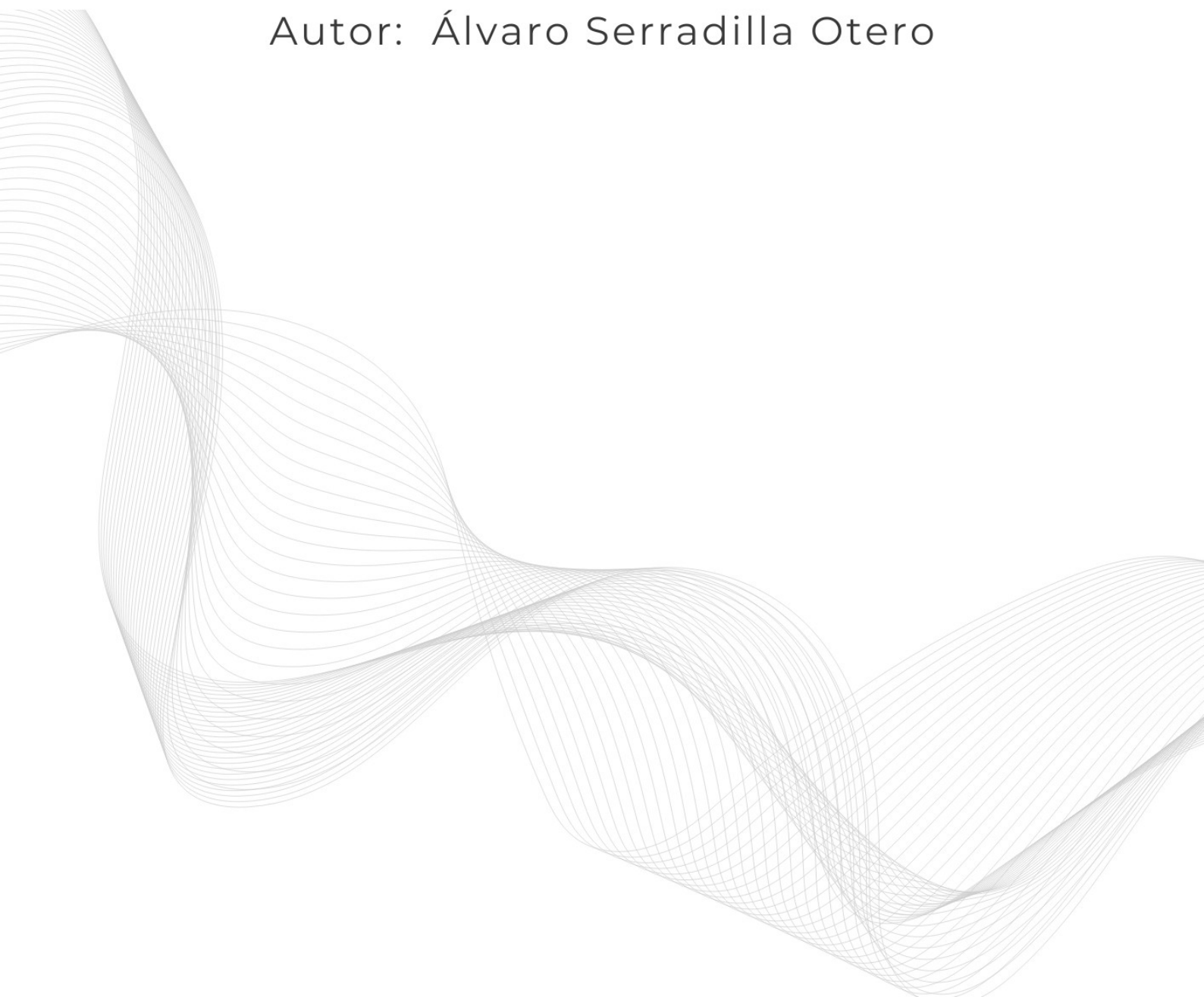


PROGRAMACIÓN
MULTIMEDIA Y DE
DISPOSITIVOS MÓVILE

DOCUMENTACIÓN TABATA

Autor: Álvaro Serradilla Otero



Índice

1. Recursos.....3

2. Descarga.....4

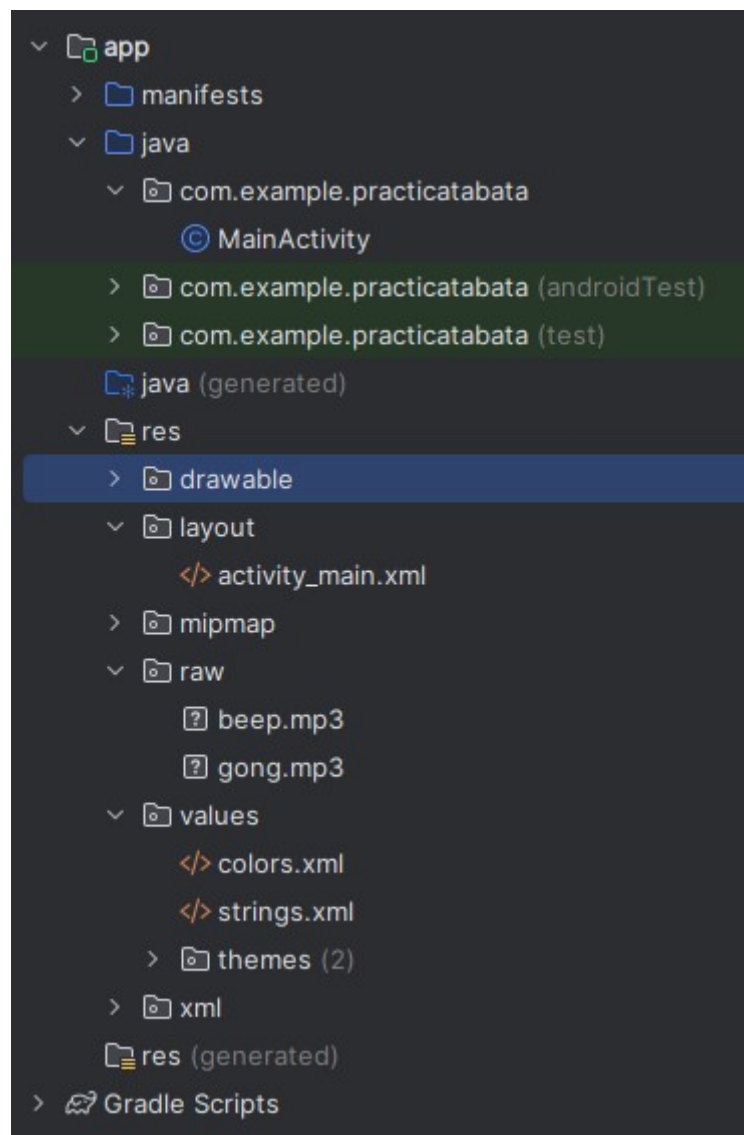
3. Clases.....5

1. Recursos

Para realizar este programa hemos seleccionado el entorno de desarrollo de Android Studio en su versión llamada Meerkat, la versión de API seleccionada a sido lollipop ya que nos permitirá comercializar nuestra app para un publico mayor.

Los lenguajes de programación escogidos fueron java y xml, el lenguaje de java se utilizara para las clases de la aplicación que nos servirá para programar las funciones del programa y el lenguaje xml lo utilizaremos para establecer todo el apartado visual.

El árbol de carpetas de nuestra aplicación quedara tal que así:



2. Descarga

Para descargar el programa hemos utilizado la plataforma de GitHub donde se puede descargar la carpeta del programa y probarlo, el enlace para descargarlo es el siguiente:

<https://github.com/AlvaroSerra24/DAM2.git>

Aquí podrás encontrar esta practica junto con otras realizadas por mi.

3. Clases

El programa cuenta con cierta variedad de clases que nos sirven para estructurar el programa de forma mas limpia, en primer lugar tenemos diversas clases que nos sirven para inicializar los diversos componentes que vamos a utilizar en el programa y ademas de configurar el botón de inicio el cual nos servirá para iniciar la función del programa, ademas de comprobar de ante mano si el programa ya esta en ejecución o no.

```
private void iniciarComponentes() { 1 usage
    start=findViewById(R.id.ButtonStart);
    main=findViewById(R.id.main);
    Series=findViewById(R.id.editTextSeries);
    Trabajo=findViewById(R.id.editTextTrabajo);
    Descanso=findViewById(R.id.editTextDescanso);
    cronometro=findViewById(R.id.textNumber);
    estado=findViewById(R.id.textWorkRest);
    left=findViewById(R.id.textSeriesLeft);
}

private void configurarBotonInicio() { 1 usage
    start.setOnClickListener( View view -> {
        if(!isRunning){
            IniciarTabata();
        }
    });
}
```

