МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОСИЙСКОЙ ФЕДЕРАЦИИ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**Лабораторная работа №2**

**По дисциплине «Технология программирования»**

**на тему**

**Разработка графического интерфейса приложения**

Группа: АВТ-808

Студент: Попов Д.А.

Новосибирск, 2020

***Задание:***

1. Познакомиться с основными графическими библиотеками Java - AWT и Swing и их основными компонентами. Изучить классы менеджеров компоновки, классы создания меню приложения, основных и диалоговых окон и т.д.

2. Для хранения генерируемых объектов использовать динамический массив объектов. Для обеспечения доступа всем элементам приложения и создания только одного экземпляра массива объектов используйте структурный паттерн Singleton.

3. При реализации шаблона Singleton используется класс Singleton. В этом классе определяется закрытый конструктор, имеется закрытая статическая переменная, в которой хранится ссылка на единственный экземпляр данного класса, а также определен статический метод доступа, возвращающий ссылку на этот экземпляр. Так же класс содержит объект коллекции и открытые методы доступа к ней. Таким образом можно в любом месте приложения получить ссылку на объект класса Singleton и доступ к коллекции объектов.

4. Доработать программу, созданную в лабораторной работе № 1:

1) поделить рабочую область окна приложения на 2 части. Визуализация переносится в одну часть окна, панель управления в другую;

2) добавить кнопки «Старт» и «Стоп» в панель управления. Они должны запускать и останавливать симуляцию соответственно. Если симуляция остановлена, то кнопка «Стоп» должна блокироваться. Если симуляция идет, то блокируется кнопка «Старт». Клавиши B и E должны функционировать по-прежнему;

3) добавить переключатель «Показывать информацию», который разрешает отображение модального диалога из 7 пункта задания;

4) добавить группу из 2 исключающих переключателей: «Показывать время симуляции» и «Скрывать время симуляции». Клавиша T должна функционировать по-прежнему;

5) используя различные менеджеры компоновки, реализовать интерфейс пользователя согласно индивидуальному заданию;

6) добавить в программу главное в меню и панель инструментов, в которых продублировать основные команды вашего интерфейса пользователя;

7) при остановке симуляции должно появляться модальное диалоговое окно (при условии, что оно разрешено) с информацией о количестве и типе сгенерированных объектов, а также времени симуляции. Вся информация выводится в элементе TextArea, недоступном для редактирования. В диалоговом окне должно быть 2 кнопки: «ОК» и «Отмена». При нажатии на «ОК» симуляции останавливается, а при нажатии на «Отмена», соответственно продолжается;

8) предусмотреть проверку данных вводимых пользователем. При вводе неверного значения обрабатывать исключительную ситуацию: выставлять значение по умолчанию и выводить диалоговое окно с сообщением об ошибке;

9) Реализовать следующие элементы управления:

- Периоды рождения объектов – текстовые поля;

- Для задания вероятностей рождения объектов комбобокс и список (шаг значений 10%);

- Дополнить интерфейс поясняющими метками.

***Описание структуры программы:***

**Класс Main** – основной класс. В нем создается рабочее окно программы, запускается таймер и в нем же добавлены слушатели событий нажатий кнопок.

**Класс Habitat** отвечает за инициализацию и обновление окна программы.

**Класс MyComponent** – наследник класса *Component*. В нем перегружен метод *paintComponent(Graphics g),* в котором происходит отрисовка компонентов в окне.

Реализован паттерн проектирования *Factory Method*, содержащий следующие компоненты:

**Интерфейс AbstractFactory** – функциональный интерфейс с единственным абстрактным методом *House create()*

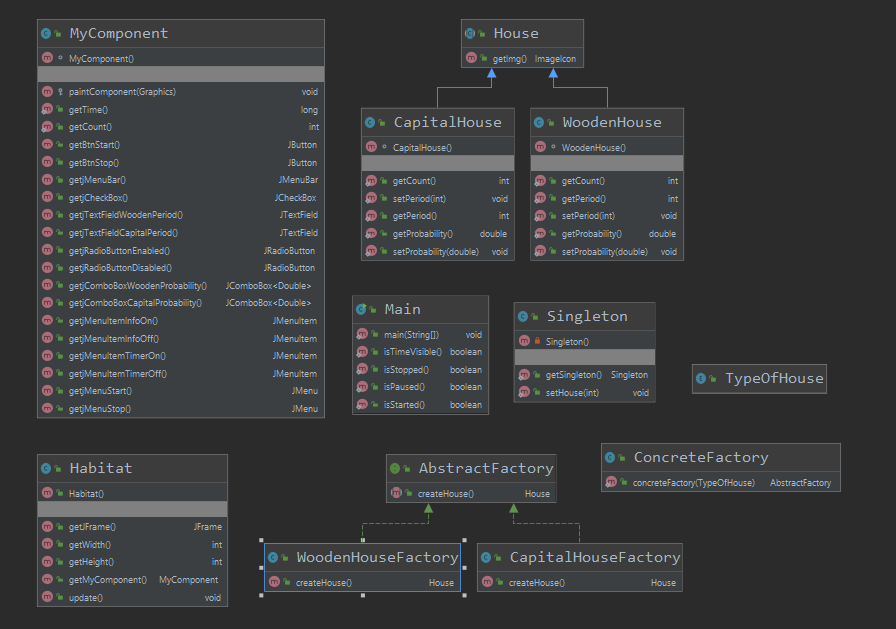
**Класс House** – абстрактный класс, содержащий поле с изображением дома и метод *getImg(),* возвращающий это изображение.

**Перечисление TypeOfHouse** содержит 2 константы типа домов: *WOODEN* и *CAPITAL*.

**Класс ConcreteFactory** имеет единственный метод *concreteFactory(TypeOfHouse typeOfHouse)* принимающий тип дома, капитальный или деревянный, и возвращающий, в зависимости от выбора, нужную фабрику.

**Классы WoodenHouseFactory** и **CapitalHouseFactory** реализуют интерфейс *AbstractFactory* и возвращают объекты готовых деревянных или капитальных домов.

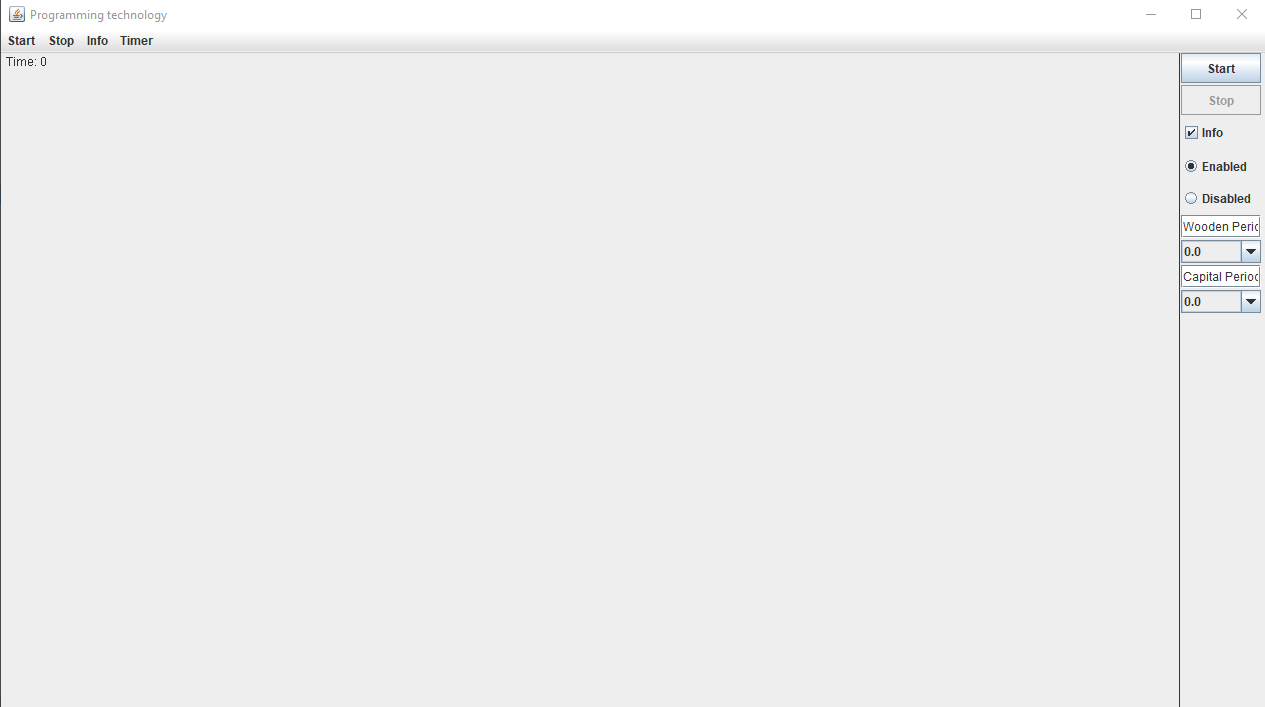
**Классы WoodenHouse** и **CapitalHouse** – наследники абстрактного класса *House*. Содержат поля периода и шанса генерации и количества сгенерированных домов. В конструкторе создается изображение дома и увеличивается счетчик количества домов текущего типа.

