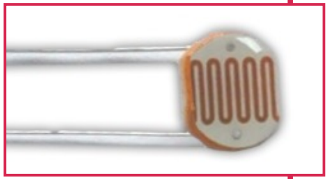
FUZE BASIC

Worksheet: 5a "Analog World"

Electricity huh! Without it our lives would be pretty dull - it is fair to say it is one of the most important, if not the most important discovery in our entire history. Without it there'd be no Apple, Sony or Samsung - no TV, heated blankets or food mixers. In fact, horrible monsters would roam the earth and we'd all have to eat more fruit and vegetables. Grim.. grim indeed. Three cheers for electricity!

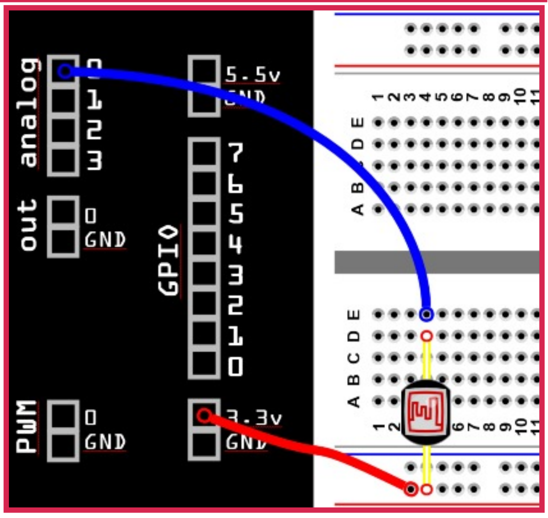
We're going to experiment with a very simple electronic component called a **L**ight **D**ependant **R**esistor or **LDR** for short. It controls the amount of electric current (the name given to flowing electricity) that goes through a circuit depending on how much light it is exposed to. We can measure how much using a very simple program. Sound like fun? Let's do it!

Grab an **LDR** (**L**ight **D**ependent **R**esistor - see pic) and wire up the breadboard as displayed . Note the **red** wire goes into the first hole along the **red** line and one end of the **LDR** goes next to it.


an LDR

Next attach a **blue** wire from analog pin zero to the next available hole where you plugged the **LDR** in.

Now enter the program listed below.



```
LOOP
CLS2
COLOUR=Raspberry
CIRCLE ( 500, 300, ANALOGREAD( 0 ) * 3 , 1 )
UPDATE
REPEAT
```

RUN the program. You should see a **raspberry red** circle. Wave your hand over the **LDR** and watch the circle change size.

The circuit passes an electric current (3.3 volts) to the **LDR** and, depending on how much light is received by the sensor more or less of this current is allowed through.

The remaining current then passes along the **blue** wire into **analog** pin 0 on the IO board.

The command **Analogread (0)** lets us read the value of this current and use it to set the radius of the circle. We have multiplied this by 3 to make it bigger.

The **CLS2** and **UPDATE** commands are required when we are using animated graphics.

For extra fun try this version;

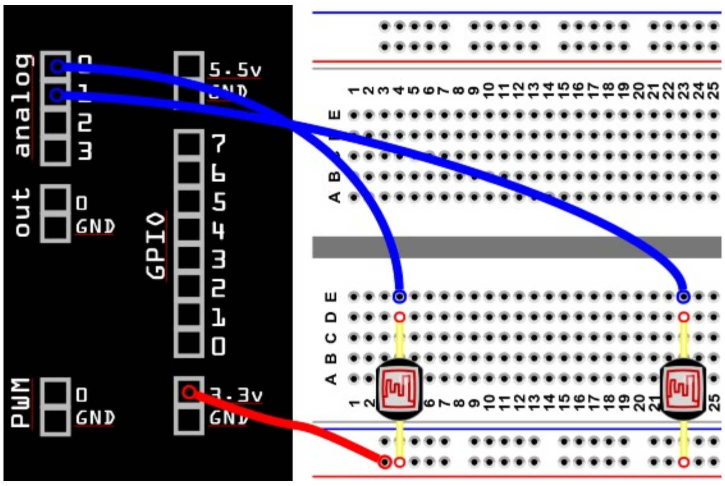
```
LOOP
COLOUR=RND (30)
CIRCLE ( MouseX, MouseY, ANALOGREAD( 0 ) * 3, 0 )
UPDATE
REPEAT
```

