Wrapping up the Cruft

Making Wrappers to Hide Complexity

Jack Reichelt

ConnectEd Code

jack@connectedcode.org

Who am I, and what do I do?

Who am I

Teacher, CTO, and Professional DM

- I teach at Presbyterian Ladies College in Sydney!
    - I teach Cambridge Computer Science to grade 9.
- I am the CTO at ConnectEd Code!
    - Connecting code with context, community, and curriculum.
- I am also a professional Dungeon Master!
    - I run roleplaying games for people! Great for team building events!

Jack Reichelt

ConnectEd Code

jack@connectedcode.org

Let’s focus on ConnectEd Code

That’s the relevant one

- We do events like:
    - Data De*Tech*tives
    - Escape Artists
    - University of Wollongong CubeSats
- These are typically aimed at high school students.
- We aim for differentiation, supporting all experience levels.

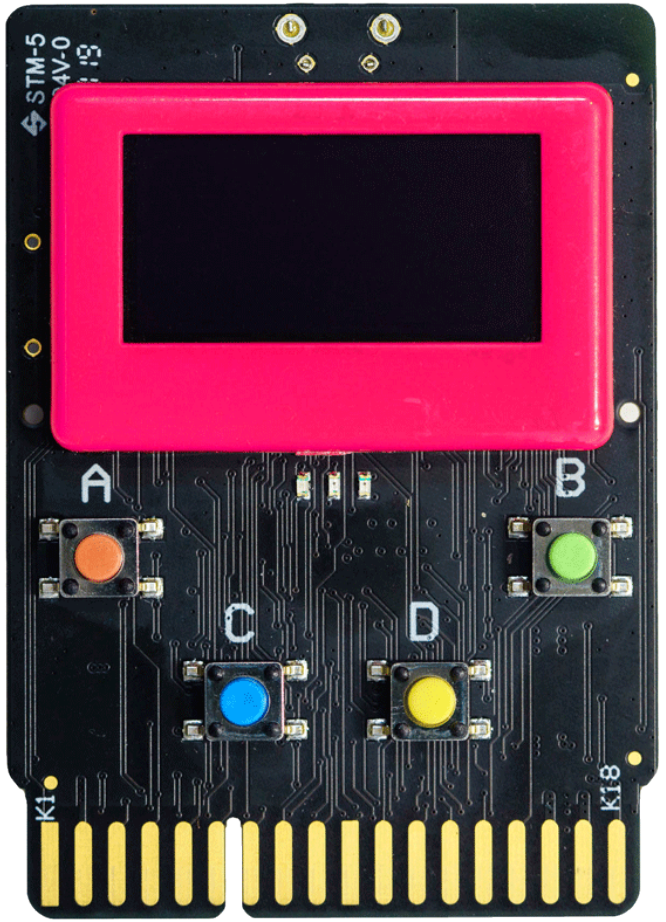
Jack Reichelt

jack@connectedcode.org

We use Kookaberries

They're great!

- Made by the AustSTEM foundation
- Run MicroPython
- Easy to use!
- Flexible!
- Fun!