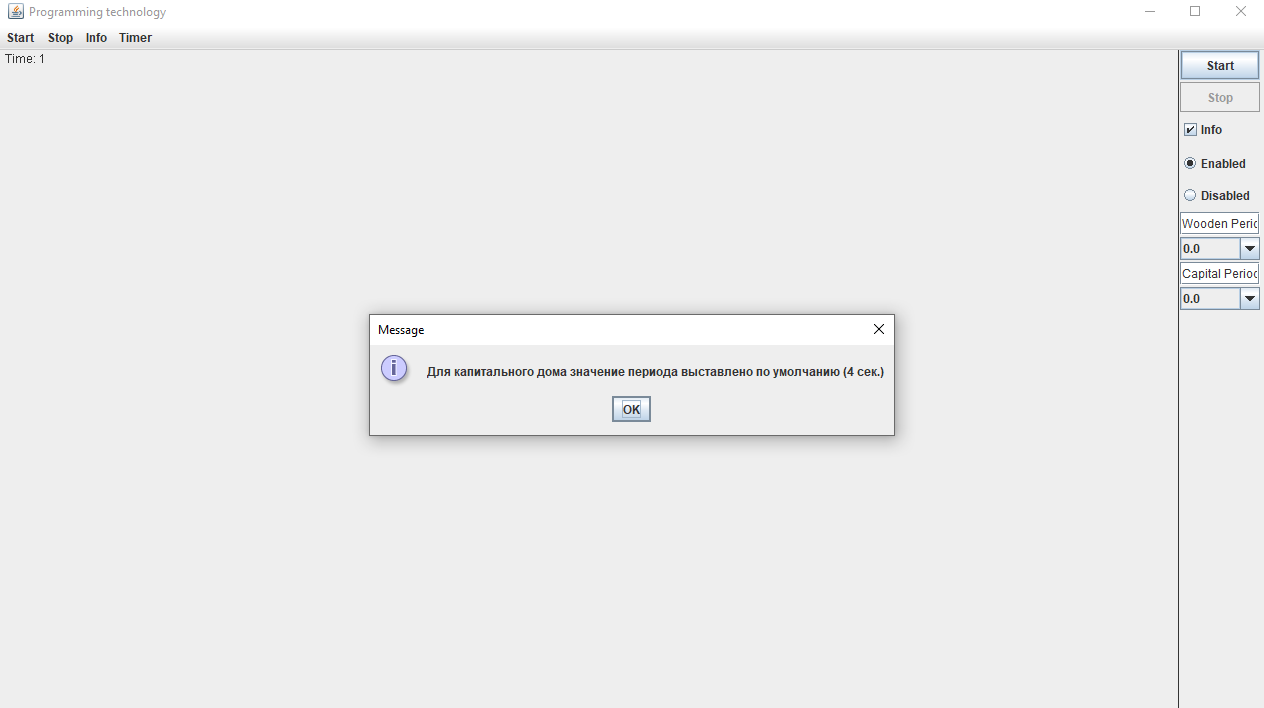


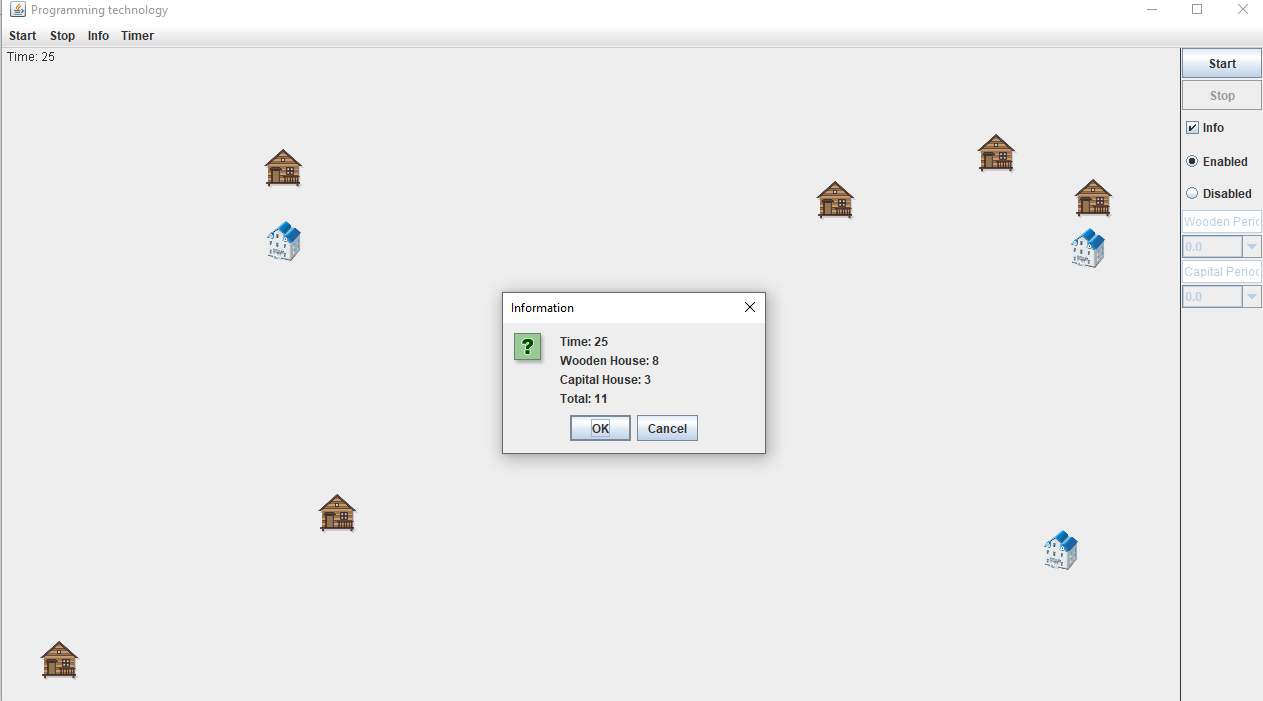
*Рис.1 – UML Диаграмма*

***Описание доработки программы:***

Область окна поделена на 2 часть, слева генерируются дома, справа представлено меню с возможностью запуска, остановки симуляции, выбора отображения информации по завершении и таймера, текстовые поля для ввода периода генерации и комбобоксы для задания вероятностей генерации. При остановке симуляции выводится диалоговое окно со статистикой, можно выбрать продолжить симуляцию или остановить. Пункты меню продублированы В меню бар сверху.









*Рис.2-5 – Демонстрация работы программы*

***Вывод:***

В процессе лабораторной работы были получены навыки по работе с библиотеками Swing и AWT, разработан графический интерфейс программы.