



**TECNOLOGICO NACIONAL DE MEXICO**

**Campus La Laguna**

Ingeniería en Sistemas



Computacionales

**DESARROLLO EN ANDROID**

SEMESTRE: Ene – Jun / 2024

GRUPO: “A” 08 – 09 Hrs

PRACTICA No. U4P01

## Juego básico con gráficos y audio

ALUMNO:

19130971 Francisco Axel Roman Cardoza

PROFESOR:

Ing. Luis Fernando Gil Vázquez

**Torreón, Coah. a 27 de Mayo de 2024**

## Situación Didáctica

Mexi-Publicidad S.A. de C.V. es una empresa de publicidad en medios electronicos que pretende aprovechar la captación de mas usuarios y visitantes a su portal web, para eso intenta aprovechar el impacto mediático de alguna noticia de relevancia plasmándola en un juego para móviles Android de una manera divertida, su estrategia es que al ocurrir una noticia de interés la empresa en un par de horas de a conocer que hay un nuevo juego relacionado con la noticia e invite a visitar su portal para descargarla, con lo que nuevos visitantes serán atraídos. Por ejemplo, ante la noticia del descubrimiento de un asteroide que pasará cerca de la Tierra se lanzó una versión de la app en la que una nave espacial desintegra el asteroide con sus misiles. Dicho juego fue un éxito sin embargo el desarrollador estelar de la empresa se retiró y Mexi-Publicidad ha solicitado a ingenieros del ITL retomar el código del proyecto para usarlo de base para nuevas versiones del juego cuando ocurran noticias relevantes.

Basicamente lo que Mexi-Publicidad necesita es que los elementos del juego tales como la nave, los misiles y asteroides sean cambiados según la noticia, poniendo en el juego las imagenes de los actores y una imagen de fondo acorde al entorno de la noticia. De esta manera con una adaptación rapida de los elementos podrá lanzar la app en un tiempo muy corto despues de ocurrida la noticia.

Indicaciones para la documentación de esta práctica.

El documento de esta práctica contendrá lo siguiente:

**ANALISIS.** Presentar en 2 columnas las imágenes originales del juego y las imágenes que sustituirán a cada una. Describir cómo se implementa la funcionalidad de reproducción del audio.

**DISEÑO.** 1. Maquetado de la Interfaz de Usuario usando la herramienta Balsamiq:


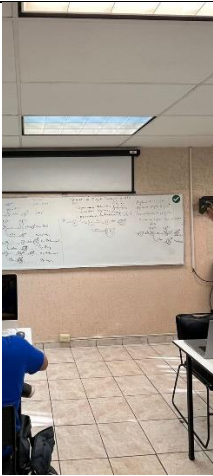










2. Diagrama UML de clases de la aplicación hecho en Visual Paradigm.

**CODIGO.** Código fuente de los archivos .java primero, después los xml de los layouts y al final el archivo de manifiesto.

**PRUEBA DE**

**EJECUCION.** Mini instructivo de usuario incluyendo capturas de pantalla de la aplicación corriendo.

Analisis

Imagen Original	Imagen Sustituta
	
	
	
	
	
	

