

“Implementación de una Aplicación Móvil en Android, Calculadora científica Para las Personas Invidentes”

UNIVERSIDAD NACIONAL JOSÉ MARÍA ARGUEDAS
FACULTAD DE INGENIERÍA
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS



Estudiantes:

- Giovanni MORCCOLLA ANCCO
- Irving ORTEGA ZARABIA

*ANDAHUAYLAS - APURÍMAC
PERÚ
JULIO, 2014*



*Universidad Nacional José María Arguedas
Identidad y Excelencia para el Trabajo Productivo y el Desarrollo*



Est. Giovanni Morccolla A. - Irving Ortega Z.

ÍNDICE

IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL EN ANDROID, CALCULADORA CIENTÍFICA PARA LAS PERSONAS INVIDENTES.....	3
1. Requerimientos Funcionales y no Funcionales	3
1.1. Requerimientos Funcionales	3
1.2. Requerimientos no Funcionales.....	4
2. Diagrama de Casos de Uso.....	5
2.1. Casos de Uso.....	5
3. Diagramas de Secuencias.....	10
4. Diseño de la Interfaz Grafica.....	12
4.1. Objetos que Forman la Interfaz.....	12
5. Diagramas de Clases.....	13
6. Desarrollo de la Aplicación Móvil.....	14
6.1. Justificación	14
6.2. Objetivos	14
6.2.1. Objetivo General.....	14
6.2.2. Objetivos Específicos.....	14
6.3. Introducción al Android.....	15
6.3.1. Que es Android	15
6.3.2. Versiones de Android.....	15
6.4. Eclipse como Entorno de Desarrollo	16
6.4.1. Vista de Eclipse	16
6.4.2. Crear un Dispositivo Virtual AVD (Android Virtual Device)	17
6.4.3. Emulador instalado.....	19
6.5. Creación de la Aplicación Móvil	20
6.5.1. Componentes de la Aplicación Móvil.....	22
6.5.2. Actividades	23
6.6. Codificación de la Aplicación	24
6.7. Ejecución de la Aplicación Móvil	38
7. Conclusiones	39
8. Referencias	40





IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL EN ANDROID, CALCULADORA CIENTÍFICA PARA LAS PERSONAS INVIDENTES



1. Requerimientos Funcionales y no Funcionales

Durante la etapa de análisis se deben definir los requerimientos funcionales y no funcionales del sistema. Igualmente, también se determinan los casos de uso posteriormente. Para nuestra aplicación tendremos los siguientes requerimientos funcionales y no funcionales:

1.1. Requerimientos Funcionales

- ✓ La aplicación deberá permitir al usuario el ingreso de dos números enteros y una operación matemática básica (suma, resta, multiplicación y división, porcentaje), luego efectuar dicha operación y mostrar su resultado.
- ✓ La aplicación deberá permitir al usuario el ingreso de un numero en decimal, el numero decimal será concatenado con punto.





Est. Giovanni Morccolla A. - Irving Ortega Z.

- ✓ La aplicación deberá permitir al usuario el ingreso de un numero entero y una operación trigonométrica (seno, coseno, tangente, arco seno, arco coseno y arco tangente), luego efectuar dicha operación y mostrar su resultado.
- ✓ La aplicación deberá permitir al usuario el ingreso de un número y una operación logarítmica (logaritmo natural y logaritmo natural), luego efectuar dicha operación y mostrar su resultado.
- ✓ La aplicación deberá permitir al usuario el ingreso de un número o dos números y una operación de potencia (número elevado al cuadrado, número elevado al cubo o número elevado otro número), luego efectuar dicha operación y mostrar su resultado.
- ✓ La aplicación deberá permitir al usuario el ingreso de un número o dos números y una operación radical (raíz cuadrada, raíz cubica o raíz n), luego efectuar dicha operación y mostrar su resultado.
- ✓ La aplicación deberá permitir al usuario el ingreso de un número o dos números y operaciones adicionales como (pi, fracción, factorial, etc.), luego efectuar dicha operación y mostrar su resultado.
- ✓ La aplicación deberá efectuar o acumular las sumas, restas, multiplicación, división, etc. Cuantas veces sea necesaria.
- ✓ La aplicación deberá permitir limpiar los números ingresados y las operaciones efectuadas.
- ✓ La aplicación deberá mostrar opciones de cálculo trigonométrica en (radianes, sexagesimales y centesimales).

1.2. Requerimientos no Funcionales

- ✓ La aplicación debe estar orientado a depósitos móviles.
- ✓ La aplicación debe permitir al usuario interactuar en un entorno móvil con sistema operativo android.
- ✓ La aplicación no almacenara datos o información por lo cual no necesita una base datos.





Est. Giovanni Morccolla A. - Irving Ortega Z.

2. Diagrama de Casos de Uso

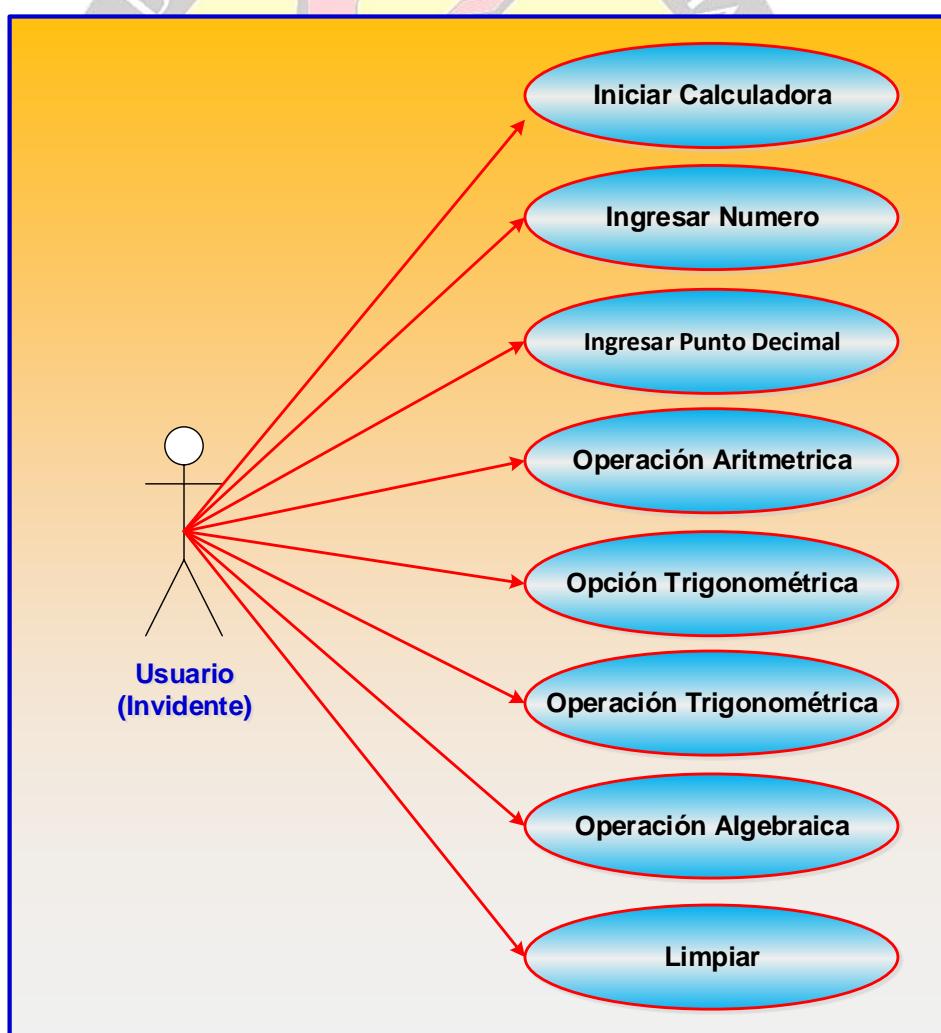
Una vez que hemos identificado como deberá funcionar la aplicación móvil Android procedemos hacer el diseño de Diagrama de casos de uso.

2.1. Casos de Uso

Un diagrama de casos de uso muestra la manera de que como los usuarios interactúan con la aplicación, para este sistema tenemos un “Actor” que es el quien utiliza la calculadora científica y los casos de uso son cada una de las operaciones que va a realizar la calculadora científica.

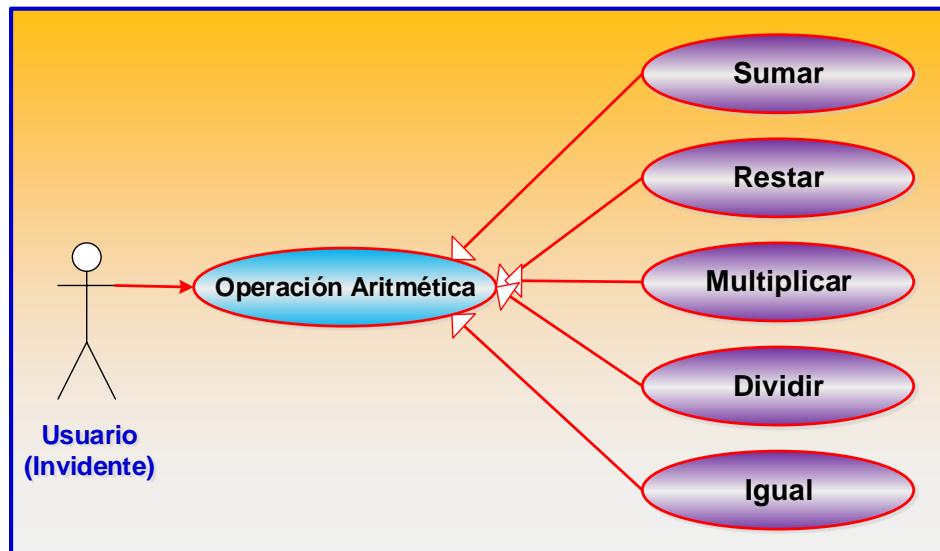
El siguiente diagrama muestra los casos de uso de la calculadora científica:

Caso de uso general:





Caso de uso (Operación Aritmética):



DESCRIPCION (Sumar, Restar, Multiplicar, Dividir): Las funcionalidades de uso son similares.

Identificador:	Sumar, Restar, Multiplicar, Dividir,
Resumen Funcionalidad General:	Operación Aritmética
Actores:	Usuario (Invíduente)
Precondición:	Ingresar un numero entero o decimal
Postcondición:	Almacenar temporalmente el numero ingresado y luego ingresar el otro numero

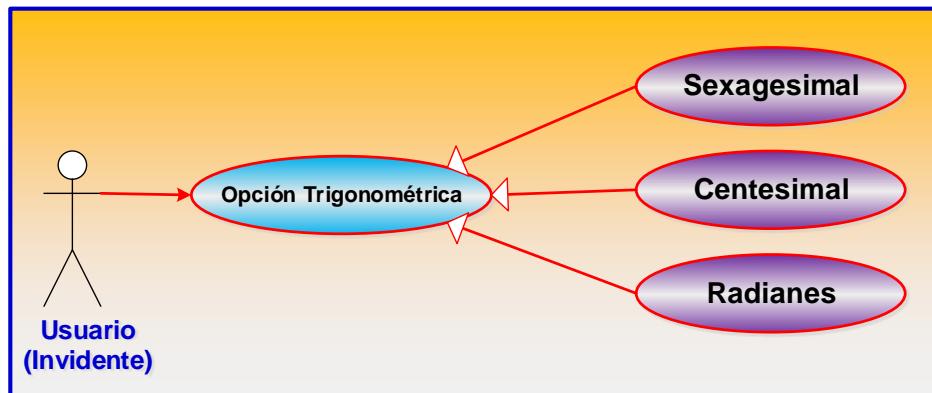
DESCRIPCION (Igual): Las funcionalidades de uso son similares en todas las operaciones.

