Lesson 3: The Big Red Button

Lesson aim:

Learn how to make buttons in processing - circles that do something when you click them.

Why:

This is a key topic, lots of programs in this term use buttons, as well as programs we use everyday, icons on phones etc. Look at the processing app as just one example.

Topics covered in this class:

- Collision detection
- Variables
- “if” statements
- Methods

Variables recap:

What they are:

A variable is a piece of information. In our case they are often numbers (we call them “`int`”s). You might also remember using `booleans` last term - do you remember what they were?

What they have:

```
int aNumber = 10;
```

- **a name:** eg “`aNumber`” you can call it almost anything you like - but no spaces!
- **a type:** eg `int`. Tell the computer what data it is: a number, boolean is true or false, etc
- **a value**: eg “10”. Depends on the type. For an `int`: a number, could be 1, -6, 1000, whatever you like! A `boolean` must be `true` or `false`.

Where do they go?

We normally “declare” them (that is, choose a type, give them a name, and give them a starting value) near the start of the program. We can use them almost wherever we like after that. If they are numbers, we can use them anywhere we would use a number.

Why use them?

There are lots of reasons to use variables. One reason is that the value of a variable can be updated, and therefore we can use variables to do cool stuff like get balls to move etc.

Example

```
int circleHosPos = 10;
void setup() { size(400,400);
   }
void draw() {
   ellipse(circleHosPos ,20,50,50);
```

```
    circleHosPos = circleHosPos +1;
}
```

Breakdown of steps:

1. Make the picture of the button (use a circle).
2. Remember from last term how to test if there is a collision with a circle and something else. (If you can't remember look below for an example)
    a. This time the "something else" is the mouse, so we are using `mouseX` and `mouseY` when we are working out the distance.
3. If the mouse collides with (is inside) the button, then we do... something. In this case, start with changing the background colour.

Collision detection:

```
// If mouse hits button:
// Check if the distance from the mouse's location to the
// middle of the circle is less than the radius of the circle.
if(dist(mouseX, mouseY, buttonCenter, buttonCenter)< buttonRadius){
   //do something
  }
```

Extra improvements:

- Make two buttons that do two different things (one makes the background red, the other green, perhaps).
- Add text to the button, saying what it does.
- Add a button to the dice program from lesson 1, and only roll when that button is pressed.
- Make a program that has a moving ball - that only moves when you press a button.
- Make a program that displays text onto the screen only when you press a button.
- Make a program that loads in a image as a background when you press a button. (If you can't remember how to load a image as a background - look at the `image()` function in the documentation).
    - Now make another button that loads in a new background.
    - And possibly a third button - use your imagination for this one.