

**Работа №1. Создание диаграммы классов**

**Цель:** получение навыков построения Class Diagram.

**Задание 1:**

Диаграмма классов в языке моделирования UML относится к структурному

типу диаграмм, используется для визуализации структуры классов в системе,

атрибутов, методов, интерфейсов и отношений между ними. Применяется при

проектировании архитектуры, документировании системы, уточнении

требований, а также для поддержки системы.

Связи между классами на диаграмме классов могут представлять отношения,

такие как ассоциация, наследование и реализация. Ассоциация показывает

отношения между классами, которые могут быть однонаправленными или

двунаправленными. Наследование показывает, как класс наследует свойства

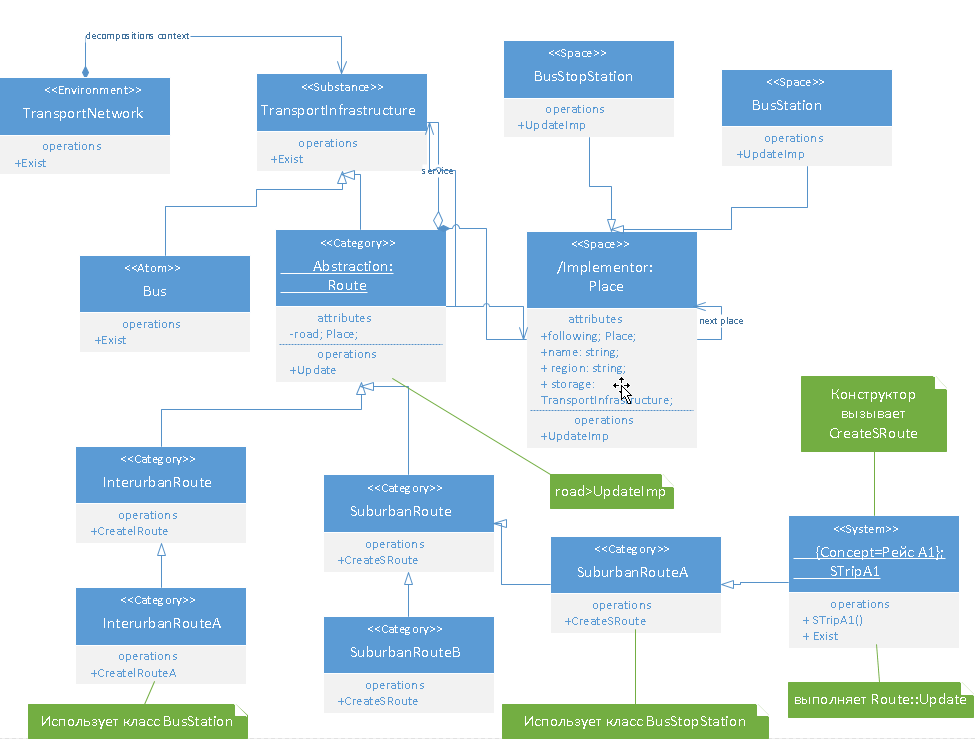
другого класса, который является его родительским классом. Реализация

показывает, как класс реализует интерфейс или абстрактный класс.

Использование диаграммы классов помогает лучше понимать структуру

системы и ее компоненты.

**Результаты выполнения работы:**



**Задание 2**

Разработайте диаграмму классов UML для системы управления

задачами. Система должна поддерживать следующий функционал:

Пользователи:

Каждый пользователь имеет уникальный идентификатор, имя и

электронную почту.

Пользователь может создавать задачи, назначать их себе или другим

пользователям, устанавливать статус (выполнено, в процессе,

отложено), устанавливать сроки выполнения.

Задачи:

Задачи имеют уникальный идентификатор, описание, статус

выполнения и срок выполнения.

Задачи могут быть созданы, просмотрены и изменены пользователями.

Категории задач:

Задачи могут быть связаны с одной или несколькими категориями

(например, работа, личные дела, обучение).

Каждая категория имеет уникальный идентификатор и название.

