Pibrella cheat sheet

Any program that uses the Pibrella needs to import the library like this:

```
import pibrella
```

Programs must be run as root, which means you need to start idle from a terminal window like this:

```
sudo idle
```

Or you can run a program you've written like this:

```
sudo python test_pibrella.py
```

Turning the 3 LEDs on and off

```
pibrella.light.red.on()             # turn on the red LED
pibrella.light.red.off()            # turn off the red LED
```

The light colours are called `red`, `yellow` and `green`.

Blinking and pulsing LEDs

```
pibrella.light.red.blink(3, 2)      # blink on for 3 secs, off for 2 secs
pibrella.light.red.pulse(2,4,2,1)   # fade in for 2 secs, on for 2 secs,
                                    # fade out 4 secs, off for 1 sec
```

Fading LEDs

```
pibrella.light.red.fade(0, 100, 2)  # From 0 to 100% in 2 seconds
```

The buzzer

```
pibrella.buzzer.buzz( frequency )   # play a frequency
pibrella.buzzer.fail()              # built in failure sound
pibrella.buzzer.success()           # built in success sound
```

Button

```
pibrella.button.read()              # read the button
```

Inputs

There are 4 inputs called a, b, c and d.

```
pibrella.input.a.read()             # read value on input a
```

Outputs

There are 4 high power outputs (that can drive a motor) called e, f, g and h.

```
pibrella.output.e.on()              # turn on output e
pibrella.output.e.off()             # turn off output e
```

Handling events

As an alternative to `polling` an input (checking the input in a loop) is to setup an event handler (like `events` in Scratch)

```
# the event handler
def button_pressed(pin):
    print("You pressed the button!")

# set the handler to trigger when the button is pressed
pibrella.button.pressed(button_pressed)

# set the handler to trigger when the button is released
pibrella.button.released(button_pressed)
```

You can also use this technique on the a, b, c and d inputs:

```
pibrella.input.a.pressed(button_pressed)
```