Anotacion ColorRes

Para esta clase puedes emplear el proyecto que hemos creado en la clase anterior. El "@ColorRes" permite obtener un entero que representa un color desde la ubicación de los recursos. La documentación de esta anotación la veremos en el siguiente enlace:

https://github.com/androidannotations/androidannotations/wiki/Resources#colorres

Añadiendo los colores

Vamos a añadir los siguientes recursos en `res >> values >> colors.xml`:

```
Key = iron_man_1
    Value = #FF4A1011
Key = iron_man_2
    Value = #FF7C2018
Key = iron_man_3
    Value = #FFD44942
Key = iron_man_4
    Value = #FFE4DE54
Key = iron_man_5
    Value = #FFA96F2D

Key = viuda_negra_1
    Value = #FF2D3D5C
Key = viuda_negra_2
    Value = #FF450000
Key = viuda_negra_3
    Value = #FF000000
Key = viuda_negra_4
    Value = #FF636162
Key = viuda_negra_5
    Value = #FF00091C

Key = capitan_america_1
    Value = #FF114FD9
Key = capitan_america_2
    Value = #FF3B7DA4
Key = capitan_america_3
    Value = #FFF4F4F3
Key = capitan_america_4
    Value = #FFFF0000
Key = capitan_america_5
    Value = #FF8B1A22

Key = thor_1
    Value = #FF264F73
Key = thor_2
    Value = #FF04C4D9
```

```
Key = thor_3
    Value = #FF03A61C
Key = thor_4
    Value = #FFA64F03
Key = thor_5
    Value = #FFA61717

Key = hulk_1
    Value = #FF022601
Key = hulk_2
    Value = #FF0ABF04
Key = hulk_3
    Value = #FF09A603
Key = hulk_4
    Value = #FF055902
Key = hulk_5
    Value = #FF010D00

Key = sprider_man_1
    Value = #FFA6242F
Key = sprider_man_2
    Value = #FF344973
Key = sprider_man_3
    Value = #FF32A9D9
Key = sprider_man_4
    Value = #FF30BAD9
Key = sprider_man_5
    Value = #FFD92B2B
```

Inicializando los recursos en el Java

Para inicializar un int de Java desde un recurso de color, debe agregar "@ColorRes" y hay tres formas:

- Utilizando el ID de recurso
- Directamente
- Directamente con más de una variable.

Usando el ID de recurso

El identificador del recurso debe estar en el parámetro del "@ColorRes":

```
@ColorRes(R.color.iron_man_1)
int ironMan1;
```

Directamente

El identificador será el nombre de la variable.

```
@ColorRes
int viuda_negra_1;
```

Directamente con más de una variable.

La misma filosofía que la opción previa pero con más de una variable.

```
@ColorRes
int viuda_negra_1, viuda_negra_1b;
```

Recuperando todos los recursos

A continuación agregaremos este código en la clase `MainActivity`:

```
@ColorRes(R.color.iron_man_1)
int ironMan1;
@ColorRes(R.color.iron_man_2)
int ironMan2;
@ColorRes(R.color.iron_man_3)
int ironMan3;
@ColorRes(R.color.iron_man_4)
int ironMan4;
@ColorRes(R.color.iron_man_5)
int ironMan5;
@ColorRes
int viuda_negra_1, viuda_negra_2, viuda_negra_3, viuda_negra_4, viud
@ColorRes
int capitan_america_1, capitan_america_2, capitan_america_3, capitan
@ColorRes
int thor_1, thor_2, thor_3,  thor_4,  thor_5;
@ColorRes
int hulk_1, hulk_2, hulk_3, hulk_4, hulk_5;
@ColorRes
int sprider_man_1, sprider_man_2, sprider_man_3, sprider_man_4, spri
```

Y actualizaremos el método bgenerateAvengerInfo:

```
private String generateAvengerInfo(String actor, String heroe, int c
    return actor + " as " + heroe + " with colors [#" + Integer.toHe
}
```

Finalmente vamos a actualizar el método anotado con `@AfterViews` y dentro del método mostraremos con el debugger todos los recursos que llaman al método anterior (generateAvengerInfo).