Diseño

Diseño de interfaz de usuario

## **Diseño de clases**

## Código

### Grafico.java

```
package mx.itl.nc21130561.u4juegoasteroideapp;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.drawable.Drawable;
import android.view.View;
import android.widget.Toast;

/**
 * Created by LGV on 25/07/2015.
 */
public class Grafico {
    private Drawable    drawable;
    private double      posX,
                      posY;
    private double      incX,
                      incY;
    private int         angulo,
                      rotacion;
    private int         ancho,
                      alto;
    private int         radioColision;
    private View        view;
    public static final int MAX_VELOCIDAD = 20;

    public Grafico ( View view, Drawable drawable ) {
        this.view = view;
        this.drawable = drawable;

        ancho = drawable.getIntrinsicWidth();
        alto  = drawable.getIntrinsicHeight();
        radioColision = ( alto + ancho ) / 4;
    }

    public void dibujarGrafico ( Canvas canvas ) {
        canvas.save ();
        int x = (int) ( posX + ancho / 2 );
        int y = (int) ( posY + alto / 2 );
        canvas.rotate ( (float) angulo, (float) x, (float) y );
        drawable.setBounds ( (int) posX, (int) posY, (int) posX + ancho, (int) posY + alto );
        drawable.draw ( canvas );
        canvas.restore();
        int rInval = (int) distanciaE ( 0, 0, ancho, alto ) / 2 + MAX_VELOCIDAD;
        view.invalidate ( x - rInval, y - rInval, x + rInval, y + rInval );
    }

    public static int getMaxVelocidad() {
        return MAX_VELOCIDAD;
    }

    public Drawable getDrawable() {
        return drawable;
    }

    public void incrementaPos () {
        posX += incX;
        if ( posX < -ancho / 2 )
            posX = view.getWidth() - ancho / 2;
        if ( posX > view.getWidth() - ancho / 2 )
            posX = -ancho / 2;
    }
}
```

```
        posY += incY;
        if ( posY < -alto / 2 )
            posY = view.getHeight() - alto / 2;
        if ( posY > view.getHeight() - alto / 2 )
            posY = -alto / 2;

        angulo += rotacion;
    }

    public double distancia ( Grafico g ) {
        return distanciaE ( posX, posY, g.posX, g.posY );
    }

    public boolean verificaColision ( Grafico g ) {
        return ( distancia ( g ) < ( radioColision + g.radioColision ) );
    }

    public static double distanciaE ( double x, double y, double x2, double y2 ) {
        return Math.sqrt((x - x2) * (x - x2) + (y - y2) * (y - y2));
    }

    public int getAncho() {
        return ancho;
    }

    public void setAncho(int ancho) {
        this.ancho = ancho;
    }

    public int getAlto() {
        return alto;
    }

    public void setAlto(int alto) {
        this.alto = alto;
    }

    public int getRadioColision() {
        return radioColision;
    }

    public void setRadioColision(int radioColision) {
        this.radioColision = radioColision;
    }

    public double getIncX() {
        return incX;
    }

    public void setIncX(double incX) {
        this.incX = incX;
    }

    public int getAngulo() {
        return angulo;
    }

    public void setAngulo(int angulo) {
        this.angulo = angulo;
    }

    public int getRotacion() {
        return rotacion;
    }
}
```

```

    public void setRotacion(int rotacion) {
        this.rotacion = rotacion;
    }

    public double getIncY() {
        return incY;
    }

    public void setIncY(double incY) {
        this.incY = incY;
    }

    public double getPosX() {
        return posX;
    }

    public void setPosX(double posX) {
        this.posX = posX;
    }

    public double getPosY() {
        return posY;
    }

    public void setPosY(double posY) {
        this.posY = posY;
    }
}

```

## **JuegoActivity.java**

```

package mx.itl.nc21130561.u4juegoasteroideapp;

import android.media.AudioManager;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.util.Log;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class JuegoActivity extends AppCompatActivity {
    private VistaJuegoView vistaJuegoView;
    private MediaPlayer mplayAudioFondo;
    private MediaPlayer mplayAudioDisparo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.juego_layout);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
            Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
            return insets;
        });
        this.setVolumeControlStream(AudioManager.STREAM_MUSIC);
        vistaJuegoView = findViewById(R.id.vistaJuegoView);
        mplayAudioFondo = MediaPlayer.create(this, R.raw.pacman);
        mplayAudioFondo.setLooping(true);
        mplayAudioDisparo = MediaPlayer.create(this, R.raw.latigo);
        vistaJuegoView.setMplayAudioDisparo(mplayAudioDisparo);
    }
}

```

```

    }

    @Override
    protected void onResume() {
        super.onResume();
        if (mplayAudioFondo != null)
            mplayAudioFondo.start();
    }

    @Override
    protected void onPause() {
        super.onPause();
        if (mplayAudioFondo != null)
            mplayAudioFondo.pause();
    }

    @Override
    protected void onDestroy() {
        if (mplayAudioFondo != null)
            mplayAudioFondo.stop();
        if (mplayAudioFondo != null)
            mplayAudioFondo.stop();

        vistaJuegoView.setCorriendo(false);
        VistaJuegoThread hilo = vistaJuegoView.getVistaJuegoThread();
        try {
            hilo.join();
        } catch (InterruptedException ex) {
            Log.e("Asteroide", ex.toString());
        }
        super.onDestroy();
    }
}