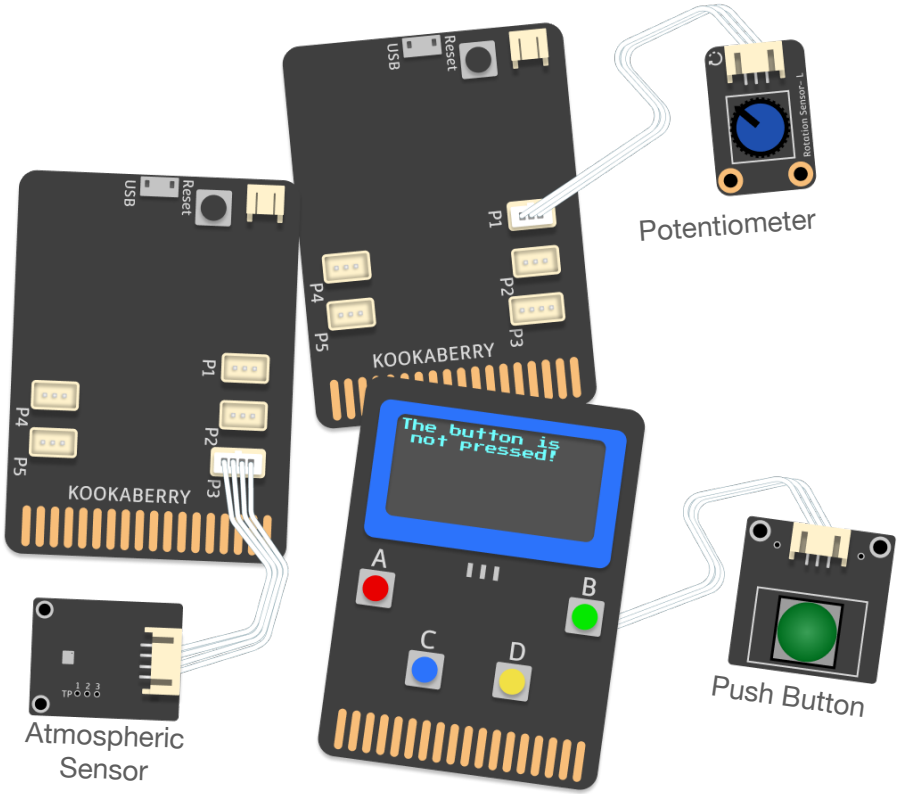
Jack Reichelt

ConnectEd Code

jack@connectedcode.org

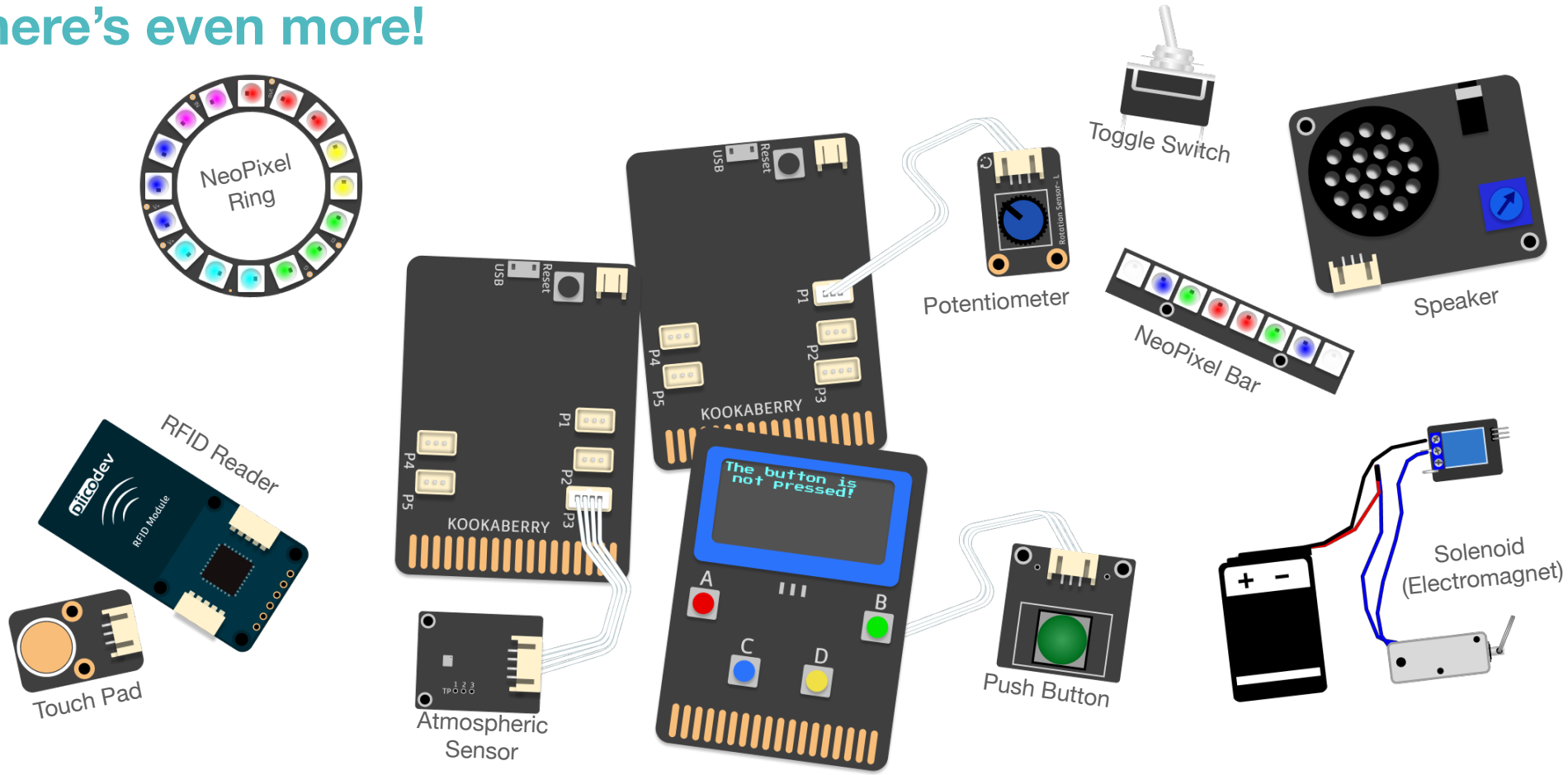
They can have peripherals!

