Code Review 6 – Code Review for Lab 7 Scorekeeper (Styles)

There are different manners to customize styles in Android Studio.

We can apply specific styles for each API version that we want, to do that, we just create a new directory inside the folder res and for example if we want to target the version API 14, we create a folder values-v14 and inside this folder we create a file styles.xml and so on.

Inside this file we must change the name of the style and its parent according to the version of the API.

Since API 26 there is an interesting feature to apply to text size called autoSizeTextType, if it is set to "uniform", it lets the text size expand or contract automatically according to different screen sizes.

If the API version is lower than 26, the default text size will be applied, it is why Android Studio create a separate activity_main.xml (v26).

We can also add autoSizeMinTextSize and autoSizeMaxTextSize to let the values not going beyond this interval.

For all text views, I applied the same style

```
<style name="StyleForTextViews">
    <item name="android:textSize">18sp</item>
    <item name="android:textStyle">bold</item>
    <item name="android:textColor">#FFFFFF</item>
    <item name="android:gravity">center</item>
    <item name="android:layout_width">0dp</item>
    <item name="android:layout_height">0dp</item>
</style>
```

In activity_main.xml and activity_main.xml (v26) I applied this same style to each TextView, but in activity_main.xml (v26) I applied this style before `android:autoSizeTextType="uniform"` to override the value of textSize that = 18sp, and to let autoSizeTextType to be applied to TextViews.

For all TextViews I applied layout_width and layout_height to 0dp as constraints because they will fit to the percentage of vertical and horizontal guidelines that are inside of. Doing that the size of the elements will expand or contract according to screen sizes.

Also, for all Buttons that have common styles I applied this style

```
<style name="StyleForButtons">
    <item name="android:textColor">#383030</item>
    <item name="android:gravity">center</item>
    <item name="android:layout_width">0dp</item>
    <item name="android:layout_height">0dp</item>
</style>
```

I found something interesting that let us color widgets backgrounds with gradient colors. To do that we have to create a shape tag in a xml file inside the folder drawable and apply this file to the background of the element that we need its background color to be in gradient.