Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский ядерный университет «МИФИ»

ОТЧЕТ

по лабораторной работе №2
по дисциплине: «Теория и технология программирования»

Выполнила: *студентка группы Б22-901* *Ерёмина Е.В.*
(подпись) (Фамилия И.О)

Проверил: . *Смирнов Д.С.*
(оценка) (подпись) (Фамилия И.О)

Москва 2025

Оглавление

Цель работы

Изучение и практическое применение порождающих паттернов проектирования для создания гибкой и масштабируемой системы генерации объектов (орков) с различными характеристиками.

Проблематика

Работа направлена на закрепление знаний о порождающих паттернах (Builder, Abstract Factory, Factory Method, Director) и их комбинировании для построения сложных объектов с гибкой конфигурацией. Результатом станет программа с графическим интерфейсом, позволяющая создавать и управлять армией орков с уникальными характеристиками и снаряжением.

Задачи

1. Изучение паттерна "Строитель" (Builder):
    - Реализовать класс Ork с настраиваемыми атрибутами (имя, сила, ловкость и т.д.).
    - Создать OrkBuilder для пошаговой конфигурации объектов орков.
    - Использовать библиотеку Faker для генерации имён орков, если они не заданы вручную.
2. Изучение паттерна "Абстрактная фабрика" (Abstract Factory):
    - Разработать интерфейс OrcGearFactory для создания снаряжения (оружие, броня, знамя).
    - Реализовать конкретные фабрики для каждого племени (MordorGearFactory, DolGuldurGearFactory и т.д.), учитывая их уникальные особенности.
3. Изучение паттерна "Фабричный метод" (Factory Method):
    - Создать интерфейс OrkBuilderFactory для генерации строителей орков (OrkBuilder), связанных с конкретными фабриками снаряжения.
4. Изучение паттерна "Директор" (Director):
    - Реализовать класс OrcDirector, управляющий процессом сборки орков.

- Определить методы для создания типовых орков (createBasicOrk, createLeaderOrk, createScoutOrk), учитывая их роль в армии.

5. Разработка графического интерфейса (Swing):
   - Создать интерфейс для выбора племени и роли орка.
   - Реализовать JTree для отображения иерархии армии (племена → орки).
   - Добавить панель с информацией о выбранном орке (имя, характеристики, снаряжение).
   - Использовать JProgressBar для визуализации характеристик.
6. Дополнительные задачи:
   - Визуализировать структуру классов с помощью диаграммы.
   - Реализовать кастомную модель для JTree на основе DefaultTreeModel.

Описание работы программы

1. До процесса кодирования была создана диаграмма классов на концептуальном уровне (Рисунок 1).

