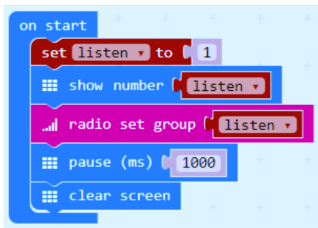


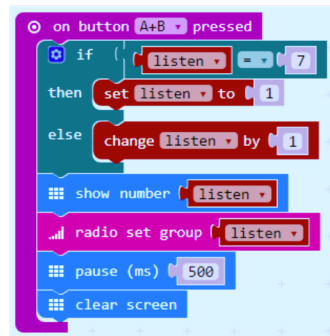
Picture Transmission

MakeCode

This is some fairly revolting code I wrote to demonstrate things like attenuation (loss of signal strength over distance) and noisy channels. There are better ways to do it, but it does the job.



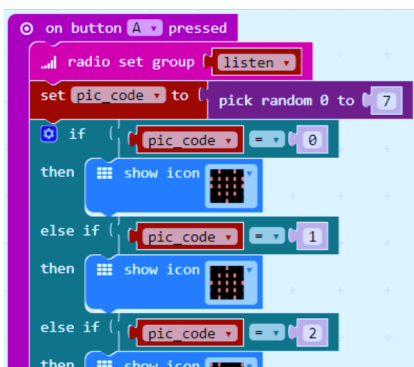
```
on start
  set listen to 1
  show number listen
  radio set group listen
  pause (ms) 1000
  clear screen
```



```
on button A+B pressed
  if listen = 7
  then
    set listen to 1
  else
    change listen by 1
  show number listen
  radio set group listen
  pause (ms) 500
  clear screen
```

The basic stuff (start/A+B) handles communications groups.

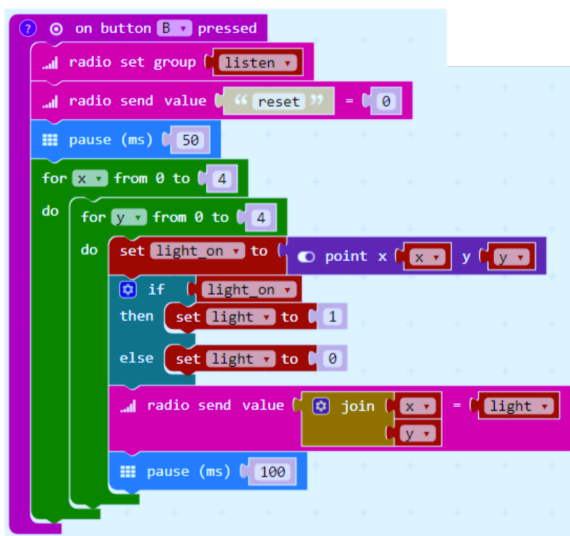
I start with all students on the same group just to drive home the idea that a single unfiltered channel is not good enough for wireless communication.



```
on button A pressed
  radio set group listen
  set pic_code to pick random 0 to 7
  if pic_code = 0
  then
    show icon [icon]
  else if pic_code = 1
  then
    show icon [icon]
  else if pic_code = 2
  then
    show icon [icon]
```

The A button does some fairly uninspiring stuff - chooses a random picture from a list of 7 and displays it on the screen.

This activity drives home the idea better if you just light up all the pixels, but it isn't as interesting for the students when outside testing out transmission ranges.



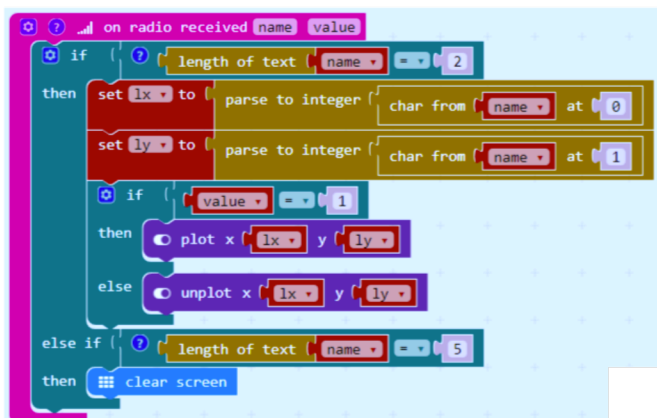
```
on button B pressed
  radio set group listen
  radio send value "reset" = 0
  pause (ms) 50
  for x from 0 to 4
  do
    for y from 0 to 4
    do
      set light on to point x x y y
      if light on
      then
        set light to 1
      else
        set light to 0
      radio send value join x = light y
      pause (ms) 100
```

The B button transmits the picture on the screen pixel by pixel with a pause in between each transmission.

The pause has two purposes: we need to give the receiving Micro:bit time to process without filling up the receive queue, and it gives the students something to watch.

It sends each pixel as a set of x/y coordinates, but would probably work just as well as a series of on/off commands. The picture on the other end would look a bit different with this method however.

A "reset" command is sent first to blank out the display.



```
on radio received name value
  if length of text name = 2
  then
    set lx to parse to integer char from name at 0
    set ly to parse to integer char from name at 1
    if value = 1
    then
      plot x lx y ly
    else
      unplot x lx y ly
  else if length of text name = 5
  then
    clear screen
```

This is the receive code, which checks what type of command has been sent (coords or reset).

If it's coordinates, it figures out what coordinate has been received and whether it's supposed to be on or off, and modifies the display accordingly.