```

### VistaJuegoThread.java

```

package mx.itl.nc21130561.u4juegoasteroideapp;

import android.util.Log;

public class VistaJuegoThread extends Thread{
    private VistaJuegoView vistaJuegoView;
    private int periodoSleep;

    public VistaJuegoThread (VistaJuegoView vistaJuegoView, int periodo) {
        super();
        this.vistaJuegoView = vistaJuegoView;
        periodoSleep = periodo;
    }

    @Override
    public void run(){
        boolean corriendo = vistaJuegoView.isCorriendo();
        while(corriendo){
            corriendo = vistaJuegoView.isCorriendo();
            vistaJuegoView.actualizarFisica();
            try{
                Thread.sleep(periodoSleep);
            } catch (InterruptedException ex){
                Log.e("Asteroide", ex.toString());
            }
        }
    }
}

```

### VistaJuegoView.java



```

package mx.itl.nc21130561.u4juegoasteroideapp;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.drawable.Drawable;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.media.MediaPlayer;
import android.util.AttributeSet;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Toast;

import java.util.List;
import java.util.Vector;

/**
 * Created by LGV on 25/07/2015.
 */
public class VistaJuegoView extends View {

    private Vector<Grafico> asteroides;
    private Drawable      drawableAsteroide [] = new Drawable [ 3 ];
    private int           numAsteroides = 5;
    private int           numFragmentos = 3;
    private Grafico       nave;
    private int           giroNave      = 0;
    private float         aceleracionNave = 2.5f ;
    private static int    PERIODO_PROCESO = 20;
    private long          ultimoProceso  = 0;
    private boolean       hayValorInicial = false;
    private float         valorInicial;

    public static final int PASO_GIRO_NAVE      = 5;
    public static final float PASO_ACELERACION_NAVE = 0.5f;

    // ***** MISIL *****
    private Grafico      misil;
    private static int    PASO_VELOCIDAD_MISIL = 12;
    private boolean       misilActivo;
    private int           distanciaMisil;

    // ***** PARA PANTALLA TACTIL *****
    private float         mX = 0, mY = 0;
    private boolean       disparo = false;

    private VistaJuegoThread vistaJuegoThread;
    private boolean corriendo;
    private MediaPlayer mplayAudioDisparo;

    //-----

    public VistaJuegoView ( Context context, AttributeSet attrs ) {
        super(context, attrs);

        Drawable drawableNave, drawableMisil;

        drawableNave = context.getResources().getDrawable(R.drawable.oswil);
        drawableMisil = context.getResources().getDrawable(R.drawable.cable);
        drawableAsteroide [0] = context.getResources().getDrawable(R.drawable.android1);
        drawableAsteroide [1] = context.getResources().getDrawable(R.drawable.android2);
        drawableAsteroide [2] = context.getResources().getDrawable(R.drawable.android3);

        nave = new Grafico(this, drawableNave);