Identificador:	Igual
Resumen Funcionalidad General:	Operación de Igualar
Actores:	Usuario (Invíduente)
Precondición:	Realizar cualquier operación con dos números ingresados.
Postcondición:	Mostrar el resultado de la operación final





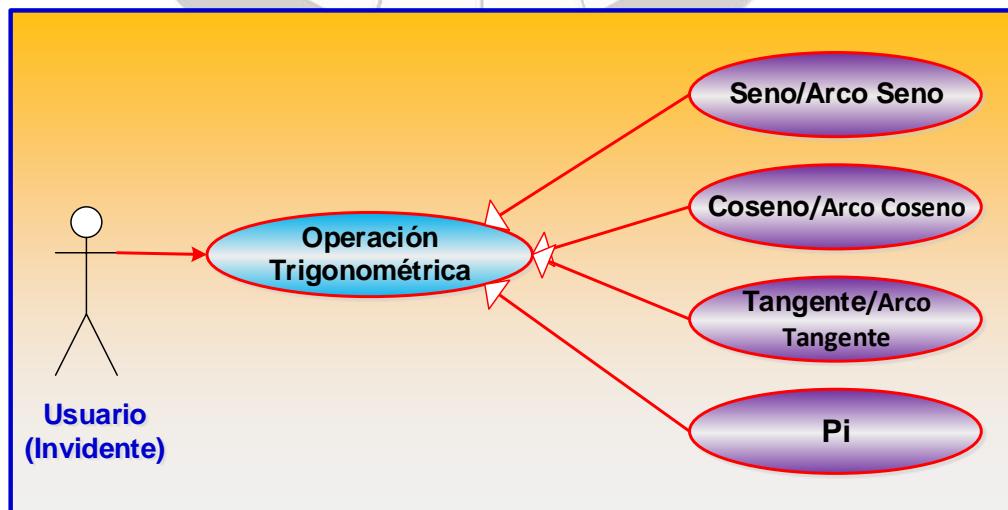
Caso de uso (Opción Trigonométrica):



DESCRIPCION (Sexagesimal, Centesimal y Radianes): Las funcionalidades de uso son similares en los tres casos.

Identificador:	Sexagesimal, Centesimal y Gradientes
Resumen Funcionalidad General:	Opción para el cálculo trigonométrico
Actores:	Usuario (Invidente)
Precondición:	Iniciar la calculadora
Postcondición:	Escoger la opción para operaciones trigo.

Caso de uso (Operación Trigonométrica):

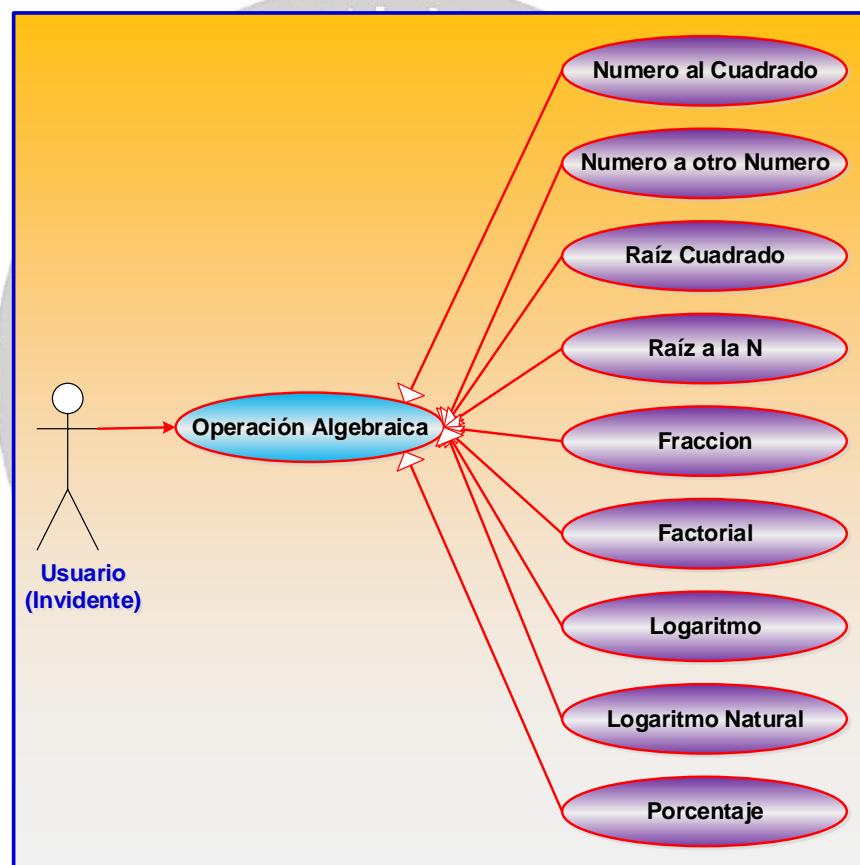




Est. Giovanni Morccolla A. - Irving Ortega Z.

DESCRIPCION (Seno, Coseno, Tangente, Arco Seno, Arco Coseno, Arco Tangente) Las funcionalidades de uso son similares en todos.	
Identificador:	Seno, Coseno, Tangente, Arco Seno, Arco Coseno, Arco Tangente
Resumen Funcionalidad General:	Escoger una opción de cálculo trigonométrico
Actores:	Usuario (Invidente)
Precondición:	Ingresar un numero
Postcondición:	Mostrar el resultado del cálculo trigonométrico

Caso de uso (Operación Algebraica):



DESCRIPCION (Número al Cuadrado, Raíz Cuadrada, Factorial, Logaritmo, Logaritmo natural, Porcentaje y Fracción): Las funcionalidades de uso son similares de todos.

Identificador:	Número al Cuadrado,, Porcentaje y Fracción
Resumen Funcionalidad General:	Operación algebraica
Actores:	Usuario (Invidente)
Precondición:	Ingresar un número entero o decimal
Postcondición:	Mostrar el resultado de la operación algebraica





Est. Giovanni Morccolla A. - Irving Ortega Z.

DESCRIPCION (Número a Otro Número, Raíz N): Las funcionalidades de uso son similares de los casos.	
Identificador:	Potencia N, Raíz N
Resumen Funcionalidad General:	Operación algebraica de Potencias
Actores:	Usuario (Invidente)
Precondición:	Ingresar un numero o decimal
Postcondición:	Mostrar el resultado de la operación Potencia

DESCRIPCION (Iniciar Calculadora)	
Identificador:	Iniciar
Resumen Funcionalidad General:	Iniciación de la calculadora
Actores:	Usuario (Invidente)
Precondición:	Encender el móvil android
Postcondición:	Iniciar la aplicación móvil

DESCRIPCION (Ingresar Número)	
Identificador:	Número
Resumen Funcionalidad General:	Ingresar número
Actores:	Usuario (Invidente)
Precondición:	Iniciar la calculadora
Postcondición:	Mostrar el número ingresado

DESCRIPCION (Ingresar Punto Decimal)	
Identificador:	Punto
Resumen Funcionalidad General:	Ingresar punto decimal
Actores:	Usuario (Invidente)
Precondición:	Ingresar número (0,1.....9)
Postcondición:	Mostrar el punto concatenado con el número

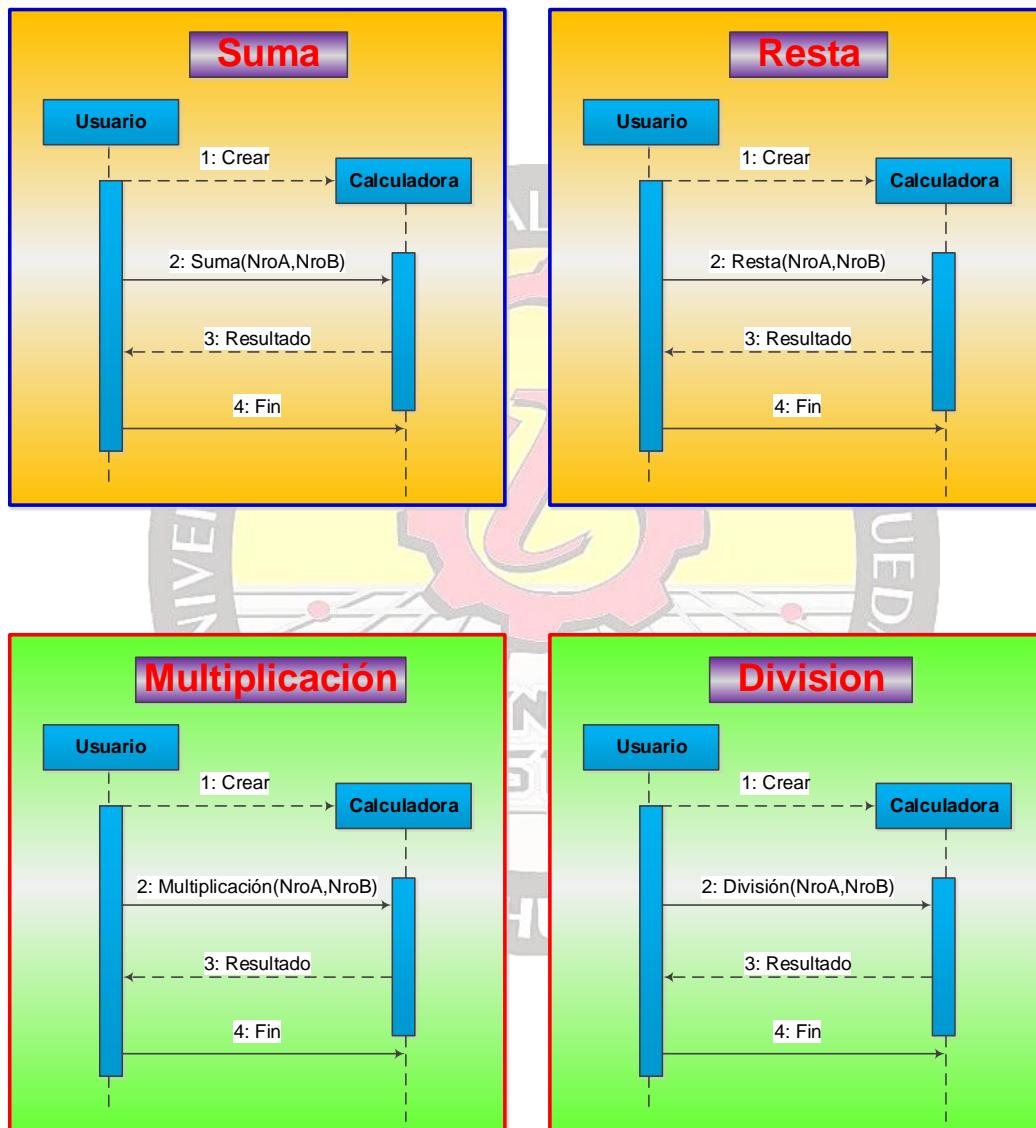
DESCRIPCION (Limpiar)	
Identificador:	Limpiar
Resumen Funcionalidad General:	Limpia los valores de las variables
Actores:	Usuario (Invidente)
Precondición:	Ingresar número o realizar cualquier operación
Postcondición:	Limpia los valores de todas las variables

