Marc Cerezo Heredero
Monlau grupo DAM

Documentación del Reto “Paleta de colores aleatoria” para Android

Tecnología utilizada durante el reto:

Para este reto he preferido utilizar Java, ya que es una herramienta que hoy día se utiliza en casi todas las empresas que trabajan este tipo de campo, a parte de ser un lenguaje el cual me gusta practicar cada día.

Visual Studio: Para la creación de la propia aplicación he utilizado Visual Studio, ya que es una herramienta con la que aún me cuesta trabajar, y con la que el reto obtendría una nueva dimensión en tanto a la dificultad planteada.

Aunque, antes de lanzarme a la creación de la aplicación desde cero en VisualStudio, hice una fase “beta” de la misma en intelliJ, donde podría crear la lógica del programa sin demasiados contra tiempos.

Documentación:

Para este programa, como en todos en los que trabajo me gusta distribuirlo en pequeñas tareas o problemas a resolver, ya que a veces, cuando miramos hacia el conjunto de problemas puede ser difícil discernir como llegar a la solución.

Por lo que lo primero que hice fue crear la interfaz del usuario y hacer que solamente el primer “textView” funcionase gracias a la escritura del color en formato “Hex” en un “EditText” recogido gracias a un botón, quedando el código de esta forma:

```
TextView textViewHex1;
EditText editTextColorHex;
textViewHex1 = (TextView) findViewById(R.id.textViewHex1);
editTextColorHex = (EditText) findViewById(R.id.editTextColorHex);
String firstHex = "";
firstHex = editTextColorHex.getText().toString();
textViewHex1.setBackgroundColor(Color.parseColor(firstHex));
```

Lograda la tarea de que nuestra aplicación pudiera modificar el color indicado por el usuario ahora era el momento de crear los otros “textView” y modificarlos, aunque esto iba a requerir de la creación de un nuevo “method” o método donde íbamos a crear la lógica para obtener colores en formato “Hex” de forma aleatoria.

Un problema constante era la aparición de las variables globales, las cuales perjudican nuestra aplicación si quisiéramos utilizar los métodos en otras aplicaciones, por lo que decidí trabajar solo con variables locales que pudieran ser llamadas por otros métodos cuando fuera necesario:

Marc Cerezo Heredero
Monlau grupo DAM

```
protected String[] randomHex() {


    String[] randomHex1;
    int Nmax = 5;
    String colorCode;

    randomHex1 = new String[Nmax];

    EditText editTextColorHex;
    TextView textViewHex1, textViewHex2
,textViewHex3,textViewHex4,textViewHex5;



    editTextColorHex = (EditText)
findViewById(R.id.editTextColorHex);
    textViewHex1 = (TextView) findViewById(R.id.textViewHex1);
    textViewHex2 = (TextView) findViewById(R.id.textViewHex2);
    textViewHex3 = (TextView) findViewById(R.id.textViewHex3);
    textViewHex4 = (TextView) findViewById(R.id.textViewHex4);
    textViewHex5 = (TextView) findViewById(R.id.textViewHex5);




    for(int y =0; y < randomHex1.length; y++){
        Random obj = new Random();
        int rand_num = obj.nextInt(0xffffff + 1);
        colorCode = String.format("#%06x", rand_num);
        randomHex1[y] = colorCode;
    }

textViewHex2.setBackgroundColor(Color.parseColor(randomHex1[0]));
        textViewHex2.setText(randomHex1[0]);

textViewHex3.setBackgroundColor(Color.parseColor(randomHex1[1]));
        textViewHex3.setText(randomHex1[1]);

textViewHex4.setBackgroundColor(Color.parseColor(randomHex1[2]));
        textViewHex4.setText(randomHex1[2]);

textViewHex5.setBackgroundColor(Color.parseColor(randomHex1[3]));
        textViewHex5.setText(randomHex1[3]);
    return randomHex1;
}
return randomHex1;
}
```

La funcion de este metodo es la de crear un array con los colores “Hex” y darselos a cada uno de los “textView” y la de indicar, a demás dentro de cada uno de estos “textView” el código de color que les está afectando.

La anterior función, si es cierto que ya funciona no va a modificar nada, ya que nunca la llamamos, por lo que creamos un nuevo método el cual la llame al principio del programa para colocar los 4 colores nada más iniciarse:

```
private void initConfig(){
    randomHex();

 }
```

Hecho esto solo nos quedaba hacer que, cuando el usuario introdujese un espacio en el "EditText" todos los "textView" se modificasen a la vez, a parte de solucionar el problema de que si el usuario introduce un texto que no es ni un código "Hex" valido o la tecla espacio el programa se cierra de forma inesperada.

Para ello, tras varios intentos acabe solventando los 2 problemas con un "try catch":

```
try {
    firstHex = editTextColorHex.getText().toString();
    textViewHex1.setBackgroundColor(Color.parseColor(firstHex));
    textViewHex1.setText(firstHex);
}catch (Exception e){
    if(firstHex.equals(" ")) {
        String[] HexRand = randomHex();

textViewHex1.setBackgroundColor(Color.parseColor(HexRand[4]));
        textViewHex1.setText(HexRand[4]);
    }
    else{editTextColorHex.setError("Invalid Color");
    }
}
```

Con esto el programa ya funciona correctamente y hace todas las exigencias dadas en el reto:

Marc Cerezo Heredero
Monlau grupo DAM