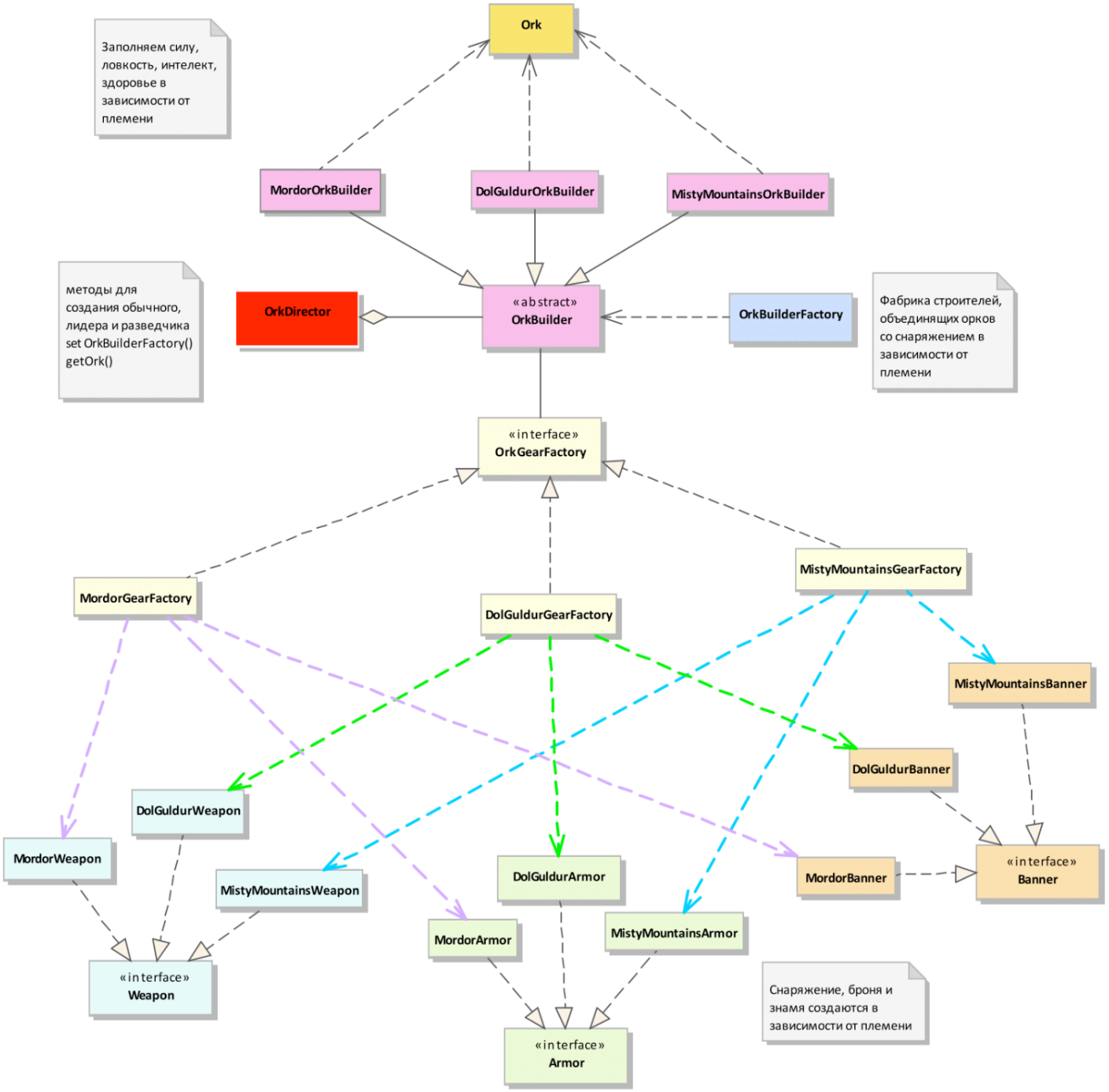

Рис. 1 Диаграмма классов на концептуальном уровне

2. Перед созданием GUI был создан wire-frame эскиз (Рисунок 2).