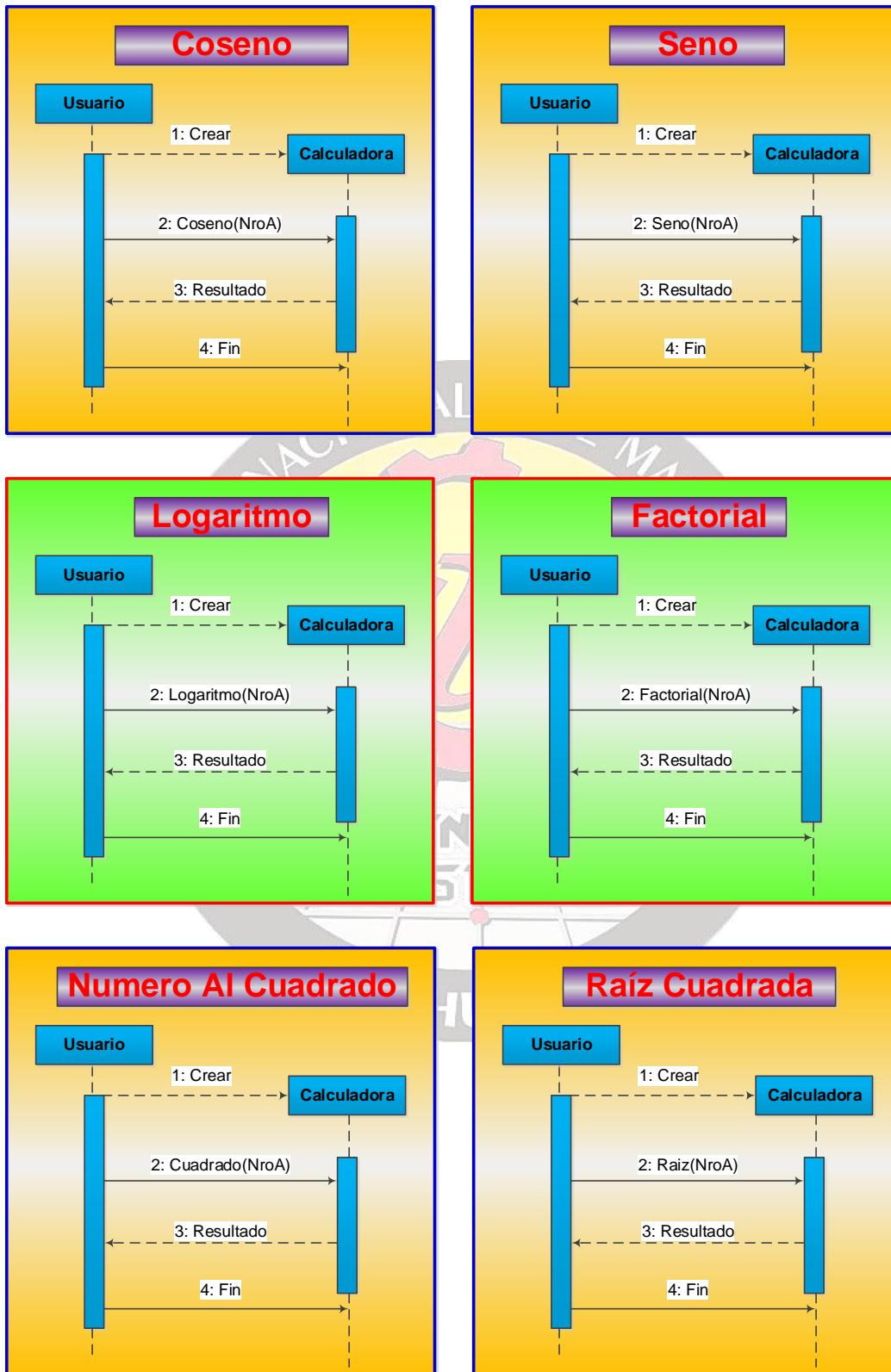




3. Diagramas de Secuencias

A continuación detallamos los diagramas de secuencia de algunos casos de uso de la aplicación móvil “Calculadora Científica” para personas invidentes



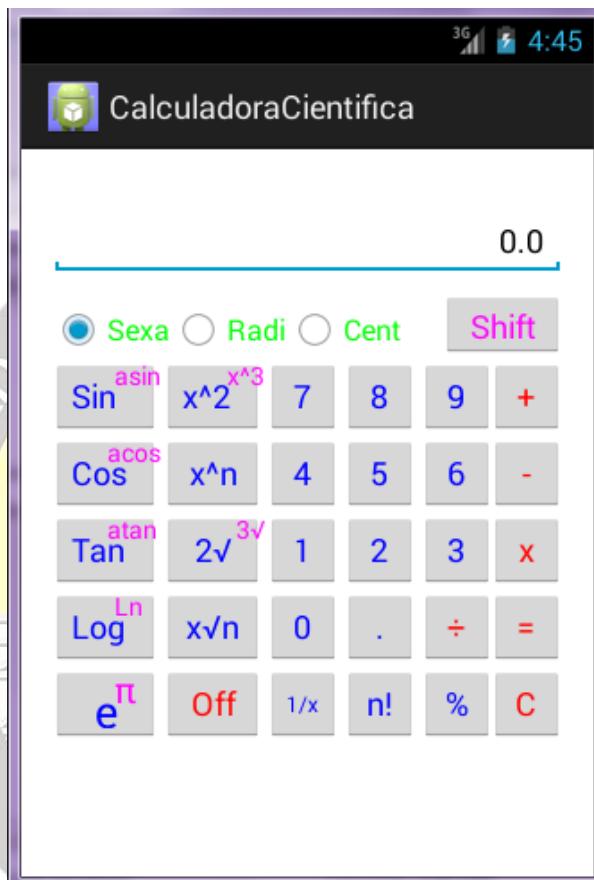




Est. Giovanni Morccolla A. - Irving Ortega Z.

4. Diseño de la Interfaz Gráfica

Para iniciar el desarrollo del software primeramente diseñamos la interfaz gráfica. La interfaz que se utiliza será similar o igual a una calculadora científica como se muestra en la siguiente gráfica.



4.1. Objetos que Forman la Interfaz

La calculadora científica incluye los siguientes objetos:

- ✓ Una ventana que permita implementar la interfaz.
- ✓ Una caja de texto (pantalla de la calculadora).
- ✓ Radio Botones (Sexag, Radia, Cente)
- ✓ Botones de órdenes distribuidos de la siguiente forma:
 - Dígitos del 0 al 9
 - Punto decimal
 - Operaciones +, -, x, /, =, %, n!, 1/x, log, cos, sen, tag, etc.
 - Operaciones de borrar o limpiar (C)





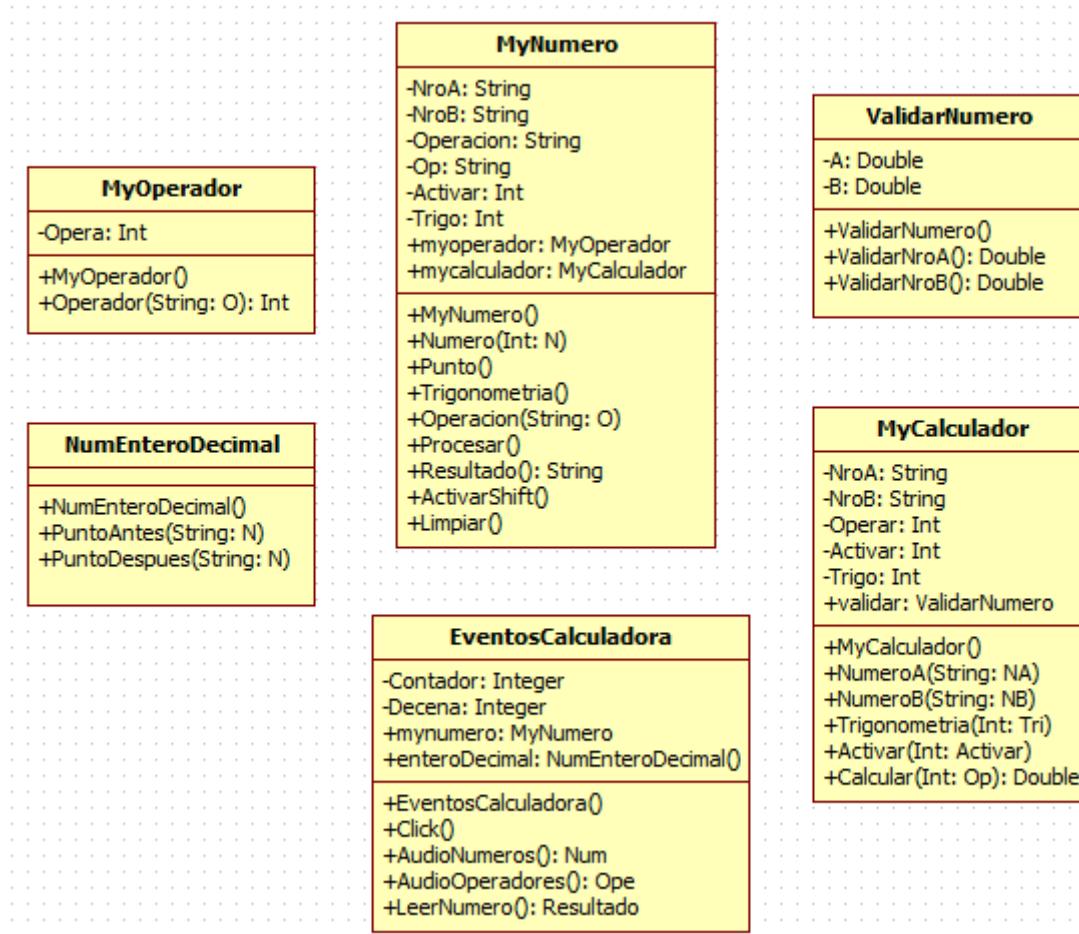
Est. Giovanni Morccolla A. - Irving Ortega Z.

5. Diagramas de Clases

Este diagrama representa la estructura de la aplicación Calculadora científica pero en cuanto a las clases, relaciones entre las clases, atributos y métodos encontramos cinco clases como:

- **EventosCalculadora**
- **MyNumero**
- **ValidarNumero**
- **MyOperacion**
- **MyCalculadora**
- **NumEnteroDecimal**

A continuación se muestra el diagrama de clases de la aplicación.





Est. Giovanni Morccolla A. - Irving Ortega Z.

6. Desarrollo de la Aplicación Móvil

6.1. Justificación

La integración de herramientas de diseño, simulación y optimización es fundamental para conocer en Android es hoy, uno de los más populares Sistemas Operativos que funcionan sobre la mayoría de los dispositivos móviles a nivel mundial tales como teléfonos inteligentes, Tablets, computadores personales, entre otros. Basado en Linux, integra diferentes librerías para el desarrollo de aplicaciones bajo el potente motor de Java.

Android incorpora infinidad de características propias de los dispositivos móviles como conexiones inalámbricas, transmisión de datos y voz, interfaces gráficas amigables con el usuario, etc. Permitiendo flexibilidad en su desarrollo y portabilidad.

Por lo anterior, Android es ideal para el desarrollo de nuevas aplicaciones, lo que demanda por parte de los programadores, un conocimiento actualizado y especializado que les permita usar esta importante herramienta.

6.2. Objetivos

6.2.1. Objetivo General

Desarrollar una aplicación móvil en Android, calculadora científica para las personas invidentes.

