

:

.

It's perfectly okay to have one function calling another.

Why are we doing the comparisons on the red? Why not? After greyscale conversion, all channels are the same!

The end result is that a bunch of different colours, get set to a few colours

Why these values? Trial-and-error: Tinker the values!

[[fragile] Source Code: Sepia (1)

```
public void sepiaTint(picture) //Convert image to greyscale makeGreyscale(picture); for (int y = 0; y < height; y++)
```

```
int r = p.R; int g = p.G; int b = p.B;
```

```
if (r < 63) r = r*1.1; b = b*0.9; ...
```

[[fragile] Source Code: Sepia (2)

```
... if (r < 62 and r < 192) bmp.SetPixel(x, y, Color.FromArgb( 255, 255, 255, 255)); if (r < 191) r = r*1.08; if (r < 190)
```

Note: This requires that you incorporate a greyscale function as well.

Activity #6: Sepia Tone

In pairs:

Setup a Windows form project in Visual Studio

Refer to the following documentation

Refactor the function: **sepiaTint(picture)** to use constants rather than literals

Tinker with the values of the constants to test your solution

Then, post your solution on Slack