Aquí tras pulsar el botón se iniciara un nuevo método llamado IniciarTabata en el cual se encarga principalmente de comprobar si los valores introducidos en los tres campos son números y que sean mayores que 0, si esto se cumple el programa iniciara la cuenta atrás del trabajo.

```
private void IniciarTabata() { 1 usage
    try{
        timeTrabajo=Integer.parseInt(Trabajo.getText().toString());
        LeftSeries=Integer.parseInt(Series.getText().toString());
        timeDescanso=Integer.parseInt(Descanso.getText().toString());

        if(LeftSeries <= 0 || timeTrabajo <= 0 || timeDescanso <= 0) {
            throw new NumberFormatException("Valores deben ser mayores a 0");
        }else{
            isRunning=true;
            CuentaTrabajo(timeTrabajo, LeftSeries);
        }
    }catch (NumberFormatException e){
        mostrarError( mensaje: "Por favor ingrese valores válidos en los campos");
    }
}

private void mostrarError(String mensaje) { 1 usage
    Toast.makeText( context: this, mensaje, Toast.LENGTH_SHORT).show();
}
```

Tras iniciar el programa el programa iniciara a ejecutar el método CuentaTrabajo, en el cual podremos ver como la interfaz del programa cambiara para hacernos saber que el programa a entrado en la fase de trabajo, ademas de que en este metodo podremos ver como el cronometro realiza la cuenta atrás del tiempo asignado.

```
private void CuentaTrabajo(int time, int LeftSeries) { 2 usages
    reproducirSonido(R.raw.beep);
    actualizarUI(estadoTexto: "WORK", Color.GREEN, LeftSeries);
    countdownTimer = new CountDownTimer( millisInFuture: time*1000, countDownInterval: 1000) {
        @Override no usages
        public void onTick(long l) {
            long tiempo= l/1000;
            cronometro.setText(String.valueOf(tiempo));
        }

        @Override no usages
        public void onFinish() { CuentaDescanso(timeDescanso, Left: LeftSeries-1); }
    }.start();
}
```

En este métodos también podemos ver que se llaman a dos métodos principalmente para reproducir un sonido de beep y actualizar la interfaz para el usuario.

```
private void reproducirSonido(int sonido) { 3 usages
    if (beep != null) {
        beep.release();
    }
    beep = MediaPlayer.create( context: this, sonido);
    beep.start();
}

private void actualizarUI(String estadoTexto, int color, int seriesRestantes) { 3 usages
    estado.setText(estadoTexto);
    main.setBackgroundColor(color);
    left.setText("Series Left: " + seriesRestantes);
}
```