```

```

nave.setIncX(Math.random() * 4 - 2);
nave.setIncY(Math.random() * 4 - 2);
nave.setAngulo(0);
nave.setRotacion(5);

misil = new Grafico ( this, drawableMisil );

asteroides = new Vector<Grafico>();
for (int i = 0; i < numAsteroides; i++) {
    Grafico asteroide = new Grafico(this, drawableAsteroide [0] );
    asteroide.setIncY(Math.random() * 4 - 2);
    asteroide.setIncX(Math.random() * 4 - 2);
    asteroide.setAngulo((int) (Math.random() * 360));
    asteroide.setRotacion((int) (Math.random() * 8 - 4));
    asteroides.add(asteroide);
}
vistaJuegoThread = new VistaJuegoThread(this, PERIODO_PROCESO);
vistaJuegoThread.start();
corriendo = true;
}

//-----

@Override
protected void onSizeChanged ( int w, int h, int oldw, int oldh ) {
    super.onSizeChanged(w, h, oldw, oldh);

    // lgv: conocer el ancho y alto en pixeles de VistaJuego ( ocupa toda la pantalla por tanto da la
    resolucion )
    Toast.makeText ( getContext(), getWidth() + "," + getHeight(), Toast.LENGTH_SHORT ).show ();

    nave.setPosX((w - nave.getAncho()) / 2);
    nave.setPosY((h - nave.getAncho()) / 2);

    for ( Grafico asteroide : asteroides ) {
        do {
            asteroide.setPosX(Math.random() * (w - asteroide.getAncho()));
            asteroide.setPosY(Math.random() * (h - asteroide.getAlto()));
        } while ( asteroide.distancia ( nave ) < ( w + h ) / 5 );
    }
}

//-----

@Override
protected void onDraw ( Canvas canvas ) {
    super.onDraw(canvas);

    synchronized ( this ) {
        for (Grafico asteroide : asteroides) {
            asteroide.dibujarGrafico(canvas);
        }
        nave.dibujarGrafico(canvas);

        if (misilActivo)
            misil.dibujarGrafico(canvas);
    }
}

//-----

protected void actualizarFisica () {

    long ahora = System.currentTimeMillis();

```

```

        double retardo = ( ahora - ultimoProceso ) / PERIODO_PROCESO;
        nave.setAngulo((int) (nave.getAngulo() + giroNave * retardo ));
        double nIncX = nave.getIncX () + aceleracionNave * Math.cos ( Math.toRadians ( nave.getAngulo() ) )
* retardo * 0;
        double nIncY = nave.getIncY () + aceleracionNave * Math.sin ( Math.toRadians ( nave.getAngulo() ) )
* retardo * 0 ;
        if ( Grafico.distanciaE ( 0, 0, nIncX, nIncY ) <= Grafico.getMaxVelocidad () ) {
            nave.setIncX ( nIncX );
            nave.setIncY ( nIncY );
        }

        nave.incrementaPos();
        for ( int i = 0; i < asteroides.size(); i++ ){
            asteroides.get (i).incrementaPos();
        }

        ultimoProceso = ahora;

        if ( misilActivo ) {
            misil.incrementaPos();
            distanciaMisil--;
            if ( distanciaMisil < 0 ) {
                misilActivo = false;
            } else {
                for ( int i = 0; i < asteroides.size () ; i++ ) {
                    if ( misil.verificaColision ( asteroides.elementAt ( i ) ) ) {
                        destruyeAsteroide ( i );
                    }
                }
            }
        }
    }

}

//-----

private void destruyeAsteroide ( int i ) {
    int tam;

    synchronized ( this ) {
        if (asteroides.get(i).getDrawable() != drawableAsteroide[2]) {
            if (asteroides.get(i).getDrawable() == drawableAsteroide[1]) {
                tam = 2;
            } else {
                tam = 1;
            }
        }
        for (int n = 0; n < numFragmentos; n++) {
            Grafico asteroide = new Grafico(this, drawableAsteroide[tam]);
            asteroide.setPosX(asteroides.get(i).getPosX());
            asteroide.setPosY(asteroides.get(i).getPosY());
            asteroide.setIncX(Math.random() * 7 - 2 - tam);
            asteroide.setIncY(Math.random() * 7 - 2 - tam);
            asteroide.setAngulo((int) (Math.random() * 360));
            asteroide.setRotacion((int) (Math.random() * 8 - 4));
            asteroides.add(asteroide);
        }
    }

    asteroides.remove(i);
    misilActivo = false;
}

}

//-----

@Override
public boolean onTouchEvent(MotionEvent event ) {