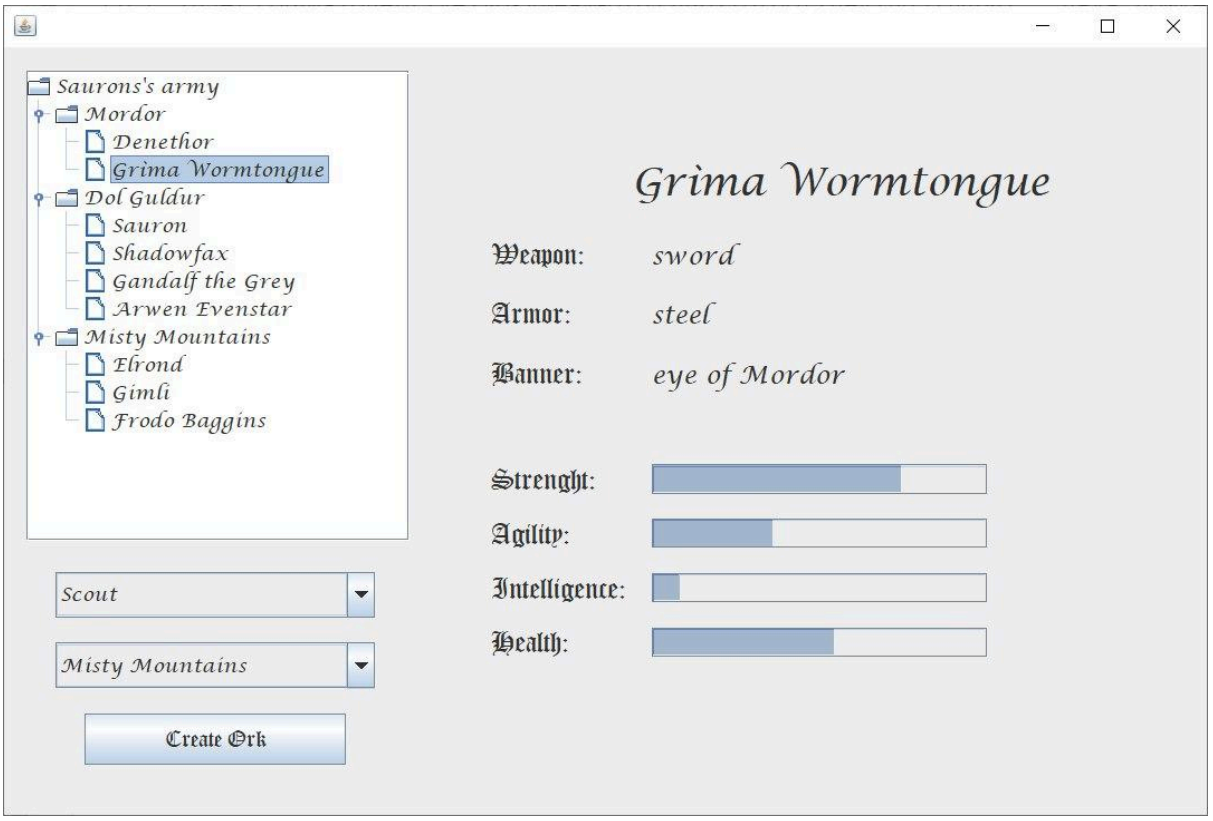

Рис. 2 wire-frame GUI

3. Скриншоты и описание работы программы.

Рис. 3 при запуске программы открывается окно, содержащее весь функционал программы

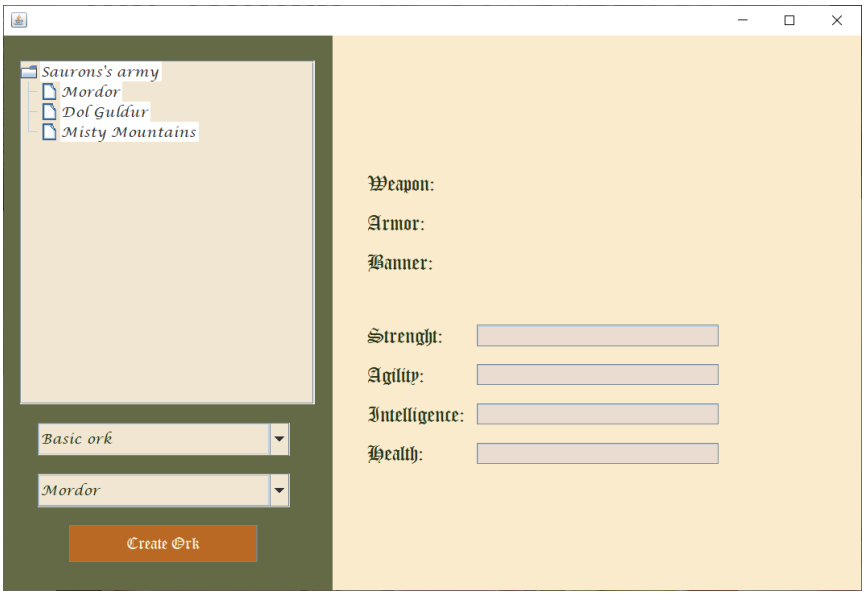

Рис. 4
пользователь
выбирает из
выпадающего
списка тип орка

Рис. 5
пользователь
выбирает из
выпадающего
списка племя
создаваемого орка

Рис. 6
после выбора
орка,
пользователь
нажимает кнопку
“Create Ork” для
его создания

Рис. 7
созданный орк
появляется в
дереве армии

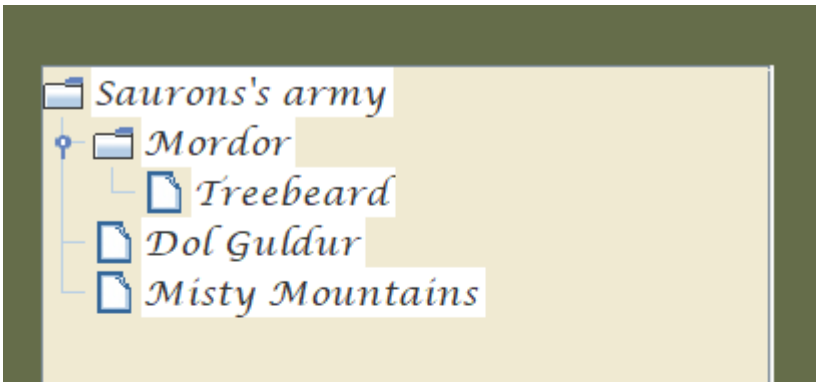

Рис. 8
при двойном нажатии на созданного орка, на поле справа появляется информация о нем