Дополнительные требования (с уточнением по видимости):

User:

id, name, email - приватные атрибуты.

create\_task - публичный метод.

assign\_task - защищенный метод.

set\_task\_status - публичный метод.

Task:

id, description, status, deadline - приватные атрибуты.

get\_priority - публичный метод.

set\_priority - защищенный метод.

link\_to\_category - публичный метод.

Category:

id, name - приватные атрибуты.

get\_category\_name - публичный метод.

Композиция:

User и Task: Каждый пользователь владеет своим списком задач.

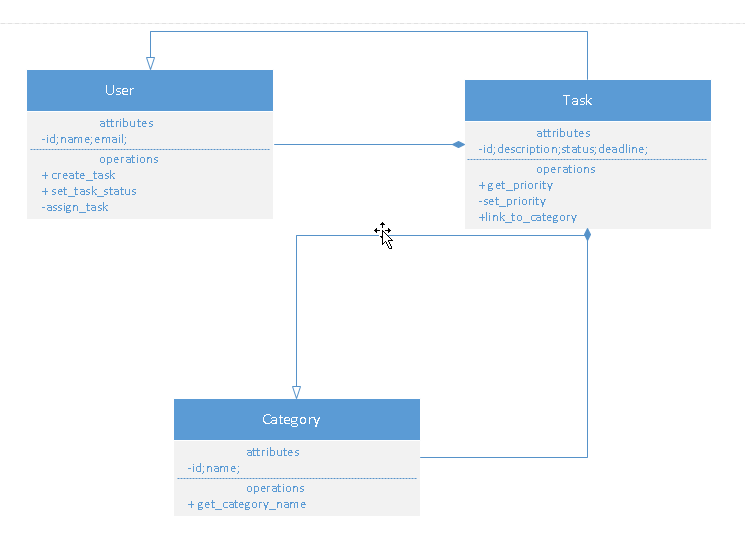
Task и Category: Задача состоит из категории.

Обобщение:

Task обобщает Category (задача - это более общий понятийный класс,

чем категория).

**Результаты выполнения работы:**



Задание 3

Разработайте диаграмму классов UML для системы школьной

библиотеки. Система предназначена для учета книг, их выдачи

студентам и управления каталогом библиотекарем.

Книги (Book):

isbn: str (private)

title: str (private)

author: str (private)

genre: str (private)

Реализуйте методы:

get\_book\_info(): void (public) - выводит информацию о книге.

check\_out\_book(student: Student): void (public) - регистрирует выдачу

книги студенту.

return\_book(): void (public) - регистрирует возврат книги в библиотеку.

Студенты (Student):

student\_id: int (private)

name: str (private)

grade: int (private)

Реализуйте методы:

borrow\_book(book: Book): void (public) - регистрирует взятие книги

студентом.

return\_book(book: Book): void (public) - регистрирует возврат книги

студентом.

get\_student\_info(): void (public) - выводит информацию о студенте.

Библиотекарь (Librarian):

librarian\_id: int (private)

name: str (private)

Реализуйте методы:

add\_book(book: Book): void (public) - добавляет книгу в каталог

библиотеки.

remove\_book(book: Book): void (public) - удаляет книгу из каталога

библиотеки.

get\_librarian\_info(): void (public) - выводит информацию о библиотекаре.

Отношения:

Агрегация:

Student агрегирует Book (студент может взять несколько книг).

Librarian агрегирует Book (библиотекарь управляет несколькими

книгами).

Ассоциация:

Student и Book: Связь между студентами и книгами (например, "взял

книгу из библиотеки").