6.2.2. Objetivos Específicos

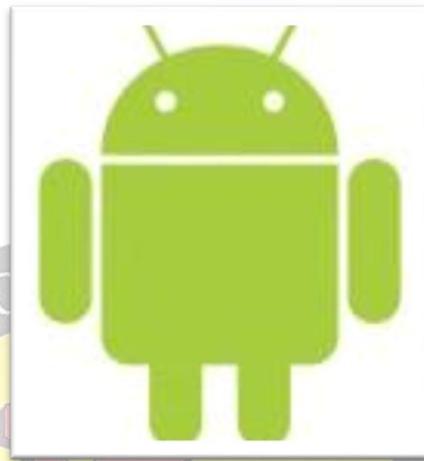
- ✓ Conocer y comprender los fundamentos de programación en Android.
- ✓ Desarrollar y compilar aplicaciones básicas para los dispositivos móviles usando el lenguaje de programación IDE ECLIPSE.
- ✓ Comprender y manipular los periféricos de entradas y salidas en Android
- ✓ Adquirir habilidades para el desarrollo de aplicaciones para Dispositivos Móviles.





6.3. Introducción al Android

6.3.1. Que es Android



Android es un sistema operativo y una plataforma software, basado en Linux para teléfonos móviles. Además, también usan este sistema operativo (aunque no es muy habitual), tablets, netbooks, reproductores de música e incluso PCs. Android permite programar en un entorno de trabajo (framework) de Java, aplicaciones sobre una máquina virtual Dalvik (una variación de la máquina de Java con compilación en tiempo de ejecución). Además, lo que le diferencia de otros sistemas operativos, es que cualquier persona que sepa programar puede crear nuevas aplicaciones, widgets, o incluso, modificar el propio sistema operativo, dado que Android es de código libre, por lo que sabiendo programar en lenguaje Java, va a ser muy fácil comenzar a programar en esta plataforma.

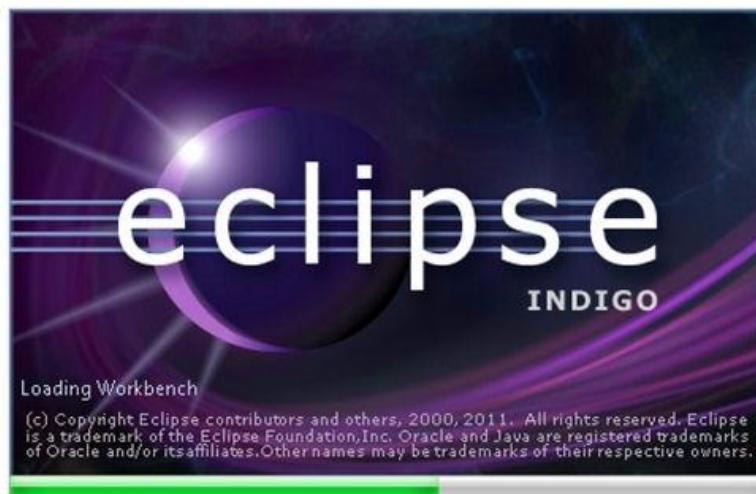
6.3.2. Versiones de Android

- ✓ Cupcake: Android Version 1.5
- ✓ Donut: Android Version 1.6
- ✓ Eclair: Android Version 2.0/2.1
- ✓ Froyo: Android Version 2.2
- ✓ Ginger Bread: Android Version 2.3
- ✓ Honey Comb: Android Version 3.0/3.4
- ✓ Ice Cream Sandwich: Android Version 4.0



Est. Giovanni Morccolla A. - Irving Ortega Z.

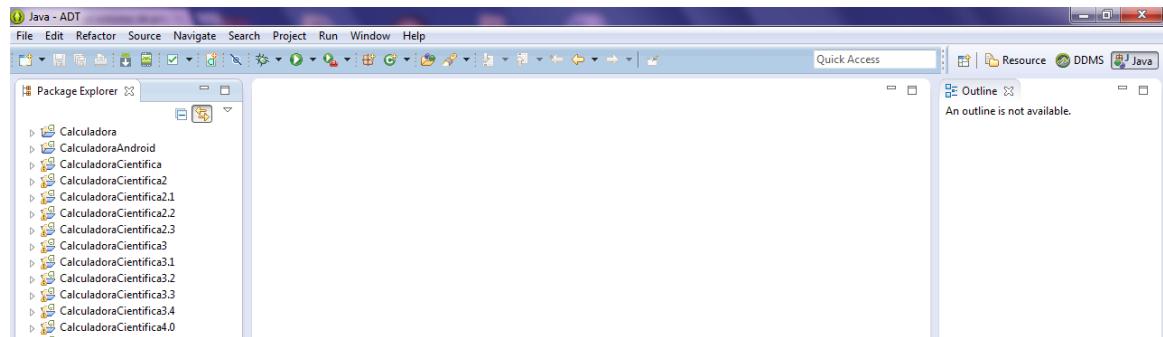
6.4. Eclipse como Entorno de Desarrollo



6.4.1. Vista de Eclipse

Lo primero que necesitaremos para poder programar en Android, es preparar el entorno de trabajo. Es necesario disponer de una versión de Eclipse Galileo 3.5 o superior para poder desarrollar nuestros proyectos. Lo segundo que necesitamos es el kit de desarrollo software para Android o Android SDK, del que se pueden encontrar varias versiones para diferentes plataformas en la página web: <http://developer.android.com/sdk/index.html>

Vista de Eclipse

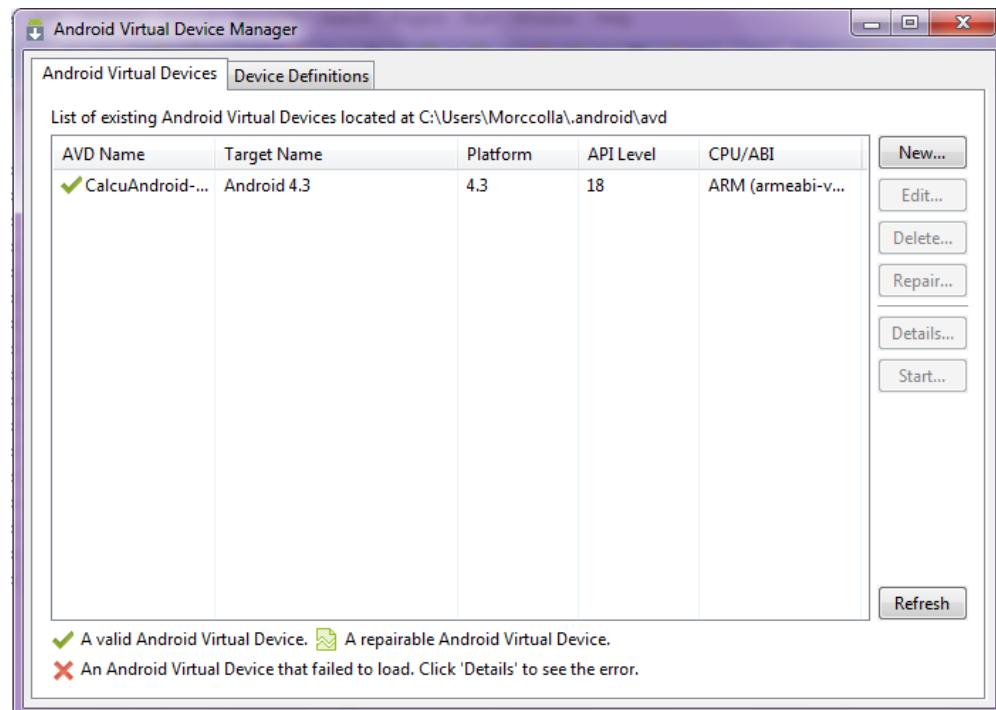




Est. Giovanni Morccolla A. - Irving Ortega Z.

6.4.2. Crear un Dispositivo Virtual AVD (Android Virtual Device)

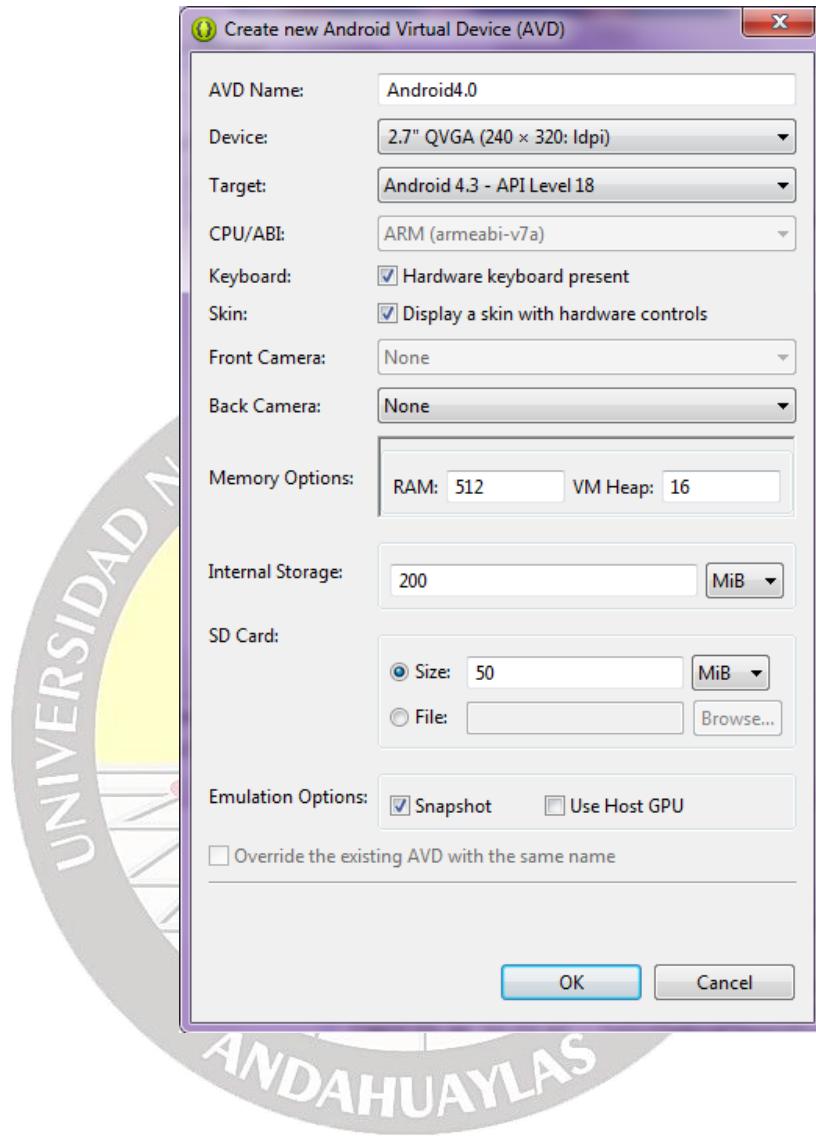
Abre Eclipse y pulsa en el botón Android Virtual Device Manager. Te aparecerá la lista con los AVD que hayas creado.



Pulsa a continuación el botón New... para crear un nuevo AVD. Aparecerá la siguiente ventana:



Est. Giovanni Morccolla A. - Irving Ortega Z.

