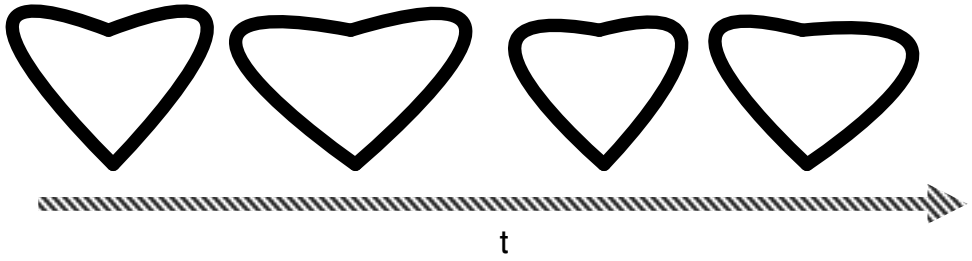
Assignment 1

1. Load pulse.csv in R and clean/convert it (produce a format that can easily be read and visualized by processing) (1p).

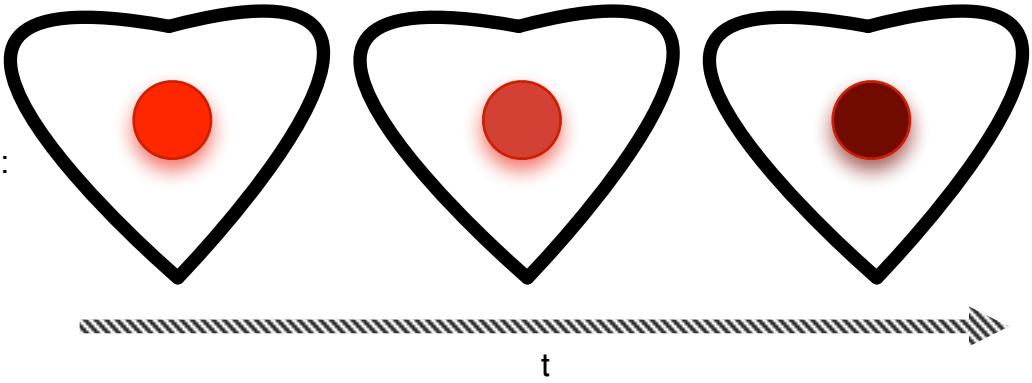
2. (In Processing) realize a visualization of a beating heart (make use of curves and animate the curves) (1p), and use this heart to visualize the pulse.csv data over time (2p).

E.g.:



3. (In Arduino) sketch a physical heart (use playmais and LEDs) (1p) and use this heart to visualize the pulse.csv data over time (i.e., send pulse.csv data over the serial connection using Processing, see tips on next page) (2p).

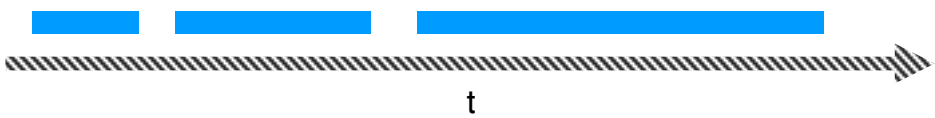
E.g.:



4. (In Arduino) use a potentiometer and an RGB LED. Use the potentiometer to continuously change the RGB LED's color (range of colors: rainbow please) (1p).

5. (In Arduino) use a potentiometer. With the potentiometer control the size of a bar (e.g. progress bar) on a webpage (use node.js to produce and connect the webpage with arduino (see tips on next page)) (2p).

E.g.:



It is a requirement that you document your results/designs by taking (whenever applicable) a short video and loading it up to Digicampus (to the folder of your group). When presenting your designs to the instructor please consider mappings, metaphors, and affordance in your descriptions.

Tips

Connect Arduino and Processing (task 3&5):

Take as a basis the SerialCallResponseASCII in Arduino->Examples->Communication. It includes the Processing part (at the end of the example as a comment). You need to copy the processing part to processing.

Connect Arduino and Web-site (task 5):

If you have node.js installed you may need to call in terminal the following to install a required package socket.io (same for serialport)

>npm install socket.io

---------- server.js ----------

```
var socketio = require('socket.io');
var serialport = require('serialport');
var SerialPort = serialport.SerialPort;
var portname = process.argv[2];

var shareData = "\n";

var myPort = new SerialPort(portname, {
 baudRate: 9600,
 options: false,
 parser: serialport.parsers.readline("\r\n")
});

myPort.on('open', function(){ console.log('port is open');});
myPort.on('close', function(){ console.log('port is closed'); });
myPort.on('error', function(){ console.log('some error - fix it'); });
myPort.on('data', function(data) {
        shareData = data + "\n";
        io.emit('news', { arduino: shareData });
        myPort.write('1'); // establish connection with arduino example
});

// server
var app = require('http').createServer(handler)
var io = require('socket.io')(app);
var fs = require('fs');

app.listen(3001);

function handler (req, res) {
  fs.readFile(__dirname + '/index.html',
  function (err, data) {
    if (err) {
      res.writeHead(500);
      return res.end('Error loading index.html');
    }
    res.writeHead(200);
    res.end(data);
  });
}

io.on('connection', function (socket) {
  socket.emit('news', { arduino: shareData });
  socket.on('my other event', function (data) {
    console.log(data);
  });
});
```

```
------------------------------------ index.html  ------------------------------------
<html>
<script src="/socket.io/socket.io.js"></script>
<script>
var shareData;
 var socket = io('127.0.0.1:3001');
 socket.on('news', function (data) {
        shareData = data;
        socket.emit('my other event', { my: 'reply' });
        document.getElementById('demo').innerHTML = shareData.arduino;
  });
</script>
<body>
        <div id='demo'>hallo arduino</div>
</body>
</html>
-----------------------------------------------------------------------------------------
```

start node.js in terminal (with arduino port) e.g.;
>node server.js /dev/tty.usbmodemfa131

visit   http://localhost:3001 to test connection