Практическая работа №4.

1. Работа с графическими изображениями.

1.1. **Задача:** Изучение теоретического материала. Закрепление полученных знаний путем разработки приложения на языке программирования Java в среде разработки eclipse.

1.2.**Тип занятий:** исследовательский.

1.3.**Содержание:** теоретический материал, задания для самостоятельного выполнения.

2.Теоретический материал к уроку

2.1. **Загрузка изображения из файла.**

Для загрузки изображения из файла используется команда:

ImageIcon ii=**new** ImageIcon("путь к нашему файлу с расширением файла")

Image img=ii.getImage();

 значение присваивается переменной типа Image(класс для работы с изображениями), пример:

ImageIcon ii=new ImageIcon("C://fon.png");

fon=ii.getImage();

В данном примере мы загружаем изображение в переменную fon, изображение у нас лежит на диске c, если мы перенесем изображение в другую папку, программа не найдет его, если изображение находится не внутри папки нашего проекта, необходимо указывать полный путь к нему, если же изображение находится в папке проекта, то достаточно указать ./ далее его имя и расширение, пример:

ImageIcon ii=new ImageIcon("src/fon.png ");

fon=ii.getImage();

2.2. **Ключевое слово this.**

Ключевое слово this – это ссылочная переменная в Java, которая ссылается на текущий объект. Его можно использовать для ссылки на переменную экземпляра текущего класса. Он может использоваться для вызова или запуска текущего конструктора классов.

2.3. Библиотеки:

javax.swing - библиотека для создания графического интерфейса для программ на языке Java.

java.awt - содержит все классы для того, чтобы создать пользовательские интерфейсы и для рисования графики и изображений.

javax.imageio - библиотека для работы с изображениями.

java.io–библиотека для работы с потоками.

2.4. Метод **drawImage**  тянет изображение в определенном расположении:

drawImage(Image img,

int x, int y,

ImageObserver observer);

x,y расположение определяют позицию для верхнего левого угла изображения, observer параметр обычно - null.

2.5. **Таймер**

Команда для задания таймера:

Timer имя\_таймера = **new** Timer(частота выполнения,**new** ActionListener() {

**public void** actionPerformed(ActionEvent e) {

блоккоманд;

}

});

Пример:Timer timer = **new** Timer(100,**new** ActionListener() {

**public void** actionPerformed(ActionEvent e) {

System.out.println("WOW!");

}

});

Запуск таймера осуществляется следующей командой: имя\_таймера.start();

Пример:timer.start();

2.6. **KeyListener – обработка событий клавиатуры**

Работая с приложением, которое имеет графический интерфейс, пользователь прибегает к помощи не только мыши, но и клавиатуры. JavaSwing даёт возможность разработчику приложения обработать различные события, которые поступают от клавиатуры в то время, когда пользователь нажимает клавиши. Давайте посмотрим, что необходимо сделать, чтобы иметь возможность слушать события клавиатуры

KeyListener имеет три метода: keyTyped, keyPressed и keyReleased.

Когда вызывается каждый из методов системой? Метод keyTyped вызывается системой каждый раз, когда пользователь нажимает на клавиатуре клавиши символы Unicode. Метод keyPressed вызывается системой в случае нажатия любой клавиши на клавиатуре. Метод keyReleased вызывается при отпускании любой клавиши на клавиатуре. Каждый раз, когда пользователь нажимает клавиши на клавиатуре и система вызывает методы keyTyped, keyPressed и keyReleased, в качестве параметра им передается объект KeyEvent, который содержит всю необходимую информацию о произошедшем событии. Отсюда можно узнать код клавиши, которая была нажата – метод getKeyCode. Стоит сказать, что события от клавиатуры будут генерироваться системой только тогда, когда компонент, который мы слушаем, находится в фокусе.

27 - код клавиши esc

37 - код клавиши стрелка влево

39 -код клавиши стрелка вправо.

2.7. **extends** это ключевое слово, предназначенное для расширения реализации какого-то существующего класса. Создается новый класс на основе существующего, и этот новый класс расширяет (extends) возможности старого.

**implements** это ключевое слово, предназначенное для реализации интерфейса (interface).

3. Задания

3.1. уровень 1

1. Откройте заготовку проекта NewYearGame.
2. Создайте 1 класс, не содержащий метод main с именем Panel

3.В классе Panel объявите 3 переменные(переменные объявляем сразу после описания класса, не вставляя ни в конструктор, ни в метод):

**private** Image img; // Закрытая Переменная класса типа изображение, в которую загружается картинка

**private** Image fon; // Закрытая Переменная класса типа изображение, в которую загружается фон

**public** **int** x = 350; // Открытая Переменная класса, используется для местоположения картинки

4. Импортируйте в класс Panel библиотеки java.awt.\*, javax.imageio.\*, javax.swing.\*, java.io.\* (для импорта библиотек перед описанием класса указываем команду import и указанное имя библиотеки, далее ставим ;).