Plug stuff in!



So many peripherals!

There's even more!



How do we do that?

We wrap up the cruft


What do I mean by cruft?

The stuff that doesn't matter

Let's look at Java for a minute

```
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

The stuff that doesn't matter

Let's look at Java for a minute

```
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

Doesn’t that look better?

Also, doesn’t it look like Python?

```
print  ("Hello, World!")
```

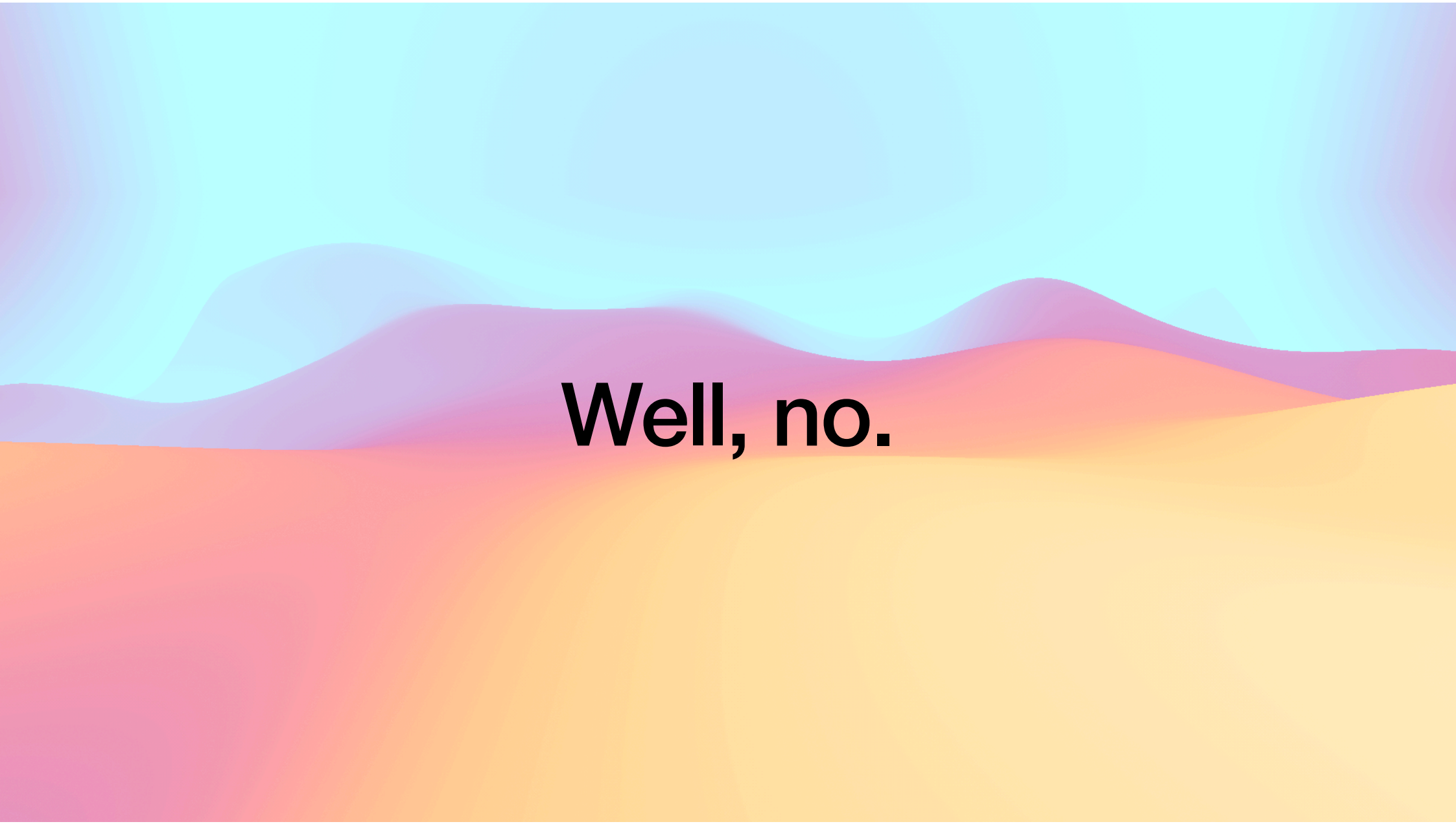
Jack Reichelt

ConnectEd Code

jack@connectedcode.org

Talk’s done!

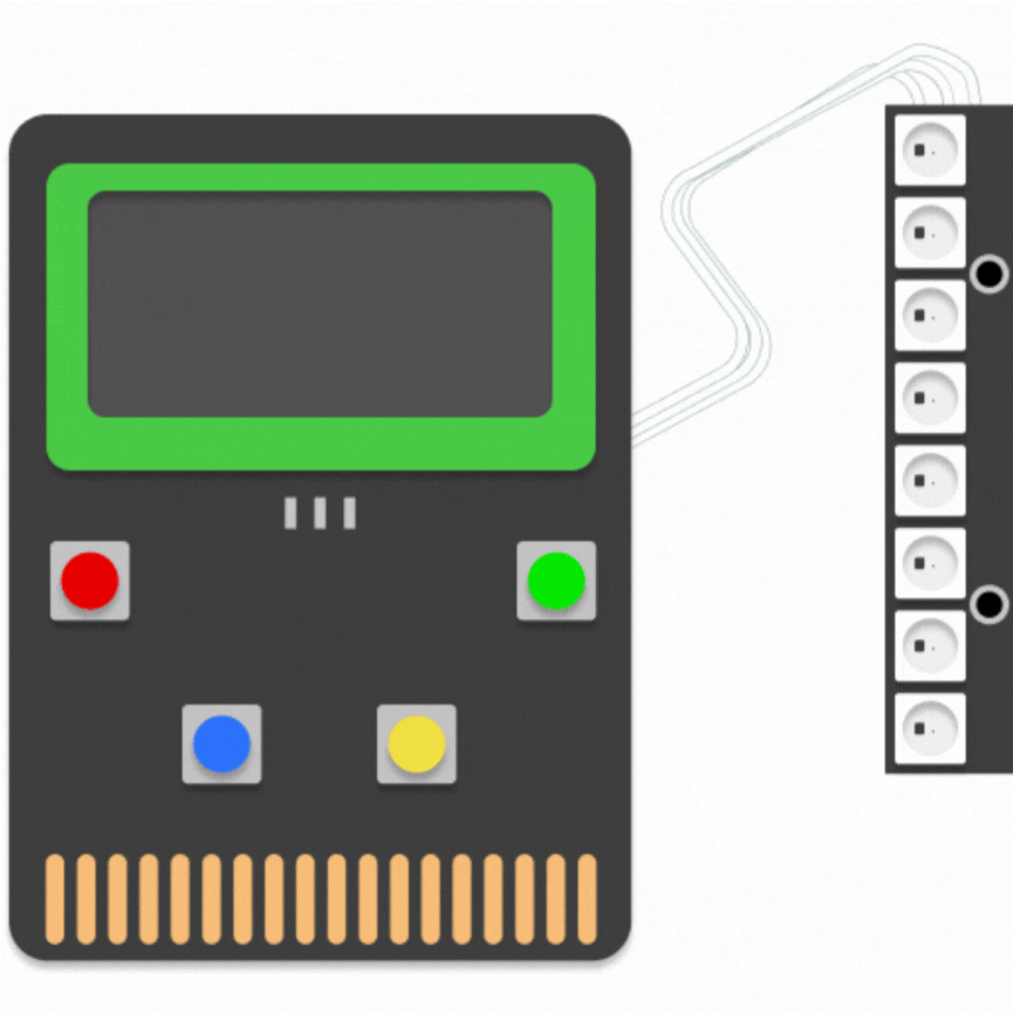
Just use Python, and you’re done… right?