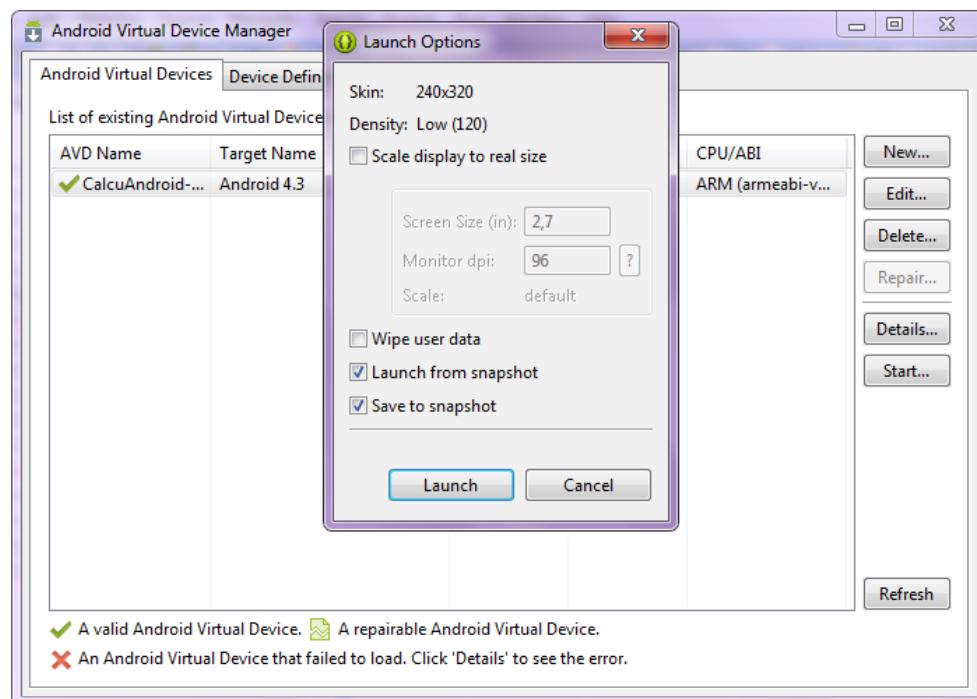


Aparecerá el dispositivo creado en la siguiente lista. Para arrancarlo seleccionalo y pulsa el botón Start.





Est. Giovanni Morccolla A. - Irving Ortega Z.



6.4.3. Emulador instalado

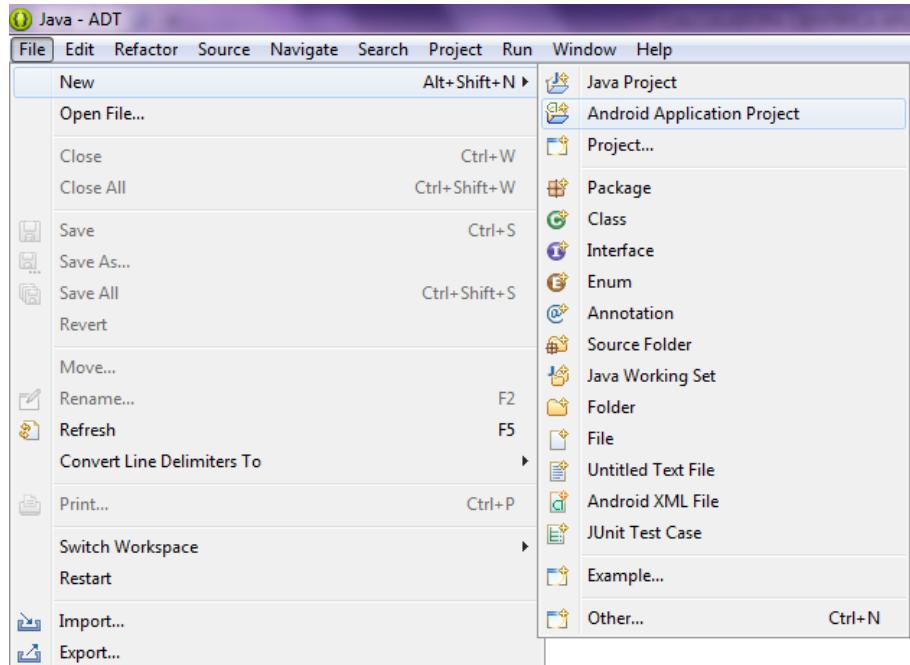
Una vez tengamos el proyecto listo para ejecutar, entra en escena el emulador de Android. Éste proporciona una vista especial para comprobar si la aplicación hace lo que se desea. A continuación se muestra la vista del emulador para la versión 4.3 de Android:



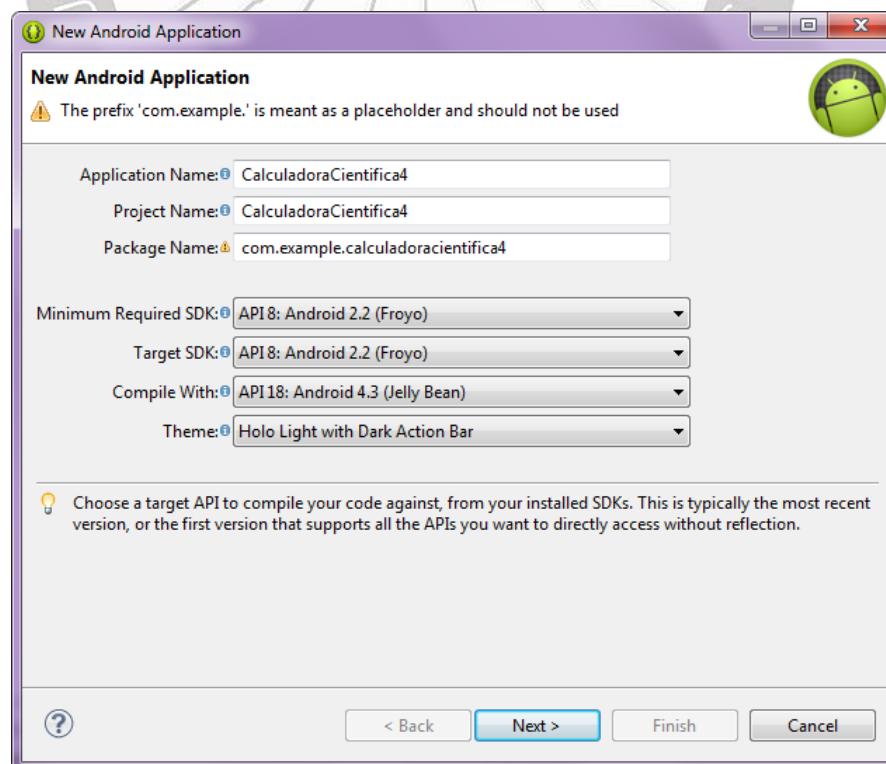


Est. Giovanni Morccolla A. - Irving Ortega Z.

6.5. Creación de la Aplicación Móvil



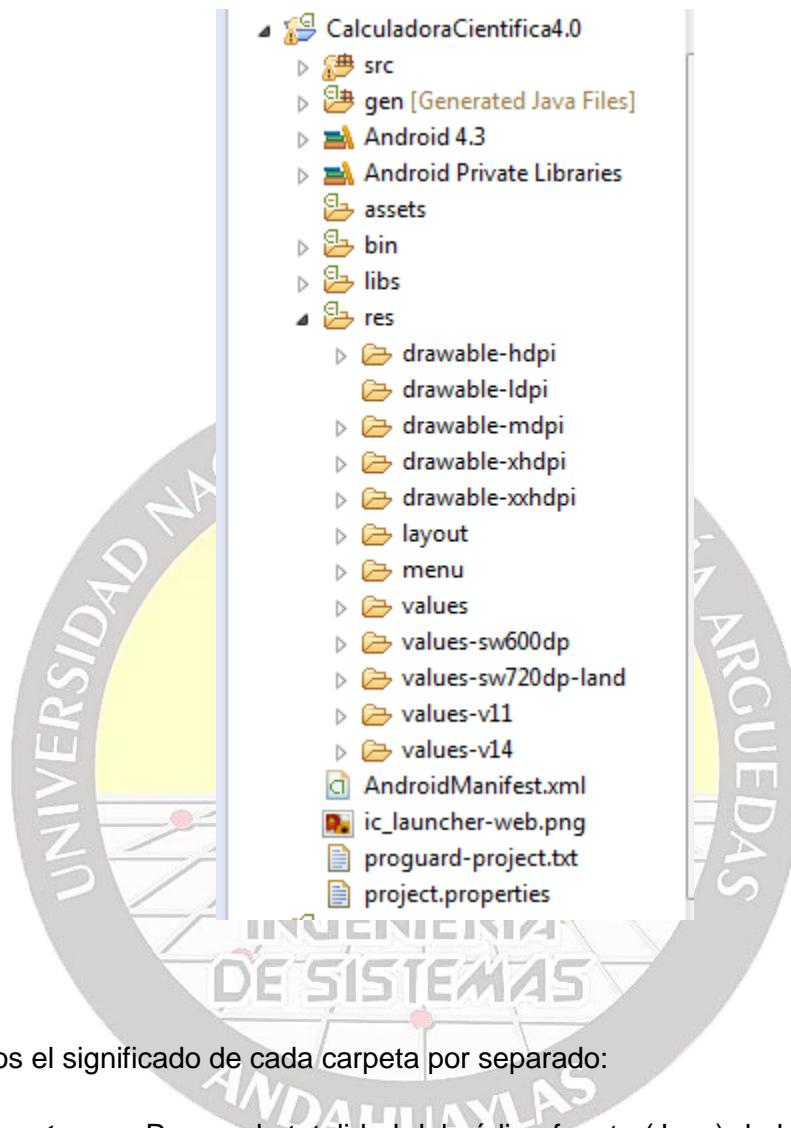
Damos el nombre del proyecto y configuramos las características





Est. Giovanni Morccolla A. - Irving Ortega Z.

Sigamos pulsando en Next eligiendo las opciones al final finish, el proyecto ya está creado.



Veamos el significado de cada carpeta por separado:

- ✓ **Carpeta src:** Recoge la totalidad del código fuente (Java) de la aplicación. En el proyecto que vamos a llevar a cabo, Eclipse generará automáticamente el código base de la ventana principal (Activity).
- ✓ **Carpeta res:** Contiene los recursos necesarios para generar una aplicación Android:
 - **res/drawable/:** Guarda las imágenes y se divide en: drawable-ldpi, drawable-mdpi y drawable-hdpi, que dependerán de la resolución del dispositivo.
 - **res/raw/:** Contiene archivos de propósito general, en otro formato que no es





Est. Giovanni Morccolla A. - Irving Ortega Z.

- XML.
- **res/layout/**: Incluye los archivos que definen el diseño de la interfaz gráfica, siempre en XML.
- **res/values/**: Guarda los datos y tipos que utiliza la aplicación, tales como colores, cadenas de texto, estilos, dimensiones...
- ✓ **Carpeta gen**: Ésta carpeta guarda un conjunto de archivos (de código Java) creados automáticamente cuando se compila el proyecto, para poder dirigir los recursos de la aplicación. El archivo R ajusta automáticamente todas las referencias a archivos y valores de la aplicación (guardados en la carpeta res).
- ✓ **Carpeta assets**: Guarda el resto de archivos necesarios para el correcto funcionamiento de la aplicación, como los archivos de datos o de configuración. La principal diferencia entre los recursos que almacena ésta carpeta y los que guarda la carpeta "res", es que los recursos de ésta última generan un identificador por recurso, identificador que se encargará de gestionar el fichero R y sólo se podrá acceder a ellos a través de determinados métodos de acceso, mientras que los recursos almacenados en la carpeta "assets" no generan identificador alguno y se accederá a ellos a través de su ruta, como se hace con cualquier otro fichero.
- ✓ **Archivo AndroidManifest.xml**: Éste archivo es uno de los más importantes de cualquier aplicación Android. Se genera automáticamente al crear el proyecto, y en él se encuentra definida la configuración del proyecto en XML (Actividades, Intents, los permisos de la aplicación, bibliotecas, etc.)