5. Сделайте класс Panel наследником от класса JPanel (контейнер) (внимание: у нас уже есть класс, просто после его имени добавляем оператор наследования и имя класса, от которого делаем наследование).

6. Для создания анимации нам понадобятся 2 изображения: фона и картинки, которую будем двигать, сохраните их под названиям:

* fon.jpg
* img.png

7. переместите их в свою папку проекта

8. Добавьте в класс Panel конструктор класса(конструктор класса добавляется внутри класса командой public имя такое же как у нашего класса() {})

9. В конструктор класса Window добавьте вызов класса Panel, нужно это сделать до команды setVisible:

Panel gameP=**new** Panel();

Container con = getContentPane();

con.add(gameP);

10.  В конструкторе класса Panel будут загружаться в программу изображение (img.png) и фона(

Пример:

// Загрузка изображения из файла

ImageIcon  fonFromFile=**new** ImageIcon("./fon.jpg");

fon=fonFromFile.getImage();

По примеру загрузите изображение второй картинки самостоятельно, то есть будет два блока с прописыванием пути к изображению и двум разным переменным будет присваиваться значение, но в каждом будет грузиться свое изображение. Внимание: это только получение изображения, если сейчас запустить программу их еще не будет на экране.

11. Теперь в класс Panel добавим метод (после окончания конструктора, но все еще внутри класса), который отрисовывает графические объекты на панели:

**public** **void** paintComponent(Graphics gr) {

// Выполнить отрисовку сначала самого окна

**super**.paintComponent(gr);

gr.drawImage(fon, 0, 0,800,600, **null**); // Рисование фона

gr.drawImage(img, x, 50,120,120, **null**); // Рисование картинки

}

12. Запустите программу на выполнение. у вас должны появиться фон и картинка.

13. Перед тем как добавлять таймер в ваш класс Panel обязательно подключите следующие библиотеки:

java.awt.event.ActionEvent;

java.awt.event.ActionListener; f

javax.swing.Timer;

14. В конструктор класса Panel добавьте таймер для перерисовки поля с частотой выполнения 50 (это означает игровое поле будет перерисовываться 20 раз в секунду). В блоке команд будет метод repaint(); // Запуск метода перерисовки поля.

15. Запустите таймер на выполнение.

16. Объявите в классе Window переменную сразу после объявления класса: **public** Panel gameP;

17. Добавьте служебный класс MyKey в класс Window с реализацией интерфейса KeyListener и подключите библиотеку java.awt.\*.:

//Обработчик событий нажатий на клавиши

**private** **class** myKey **implements** KeyListener

{

// Метод, который срабатывает при нажатии

**public** **void** keyPressed(KeyEvent e)

{

// Получение кода нажатой клавиши

**int** key\_ = e.getKeyCode();

// Выход из программы, если нажат - Esc

**if** (key\_==27) System.*exit*(0);

**else** **if** (key\_==37) // Если нажата стрелка влево

{

// Контроль перемещения влево за пределы окна

**if** (gameP.x-30>-48) gameP.x-=30;

**else** gameP.x=752;

}

**else** **if** (key\_==39) // Если нажата стрелка вправо

{

// Контроль перемещения вправо за пределы окна

**if** (gameP.x+30<752) gameP.x+=30;

**else** gameP.x=-48;

}

}

**public** **void** keyReleased(KeyEvent e) {}

**public** **void** keyTyped(KeyEvent e) {}

}

18. В конструктор класса Window перед установкой фокуса setFocusable подключите обработчик события для клавиатуры к окну следующей строчкой: addKeyListener(**new** myKey());

19. Найдите блок команд: Panel gameP=**new** Panel();

Container con = getContentPane();

con.add(gameP);

Измените таким образом, чтобы использовалась глобальная переменная класса.

20. Запустите программу на выполнение и нажмите клавишу esc, осуществился ли выход из программы? если да, запустите еще раз и теперь проверьте работу клавиш стрелка влево и вправо, если ловец перемещается по экрану - все правильно.

21. Добавьте еще одно изображение, чтобы можно было управлять еще одним предметом.

22. измените управление на клавиши A и S для второй картинки, чтобы узнать код клавиш можно вывести их на экран в классе MyKey командой System.out.println(e.getKeyCode());