Well, no.

Let's look bigger

And flash some lights!

- We want to make this!
- The first three NeoPixels light up red, green, and blue in turn.
- They stay on for another moment, then clear!
- Do this forever!



Jack Reichelt

ConnectEd Code

jack@connectedcode.org

What's the old code?

This is the normal way

```
from machine import Pin          # A whole bunch of imports
from neopixel import NeoPixel
from time import sleep

np = NeoPixel(Pin("P1"), 8)      # Mysterious code initialising the NeoPixel bar

while True:
    np[0] = (255, 0, 0)          # Set the first pixel to red
    np.write()                   # Display the change
    sleep(.5)                    # Wait for half a second
    np[1] = (0, 255, 0)          # Set the second pixel to green
    np.write()
    sleep(.5)
    np[2] = (0, 0, 255)          # Set the third pixel to blue
    np.write()
    sleep(.5)
    np.fill(0, 0, 0)             # Set all pixels to "black"/clear
    np.write()
    sleep(.5)
```

What's the new code?

All wrapped up!

```
from connectedcode import *          # Just one import

bar = connect_bar_light("P1")        # Says what it does

while True:
    bar.set_pixel(0, (255, 0, 0))    # No indexing, no write call.
    sleep(.5)                        # Waiting stays the same
    bar.set_pixel(1, (0, 255, 0))
    sleep(.5)
    bar.set_pixel(2, (0, 0, 255))
    sleep(.5)
    bar.clear()                      # Plain English
    sleep(.5)
```

What about a bigger example

I2C RFID Reader - 273 lines

Wrap that up!

I2C RFID Reader - 6 lines

```
from connectedcode import *              # The same import

reader = connect_rfid_reader()           # Connects with no arguments

while True:
    tag = reader.read_tag_name()         # Returns a string or None
    if tag:
        print(tag)
```


So how do I wrap things up?

1. What do they want to do?

Focus on outcomes

- Make a device do something
- Check a condition
- Update the state
- Get input or produce output

2. What bits make sense?

Things to build off

- Good function names
- Clear steps
- if statements
- Variables (eventually)

3. What gets in the way?

Try to limit these

- Bad names
- Extra steps
- “Unnecessary” arguments
- Protocols

4. Consistency is key

Keep it familiar

- Build on what they know
- Reinforce lessons
- Maintain patterns
  - I keep everything as `subject.verb(object)`


How have these wrappers gone?

How have these wrappers gone?

What results did we see?

- We've had four major events with these wrappers.
- Before 26% liked coding, 39% didn't (35% neutral).
- After **55%** liked coding, only **15%** didn't!
- Lots of positive feedback - we made it approachable.

Jack Reichelt

ConnectEd Code

jack@connectedcode.org


How can you do this too?

How can you do this too?

Bring this to your class!

- Check out the demo wrappers on Github!

  `ConnectEd-Code/pycon-wrappers`

- Make your own wrappers! Email me for advice!

- Email **info@connectedcode.org** to bring us to you!

- Check out Kookaberries at **learn.auststem.com.au**

Jack Reichelt

ConnectEd Code

jack@connectedcode.org