PROGRAMACION ORIENTADA A OBJETOS

PROPUESTA DE PROYECTO

Semestre 2014-2015/I

[No de proyecto](#h.z8dwovkpuhl8)

[Nombre del proyecto](#h.mnxc81u1z897)

[Integrantes del equipo](#h.dntok8juv9p3)

[Objetivo del proyecto](#h.bsu9evf8nxs1)

[Descripción del proyecto](#h.eivcmejh9ing)

[Descripción e imágenes de cada nivel](#h.dipcv9h52uj1)

[ivel Principiante](#h.30shymbyey24)

[Nivel Intermedio](#h.lp6l46qd1968)

[Nivel Avanzado](#h.iwgxt4txxcht)

[Diagrama de clases UML](#h.g0vt53pc7r37)

[Características y comportamiento de cada clase](#h.mtuw3jfplyj6)

[Herencia y polimorfismo](#h.k1a372vuic1q)

[Cronograma de actividades (plan de trabajo)](#h.30zz6o5p35xo)

[Bitácora de actividades (historial)](#h.rr6kuez6yf7u)

1. No de proyecto

201

1. Nombre del proyecto

Mundo Ninja

1. Integrantes del equipo

Ricardo Jose Huizar Lopez Clave: 209037

Cristian Eduardo Lopez Amaro Clave: 212361

1. Objetivo del proyecto
2. -Probar y desarrollar las habilidades del jugador para calcular la distancia, altura maxima, y velocidad que puede adquirir un objeto con un cierto impulso.
3. -probar la cantidad de veces que el jugador puede repetir dicha accion sin perder.
4. Descripción del proyecto

El juego consiste en un ninja el cual según la historia, está en un entrenamiento, el entrenamiento es ir saltando unos postes los cuales están parados sobre un rio de forma vertical, con la barra espaciadora el jugador podrá saltar solo con cierto impulso, si saltas mas o menos de lo debido el ninja podría caer al agua o chocar con un pajaro, e incluso si no llegas a saltar el enemigo pulpo puede hacer que pierdas.

1. Descripción e imágenes de cada nivel

El primer tiene la menor dificultad, en el cual solo se deben saltar los postes.

En el segundo se agrega el enemigo Pulpo el cual esta nadando en el agua y regularmente brinca en donde estas para atacarte.

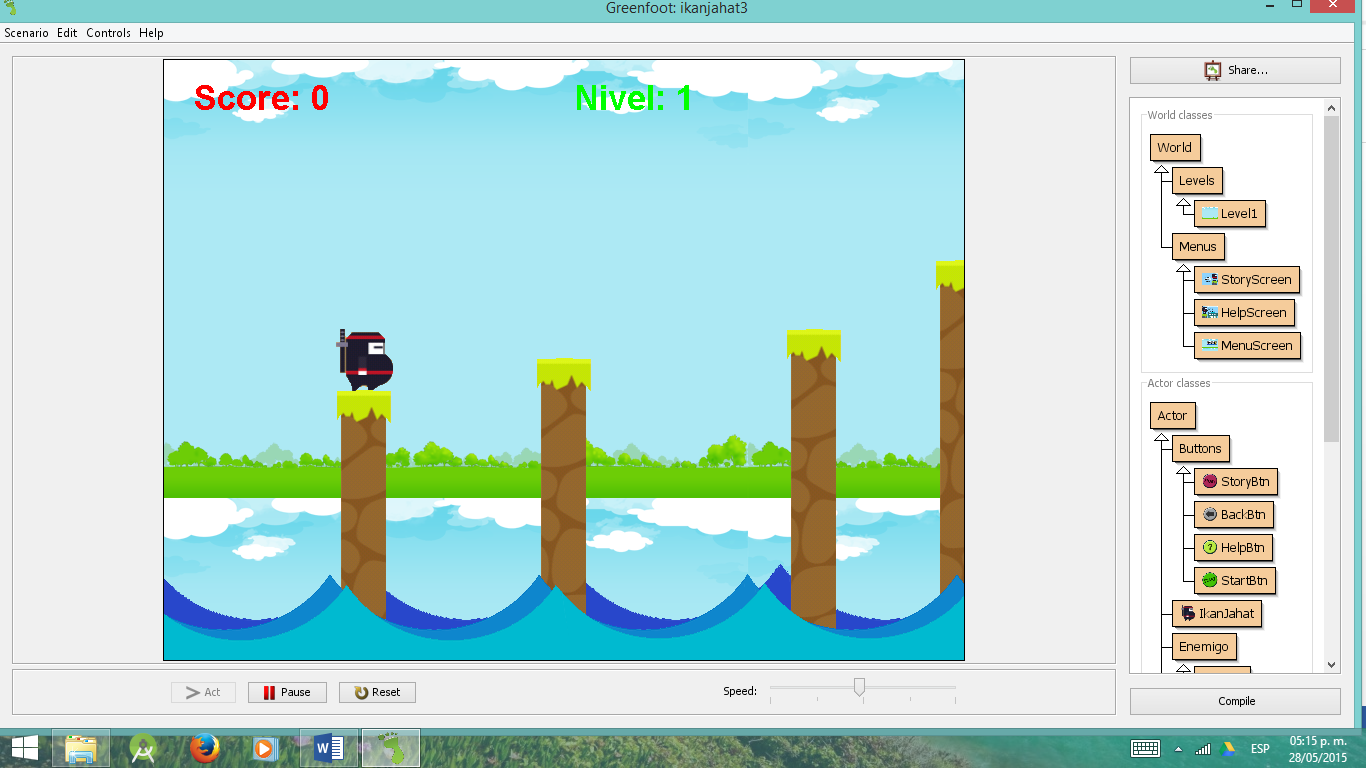
El tercer nivel es un poco mas complicado, ya que se agrega un enemigo ave, el cual pasa volando.