6.5.1. Componentes de la Aplicación Móvil

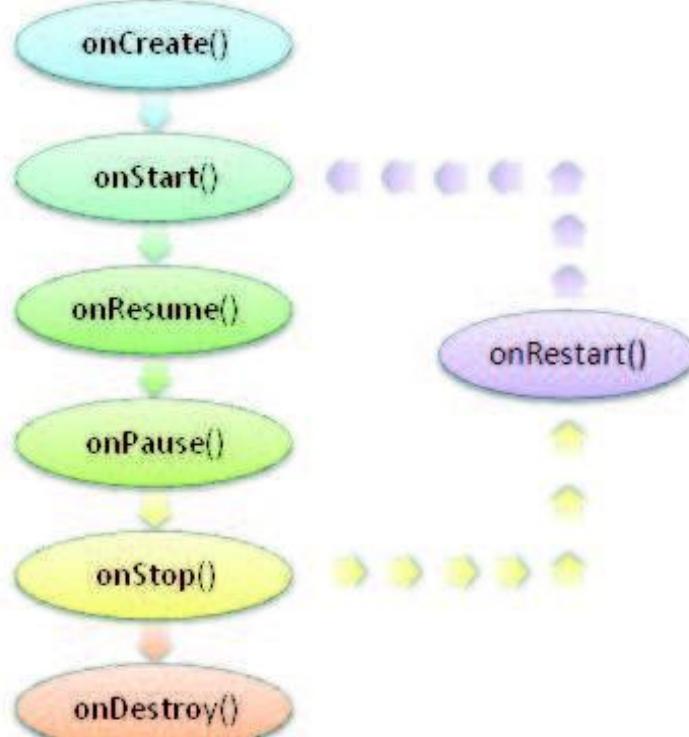
Los dispositivos tienen un único foco, la ejecución principal, que es la aplicación que está visible en la pantalla, pero puede tener varias aplicaciones en un segundo plano, cada una con su propia pila de tareas. La pila de tareas es la secuencia de ejecución de procesos en Android. Se componen de actividades que se van apilando según son invocadas, y solo pueden terminarse cuando las tareas que tiene encima están terminadas, o cuando el sistema las destruye porque necesita memoria, por lo que tienen que estar preparadas para terminar en cualquier momento. El sistema siempre eliminará la actividad que lleve más tiempo parada.





6.5.2. Actividades

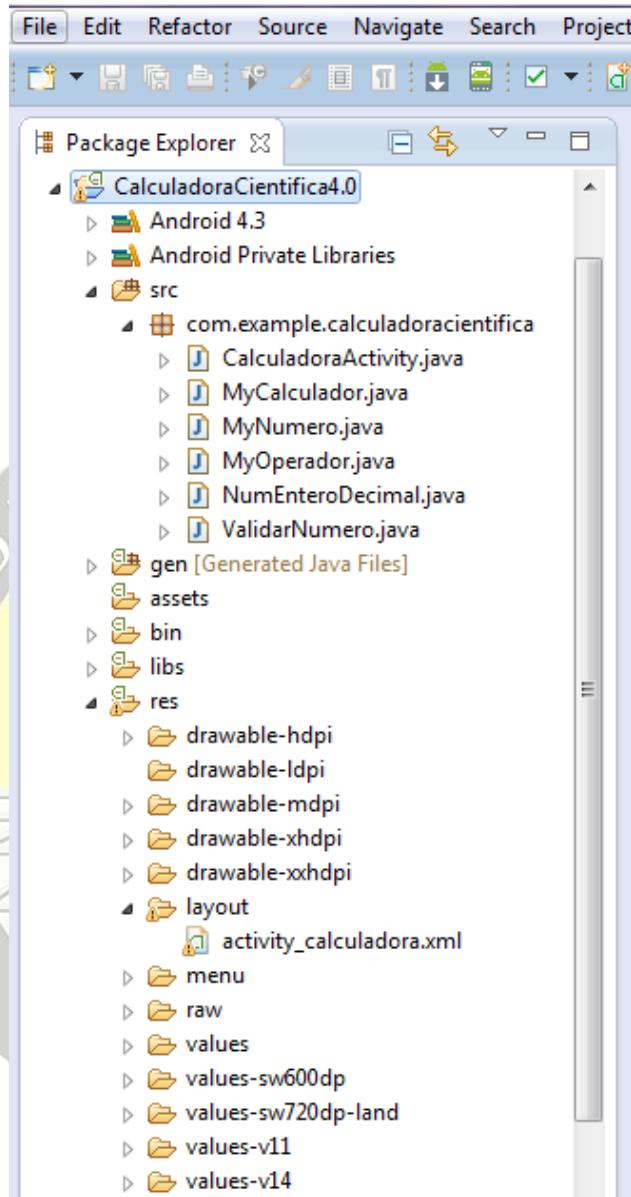
Una actividad (o Activity) es la componente principal encargada de mostrar al usuario la interfaz gráfica, es decir, una actividad sería el equivalente a una ventana, y es el medio de comunicación entre la aplicación y el usuario. Se define una actividad por cada interfaz del proyecto. Los elementos que se muestran en ella deben ser definidos en el fichero xml que llevan asociado (que se guarda en ./res/layout) para poder ser tratados en la clase NameActivity.class, que hereda de la clase Activity.





Est. Giovanni Morccolla A. - Irving Ortega Z.

6.6. Codificación de la Aplicación



El proyecto está basado a programación orientado a objetos lo cual Creamos cinco clases para desarrollar la aplicación de calculadora.

Interfaz de la Aplicación (Activity_calculadora.xml)



Clases:

- ✓ CalculadoraActiviy
- ✓ MyNumero
- ✓ MyOperaador
- ✓ MyCalculador
- ✓ ValidarNumero
- ✓ NumEnteroDecimal

Interfaz “Activity_calculadora.xml”: Es el diseño para que interactúe el usuario al momento de utilizar la calculadora en un dispositivo móvil.

The screenshot shows the Android Studio interface with two main windows. The top window displays the graphical layout of the activity_calculadora.xml file, showing a scientific calculator interface with a numeric keypad, arithmetic operators (+, -, ×, ÷, =), trigonometric functions (Sin, Cos, Tan), logarithmic functions (Log, LogN), and other mathematical operations (x², x³, √, !). The bottom window shows the corresponding XML code for the RelativeLayout structure, defining the layout parameters and components like the EditText for numbers and the RadioGroup for units (Sexo, Radi, Cent).

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".CalculadoraActivity" >

    <EditText
        android:id="@+id/Numero"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"
        android:cursorVisible="false"
        android:gravity="right" >
    </EditText>

    <RadioGroup
        android:id="@+id/radioGroup1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/Numero"
        android:layout_below="@+id/Numero"
        android:layout_marginTop="14dp" >
        <!-- Other radio buttons for Sexo, Radi, Cent -->
    </RadioGroup>

    <!-- Buttons for Sin, Cos, Tan, Log, etc. -->
    <!-- Buttons for x², x³, √, ! -->
    <!-- Buttons for +, -, ×, ÷, = -->
    <!-- Buttons for Shift, Off, etc. -->
</RelativeLayout>
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

Clase “CalculadoraActiviy”: Esta clase se crea por defecto, el código fuente está programado para recibir los eventos o escuchar del interfaz que va pulsar los botones el usuario, también visualiza los números y operaciones calculados y también se programa para emitir la voz o leer el numero ingresado o calculado.

```
//Autor: Giovanni Morccolla Ancco
package com.example.calculadorcientifica;

import android.media.MediaPlayer;

public class CalculadoraActivity extends Activity implements OnClickListener{

    EditText Numero;
    MediaPlayer md;
    private int tamNumeroAntes,decena,contador=0;
    TextView t;
    //*****
    MyNumero mynumero = new MyNumero();
    NumEnteroDecimal enteroDecimal = new NumEnteroDecimal();
    //*****
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_calculadora);
        //*****
        this.Numero =(EditText)findViewById(R.id.Numero);
        this.Numero.setText("0.0");
        tamNumeroAntes=-1;
        //*****
        final Button Cero=(Button)findViewById(R.id.Cero);
        final Button Uno=(Button)findViewById(R.id.Uno);
        final Button Dos=(Button)findViewById(R.id.Dos);
        final Button Tres=(Button)findViewById(R.id.Tres);
        final Button Cuatro=(Button)findViewById(R.id.Cuatro);
        final Button Cinco=(Button)findViewById(R.id.Cinco);
        final Button Seis=(Button)findViewById(R.id.Sies);
        final Button Siete=(Button)findViewById(R.id.Siete);
        final Button Ocho=(Button)findViewById(R.id.Ocho);
        final Button Nueve=(Button)findViewById(R.id.Nueve);
        final Button Punto=(Button)findViewById(R.id.Punto);

        //*****
        public void audioNumeros(int numero){
            switch (numero){
                case 0:
                    md = MediaPlayer.create(this, R.raw.cero);
                    if(tamNumeroAntes<0)
                    {
                        md.start();
                    }
                    break;
                case 1:
                    if(decena>1)
                    {
                        md = MediaPlayer.create(this, R.raw.y);
                        md.start();
                        SystemClock.sleep(100);
                    }
                    md = MediaPlayer.create(this, R.raw.uno);
                    md.start();
                    break;
                case 2:
                    if(decena>1)
                    {
                        md = MediaPlayer.create(this, R.raw.y);
                        md.start();
                        SystemClock.sleep(100);
                    }
            }
        }
    }
}
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
        md = MediaPlayer.create(this, R.raw.dos);
        md.start();
        break;
    case 3:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();
            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.tres);
        md.start();
        break;
    case 4:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();
            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.cuatro);
        md.start();
        break;
    case 5:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();
            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.cinco);
        md.start();
        break;
    case 6:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();
            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.seis);
        md.start();
        break;
    case 7:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();
            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.siete);
        md.start();
        break;
    case 8:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();
            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.ocho);
        md.start();
        break;
    case 9:
        if(decena>1)
        {
            md = MediaPlayer.create(this, R.raw.y);
            md.start();

            SystemClock.sleep(100);
        }
        md = MediaPlayer.create(this, R.raw.nueve);
        md.start();
        break;
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
case 9:  
    if(decena>1)  
    {  
        md = MediaPlayer.create(this, R.raw.y);  
        md.start();  
        SystemClock.sleep(100);  
    }  
    md = MediaPlayer.create(this, R.raw.nueve);  
    md.start();  
    break;  
case 10:  
    md = MediaPlayer.create(this, R.raw.dies);  
    md.start();  
    break;  
case 11:  
    md = MediaPlayer.create(this, R.raw.once);  
    md.start();  
    break;  
case 12:  
    md = MediaPlayer.create(this, R.raw.doce);  
    md.start();  
    break;  
case 13:  
    md = MediaPlayer.create(this, R.raw.trece);  
    md.start();  
    break;  
case 14:  
    md = MediaPlayer.create(this, R.raw.catorce);  
    md.start();  
    break;  
case 15:  
    md = MediaPlayer.create(this, R.raw.quince);  
    md.start();  
    break;  
  
case 16:  
    md = MediaPlayer.create(this, R.raw.diciseis);  
    md.start();  
    break;  
case 17:  
    md = MediaPlayer.create(this, R.raw.dicisiete);  
    md.start();  
    break;  
case 18:  
    md = MediaPlayer.create(this, R.raw.diciocho);  
    md.start();  
    break;  
case 19:  
    md = MediaPlayer.create(this, R.raw.dicinove);  
    md.start();  
    break;  
case 20:  
    md = MediaPlayer.create(this, R.raw.vente);  
    md.start();  
    break;  
case 30:  
    md = MediaPlayer.create(this, R.raw.treinta);  
    md.start();  
    break;  
case 40:  
    md = MediaPlayer.create(this, R.raw.cuarenta);  
    md.start();  
    break;  
case 50:  
    md = MediaPlayer.create(this, R.raw.cincuenta);  
    md.start();  
    break;  
  
case 60:  
    md = MediaPlayer.create(this, R.raw.secenta);  
    md.start();  
    break;  
case 70:  
    md = MediaPlayer.create(this, R.raw.setenta);  
    md.start();  
    break;
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
        case 80:
            md = MediaPlayer.create(this, R.raw.ochenta);
            md.start();
            break;
        case 90:
            md = MediaPlayer.create(this, R.raw.noventa);
            md.start();
            break;
        case 100:
            md = MediaPlayer.create(this, R.raw.cien);
            md.start();
            break;
        case 200:
            md = MediaPlayer.create(this, R.raw.doscientos);
            md.start();
            break;
        case 300:
            md = MediaPlayer.create(this, R.raw.trecientos);
            md.start();
            break;
        case 400:
            md = MediaPlayer.create(this, R.raw.cuatrocientos);
            md.start();
            break;
        case 500:
            md = MediaPlayer.create(this, R.raw.quinientos);
            md.start();
            break;
        case 600:
            md = MediaPlayer.create(this, R.raw.seiscientos);
            md.start();
            break;
        case 700:
            md = MediaPlayer.create(this, R.raw.setecientos);
            md.start();
            break;
        case 800:
            md = MediaPlayer.create(this, R.raw.ochocientos);
            md.start();
            break;
        case 900:
            md = MediaPlayer.create(this, R.raw.novecientos);
            md.start();
            break;
        case 1000:
            md = MediaPlayer.create(this, R.raw.mil);
            md.start();
            break;
    }
}
//*****
public void audioOperadores(int Operar){
    switch(Operar){
        case 1: //Suma
            md = MediaPlayer.create(this, R.raw.suma);
            md.start();
            break;
        case 2: //Resta
            md = MediaPlayer.create(this, R.raw.resta);
            md.start();
            break;
        case 3: //Multiplicacion
            md = MediaPlayer.create(this, R.raw.multiplicacion);
            md.start();
            break;
        case 4: //Division
            md = MediaPlayer.create(this, R.raw.division);
            md.start();
            break;
        case 5: //Punto
            md = MediaPlayer.create(this, R.raw.punto);
            md.start();
            break;
    }
}
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
        case 6: //Igual
            md = MediaPlayer.create(this, R.raw.igual);
            md.start();
            break;
        case 7: //X al Cuadrado
            md = MediaPlayer.create(this, R.raw.xalcuad);
            md.start();
            break;
        case 8: //X al Cubo
            md = MediaPlayer.create(this, R.raw.xalcub);
            md.start();
            break;
        case 9: // X a la N
            md = MediaPlayer.create(this, R.raw.xalan);
            md.start();
            break;
        case 10: //Raiz Cuadrada
            md = MediaPlayer.create(this, R.raw.raizcuad);
            md.start();
            break;
        case 11: //Raiz Cubica
            md = MediaPlayer.create(this, R.raw.raizcub);
            md.start();
            break;
        case 12: //Raiz Cubica
            md = MediaPlayer.create(this, R.raw.raizn);
            md.start();
            break;
        case 13: //Fraccion
            md = MediaPlayer.create(this, R.raw.fraccion);
            md.start();
            break;
    }
    //*****
    @Override
    public void onClick(View Click) {
        switch(Click.getId()){
            //*****
            case R.id.Cero:
                contador++;
                this.audioNumeros(0);
                if(contador==2)
                {
                    this.mynumero.Numero("0");
                    this.Numero.setText(""+mynumero.Resultado());
                    contador=0;
                }
                break;
            //*****
            case R.id.Uno:
                contador++;
                this.audioNumeros(1);
                if(contador==2)
                {
                    this.mynumero.Numero("1");
                    this.Numero.setText(""+mynumero.Resultado());
                    contador=0;
                }
                break;
            //*****
            case R.id.Dos:
                contador++;
                this.audioNumeros(2);
                if(contador==2)
                {
                    this.mynumero.Numero("2");
                    this.Numero.setText(""+mynumero.Resultado());
                    contador=0;
                }
                break;
        }
    }
    //*****
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
        case R.id.Tres:
            contador++;
            this.audioNumeros(3);
            if(contador==2)
            {
                this.mynumero.Numero("3");
                this.Numero.setText(""+mynumero.Resultado());
                contador=0;
            }
            break;
        //*****
        case R.id.Cuatro:
            contador++;
            this.audioNumeros(4);
            if(contador==2)
            {
                this.mynumero.Numero("4");
                this.Numero.setText(""+mynumero.Resultado());
                contador=0;
            }
            break;
        //*****
        case R.id.Cinco:
            contador++;
            this.audioNumeros(5);
            if(contador==2)
            {
                this.mynumero.Numero("5");
                this.Numero.setText(""+mynumero.Resultado());
                contador=0;
            }
            break;
        //*****
    public void LeerNumero(String resultado) {
    // TODO Auto-generated method stub

    int numeroPuntoAntes,hnumeroPuntoAntes;
    int tamNumeroDespues,numeroPD=0;
    String numeroPuntoDespues;

    NumEnteroDecimal numero = new NumEnteroDecimal();
    numeroPuntoAntes = Integer.parseInt(numero.PuntoAntes(resultado));
    numeroPuntoDespues = numero.PuntoDespues(resultado);

    tamNumeroAntes = numero.PuntoAntes(resultado).length();
    tamNumeroDespues = numero.PuntoDespues(resultado).length();

    hnumeroPuntoAntes = numeroPuntoAntes;
    while(tamNumeroAntes>0){
        tamNumeroAntes--;
        if(hnumeroPuntoAntes>10 && hnumeroPuntoAntes<20)
        {
            audioNumeros(hnumeroPuntoAntes);
            SystemClock.sleep(800);
            break;
        }
        else
        {
            numeroPuntoAntes = (int) (hnumeroPuntoAntes/Math.pow(10, tamNumeroAntes));
            hnumeroPuntoAntes = (int) (hnumeroPuntoAntes%Math.pow(10, tamNumeroAntes));
            if(tamNumeroAntes==1)
            {
                decena=numeroPuntoAntes;
            }
            audioNumeros((int)(numeroPuntoAntes*(Math.pow(10, tamNumeroAntes))));  
SystemClock.sleep(800);
        }
    }
    tamNumeroAntes=-1;
    decena=1;
    if(Integer.parseInt(numero.PuntoAntes(resultado))==0)
    {
        audioNumeros(0);
        SystemClock.sleep(800);
    }
}
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
        }
        if(Integer.parseInt(numeroPuntoDespues)!=0){
            audioOperadores(5);
            SystemClock.sleep(200);
            hnumeroPuntoAntes=Integer.parseInt(numeroPuntoDespues);
            while(tamNumeroDespues>0){
                numeroPD = (int) (hnumeroPuntoAntes/Math.pow(10, tamNumeroDespues-1));
                hnumeroPuntoAntes = (int) (hnumeroPuntoAntes%Math.pow(10, tamNumeroDespues-1));
                audioNumeros(numeroPD);
                SystemClock.sleep(400);
                tamNumeroDespues--;
            }
        }
        numero.PuntoAntes(resultado);
        t.setText(numero.PuntoAntes(resultado)+"      "+numero.PuntoDespues(resultado)+"
```

Clase “MyNumero”: Esta clase es la principal para realizar toda las acciones de la calculadora lo cual contiene propiedades y Constructores, Procedimientos: Numero (), Punto (), Operación (), ActivarShift (), Trigonometría (), Limpiar (). Funciones: Resultado () .

```
MyNumero.java
package com.example.calculadorcientifica;

public class MyNumero {
    private String NroA;
    private String NroB;
    private String Op;
    private int Operacion;
    private int Activar;
    private int Trigo;
    //*****
    public MyNumero(){
        this.NroA = "";
        this.NroB = "";
        this.Op = "";
        this.Operacion = 0;
        this.Activar = 0;
        this.Trigo = 1;
    }
    //*****
    MyOperador myoperador = new MyOperador();
    MyCalculador mycalculador = new MyCalculador();
    //*****
    public void Numero(String N){
        if(Operacion==0)
            if(NroA!="" && Op!="")
                { NroA = ""; Op=""; NroA = NroA + String.valueOf(N); }
            else
                NroA = NroA + String.valueOf(N);
        else
            NroB = NroB + String.valueOf(N);
    }
    //*****
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
public void Punto(){
    if(Operacion==0){
        if(!NroA.contains(".")) && NroA!="")
            NroA = NroA + ".";
    }
    else
        if(!NroB.contains(".")) && NroB!="")
            NroB = NroB + ".";
}
//*****
public String Resultado(){
    String R;
    if(Operacion==0)
        R = NroA;
    else
        R = NroB;
    if(NroA=="" && NroB=="")
        R = "0.0";
    return (R);
}
//*****
public void Operacion(String O) {
    Op = O;
    Operacion = this.myoperador.Operador(this.Op);
}
//*****
public void ActivarShift(){
    Activar = 1;
    this.mycalculador.Activar(this.Activar);
}
//*****
public void Trigonometria(int T) {
    this.Trigo = T;
}
//*****
public void Procesar(){
    double C;

    this.mycalculador.NumeroA(this.NroA);
    this.mycalculador.NumeroB(this.NroB);
    this.mycalculador.Trigonometria(this.Trigo);

    C = this.mycalculador.Calcular(this.Operacion);

    this.NroA = String.valueOf(C);
    this.NroB = "";
    this.Operacion = 0;
}
//*****
public void Limpiar(){
    this.NroA = "";
    this.NroB = "";
    this.Op = "";
    this.Operacion = 0;
    this.Activar = 0;
    this.mycalculador.Activar(this.Activar);
}
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

Clase "MyOperador": Esta clase es para seleccionar las operaciones que va realizar el usuario lo cual contiene un Procedimiento: Operador ():

