Práctica final

Recuperaremos la práctica del curso anterior e iremos haciendo las modificaciones.

CustomView

Añadir en `ImageView imageView;` la anotación de `ViewById`:

```
@ViewById(R.id.iv_avenger_avatar)
ImageView imageView;
```

Y borrar la línea:

```
imageView = findViewById(R.id.iv_avenger_avatar);
```

Añadir en `CustomTextView_ customTextView;` la anotación de `ViewById`, recordar que al tener la "_" se está empleando la clase generada por AndroidAnnotations, por lo que se deberá eliminar, quedando:

```
@ViewById(R.id.ctv_avenger_info)
CustomTextView customTextView;
```

Y borrar la línea:

```
customTextView = findViewById(R.id.ctv_avenger_info);
```

Añadir en `View custom_avenger_view;` la anotación de `ViewById`:

```
@ViewById
View custom_avenger_view;
```

Y borrar la línea:

```
custom_avenger_view = findViewById(R.id.ctv_avenger_info);
```

A continuación haremos el evento para los botones. Como hay que añadir una animación, primero la añadiremos a los recursos de la aplicación, en la carpeta `anim`, la animación será `blink.xml`

```
@AnimationRes(R.anim.blink)
Animation blink;
```

Tras ello, instnciaremos con AndroidAnnotation los `Button b1, b2, b3, b4, b5;` con:

```
    @ViewById
    Button button1, button2, button3, button4, button5;
```

Y posteriormente añadiremos el evento de onTouch:

```
    @Touch({R.id.button1, R.id.button2, R.id.button3, R.id.button4, R.id
    void bOperationsButtons(View v, MotionEvent event) {
        if (MotionEvent.ACTION_DOWN == event.getAction()) {
            v.setAnimation(blink);

            int color = generateColor( v.getId() );
            custom_avenger_view.setBackgroundColor(color);
        } else {
            v.clearAnimation();
        }
    }

    private int generateColor(int id) {
        int color;

        switch (id) {
            case R.id.button1:
                color = colors[0];
                break;
            case R.id.button2:
                color = colors[1];
                break;
            case R.id.button3:
                color = colors[2];
                break;
            case R.id.button4:
                color = colors[3];
                break;
            default:
                color = colors[4];
                break;
        }

        return color;
    }
```

MainActivity

Primero cambiaremos la instanciación de los botones (`bEnglishChange`, `bSpannishChange`):

```
    @ViewById(R.id.b_english_change)
    Button bEnglishChange;
    @ViewById(R.id.b_spanish_change)
```

```
Button bSpannishChange;
```

Y luego eliminaremos tanto su instanciación como su `setOnClickListener`.

A continuación a los métodos `changeToEnglish()` y `changeToSpanish()` deberemos añadir la repectiva anotación del evento `OnClick`:

```
@Click(R.id.b_english_change)
void changeToEnglish() {
    changeTextFromToolbar(english.getAppName(), english.getBSelectEn
    changeCustomView(Constants.EN);
}

@Click(R.id.b_spanish_change)
void changeToSpanish() {
    changeTextFromToolbar(spanish.getAppName(), spanish.getBSelectEn
    changeCustomView(Constants.ES);

}
```

Posteriormente haremos lo mismo con los `private ImageButton ib_next, ib_back;`. Dejaremos esto:

```
@ViewById
ImageButton ib_next, ib_back;
```

Acontinuación eliminaremos la instanciación y el listener del método `initAfterViews()`. Y los métodos `nextAvenger` y `backAvenger` los anotaremos con `@Click(R.id.ib_next)` y `@Click(R.id.ib_back)` respectivamente.

En cuanto al `private Toolbar toolbar;` borraremos el `private` y añadiremos el `@ViewById`:

```
@ViewById
Toolbar toolbar;
```

Y borraremos

```
toolbar = findViewById(R.id.toolbar);
```