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

2.2 **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 (в папке src), то достаточно указать src/ далее его имя и расширение, пример:

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

fon=ii.getImage();

2.3. **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.

2.4. 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.

2.5. 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.

3. Задания

3.1. уровень 1

1. Откройте ваш проект Game.
2. Создайте еще два класса, не содержащих метод main:

Pole- для игрового поля

Predmet- для подарков, которые будут в игре

1. В классе Pole объявите 4 переменные(переменные объявляем сразу после описания класса, не вставляя ни в конструктор, ни в метод):

**private** Image catcher; // Закрытая Переменная класса типа изображение, в которую загружается ловец

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

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

**private** Image endGame; // Изображение Окончания игры

1. Импортируйте в класс Pole библиотеки java.awt.\*, javax.imageio.\*, javax.swing.\*, java.io.\* (для импорта библиотек перед описанием класса указываем команду import и указанное имя библиотеки, далее ставим ;).
2. Сделайте класс Pole 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).
3. Для создания игры нам понадобятся изображения шапки для ловли предметов, изображение фона, конца игры и 7 предметов, сохраните их под названиям:

* fon.png
* endGame.png
* catcher.png
* predmet0.png
* predmet1.png
* predmet2..png
* predmet3.png
* predmet4.png
* predmet5.png
* predmet6.png

1. переместите их в свою папку проекта
2. Добавьте в класс Pole конструктор класса(конструктор класса добавляется внутри класса командой public имя такое же как у нашего класса() {})
3. В конструктор класса Window вместо вызова метода класса DrawSquare добавьте вызов класса Pole, нужно это сделать до команды setVisible:

Pole gameP=**new** Pole();

Container con = getContentPane();

con.add(gameP);

1. в конструкторе класса Pole будут загружаться в программу изображение шапки фона(fon.png) и окончания игры(endGame.png).

Пример:

// Загрузка изображения ловца из файла

ImageIcon catcherFromFile=**new** ImageIcon("src/catcher.png");

catcher=catcherFromFile.getImage();

По примеру загрузите изображение фона и окончания игры самостоятельно, то есть будет три блока с прописыванием пути к изображению и трем 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.

1. Теперь в класс Pole 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(catcher, x, 465,120,120, **null**); // Рисование ловца

}

1. Запустите программу на выполнение. у вас должны появиться фон и ловец.
2. Импортируйте в класс Predmet библиотеку java.awt.\* (читаем выше как импортировать п.4).
3. Объявите в классе Predmet три переменные (также после описания класса вне методов и конструктора):

**public** Image img; // Изображение предмета

**public int** x,y; // Положение предмета на игровом поле, в пикселях, x - отступ слева, y - отступ сверху

**public** Boolean act; // Переменная логического типа, показывающая активность предмета, есть он игровом поле или нет

1. Добавьте конструктор класса с параметром для передачи изображения из вызывающего класса (прямо копируем блок в наш класс после переменных):

// Конструктор класса

**public** Predmet(Image img){

**this**.img = img; // Передача изображения из круглых скобок Конструктора класса в переменную класса

act=**false**; // Изначально делаем предмет неактивным, отсутствующим на игровом поле

}

1. Добавьте 2 метода в класс Predmet:

// Метод, выполняющий отрисовку предмета на игровом поле, если он активен

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

if (act==true){

gr.drawImage(img,x,y,120,120,null); // Рисование изображения

}

}

// Метод вывода предмета сверху игрового поля

**public void** start(){

y = 0; // Отступ cверху в пикселях

x = (**int**)(Math.*random*()\*700); // Отступ слева в пикселях, получаем случайным образом от 0 до 700

act = **true**;

}

1. Теперь когда есть метод start(), добавьте его вызов в конструкторе класса Predmet, для вызова метода используется:

имя метода(параметры); //в нашем случае параметры отсутствуют , значит нужно просто указать имя, круглые скобки после него и точку с запятой.

1. Запустите программу на выполнение. Ничего не изменилось, потому что мы только описали класс, но не добавили загрузку изображений.
2. В класс Pole добавьте описание массива типа нашего класса Predmet:

**private** Predmet[] gamePredmet; // Массив предметов добавляем в начало класса после всех уже объявленных переменных

20) и в конструктор класса Pole добавьте в цикле загрузку предметов:

// Загрузка семи изображений предметов

gamePredmet = **new** Predmet[7];

**for** (**int** i=0;i<7;i++)

}

21) В метод paintComponent добавьте отрисовку предметов:

// Цикл, который отображает предметы на игровом поле

**for** (**int** i=0;i<7;i++) {

gamePredmet [i].draw(gr); // Отображение подарка

}

22) Запустите программу на выполнение, подарки будут появляться в разных местах.