Por ultimo el cuarto nivel tiene un enemigo fijo, consiste en una cadena de espinas la cual no te permite saltar completamente.

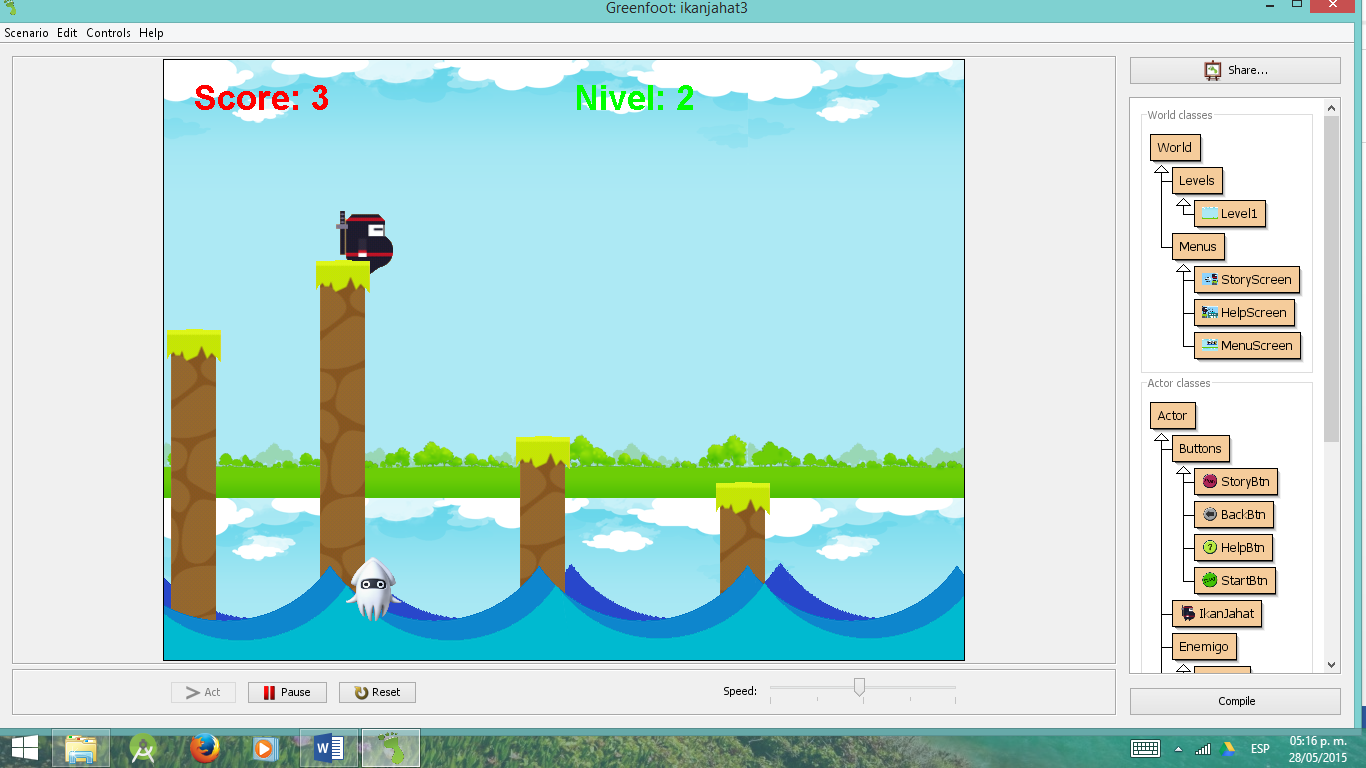
Deberás ser muy hábil para pasar todos estos niveles