Y por ultimo tenemos el método Cuenta Descanso el cual es llamado cuando la cuenta atrás del trabajo termina, en este método podemos ver que se asemeja mucho con el método de CuentaTrabajo salvo que la interfaz tiene ciertos cambio y que al terminar la cuenta atrás del descanso debe comprobar si las series que quedan, en caso de que dicha variable no sea 0 deberá volver a ejecutar el método CuentaTrabajo y en caso de que sea 0 realizar los cambios necesario en la interfaz y reproducir un sonido de gong, ademas de hacerle saber al programa que este ya no se esta ejecutando por si quiere volver a pulsar el botón el programa se vuelva a ejecutar como es debido.

```

private void CuentaDescanso(int timeDescanso, int Left) { 1 usage
    reproducirSonido(R.raw.beep);
    actualizarUI( estadoTexto: "REST",Color.RED,Left);
    timeTrabajo = Integer.parseInt(Trabajo.getText().toString());

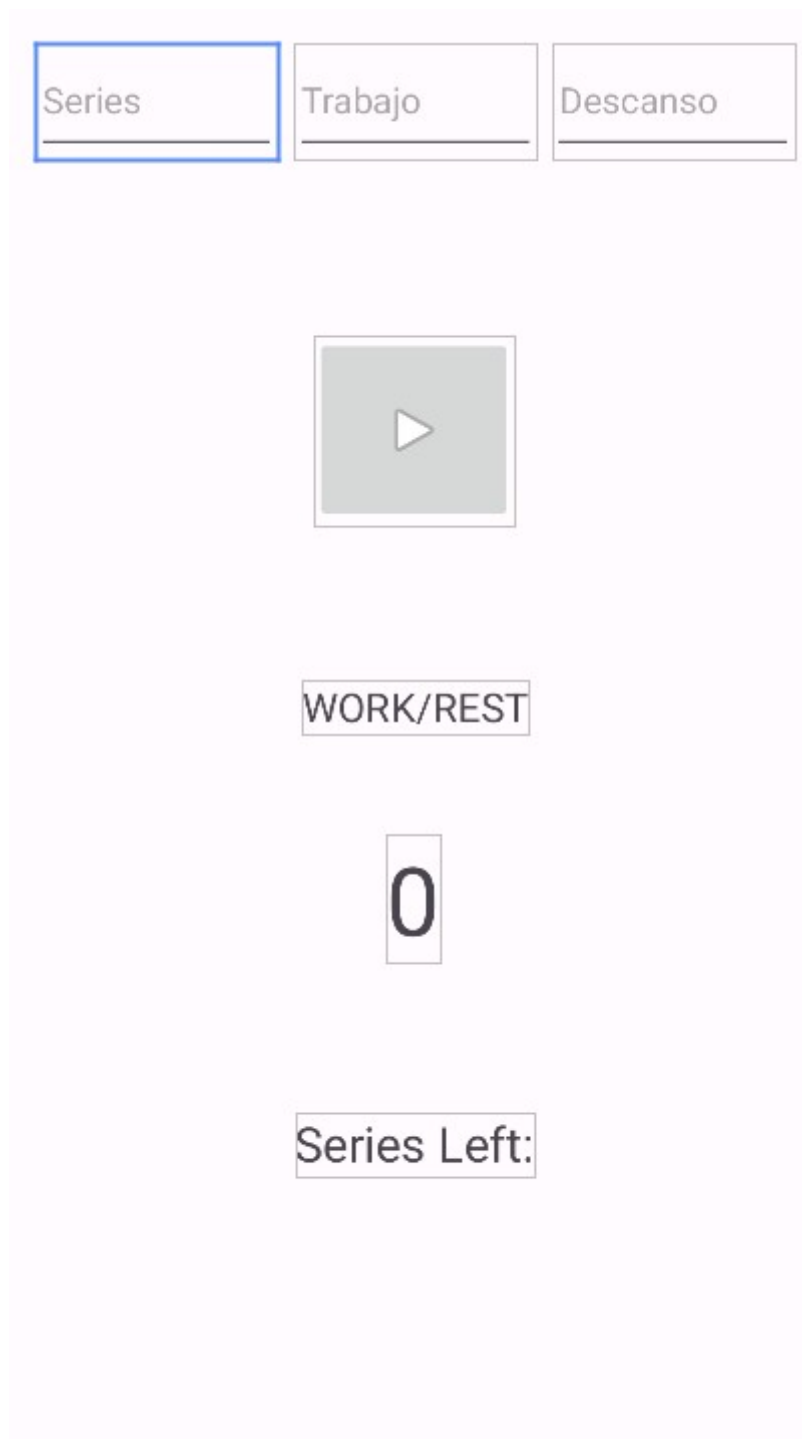
    countdownTimer = new CountDownTimer( millisInFuture: timeDescanso*1000, countDownInterval: 1000) {
        @Override no usages
        public void onTick(long l) {
            long tiempo= l/1000;
            cronometro.setText(String.valueOf(tiempo));
        }

        @Override no usages
        public void onFinish() {
            if(Left!=0){
                CuentaTrabajo(timeTrabajo, Left);
            }else{
                reproducirSonido(R.raw.gong);
                actualizarUI( estadoTexto: "FINISH",Color.GRAY,Left);
                isRunning = false;
            }
        }
    }.start();
}

```

4. Interfaz

Para la interfaz hemos optado por tres campos de texto en la parte superior de la ventana para rellenar con el tiempo que queramos que dure el trabajo y descanso y el numero de series a realizar, ademas de esto contamos con un texto el cual nos representara si nos encontramos en la cuenta atrás del trabajo o descanso acompañado en la parte inferior de otro texto el cual nos ira diciendo el numero de series que nos falta, y para terminar el programa cuenta con un texto en el centro de este para ver la cuenta atrás y encima de este tendremos el botón para iniciar el programa.



The image shows a UI mockup for a timer application. At the top, there are three input fields labeled "Series", "Trabajo", and "Descanso". Below these fields is a large play button icon. Under the play button is a label "WORK/REST". Below that is a large digital display showing the number "0". At the bottom is a label "Series Left:".

5. Funcionalidad

En este programa mediante la introducción del tiempo de descanso y trabajo seguido del numero de series se nos iniciara una cuenta atrás de trabajo seguido de una cuenta atrás de descanso hasta que el numero de series llegue a cero, esto nos servirá para realizar diversos tipos de ejercicio, acompañado de este un sonido de beep cada vez que inicia una cuenta atrás y con el cambio de color a verde si es para trabajo y rojo si es para el descanso, ademas cuando finalice el fondo de la pantalla pasara a gris acompañado de un sonido de gong.

