Seaborn Workshop

To remedy this, seaborn provides an interface to the husl system, which also makes it easy to select evenly spaced hues while keeping the apparent brightness and saturation much more uniform.

```
sns.palplot(sns.color_palette("husl", 8))
```

There is similarly a function called `husl_palette()` that provides a more flexible interface to this system.

Using categorical Color Brewer palettes

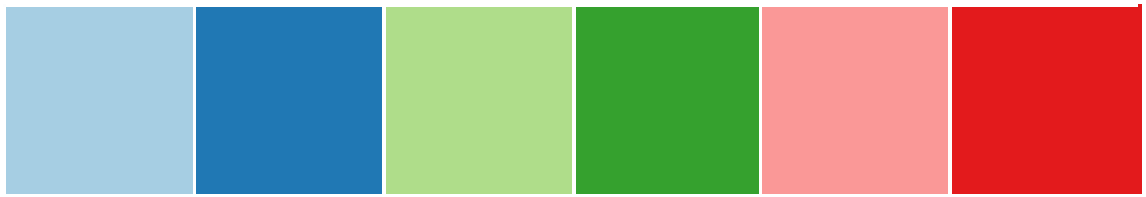
- Another source of visually pleasing categorical palettes comes from the Color Brewer tool (which also has sequential and diverging palettes, as well see below).
- These also exist as matplotlib colormaps, but they are not handled properly.
- In seaborn, when you ask for a qualitative Color Brewer palette, youll always get the discrete colors, but this means that at a certain point they will begin to cycle.

Seaborn Workshop

- A nice feature of the **Color Brewer** website is that it provides some guidance on which palettes are color blind safe.
- There are a variety of [kinds](http://en.wikipedia.org/wiki/Color_blindness) of color blindness, but the most common variant leads to difficulty distinguishing reds and greens.
- Its generally a good idea to avoid using red and green for plot elements that need to be discriminated based on color.

Seaborn Workshop

```
sns.palplot(sns.color_palette("Paired"))
```



```
sns.palplot(sns.color_palette("Set2", 10))
```

Seaborn Workshop

- To help you choose palettes from the Color Brewer library, there is the `choose_colorbrewer_palette()` function.
- This function, which must be used in an IPython notebook, will launch an interactive widget that lets you browse the various options and tweak their parameters.

Seaborn Workshop

Of course, you might just want to use a set of colors you particularly like together. Because `color_palette()` accepts a list of colors, this is easy to do.

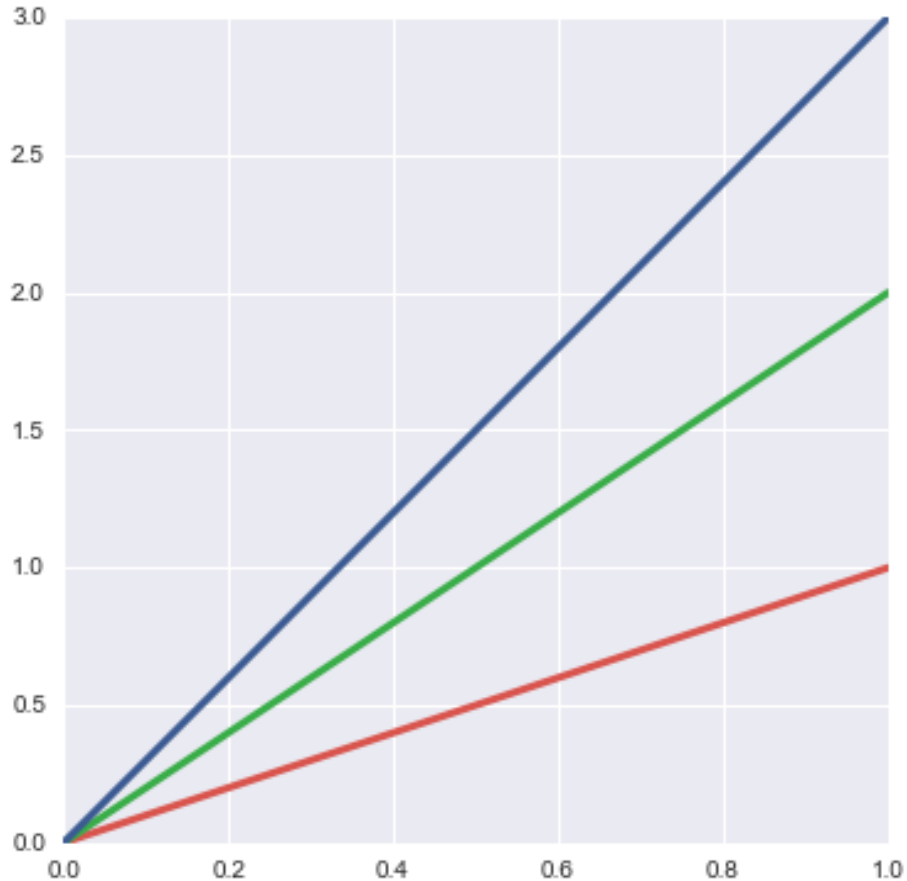
```
flatui = ["#9b59b6", "#3498db", "#95a5a6", "#e74c3c", "
sns.palplot(sns.color_palette(flatui))
```

xkcd

- A while back, xkcd ran a crowdsourced effort to name random RGB colors.
- This produced a set of 954 named colors, which you can now reference in seaborn using the `xkcd_rgb` dictionary:

Seaborn Workshop

```
plt.plot([0, 1], [0, 1], sns.xkcd_rgb["pale red"], lw=3
plt.plot([0, 1], [0, 2], sns.xkcd_rgb["medium green"],
plt.plot([0, 1], [0, 3], sns.xkcd_rgb["denim blue"], lw
```



Seaborn Workshop

If you want to spend some time picking colors, this interactive visualization may be useful. In addition to pulling out single colors from the `xkcd_rgb` dictionary, you can also pass a list of names to the `xkcd_palette()` function.

```
colors = ["windows blue", "amber", "greyish", "faded gr
sns.palplot(sns.xkcd_palette(colors))
```