Welcome to the Matrix

Welcome to the Dot Matrix

with Processing.js

Key URLs

This document http://bit.ly/29H7sNj

Original Processing (downloadable) for performances

https://processing.org/

Best Online Editor for Processing

http://sketchpad.cc/

Alternative Online Editor

http://js.do/blog/processing/editor/

```
void setup() {
  size(32, 16);
}
```

Establishes ‘canvas’ size

Corresponds with the number of pixels of the Matrix display

```
void setup() {
  size(32, 16);
}

void draw() {
   point(16,8);
}
```

Draws a single pixel

- Uses convention of X,Y
- X is number of pixels from the left
- Y is number of pixels from the top

```
void setup() {
  size(32, 16);
}

void draw() {
   line(0,0,32,16);
}
```

Draws a line

- Uses convention of x,y twice
- First the start point, x1, y1
- Then the end point, x2,y2
- Altogether, looks like…
    - line(x1,y1,x2,y2)

```
void setup() {
  size(32, 16);
}

void draw() {
   line(0,0,32,16);
   line(0,16,32,0);
}
```

You can draw multiple lines

- Just write multiple *line* commands
- Each one draws on top of the last

```
void setup() {
  size(32, 16);
  stroke(255,0,0);
  fill(255,0,0);
  background(0,0,0);
}

void draw() {
   line(0,0,32,16);
   line(0,16,32,0);
}
```

Let’s choose some colors

- background() sets the canvas color
- color() sets the foreground color
- fill() sets the color shapes are filled in
- Note: these are initialised in setup()

- Any color can be specified as a combination of Red Green and Blue
- In each case, the format (r,g,b) is used

```
void setup() {
  size(32, 16);
  stroke(255,0,0);
  fill(255,0,0);
  background(0,0,0);
}

void draw() {
   rect(1,1,31,15);
}
```

We can draw a filled shape

- Here, the coordinates are the left top corner, and the right bottom corner
- rect(left, top, right, bottom)

```
int barsLeft=1;
int barsRight=31;
int barsTop = 1;
int barsBottom = 7;

void setup() {
  size(32, 16);
  color(255,0,0);
  fill(255,0,0);
}

void draw() {
  rect(barsLeft, barsTop, barsRight,
barsBottom);
}
```

We can use ‘Variables’

Here, we specify the graph ‘bounds’

- left, top, right, bottom

Why use a variable?

We’ll need to refer to these same numbers a lot as the code gets more complicated

```
int barsLeft=1;
int barsRight=31;
int barsTop = 1;
int barsBottom = 7;

int maxBarWidth = barsRight - barsLeft;

int numBars = 8;
int barHeight = 1;
float[] barLengths = {
      0.5, 0.75, 1.0, 0.5,
      0.75, 1.0, 0.5, 0.75
};

void setup() {
  size(512, 256);
  stroke(255,0,0);
  fill(255,0,0);
  background(0,0,0);
}

void draw() {
  scale(16);
  int barIndex = 0;
  while(barIndex < numBars){
    int barY = barsTop + (barHeight * barIndex);
    rect(barsLeft,barY, (barLengths[barIndex] *
maxBarWidth), barHeight - 1);
    barIndex = barIndex + 1;
  }
}
```

Let's use an Array Variable

Arrays are lists of items.

- The barLengths array contains 'floating point' numbers
- The numbers contained can be any fraction from 0.0 to 1.0
- They describe the fraction of the available *maxBarWidth* which should be shown

Try changing the numbers to ...

...create a staircase shape.

...reduce or increase the number of stairs.

```
int barsLeft=1;
int barsRight=31;
int barsTop = 1;
int barsBottom = 7;

int maxBarWidth = barsRight - barsLeft;

int numBars = 8;
int barHeight = 1;
float[] barLengths = {
      0.5, 0.75, 1.0, 0.5,
      0.75, 1.0, 0.5, 0.75
};

void setup() {
  size(512, 256);
  stroke(255,0,0);
  fill(255,0,0);
  background(0,0,0);
}

void draw() {
  scale(16);
  int barIndex = 0;
  while(barIndex < numBars){
    int barY = barsTop + (barHeight * barIndex);
    rect(barsLeft,barY, (barLengths[barIndex] *
maxBarWidth), barHeight - 1);
    barIndex = barIndex + 1;
  }
}
```

Let's use an Array Variable

Arrays are lists of items.

- The barLengths array contains 'floating point' numbers
- The numbers contained can be any fraction from 0.0 to 1.0
- They describe the fraction of the available *maxBarWidth* which should be shown

Try changing the numbers to ...

...create a staircase shape.

...reduce or increase the number of stairs.

Final Graph Code

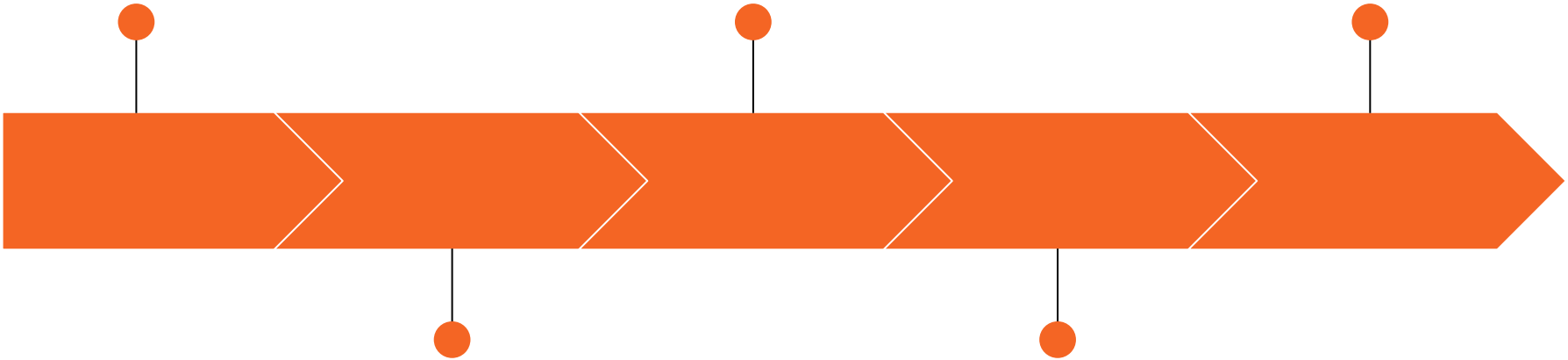
See http://bit.ly/29HbHsh

This code detects keypresses and counts them as ‘votes’ which are shown in a bar graph drawn on a 32x16 grid

Setting Canvas Size

Setting Colors

Handling ‘input events’



Drawing points, lines and filled shapes

Repeating with a loop

Next steps

Firing up three readers

Still a problem here, which no-one seems to be able to solve

Conceiving the different modes (as well as voting)

Clock, Lesson Countdown, Test Countdown, Year-specific scheduling

Testing and iterating

Once it’s installed, you learn loads more about what you should have done