The **CIRCLE** command draws a circle at the location specified in pixels. 500 pixels across and 300 pixels up or in the second example we're using the mouse X & Y position. Then we use **ANALOGREAD (0) * 3** to define the radius of the circle (this how big the circle is from the middle to the outside) and the **1** at the very end tells it to draw a filled in circle as opposed to **0** which just draws an outline. Notice we also deleted the **CLS2** statement so it stops clearing the screen.

FUZE BASIC

Worksheet: 5b

“Bear with me..”



Add a second LDR as shown on the left. The second blue wire goes from the top of the LDR to analog pin 1.

It is best to have a bit of distance between the two LDRs so place them as far apart as you can. However, notice that the red and blue lines have a break in them so stay on the left of this.

Now we can read two LDRs individually.

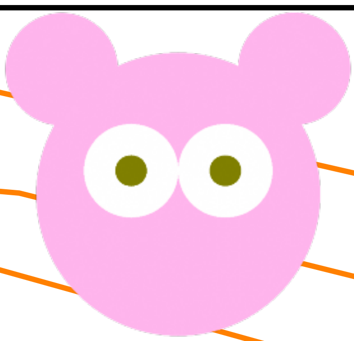
```
LOOP
CLS2
COLOUR=Raspberry
CIRCLE ( 300, 300, ANALOGREAD( 0 ) * 2, 1 )
COLOUR=Teal
CIRCLE ( ( 600, 300, ANALOGREAD( 1 ) * 2, 1 )
UPDATE
REPEAT
```

Open the Editor (F2 if in immediate mode). Delete any code you have in there so you have a blank page.

Enter the code as shown on the left then RUN with F3

As before the CIRCLE command is used to draw two circles with their radius defined by the values being read by ANALOG pins 0 and 1. Notice the horizontal (X) position has been changed so that both circles fit on the screen.

```
FULLSCREEN=1
LOOP
CLS2
X = GWIDTH / 2
Y = GHEIGHT / 2
SIZE = ANALOGREAD( 0 )
COLOUR=Brown
CIRCLE ( X, Y, SIZE * 3, 1 )
COLOUR=White
CIRCLE ( X - SIZE , Y + SIZE / 2, SIZE, 1 )
CIRCLE ( X + SIZE , Y + SIZE / 2, SIZE, 1 )
COLOUR = Olive
CIRCLE ( X - SIZE , Y + SIZE / 2, SIZE / 3, 1 )
CIRCLE ( X + SIZE , Y + SIZE / 2, SIZE / 3, 1 )
UPDATE
REPEAT
```



Start a fresh program and then enter the code as shown on the left then **RUN with F3.**

We start by setting full screen an starting a main LOOP.

GWIDTH and GHEIGHT tell us the width and height of the display in pixels. We store these values, divided by 2, in the X and Y. We divide them 2 to give us the exact centre of the screen us the pixel centre of the screen.

Then we set the SIZE variable to store value coming into ANALOGREAD (0).

Next we draw a sequence of circles in the centre of the screen and use the SIZE variable to position eyes to the left (X - SIZE) and right (X + SIZE) of the centre of the screen. We do the same for the height (Y + SIZE).

Even more circles are drawn to be the pupils.

ADVANCED CHALLENGE:
Now you know how to position a circle on screen and change its colour, could you add ears and a mouth?

You could try using the **ELLIPSE (x, y, width, height, fill)** command for the mouth - maybe one on top of another? How about making the pupils move by using a second LDR? (Ok that one is really tough but you can do it!)