```

```

        super.onTouchEvent(event);
        float x = event.getX ();
        float y = event.getY ();

        switch ( event.getAction() ) {
            case MotionEvent.ACTION_DOWN :
                disparo = true;
                break;
            case MotionEvent.ACTION_MOVE :
                float dx = Math.abs ( x - mX );
                float dy = Math.abs ( y - mY );
                if ( dy < 6 && dx > 6 ) {
                    giroNave = Math.round ( ( x- mX ) / 2 );
                    disparo = false;
                } else if ( dx < 6 && dy > 6 ) {
                    aceleracionNave = Math.round ( ( mY - y ) / 25 );
                    disparo = false;
                }
                break;
            case MotionEvent.ACTION_UP :
                giroNave = 0;
                aceleracionNave = 0;
                if ( disparo ) {
                    activaMisil();
                }
                break;
        }
        mX = x;
        mY = y;
        return true;
    }

    //-----

    private void activaMisil () {
        misil.setPosX ( nave.getPosX() + nave.getAncho() / 2 - misil.getAncho () / 2 );
        misil.setPosY(nave.getPosY() + nave.getAlto() / 2 - misil.getAlto() / 2);
        misil.setAngulo(nave.getAngulo());
        misil.setIncX(Math.cos(Math.toRadians(misil.getAngulo())) * PASO_VELOCIDAD_MISIL);
        misil.setIncY(Math.sin(Math.toRadians(misil.getAngulo())) * PASO_VELOCIDAD_MISIL);
        distanciaMisil = (int) Math.min ( this.getWidth() / Math.abs ( misil.getIncX() ),
                                           this.getHeight() / Math.abs ( misil.getIncY () )
                                           ) - 2;

        misilActivo = true;
        //reproducir audio de disparo
        if (mplayAudioDisparo !=null)
            mplayAudioDisparo.start();
    }

    //-----

    public boolean isCorriendo() {
        return corriendo;
    }

    public void setCorriendo(boolean corriendo) {
        this.corriendo = corriendo;
    }

    //-----

    public VistaJuegoThread getViewaJuegoThread() {
        return vistaJuegoThread;
    }

    //-----

```

```
public void setMplayAudioDisparo(MediaPlayer mplayAudioDisparo) {
    this.mplayAudioDisparo = mplayAudioDisparo;
}
```

```
//-----
```

```
}
```

## Juego\_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".JuegoActivity">

    <mx.itl.nc21130561.u4juegoasteroideapp.VistaJuegoView
        android:id="@+id/vistaJuegoView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@drawable/salon" />

</LinearLayout>
```

## AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

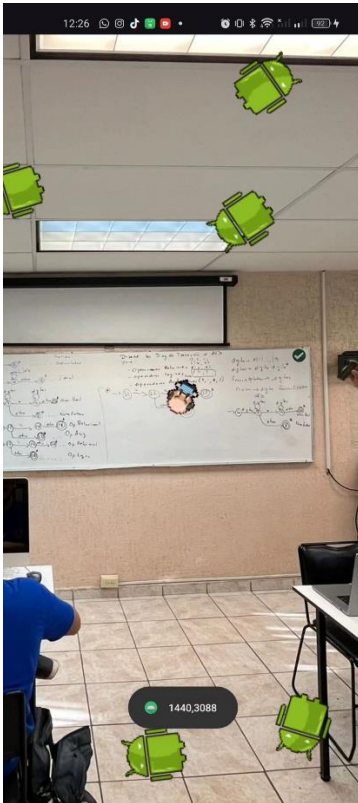
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.U4JuegoAsteroideApp"
        tools:targetApi="31">
        <activity
            android:name=".JuegoActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

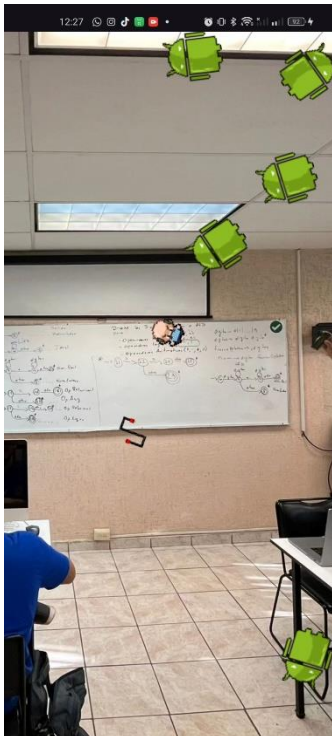
</manifest>
```