Se el mejor ninja!

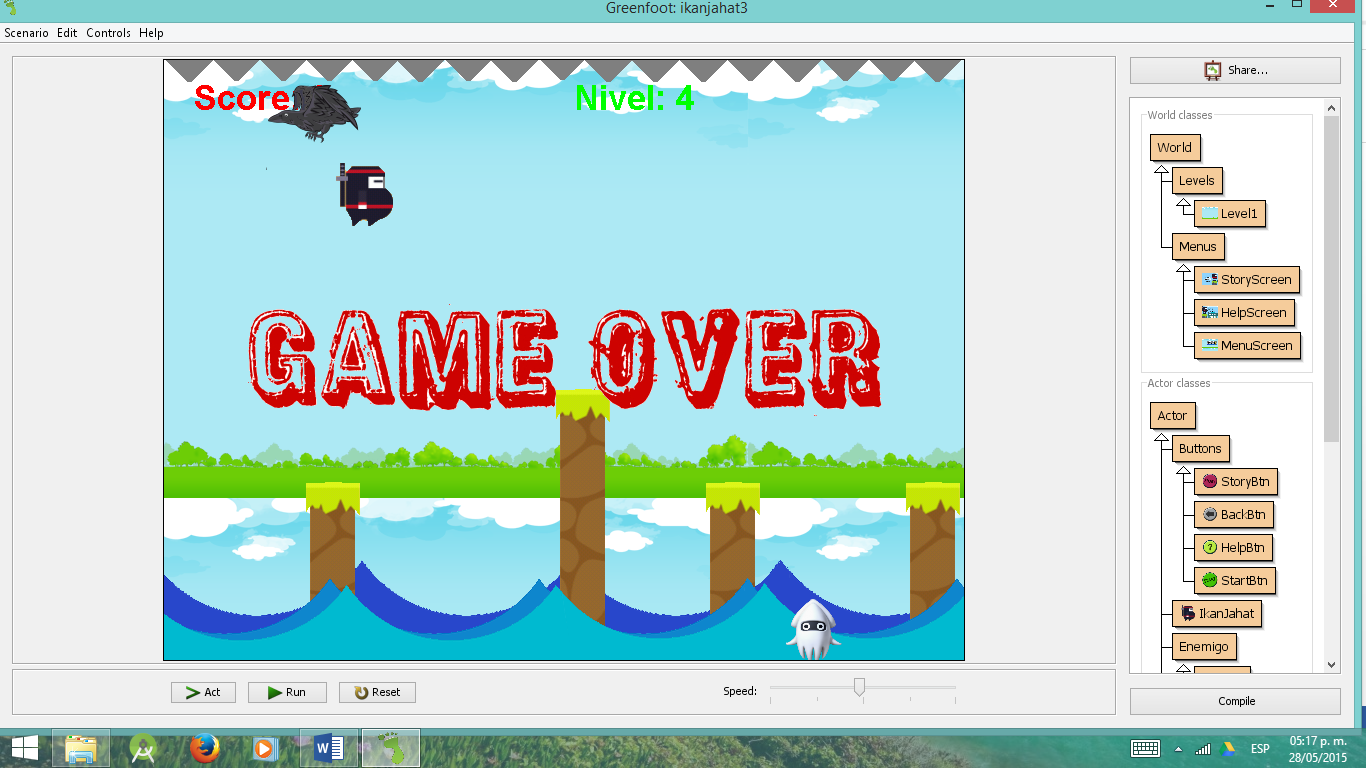
* Nivel Principiante



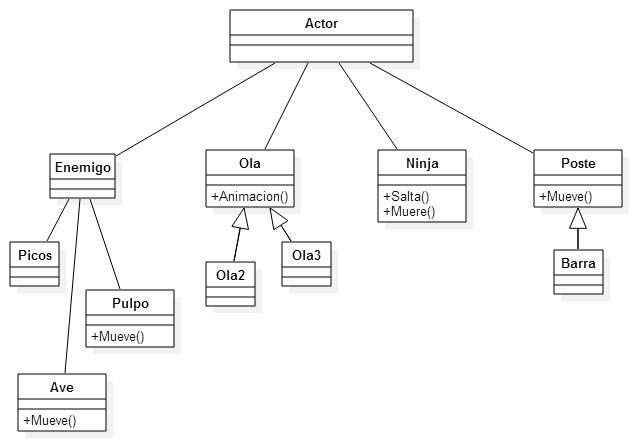
* Nivel Intermedio



* Nivel Avanzado



1. Diagrama de clases UML



1. Características y comportamiento de cada clase

|  |  |
| --- | --- |
| Nombre de la clase: | Levels |
| Características: | Clase principal en donde se crean todos los objetos |
|  |  |
|  |  |
| Comportamiento: |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Nombre de la clase: | Ikanjahat3 |
| Características: | Clase del Ninja (Ikanjahat en Chino), aquí es en donde se lleva el conteo de los puntos, el movimiento, crea enemigos, en si es la base del juego. |

|  |  |
| --- | --- |
| Nombre de la clase: | Enemigos |
| Características: | Padre de las clases Pulpo y Ave |
|  |  |
|  |  |
| Comportamiento: | Son los enemigos de ninja, al tocarse se detiene el juego. |
|  |  |
|  |  |

*”*

1. Herencia y polimorfismo

Las clases Ave, Pico y Pulpo heredan de la clase Enemigo

Todos Atacan pero responden de manera diferente respecto al ninja.

1. Cronograma de actividades (plan de trabajo)

|  |  |  |
| --- | --- | --- |
| Fecha de Inicio | Fecha de Término | Actividad por realizar |