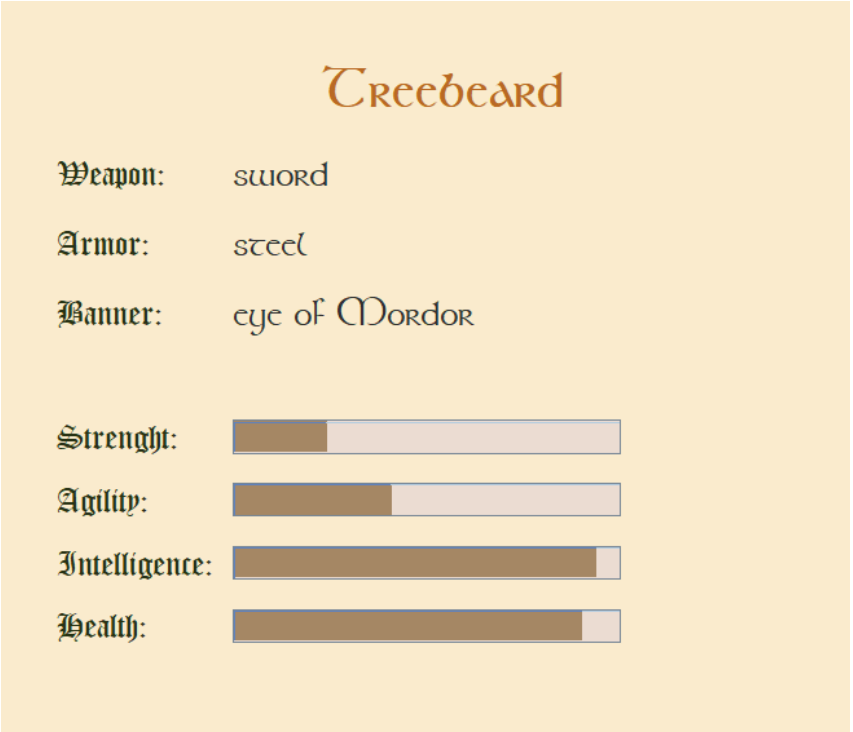

Рис. 9
характеристики и снаряжение орков зависят от племени и типа орка

4. Сборка проекта осуществляется в системе Maven. Ключевые выдержки из файла pom.xml

```
<dependencies>
        <dependency>
            <groupId>com.github.javafaker</groupId>
            <artifactId>javafaker</artifactId>
            <version>1.0.2</version>
        </dependency>
    </dependencies>
```

```
    <build>
       <plugins>
           <plugin>
               <groupId>org.apache.maven.plugins</groupId>
               <artifactId>maven-shade-plugin</artifactId>
               <version>3.2.4</version>
               <executions>
                   <execution>
                       <phase>package</phase>
                       <goals>
                           <goal>shade</goal>
                       </goals>
                       <configuration>

<finalName>LordOfTheRings-1.0-RELEASE-fat</finalName>
                           <transformers>
                                                                   <transformer
implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTra
nsformer">

<mainClass>mephi.b22901.kateero.laba2.Laba2</mainClass>
                               </transformer>
                           </transformers>
                       </configuration>
                   </execution>
               </executions>
           </plugin>
       </plugins>
   </build>
```

Ссылки на работу

1. На репозиторий https://github.com/kateero/lordOfTheRings
2. На релиз https://github.com/kateero/lordOfTheRings/releases/tag/v1.0
3. На скачивание jar-файла https://github.com/kateero/lordOfTheRings/releases/download/v1.0/LordOfTheRings-1.0-RELEASE-fat.jar
4. Команда для запуска файла

```
java -jar путь до файла\LordOfTheRings-1.0-RELEASE-fat.jar
```

Выводы

В ходе выполнения лабораторной работы были успешно изучены и применены ключевые порождающие паттерны проектирования, такие как *Строитель, Абстрактная фабрика, Фабричный метод и Директор*, что позволило создать гибкую и расширяемую систему для генерации орков с уникальными характеристиками и снаряжением в зависимости от их племени и роли.

Реализация графического интерфейса на Swing наглядно продемонстрировала практическую пользу этих паттернов, упростив процесс конфигурации и визуализации объектов, а также закрепив понимание их взаимодействия в реальных проектах.