|  |
| --- |
| Изображение выглядит как текст, коллекция картинок, фарфор  Автоматически созданное описание  МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
| Институт кибербезопасности и цифровых технологий |
| Кафедра КБ-2 «Прикладные информационные технологии» |

**ЗАЧТЕНО**

\_\_\_\_\_\_\_\_\_\_ /Лесько С.А./

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3  
по дисциплине  
«Технологии и методы программирования»**

***(наименование дисциплины)***

**Выполнил**

Студент 3 курса: Жидков Г.С.

Группы: БИСО-01-20

Специальности: 10.05.04

Москва 2023

**Оглавление**

[Стратегия (Strategy) 3](#_Toc133854309)

[Шаблонный метод (Template Method) 4](#_Toc133854310)

[Приложение 5](#_Toc133854311)

Стратегия (Strategy)

На Рисунке 1 представлена UML-диаграмма реализации паттерна «Стратегия», код диаграммы представлен в Приложении. Исходный код реализации на C# представлен в GitHub-репозитории, который указан в Приложении.

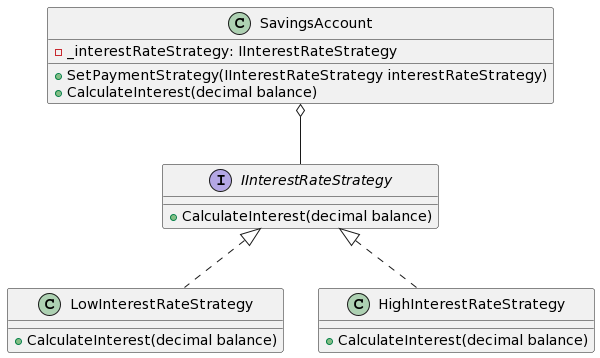


Рисунок 1 – Диаграмма для паттерна «Стратегия»

*Описание.*

IInterestRateStrategy - интерфейс стратегии с одним методом, который необходимо реализовать наследникам

LowInterestRateStrategy – конкретная стратегия для расчета низкой процентной ставки, реализует интерфейс IInterestRateStrategy

HighInterestRateStrategy – конкретная стратегия расчета высокой процентной ставки, реализует интерфейс IInterestRateStrategy

SavingsAccount - контекст, который используется клиентом для реализации стратегии, содержит информацию о текущей стратегии

Шаблонный метод (Template Method)

На Рисунке 2 представлена UML-диаграмма реализации паттерны «Шаблонный метод», код диаграммы представлен в Приложении. Исходный код реализации на C# представлен в GitHub-репозитории, который указан в Приложении.

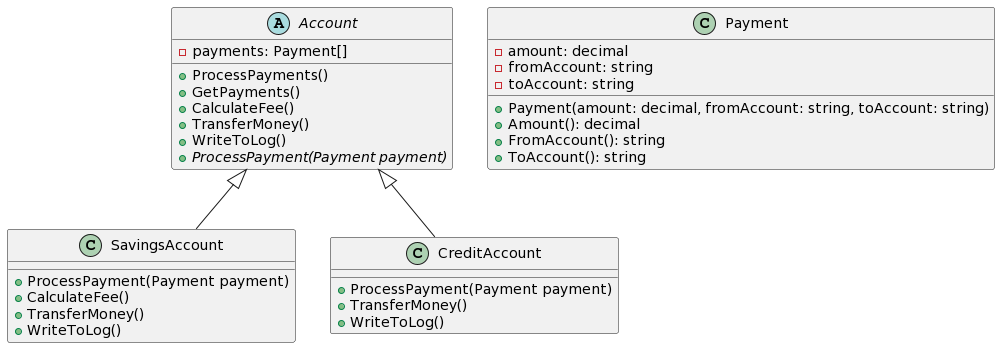


Рисунок 2 – Диаграмма для паттерна «Шаблонный метод»

*Описание.*

Payment - класс платежа, содержит сумму, информацию о получателе и отправителе.

Account - базовый абстрактный класс Account, который содержит основной алгоритм обработки платежей и объявляет абстрактные методы.

SavingsAccount - класс сберегательного счета, наследует и реализует класс Account

CreditAccount - класс кредитного счета, наследует и реализует класс Account

Приложение

Ссылка на репозиторий GitHub:<https://github.com/st-georgy/TMP>

**Исходный код диаграммы паттерна «Стратегия»**

@startuml

interface IInterestRateStrategy {

+CalculateInterest(decimal balance)

}

class LowInterestRateStrategy {

+CalculateInterest(decimal balance)

}

class HighInterestRateStrategy {

+CalculateInterest(decimal balance)

}

class SavingsAccount {

-\_interestRateStrategy: IInterestRateStrategy

+SetPaymentStrategy(IInterestRateStrategy interestRateStrategy)

+CalculateInterest(decimal balance)

}

SavingsAccount o-- IInterestRateStrategy

IInterestRateStrategy <|.. LowInterestRateStrategy

IInterestRateStrategy <|.. HighInterestRateStrategy

@enduml

**Исходный код диаграммы паттерна «Шаблонный метод»**

@startuml

abstract class Account {

-payments: Payment[]

+ProcessPayments()

+GetPayments()

+CalculateFee()

+TransferMoney()

+WriteToLog()

{abstract} +ProcessPayment(Payment payment)

}

class SavingsAccount {

+ProcessPayment(Payment payment)

+CalculateFee()

+TransferMoney()

+WriteToLog()

}

class CreditAccount {

+ProcessPayment(Payment payment)

+TransferMoney()

+WriteToLog()

}

class Payment {

-amount: decimal

-fromAccount: string

-toAccount: string

+Payment(amount: decimal, fromAccount: string, toAccount: string)

+Amount(): decimal

+FromAccount(): string

+ToAccount(): string

}

Account <|-- SavingsAccount

Account <|-- CreditAccount

@enduml