```
MyOperador.java
package com.example.calculadorcientifica;

public class MyOperador {
    private int Opera;
    //*****
    public MyOperador(){
        this.Opera = 0;
    }
    //*****
    public int Operador(String O){
        if(O.compareTo("+")==0) Opera = 1; // Suma
        if(O.compareTo("-")==0) Opera = 2; // Resta
        if(O.compareTo("*")==0) Opera = 3; // Multiplicacion
        if(O.compareTo("/")==0) Opera = 4; // Division
        if(O.compareTo("cu")==0) Opera = 5; // Cuadrado de un Numero
        if(O.compareTo("po")==0) Opera = 6; // Potencia de un Numero a otro Numero
        if(O.compareTo("ra")==0) Opera = 7; // Raiz Cuadrada de un Numero
        if(O.compareTo("rn")==0) Opera = 8; // Raiz N de un Numero
        if(O.compareTo("fr")==0) Opera = 9; // Fraccion de un Numero
        if(O.compareTo("pi")==0) Opera = 10; // Valor de PI
        if(O.compareTo("lo")==0) Opera = 11; // Logaritmo de un Numero
        if(O.compareTo("e")==0) Opera = 12; // Valor de Epsilo
        if(O.compareTo("fa")==0) Opera = 13; // Factorial de un Numero
        if(O.compareTo("pr")==0) Opera = 14; // Porcentaje de un Numero
        if(O.compareTo("cos")==0) Opera = 15; // Coseno de un Numero
        if(O.compareTo("sen")==0) Opera = 16; // Seno de un Numero
        if(O.compareTo("tan")==0) Opera = 17; // Tangente de un Numero

        return (Opera);
    }
    //*****
}
```

Clase "MyCalculador": Esta clase es para procesar o calcular la operación ingresada por el usuario tiene Métodos: NumeroA(), NumeroB(), Activar(), Trigonometría(). Y un Procedimiento: Calcular () .





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
MyCalculador.java
package com.example.calculadorcientifica;

public class MyCalculador {
    private int Operar;
    private String NroA;
    private String NroB;
    private int Activar;
    private int Trigo;
    //*****
    ValidarNumero validar = new ValidarNumero();
    //*****
    public MyCalculador(){
        this.Operar = 0;
        this.NroA = "";
        this.NroB = "";
        this.Activar = 0;
        this.Trigo = 0;
    }
    //*****
    public void NumeroA(String nroA) {
        NroA = nroA;
    }
    //*****
    public void NumeroB(String nroB) {
        NroB = nroB;
    }
    //*****
    public void Activar(int activar) {
        Activar = activar;
    }
    //*****
    public void Trigonometria(int Tri) {
        Trigo = Tri;
    }
    //*****
    public double Calcular(int Op){
        Operar = Op;
        double A, B, C = 0;
        int i, f;
        A = this.validar.ValidarNroA(this.NroA);
        B = this.validar.ValidarNroA(this.NroB);

        // Convertimos el numero para calcular en Radianes o sexagesimal o centesimal
        if(this.Trigo==1 && (Operar==15||Operar==16||Operar==17)) A = (A*Math.PI)/180;
        //if(this.Trigo==3 && (Operar==15||Operar==16||Operar==17)) A = A;
        if(this.Trigo==3 && (Operar==15||Operar==16||Operar==17)) A = (A*Math.PI)/200;

        switch(this.Operar){
            //-
            case 1:   C = A + B;                                break;
            //-
            case 2:   C = A - B;                                break;
            //-
            case 3:   C = A * B;                                break;
            //-
            case 4:   C = A / B;                                break;
            //-
            case 5:   if(Activar!=1)
                        C = Math.pow(A,2);
                    else
                        C = Math.pow(A,3);                break;
            //-
            case 6:   C = Math.pow(A,B);                          break;
            //-
        }
    }
}
```





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
        case 7:    if(Activar!=1)
                    C = Math.pow(A,0.5);
                else
                    C = Math.pow(A,(1.0/3));           break;
        //-----
        case 8:    C = Math.pow(A,(1.0/B));           break;
        //-----
        case 9:    C = (1.0)/A;                      break;
        //-----
        case 10:   C = Math.PI;                      break;
        //-----
        case 11:   if(Activar!=1)
                    C = Math.Log10(A);
                else
                    C = Math.Log(A);             break;
        //-----
        case 12:   if(Activar!=1)
                    C = Math.E;
                else
                    C = Math.PI;                 break;
        //-----
        case 13:   i=1; f=1;
                    while(i<=A) { f=f*i; i++; } C = f;      break;
        //-----
        case 14:   C = A/100;                      break;
        //-----
        case 15:   if(Activar!=1)
                    C = Math.cos(A);
                else
                    C = Math.acos(A);            break;
        //-----
        case 16:   if(Activar!=1)
                    C = Math.sin(A);
                else
                    C = Math.asin(A);           break;
        //-----
        case 17:   if(Activar!=1)
                    C = Math.tan(A);
                else
                    C = Math.atan(A);          break;
        //-----
    }
    return (C);
}
```

ANDAHUAYLAS

Clase “ValidarNúmero”: Esta clase es para validar el número ingresado por el usuario contiene constructores y Funciones: ValidarNroA(), ValidarNroB().





Est. Giovanni Morccolla A. - Irving Ortega Z.

```
ValidarNumero.java X
package com.example.calculadoracientifica;

public class ValidarNumero {

    private double A;
    private double B;
    //*****
    public ValidarNumero(){
        this.A = 0.0;
        this.B = 0.0;
    }
    //*****
    public double ValidarNroA(String nroA){

        try { A = Double.parseDouble(nroA);
        }
        catch(Exception e){ A = 0; }

        return (A);
    }
    //*****
    public double ValidarNroB(String nroB){

        try { B = Double.parseDouble(nroB);
        }
        catch(Exception e){ B = 0; }

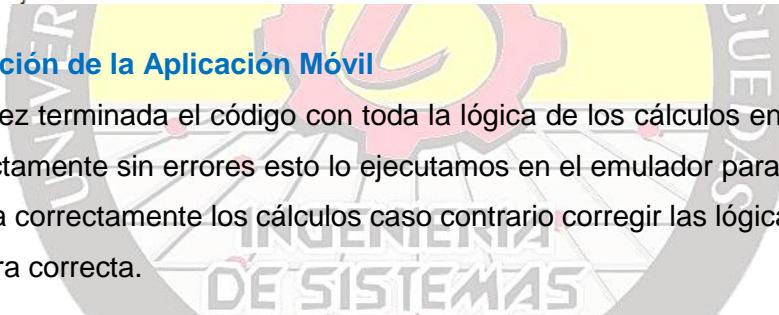
        return (B);
    }
}
```

Clase NumEnteroDecimal: esta clase es para separar números parte entera y parte decimal par que pueda leer o emitir el sonido.





Est. Giovanni Morccolla A. - Irving Ortega Z.

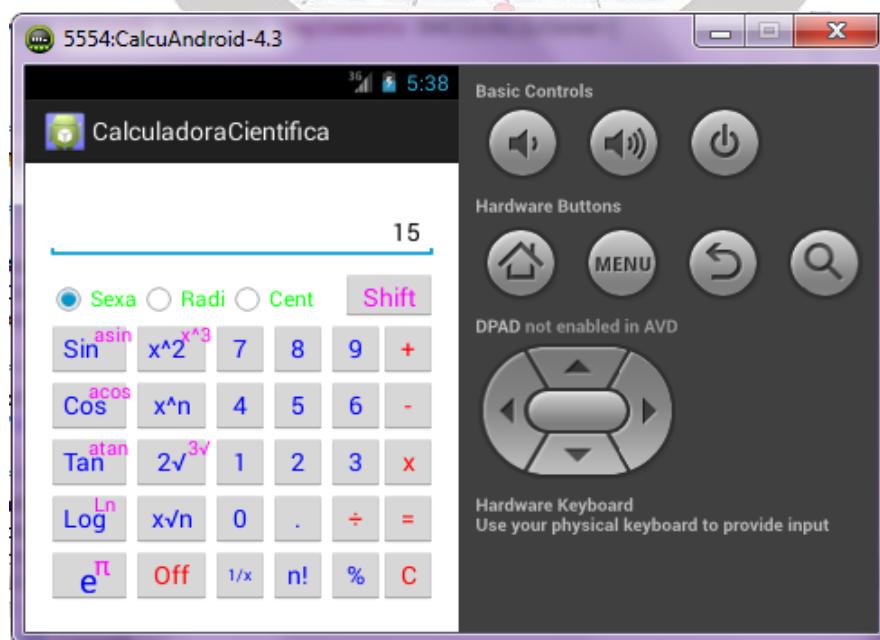


```
NumEnteroDecimal.java
package com.example.calculadoracientifica;
public class NumEnteroDecimal {

    public NumEnteroDecimal(){
    }
    //*****Este método captura los numeros antes del punto*****
    public String PuntoAntes(String n){
        int i=0, j=0;
        String num, strEnter = "";
        num = n;
        for(i=0; i<num.length();i++){
            if(num.charAt(i)=='.'){
                j = 1;
            }
            if(j==0){
                strEnter+=num.charAt(i);
            }
        }
        return strEnter;
    }
    //*****Este método captura los numeros despues del punto*****
    public String PuntoDespues(String n){
        int i=0,j=0;
        String num,strDecimal="";
        num = n;
        for(i=0; i<num.length();i++){
            if(num.charAt(i)=='.'){
                j = 1;
            }
            if(j==1 && num.charAt(i)!='.'){
                strDecimal+=num.charAt(i);
            }
        }
        return strDecimal;
    }
}
```

6.7. Ejecución de la Aplicación Móvil

Una vez terminada el código con toda la lógica de los cálculos en toda las clases correctamente sin errores esto lo ejecutamos en el emulador para poder probar si realiza correctamente los cálculos caso contrario corregir las lógicas de cálculo de manera correcta.





Est. Giovanni Morccolla A. - Irving Ortega Z.

7. Conclusiones

- ✓ El desarrollo de aplicaciones móviles en Android puede ser explorado en muchos campos o temas para beneficios y comodidad de los usuarios finales, ofrecer una ayuda valiosa a la sociedad en conjunto.
- ✓ Para desarrollar la aplicación móvil en Android es necesario realizar todo el estudio de los requerimientos y diseño, y teniendo estas en claro se logra lo que se va implementar y así no se presenten errores durante el desarrollo.
- ✓ Para desarrollar la aplicación móvil en Android fue necesario estudiar, investigar y analizar las clases, métodos, funciones, reproducción de audios, etc. para realizar la codificación de la aplicación.
- ✓ Las herramientas usadas para desarrollar aplicaciones móviles en Android es Emulador Virtual ADV (Android Virtual Device), SDK de Android e IDE Eclipse, que proveen una serie de características para facilitar el desarrollo de aplicaciones móviles en Android.
- ✓ El sistema operativo Android evoluciona para brindar a los desarrolladores nuevas características que permiten crear y desarrollar aplicaciones cada vez más complejas y completas, para satisfacer las necesidades de los usuarios finales.
- ✓ El emulador virtual ADV (Android Virtual Device), es una herramienta completa y sofisticada que facilita el desarrollo de aplicaciones móviles en Android, gracias a que permite ejecutarlas durante el proceso de creación de aplicación.





Est. Giovanni Morccolla A. - Irving Ortega Z.

8. Referencias

- [1] David Robledo Fernández. **Desarrollo de aplicaciones para Android II**
<http://descargas.pntic.mec.es/mentor/visitas/DesarrolloAppsAndroidII.pdf>
- [2] Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo, Pilar Torralbo, Álvaro Zapata. **Introducción a Android.**
<http://pendientedemigracion.ucm.es/info/tecnomovil/documentos/android.pdf>
- [3] <http://www.olimpiadasandroid.com/pdf/preparate-para-las-olimpiadas-android.pdf>
- [4] <http://developer.android.com/sdk/index.html>
- [5] [http://www.exabyteinformatica.com/uoc/Informatica/Tecnologia_y_desarrollo_en_dispositivos_moviles/Tecnologia_y_desarrollo_en_dispositivos_moviles_\(Modulo_4\).pdf](http://www.exabyteinformatica.com/uoc/Informatica/Tecnologia_y_desarrollo_en_dispositivos_moviles/Tecnologia_y_desarrollo_en_dispositivos_moviles_(Modulo_4).pdf)
- [6] <http://www.adrformacion.com/cursos/android/leccion1/tutorial3.html>
- [7] <http://www.taringa.net/posts/apuntes-y-monografias/17190144/Crear-y-configurar-nuestro-emulador-Android.html>
- [8] <http://geekytheory.com/tutorial-sonidos-en-android-mediaplayer-y-soundpool/>
- [9] <https://www.youtube.com/watch?v=hbplpL5nvc>
- [10] <http://jomobiletutorials.blogspot.com/2013/05/reproducir-archivo-de-audio-o-video-en.html>