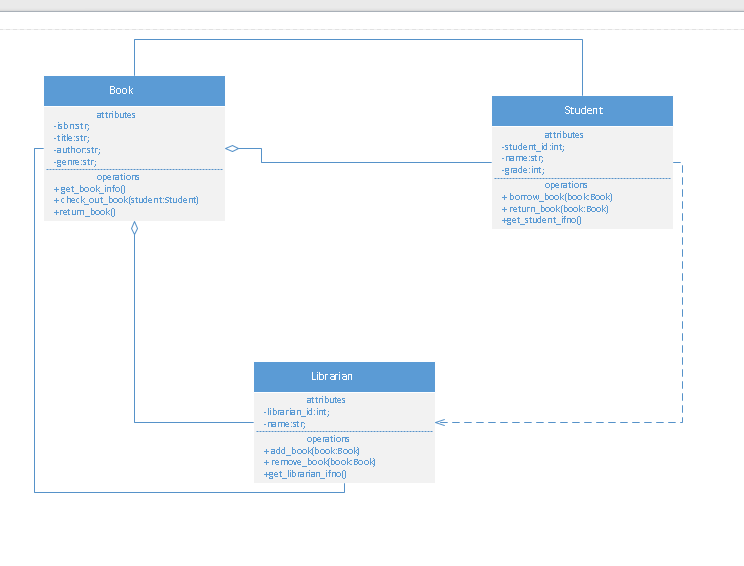
Librarian и Book: Связь между библиотекарем и книгами (например,

"добавил книгу в каталог").

Зависимость:

Student зависит от Librarian при взятии или возврате книги.

**Результаты выполнения работы:**



Задание 4

Класс MyWindow уточняет абстрактный базовый класс Window.

MyWindow состоит (композиция) из кнопки класса Button и надписи

класса Label.

Отобразите на диаграмме классов:

а. Класс Label имеет закрытый атрибут text типа String и общедоступную

операцию setText c параметром text типа String.

б. Композиция между MyWindow и Button называется HoldsButton.

Полюс со стороны кнопки имеет имя okButton, кратность 1. Композиция

между MyWindow и Label называется HoldsLabel. Название полюса со

стороны Label: textLabel, кратность 1.

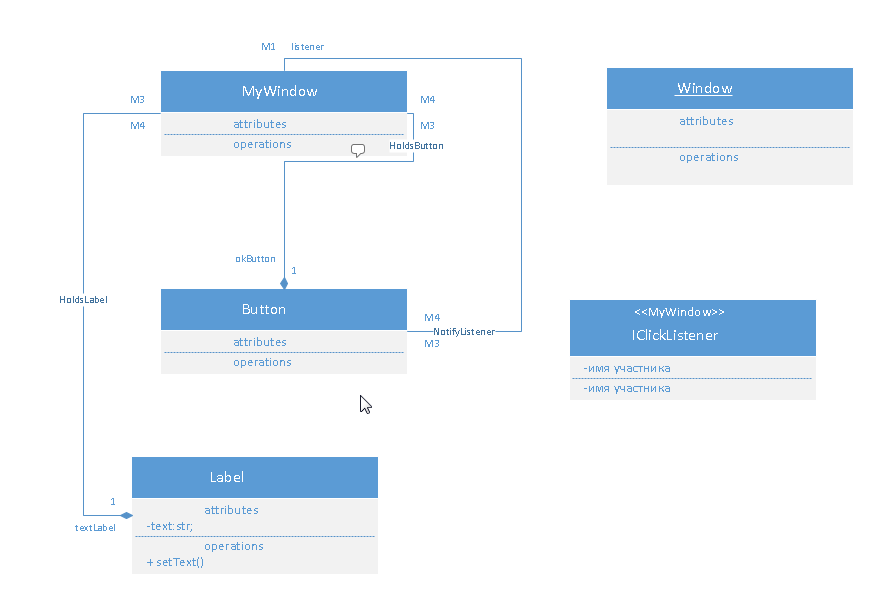
в. Класс MyWindow реализует интерфейс IClickListener для реакции на

нажатие кнопки. Отобразите на диаграмме, что между классом Button и

MyWindow есть ассоциация с именем NotifyListener с направлением от

кнопки к окну. Укажите, что полюс со стороны окна называется listener.

**Результаты выполнения работы:**



Задание 5

Постройте диаграмму классов, описывающей реализацию систем продаж

товаров по каталогу.

Результат выполнения работы