## Prueba de Ejecución

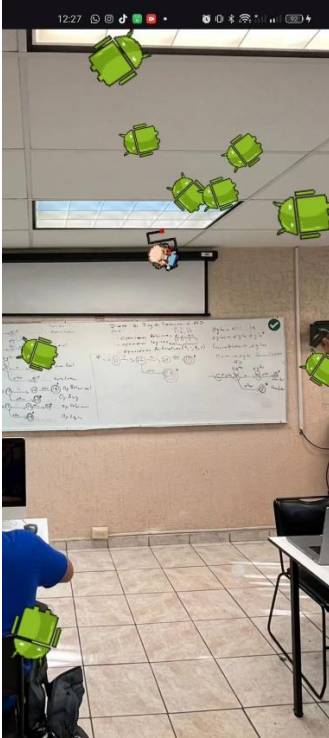
Una vez iniciado el juego podremos ver a nuestro personaje y a diferentes enemigos rodeándolo



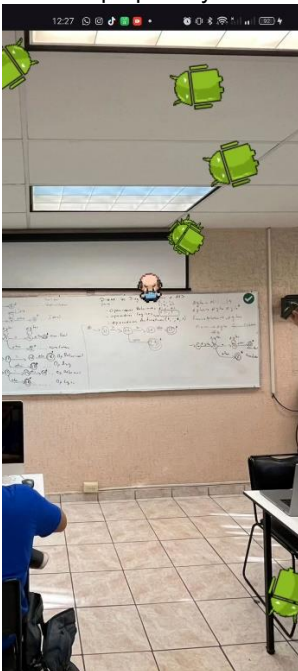
Podemos eliminar a los enemigos presionando sobre la pantalla y nuestro personaje comenzara a lanzar cables HDMI con los cuales se defendera



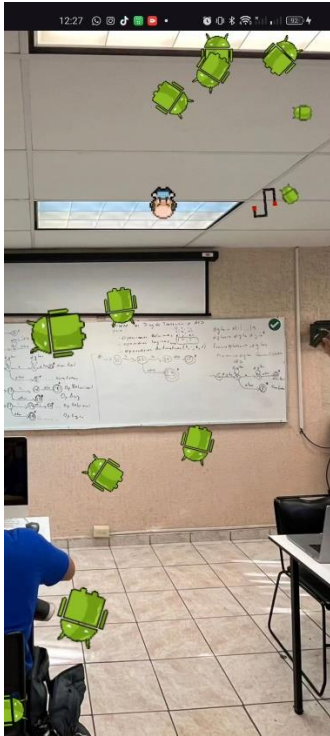
Cuando un enemigo es alcanzado por el cable HDMI se divide en 3 partes haciéndose mas pequeño



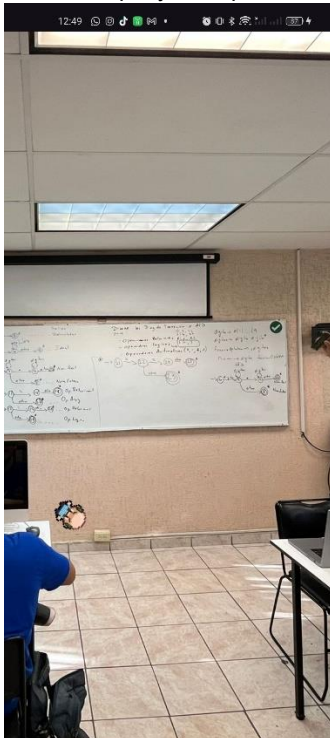
Cuando ese enemigo es más pequeño que el anterior y vuelve a ser alcanzado por un cable HDMI se vuelve a dividir haciéndose aun más pequeño y dividiéndose en 3 enemigos mas pequeños



Una vez que los enemigos se han dividido una segunda vez y son lo mas pequeños posibles podemos lanzar un cable HDMI para poder eliminarlos por completo



Así hasta que ya no quede ningún enemigo en la pantalla





## Fuentes de información

L. F. Gil Vazquez. "Cátedra Digital - Enero Junio 2024: Ingresar al sitio". Cátedra Digital - Enero Junio 2024. Accedido el 25 de mayo de 2024. [En línea]. Disponible: <https://catedig.itlalaguna.edu.mx/mod/forum/view.php?id=3366>

-oOo-