| 20 Abril | 28 Mayo | Codigo |
| 28 Abril | 28 Mayo | Manual de Programador |
| 28 Mayo | 28 Mayo | video |
| 28 Mayo | 28 Mayo | Link A GreenFoot |

*\* Para añadir más renglones a la tabla debe seleccionar primero la tabla y después desde el menú “****Tabla****” seleccionar la opción “****Insertar una fila debajo****”*

1. Bitácora de actividades (historial)

Esta parte será llenada durante la elaboración del proyecto. Aquí se deben describir cada una de las actividades realizadas desde la propuesta hasta la entrega del proyecto.

|  |  |
| --- | --- |
| Fecha (dd/mm/aa) | Descripción de la actividad realizada. |
| 05 Abril 2015 | Propuesta de proyecto  Todo comenzó con una idea de juego, se elaboró la propuesta de proyecto del juego Mundo Ninja. |
| 20 Abril 2015 | Codigo  Comenzamos a escribir el código, es la parte mas compleja del proyecto. |
| 28 Mayo | Manual de Programador y de usuario  Los manuales se realizaron al finalizar el proyecto. |
|  | Video  El video fue grabado al finalizar el proyecto, para mostrar todas sus funciones. |
| 28 Mayo | Link A GreenFoot  El link a GreenFoot fue obtenido este dia al terminar el proyecto. |
|  |  |

**Class ninjaWorld**

java.lang.Object

greenfoot.World

**ninjaWorld**

public class **ninjaWorld**extends greenfoot.World

|  |  |
| --- | --- |
| **Field Summary** | |
| Ninja | [**n1**](about:blankninjaWorld.html#n1) |
| greenfoot.GreenfootImage | [**nImage**](about:blankninjaWorld.html#nImage) |
| poste | [**p1**](about:blankninjaWorld.html#p1) |
| poste | [**p2**](about:blankninjaWorld.html#p2) |
| poste | [**p3**](about:blankninjaWorld.html#p3) |
| poste | [**p4**](about:blankninjaWorld.html#p4) |
| greenfoot.GreenfootImage | [**pImage**](about:blankninjaWorld.html#pImage) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**ninjaWorld**](about:blankninjaWorld.html#ninjaWorld())()           Clase mundo es donde se crean todos los objetos y se agregan a proyecto en una posicion, tambien se hacen las respectivas llamadas. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**act**](about:blankninjaWorld.html#act())()           Act es el metodo que se repite siempre en cada objeto como si fuera un ciclo. |
| void | [**createPipes**](about:blankninjaWorld.html#createPipes())()           Metodo que agrega un poste con su respectiva barra al mundo. |

|  |
| --- |
| **Methods inherited from class greenfoot.World** |
| addObject, getBackground, getCellSize, getColorAt, getHeight, getObjects, getObjectsAt, getWidth, numberOfObjects, removeObject, removeObjects, repaint, setActOrder, setBackground, setBackground, setPaintOrder, showText, started, stopped |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

**n1**

public Ninja **n1**

**nImage**

public greenfoot.GreenfootImage **nImage**

**p1**

public poste **p1**

**p2**

public poste **p2**

**p3**

public poste **p3**

**p4**

public poste **p4**

**pImage**

public greenfoot.GreenfootImage **pImage**

|  |
| --- |
| **Constructor Detail** |

**ninjaWorld**

public **ninjaWorld**()

Clase mundo es donde se crean todos los objetos y se agregan a proyecto en una posicion, tambien se hacen las respectivas llamadas.

|  |
| --- |
| **Method Detail** |

**act**

public void **act**()

Act es el metodo que se repite siempre en cada objeto como si fuera un ciclo. En este caso se utiliza para mover los postes en caso de que el ninja no este tocando alguna barra la cual se crea en la creacion de un poste tambien

**Overrides:**

act in class greenfoot.World

**createPipes**

public void **createPipes**()

Metodo que agrega un poste con su respectiva barra al mundo. Se adapta a la configuracion de tamaño de juego.

**Class Ninja**

java.lang.Object

greenfoot.Actor

**Ninja**

public class **Ninja**extends greenfoot.Actor

Aqui se encuentra la mayor parte del codigo del juego, mas adelante se describira cada metodo. incluye la constante que ayuda a simular la gravedad. y tambien una variable que ayuda a que el ninja solo pueda dar un brinco en el aire.

**Author:**

Ricardo Huizar

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**Ninja**](about:blankNinja.html#Ninja())() |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**act**](about:blankNinja.html#act())()           Metodo principal que manda llamar la clase mundo. |
| void | [**brincar**](about:blankNinja.html#brincar())()           Metodo que sirve para hacer que el Ninja salte |
| void | [**caer**](about:blankNinja.html#caer())()           Metodo que sirve para que el jugador caiga si no hay Barra debajo de el. |
| void | [**checaPierde**](about:blankNinja.html#checaPierde())()           Metodo que sirve para checar si el Ninja esta en contacto con algun enemigo o se ahogó en el agua. |
| int | [**getXNinja**](about:blankNinja.html#getXNinja())()           Metodo que sirve para enviar la coordenada en X. |
| int | [**getYNinja**](about:blankNinja.html#getYNinja())()           Metodo que sirve para enviar la coordenada en X. |
| boolean | [**siTocaBarra**](about:blankNinja.html#siTocaBarra())()           Metodo que sirve para saber si el Ninja esta tocando una barra, se utiliza en el act. |

|  |
| --- |
| **Methods inherited from class greenfoot.Actor** |
| addedToWorld, getImage, getIntersectingObjects, getNeighbours, getObjectsAtOffset, getObjectsInRange, getOneIntersectingObject, getOneObjectAtOffset, getRotation, getWorld, getX, getY, intersects, isAtEdge, isTouching, move, removeTouching, setImage, setImage, setLocation, setRotation, turn, turnTowards |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

**Ninja**

public **Ninja**()

|  |
| --- |
| **Method Detail** |

**act**

public void **act**()

Metodo principal que manda llamar la clase mundo. aqui se encuentra las condiciones para poder brincar y la funcion para caer.

**Overrides:**

act in class greenfoot.Actor

**brincar**

public void **brincar**()

Metodo que sirve para hacer que el Ninja salte

**caer**

public void **caer**()

Metodo que sirve para que el jugador caiga si no hay Barra debajo de el.

**checaPierde**

public void **checaPierde**()

Metodo que sirve para checar si el Ninja esta en contacto con algun enemigo o se ahogó en el agua.

**getXNinja**

public int **getXNinja**()

Metodo que sirve para enviar la coordenada en X.

**getYNinja**

public int **getYNinja**()

Metodo que sirve para enviar la coordenada en X.

**siTocaBarra**

public boolean **siTocaBarra**()

Metodo que sirve para saber si el Ninja esta tocando una barra, se utiliza en el act.

**Class poste**

java.lang.Object

greenfoot.Actor

**poste**

public class **poste**extends greenfoot.Actor

En esta clase tendremos todo lo relacionado con el poste.

**Author:**

Ricardo Huizar

|  |  |
| --- | --- |
| **Field Summary** | |
| Barra | [**barra**](about:blankposte.html#barra) |
| greenfoot.GreenfootImage | [**bImage**](about:blankposte.html#bImage) |
| greenfoot.GreenfootImage | [**pImage**](about:blankposte.html#pImage) |
| int | [**PIPE\_SPEED**](about:blankposte.html#PIPE_SPEED) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**poste**](about:blankposte.html#poste())() |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**act**](about:blankposte.html#act())()           Metodo act que manda llamar el metodo eliminarFinal. |
| void | [**addBarra**](about:blankposte.html#addBarra())()           Metodo que sirve para agregar una barra al mundo la cual estara en la parte superior del poste |
| void | [**detener**](about:blankposte.html#detener())()           metodo que sirve para detener el movimiento de las pipas |
| void | [**eliminarFinal**](about:blankposte.html#eliminarFinal())()           Metodo que elimina los postes al llegar al final de la pantalla |
| void | [**mover**](about:blankposte.html#mover())()           Metodo que sirve para Comenzar a mover las pipas. |

|  |
| --- |
| **Methods inherited from class greenfoot.Actor** |
| addedToWorld, getImage, getIntersectingObjects, getNeighbours, getObjectsAtOffset, getObjectsInRange, getOneIntersectingObject, getOneObjectAtOffset, getRotation, getWorld, getX, getY, intersects, isAtEdge, isTouching, move, removeTouching, setImage, setImage, setLocation, setRotation, turn, turnTowards |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

**barra**

public Barra **barra**

**bImage**

public greenfoot.GreenfootImage **bImage**

**pImage**

public greenfoot.GreenfootImage **pImage**

**PIPE\_SPEED**

public int **PIPE\_SPEED**

|  |
| --- |
| **Constructor Detail** |

**poste**

public **poste**()

|  |
| --- |
| **Method Detail** |

**act**

public void **act**()

Metodo act que manda llamar el metodo eliminarFinal.

**Overrides:**

act in class greenfoot.Actor

**addBarra**

public void **addBarra**()

Metodo que sirve para agregar una barra al mundo la cual estara en la parte superior del poste

**detener**

public void **detener**()

metodo que sirve para detener el movimiento de las pipas

**eliminarFinal**

public void **eliminarFinal**()

Metodo que elimina los postes al llegar al final de la pantalla

**mover**

public void **mover**()

Metodo que sirve para Comenzar a mover las pipas.