Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Алтайский государственный технический университет

им. И.И. Ползунова»

Факультет (институт) Информационных технологий

Кафедра Прикладная математик

Отчет защищен с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.М. Старолетов

(подпись преподавателя) (инициалы, фамилия)

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

Отчет

по лабораторной (практической) работе № 2

по дисциплине Архитектурное проектирование и паттерны программирования

(наименование дисциплины)

ЛР 09.03.04.10.000 ОТ

(обозначение документа)

Студенты группы ПИ-42 Е.А. Колесников

(инициалы, фамилии)

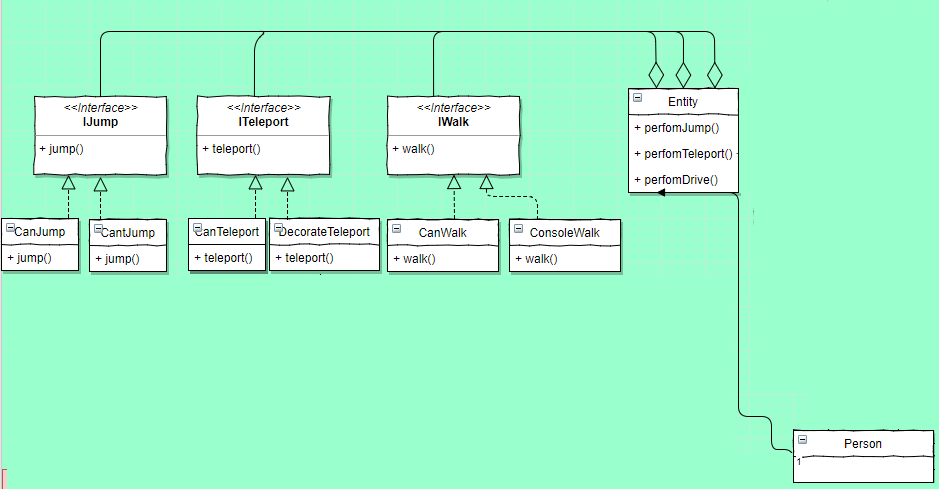
Преподаватель Доцент к. канд. физ.-мат. наук С.М.Старолетов

(должность, ученое звание) (инициалы, фамилия)

Барнаул 2017

**Задание:** реализовать паттерны Делегирование и GRASP в NeytronGame

**UML диаграмма:**

****

public interface IJump {  
 //Реализация прыжка  
 public void jump(Entity e);  
}

public interface ITeleport {  
 //Реализация телепорта  
 public void teleport(Entity e, int orientation);  
 public int getVelocityX();  
 public int getVelocityY();  
 public void setVelocityX(int velocityX);  
 public void setVelocityY(int velocityY);  
  
}

public interface IWalk {  
 public void walk(Entity e, int orientation);  
}

public class CanJump implements IJump {  
 public void jump(Entity e) {  
 // X := X0 + t \* V \* Cos(A);  
// Y := Y0 + t \* V \* Sin(A) - g \* Sqr(t);  
// X, Y - искомые координаты.  
// Gdx.graphics.getDeltaTime() - текущее время.  
// X0, Y0 - координаты начала прыжка.  
// e.velocity.x - начальная скорость.  
// A - угол прыжка (в радианах).  
// g - ускорение свободного падения (9.81).  
 // double new\_position\_x = e.velocity.x \* Gdx.graphics.getDeltaTime() \* Math.cos(70);  
 double new\_position\_y = e.velocity.y \* Gdx.*graphics*.getDeltaTime() \* Math.*sin*(70) - 9.8 \* Math.*sqrt*(Gdx.*graphics*.getDeltaTime());  
  
 e.setPosition(new Vector3(e.getPosition().x/\*+(float)new\_position\_x\*/, e.getPosition().y + (float) new\_position\_y, 0));  
  
 e.getPlayer().y = e.getPosition().y;  
 e.onGround = false;  
  
 }  
}

public class CantJump implements IJump {  
 public void jump(Entity e) {  
 }  
}

public class CanWalk implements IWalk {  
 public int frames = 1;  
  
 @Override  
 public void walk(Entity e, int orientation) {  
 e.getPosition().x += e.velocity.y \* orientation \* Gdx.*graphics*.getDeltaTime();  
 e.getPlayer().x = e.getPosition().x;  
 if (frames == 11)  
 frames = 1;  
 switch (orientation) {  
 case 1:  
 e.setPlayerImage(new Texture(Gdx.*files*.internal(e.getPath\_player\_image()+"/" + (frames++) + ".png")));  
 break;  
 case -1:  
 e.setPlayerImage(new Texture(Gdx.*files*.internal(e.getPath\_player\_image()+"/" + (frames++) + "l.png")));  
 break;  
 }  
  
 }  
}

public class CantWalk implements IWalk {  
 @Override  
 public void walk(Entity e, int orientation) {  
  
 }  
}

public class CanTeleport implements ITeleport {  
 private int velocityX;  
 private int velocityY;  
  
 @Override  
 public int getVelocityX() {  
 return velocityX;  
 }  
  
 @Override  
 public void setVelocityX(int velocityX) {  
 this.velocityY=velocityX;  
 }  
  
 @Override  
 public void setVelocityY(int velocityY) {  
 this.velocityY=velocityY;  
 }  
  
 @Override  
 public int getVelocityY() {  
 return velocityY;  
 }  
  
 public CanTeleport() {  
 velocityX=400;  
 velocityY=0;  
 }  
 public CanTeleport(int velocity) {  
 this.velocityX = velocity;  
 }  
 @Override  
 public void teleport(Entity e, int orientation) {  
 e.setPosition(new Vector3(e.getPosition().x + velocityX \* orientation, e.getPosition().y+velocityY, 0));  
 }  
}

**Тесты:**

Ходьба:



Прыжок:

