

INTRODUCTION

P5

PETRA ISENBERG

INFOVIS

YOUR CLASS PROJECT

LETS GET CODING

CODING ENVIRONMENT

p5.js

[Download](#) * [Start](#) * [Reference](#) * [Libraries](#) * [Learn](#) * [Community](#)

Hello! p5.js is a JavaScript library that starts with the original goal of [Processing](#), to make coding accessible for artists, designers, educators, and beginners, and reinterprets this for today's web.

Using the original metaphor of a software sketchbook, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas, you can think of your whole browser page as your sketch! For this, p5.js has add-on [libraries](#) that make it [easy to interact](#) with other HTML5 objects, including text, input, video, webcam, and sound.

p5.js is a new interpretation, not an emulation or port, and it is in active development. An official editing environment is coming soon, as well as many more features!

p5.js was created by [Lauren McCarthy](#) and is developed by a community of collaborators, with support from the [Processing Foundation](#) and [NYU ITP](#). © [Info](#).



Processing

[Cover](#)[Download](#)[Exhibition](#)[Reference](#)[Libraries](#)[Tools](#)[Environment](#)[Tutorials](#)[Examples](#)[Books](#)[Handbook](#)[Overview](#)[People](#)[Shop](#)[» Forum](#)[» GitHub](#)

Welcome to Processing 3! Dan explains the new features and changes; the links Dan mentions are on the [Vimeo page](#).

» [Download Processing](#)

» [Browse Tutorials](#)

» [Visit the Reference](#)

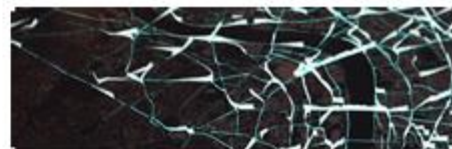
Processing is a flexible software sketchbook and a language for learning how to code within the context of the visual arts. Since 2001, Processing has promoted software literacy within the visual arts and

» [Exhibition](#)



[Fluid Leaves](#)

by Reinoud van Laar



[cf.city flows](#)

by Till Nagel and Christopher Pietsch



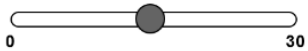
SOME EXAMPLES OF PAST STUDENT PROJECTS WITH THE DATA & P5

MEMBER HUNTER --- FIND THE NEXT COMMITTEE BOARD

by ranking past experience and publications

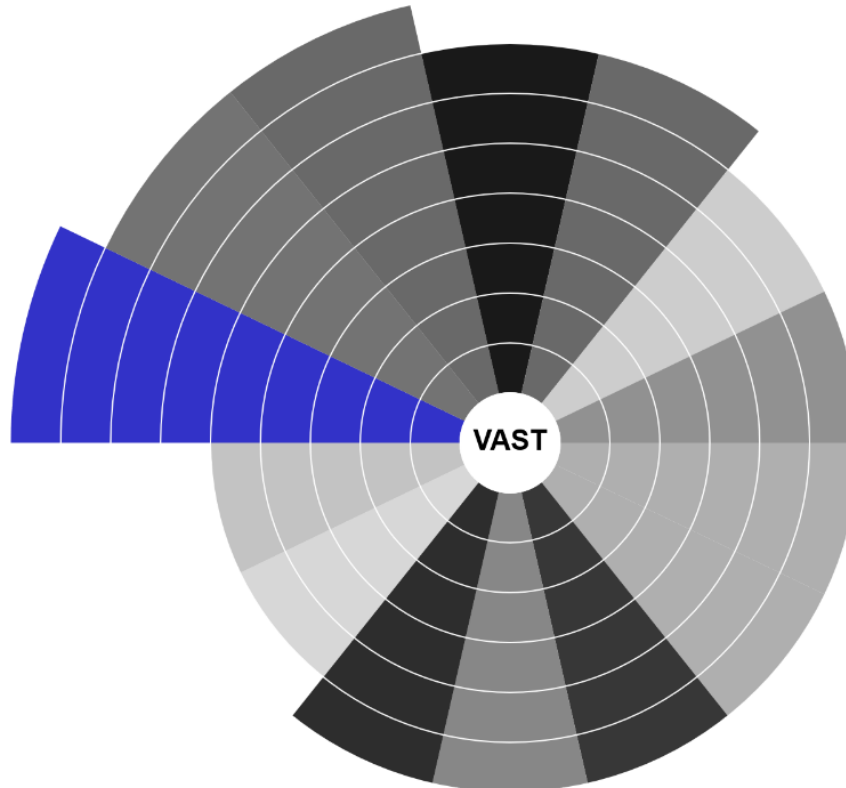
Vis	VAST	InfoVis	SciVis
-----	------	---------	--------

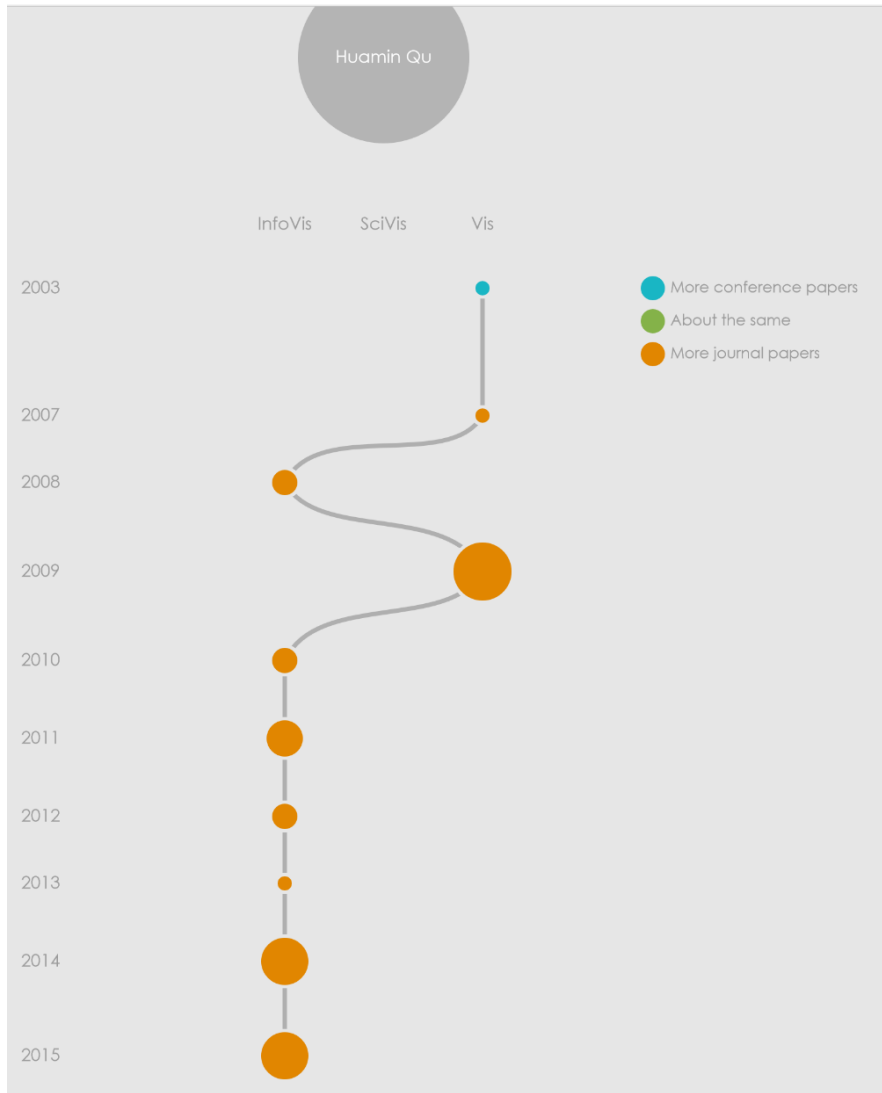
Recommended Authors for Next Committe



North, C. 9 papers

- Ribarsky, W.
- Huamin Qu
- Maciejewski, R.
- Koch, S.
- Xiaoyu Wang
- Silva, C.T.
- Shixia Liu
- Schreck, T.
- Ertl, T.
- Endert, A.
- Ebert, D.S.
- Wenwen Dou
- Perer, A.

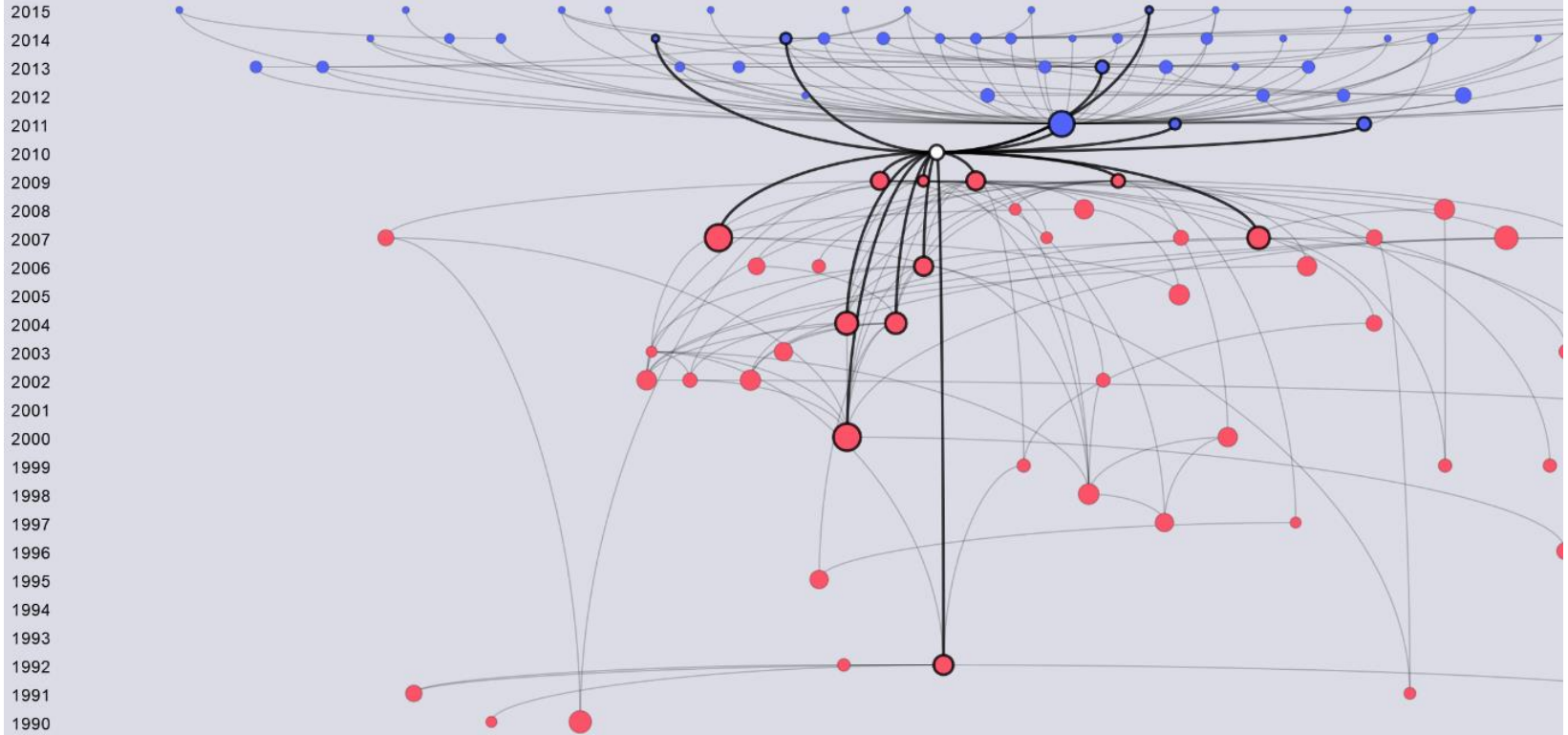




How can we support paper writing with citation information?

RefTree - A tool to help finding and exploring related articles

History



DOWNLOAD

Get your favorite
text editor

On windows, e.g.
Notepad++

p5*.js



The screenshot shows the p5.js website's download page. A red arrow points to the 'Reference' link in the left-hand navigation menu. The main content area is titled 'Download' and includes a 'Complete Library' section with a box for 'p5.js complete' which lists included files and the version (0.5.16). Below this is a 'Single Files' section with three boxes: 'p5.js' (single file, full uncompressed version), 'p5.min.js' (single file, compressed version), and 'CDN' (link to a statically hosted file).







Home Download
Download Complete Library
Start
Reference p5.js complete
Learn *
Includes:
p5.js, p5.dom.js, p5.sound.js, and an example project
Version 0.5.16 (October 11, 2017)
Examples
Books
Community
Forum
GitHub
Twitter

Single Files

p5.js Single file: Full uncompressed version	p5.min.js Single file: Compressed version	CDN Link: Statically hosted file
--	---	--

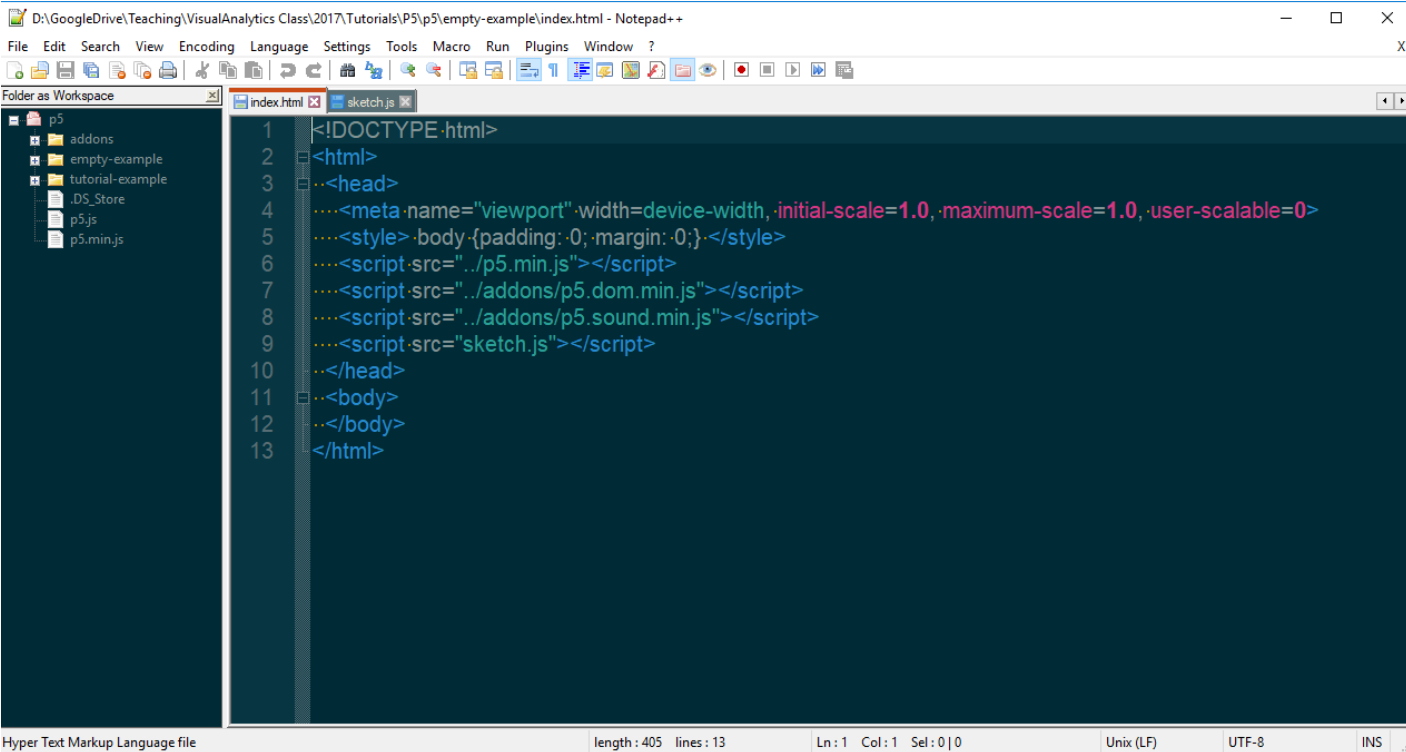
P5 COMPLETE

- Extract into a folder
- Copy the empty example
- Rename the empty example to something useful, e.g. “tutorial-example”

 addons	19/11/2017 22:21	File folder	
 empty-example	19/11/2017 22:21	File folder	
 tutorial-example	19/11/2017 22:22	File folder	
 .DS_Store	19/11/2017 22:21	DS_STORE File	7 KB
 p5.js	19/11/2017 22:21	JavaScript File	2.500 KB
 p5.min.js	19/11/2017 22:21	JavaScript File	1.159 KB

OPTIONAL - NOTEPAD++

- File -> Open folder as workspace

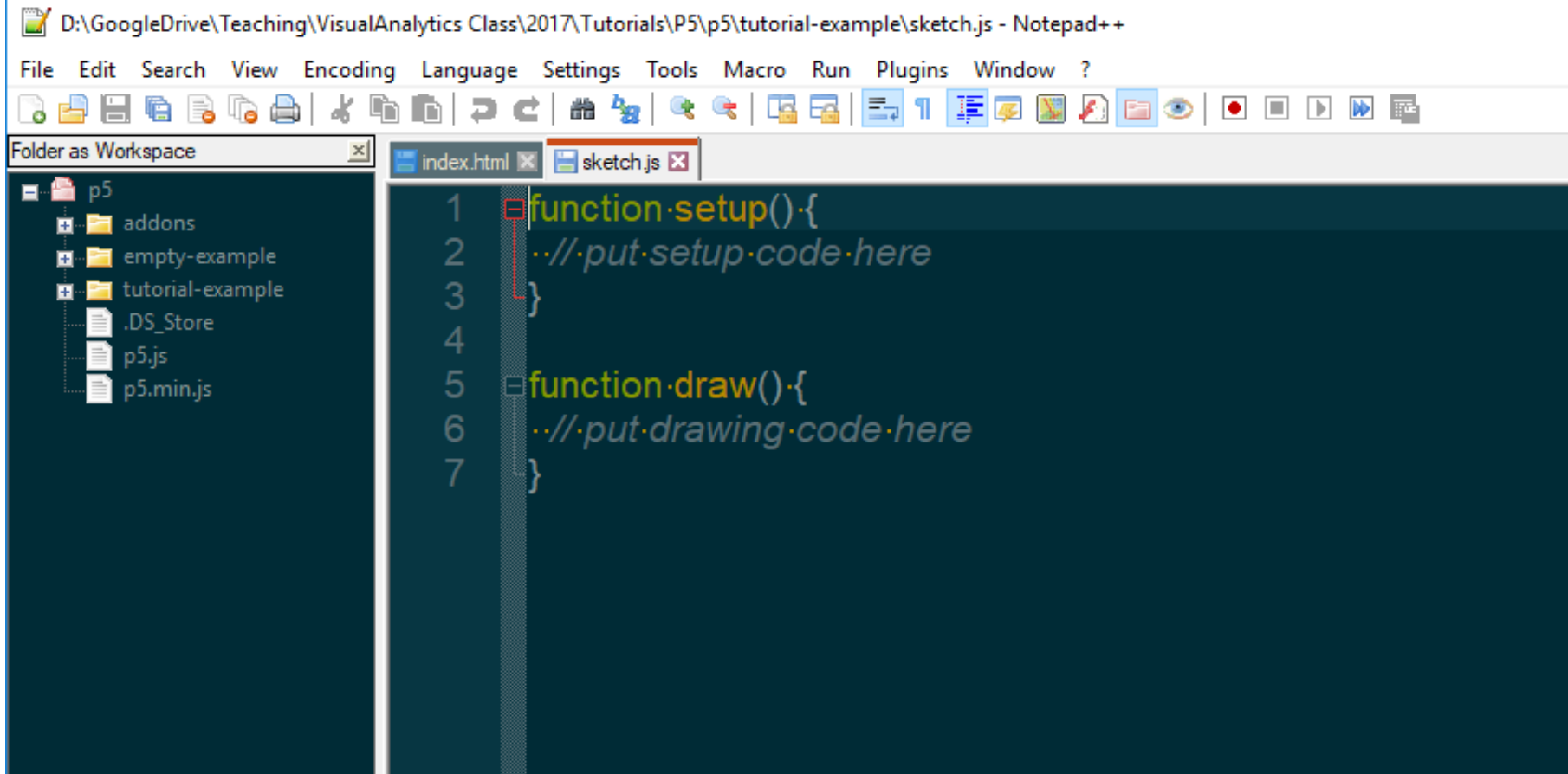


The screenshot shows the Notepad++ interface with a workspace. The left sidebar displays a folder tree for 'p5' containing subfolders 'addons', 'empty-example', and 'tutorial-example', along with files '.DS_Store', 'p5.js', and 'p5.min.js'. The main editor area shows the content of 'index.html' with the following code:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta name="viewport" width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0>
5 <style> body {padding:0; margin:0;} </style>
6 <script src="..p5.min.js"></script>
7 <script src="..addons/p5.dom.min.js"></script>
8 <script src="..addons/p5.sound.min.js"></script>
9 <script src="sketch.js"></script>
10 </head>
11 <body>
12 </body>
13 </html>
```

The status bar at the bottom indicates: Hyper Text Markup Language file, length: 405 lines: 13, Ln: 1 Col: 1 Sel: 0|0, Unix (LF), UTF-8, INS.

START







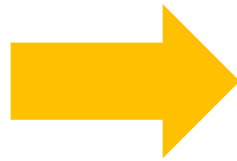
The image shows a Notepad++ window titled "D:\GoogleDrive\Teaching\VisualAnalytics Class\2017\Tutorials\P5\p5\tutorial-example\sketch.js - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations and editing. The left sidebar shows a "Folder as Workspace" view of a directory named "p5", containing subfolders "addons", "empty-example", and "tutorial-example", and files ".DS_Store", "p5.js", and "p5.min.js". The main editor area shows the "sketch.js" file with the following code:

```
1 function setup(){
2   ..//.put.setup.code.here
3 }
4
5 function draw(){
6   ..//.put.drawing.code.here
7 }
```






DATA & LIBRARIES FOLDERS

Name ^

-  libraries
-  desktop.ini
-  index.html
-  sketch.js



Name ^

-  data
-  libraries
-  desktop.ini
-  index.html
-  sketch.js

COPY DATA FILE

- Go to vispubdata.org -> download complete dataset->export as .csv.
- Put the csv into the data folder
- Copy p5-min.js into libraries folder

- Start webservice
- E.g. `python -m http.server`
(python 3)

CHANGE HTML FILE

- remove the `<script>` lines with `../addons`
- Add `libraries/` in instead of `../` in the line with `p5.min.js`
- Add a title if you want

```
<!DOCTYPE html>
<html lang="">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>p5.js example</title>
    <style> body {padding: 0; margin: 0;} </style>
    <script src="libraries/p5.min.js"></script>
    <script src="sketch.js"></script>
  </head>
```

```
1 var w = 1300;
2 var h = 1000;
3
4 function setup() {
5
6   createCanvas(w,h);
7   background(210);
8
9 }
10
11 function draw(){
12 }
13
14
15
```

Ctrl+Shift+R for reloading a refreshed js

```
1 var w = 1300;
2 var h = 1000;
3
4 function preload(){
5     //code here is executed before setup()
6     table = loadTable("data/IEEE VIS papers 1990-2018 - Main dataset.csv","csv","header");
7 }
8
9 function setup() {
10
11     createCanvas(w,h);
12     background(210);
13
14 }
15
16 function draw(){
17 }
18
```

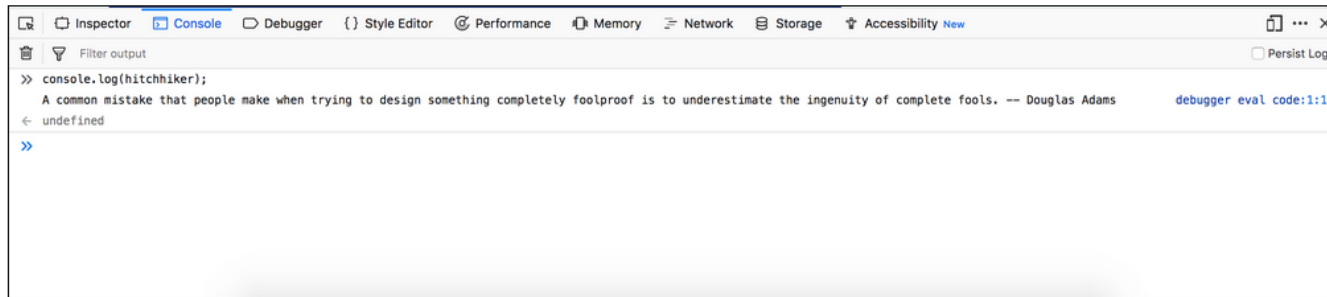
HOW CAN WE SEE THAT EVERYTHING IS OK?

Firefox

To open the Web Console:

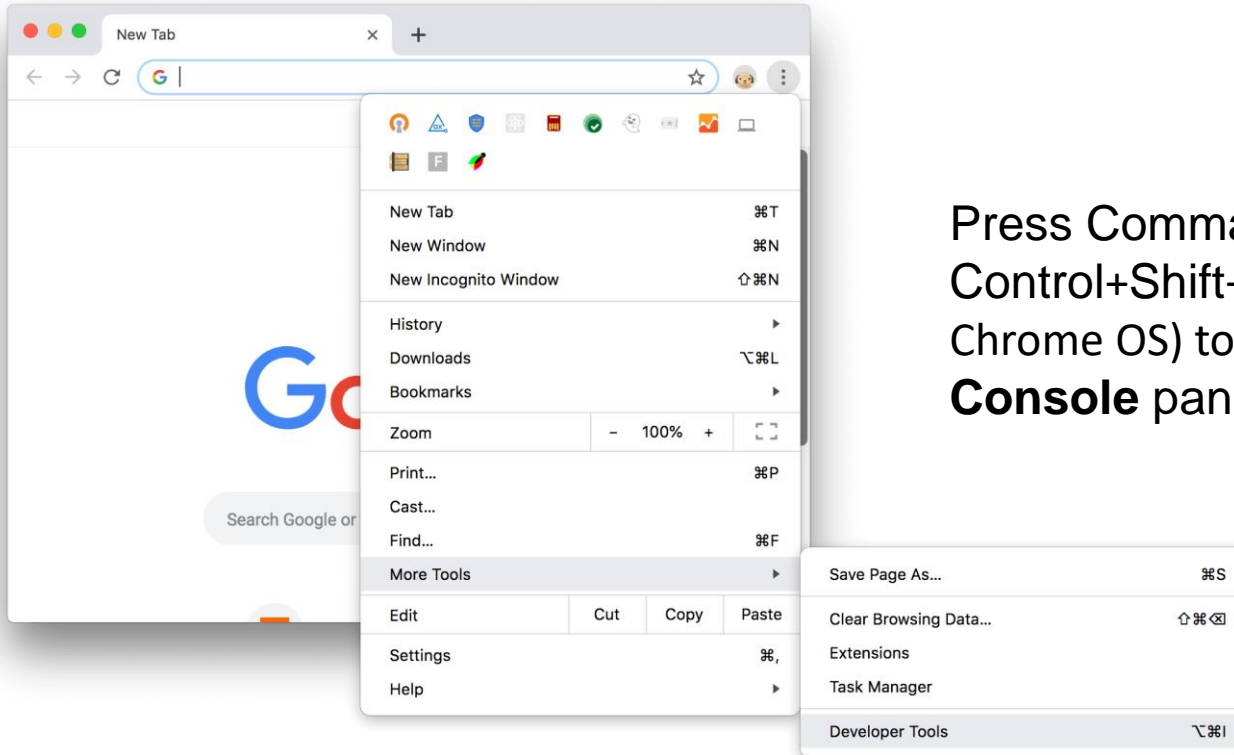
- either select "Web Console" from the Web Developer submenu in the Firefox Menu (or Tools menu if you display the menu bar or are on Mac OS X)
- or press the `Ctrl Shift K` (`Command Option K` on OS X) keyboard shortcut.

The [Toolbox](#) will appear at the bottom of the browser window, with the Web Console activated (it's just called "Console" in the [DevTools toolbar](#)):



HOW CAN WE SEE THAT EVERYTHING IS OK?

Chrome












Press Command+Option+J (Mac) or Control+Shift+J (Windows, Linux, Chrome OS) to jump straight into the **Console** panel.


CHECK THE DATA LOADED OK


Add in setup() & check output in console by reloading index.html on your browser

```
console.log(table.getRowCount() + " total rows in table");  
console.log(table.getColumnCount() + " total columns in table");
```

 Inspector  Console  Debugger  Style Editor  Performance  Memory  Network  Storage

  Filter output

 Use of the orientation sensor is deprecated.

 Use of the motion sensor is deprecated.

3101 total rows in table

18 total columns in table

 |

SUBSET THE DATA

For this tutorial we will only look at 2018 data

Note: for your project you should also only use papers of type “J” and “C”

-> I suggest to remove the “M” papers manually out of the dataset


```
11 function setup() {  
12  
13   createCanvas(w,h);  
14   background(210);  
15  
16   console.log(table.getRowCount() + " total rows in table");  
17   console.log(table.getColumnCount() + " total columns in table");  
18  
19   datarows = table.findRows("2018","Year");  
20   console.log("2018 paper count: "+datarows.length);  
21  
22 }  
23
```

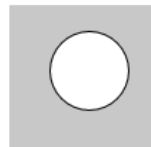
**NEXT: REPRESENT EACH 2018
PUBLICATION**

Reference

Search the API

ellipse()

Example



```
ellipse(56, 46, 55, 55);
```

edit

reset

copy

Description

Draws an ellipse (oval) to the screen. An ellipse with equal width and height is a circle. By default, the first two parameters set the location, and the third and fourth parameters set the shape's width and height. If no height is specified, the value of width is used for both the width and height. If a negative height or width is specified, the absolute value is taken. The origin may be changed with the `ellipseMode()` function.

Syntax

```
ellipse(x, y, w, [h])
```

```
ellipse(x, y, w, h, detail)
```

ellipseMode()

The default

Example



```
ellipseMode(RADIUS); // Set ellipseMode to RADIUS  
fill(255); // Set fill to white  
ellipse(50, 50, 30, 30); // Draw white ellipse using RADIUS  
  
ellipseMode(CENTER); // Set ellipseMode to CENTER  
fill(100); // Set fill to gray  
ellipse(50, 50, 30, 30); // Draw gray ellipse using CENTER
```



```
ellipseMode(CORNER); // Set ellipseMode is CORNER  
fill(255); // Set fill to white  
ellipse(25, 25, 50, 50); // Draw white ellipse using CORNER  
  
ellipseMode(CORNERS); // Set ellipseMode to CORNERS  
fill(100); // Set fill to gray
```

Add this function above the draw function

```
function drawPublication(x,y,size){  
  //this will draw the representation of one publication  
  stroke (255);  
  fill(210);  
  ellipse(x,y,size);  
}
```

```
function draw() {  
    // put drawing code here  
  
    var pubDrawingSize = 50;  
    var x = 0;  
    var y = pubDrawingSize * 0.5;  
  
    for (var i = 0; i < table.getRowCount(); i++)  
    {  
        x = x + pubDrawingSize;  
        if( x > w - pubDrawingSize){  
            //so we don't draw over the edge on the right side  
            x = pubDrawingSize; //we put the center of the next circle back to the left  
            //but we also need to move to the next line of circles  
            y = y + pubDrawingSize;  
        }  
  
        drawPublication(x,y, pubDrawingSize);  
    }  
}
```

Press reload on the website and check the result

EXERCISE

- Look at the reference manual for P5
- Search for the `text()` function
- Figure out a way to draw a number in the center of each circle that shows the paper's publication year (column "Year")

<https://p5js.org/reference/>

NAÏVE VERSION

```
function drawPublication(x,y,size,label){  
  //this will draw the representation of one publication  
  stroke (255);  
  fill(210);  
  ellipse(x,y,size);  
  text(label,x,y);  
}
```

In the draw function:

```
  y = y + pubDrawingSize;  
}  
  
drawPublication(x,y,pubDrawingSize,datarows[i].get("Year"););  
}
```



Looks bad – WHY?

```
function drawPublication(x,y,size,label){  
  //this will draw the representation of one publication  
  stroke (255);  
  fill(210);  
  ellipse(x,y,size);  
  fill(255);  
  noStroke();  
  text(label,x,y);  
}
```

Better but still unsatisfying
What next?

```
function drawPublication(label,x,y,size){
  //this will draw the representation of one publication in the dataset
  stroke(255);
  fill(210);
  ellipse(x,y,size);
  fill(255);
  noStroke();
  textAlign(CENTER,CENTER);
  text(label,x,y);
}
```

NICE!

...but not very meaningful since it's
only about 2018 papers anyways

**WHICH DATA COULD WE DISPLAY
INSTEAD IN THE CENTER?**

EXERCISE

- Look at your dataset
- Find the column that displays the conference
- Show the conference in the center of the circle
- Change the size of the circle to 90px

<https://p5js.org/reference/>

WHAT QUANTITATIVE DATA DO WE HAVE OR CAN WE DERIVE?

Now we need to do some data crunching


```
function setup() {  
  
  createCanvas(w,h);  
  background(210);  
  
  console.log(table.getRowCount() + " total rows in table");  
  console.log(table.getColumnCount() + " total columns in table");  
  
  datarows = table.findRows("2018","Year");  
  console.log("2018 paper count: "+datarows.length);  
  
  calculateAuthorCounts();  
  
}
```

```
function calculateAuthorCounts()
{
  authorCount = [];
  for(var i = 0; i < datarows.length;i++)
  {
    authorCount.push(split(datarows[i].get("AuthorNames-Deduped"),";").length);
  }
  minAuthorCount = min(authorCount);
  maxAuthorCount = max(authorCount);
}
```

Here we calculate once how many authors a paper has
We store the counts in an array called authorCount

Now preparing for drawing out first data glyph

```
function drawLineAtAngle(startx, starty, length, angleInDegrees)
```

```
//this function allows to draw a line of a specific length from a start point
```

```
//outwards at an angle
```

```
angleMode(DEGREES);
```

```
//some basic math skills from highschool
```

```
endx = startx + length*cos(angleInDegrees);
```

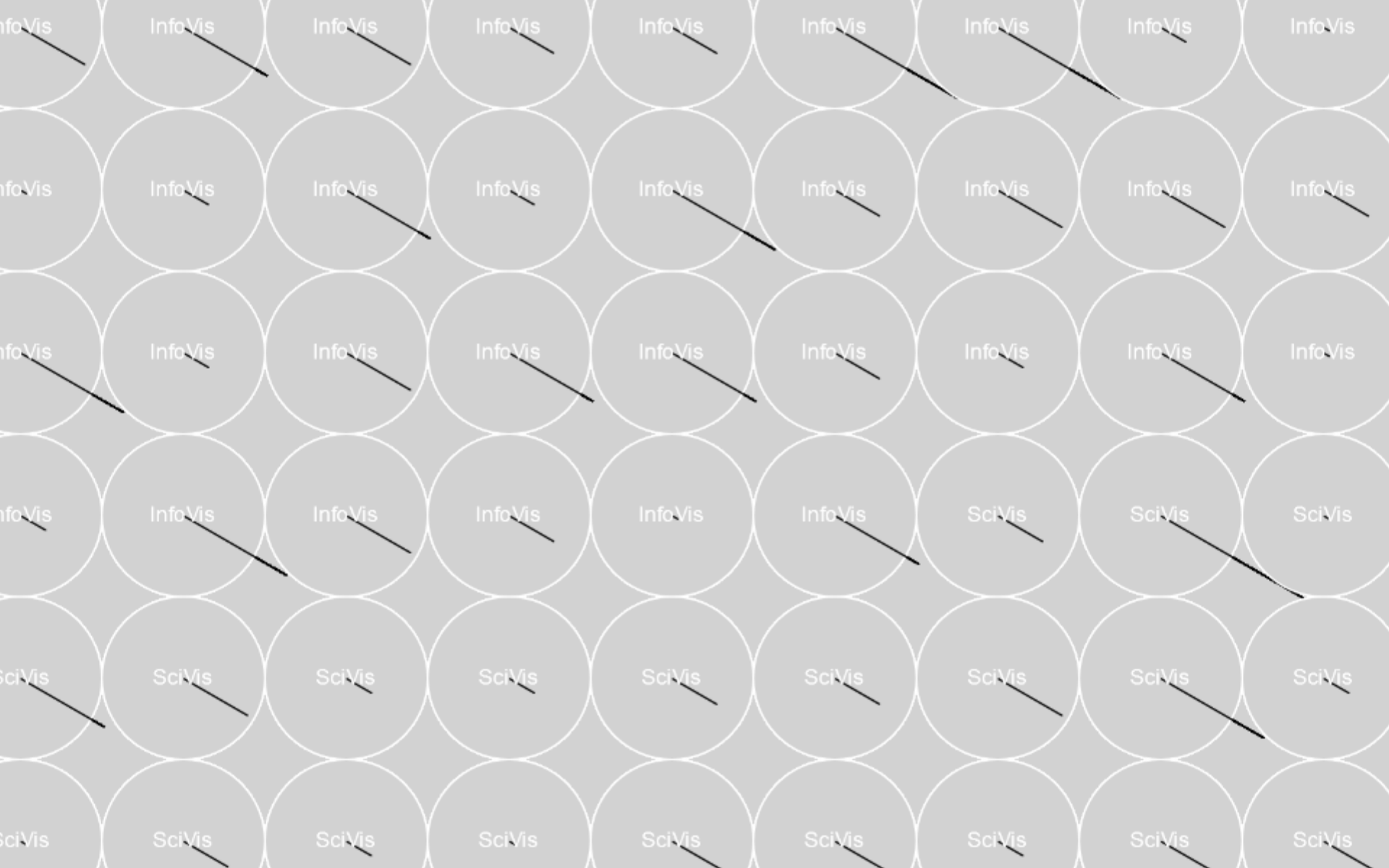
```
endy = starty + length*sin(angleInDegrees);
```

```
line(startx,starty,endx,endy);
```

```
function drawPublication(x,y,size,label,pubIndex){
  //this will draw the representation of one publication
  stroke(255);
  fill(210);
  ellipse(x,y,size);

  //draw the lines representing data
  stroke(0);
  //we draw a line to represent the number of authors
  drawingAngle = 30;
  lineLength = map(authorCount[pubIndex],minAuthorCount,maxAuthorCount,3,size); //we want the line at minimum 3 pixels long
  drawLineAtAngle(x,y,lineLength,drawingAngle);

  fill(255);
  noStroke();
  textAlign(CENTER,CENTER);
  text(label,x,y);
}
```



InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

InfoVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

SciVis

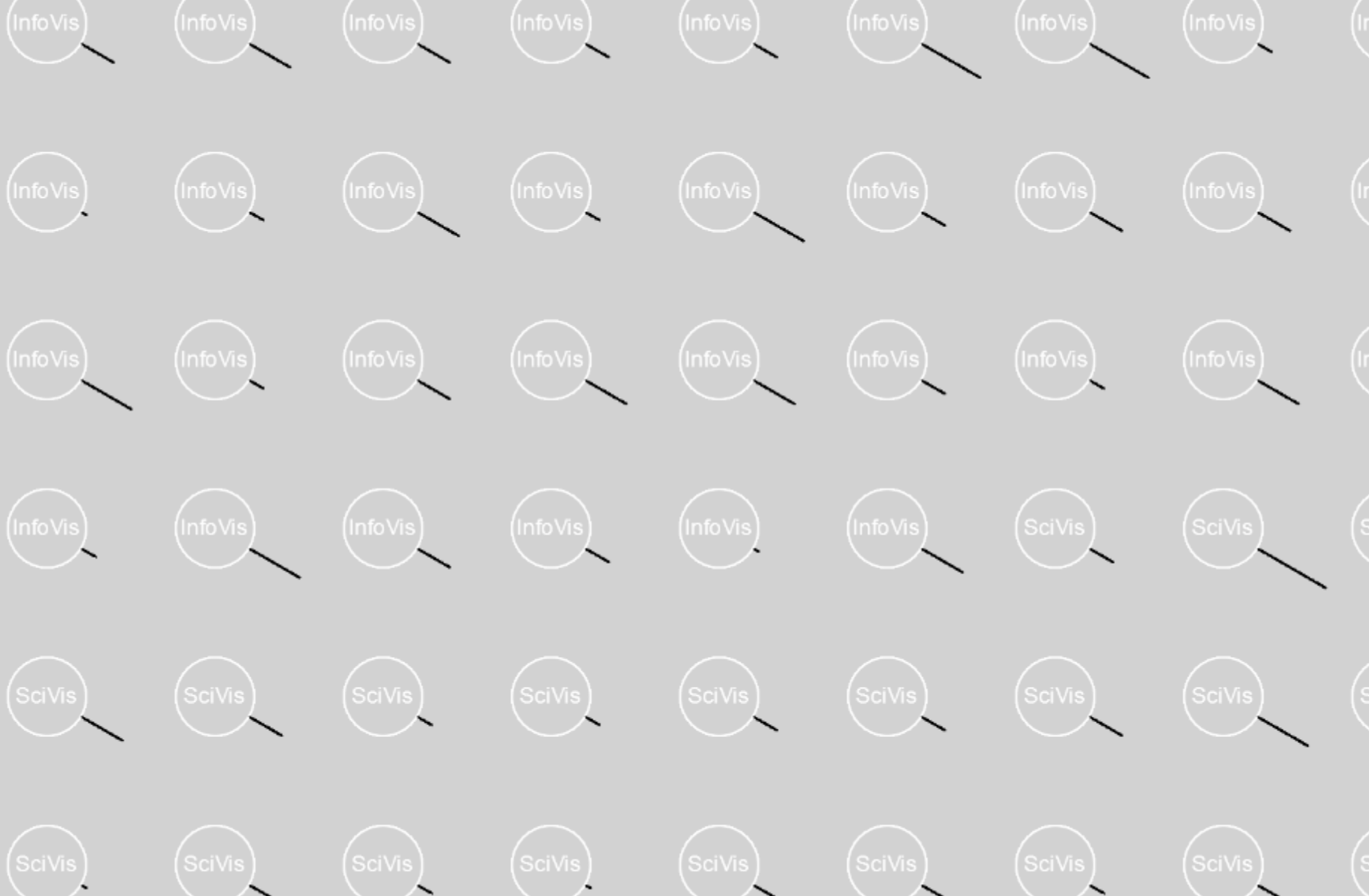
SciVis

SciVis

SciVis

SciVis

SciVis



InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis

InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis

InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis

InfoVis InfoVis InfoVis InfoVis InfoVis InfoVis SciVis SciVis

SciVis SciVis SciVis SciVis SciVis SciVis SciVis SciVis

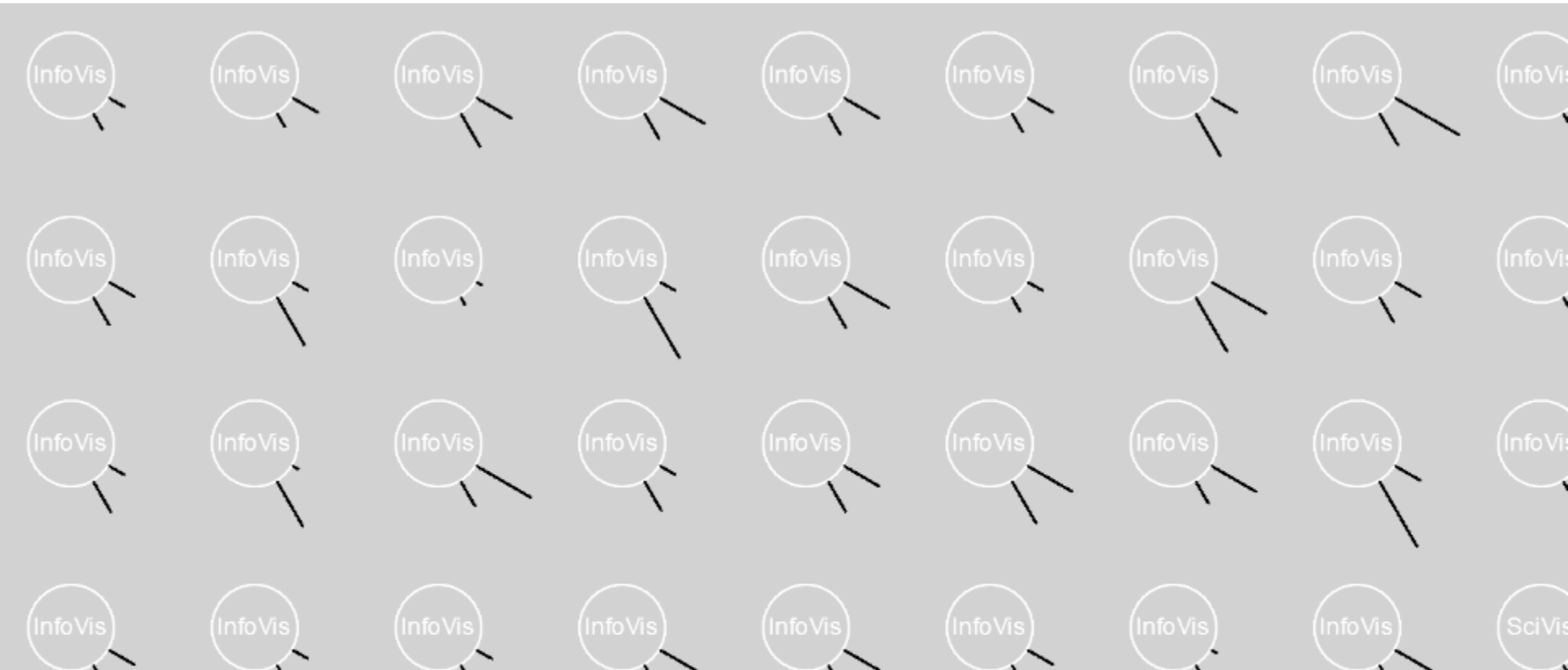
SciVis SciVis SciVis SciVis SciVis SciVis SciVis SciVis

Try to achieve this:

```
function drawPublication(x,y,size,label,pubIndex){  
    size = max(textWidth("InfoVis"),textWidth("VAST"),textWidth("SciVis"))+ 5;  
  
    var linestartx = x + size * 0.5 * cos(drawingAngle);  
    var linestarty = y + size * 0.5 * sin(drawingAngle);  
  
    drawLineAtAngle(linestartx,linestarty,lineLength,drawingAngle);  
}
```

YOU:

- Add a function “calculateTitleLength”
- Add a line to each glyph that shows the length of the title of each paper




```
function calculateTitleLength(){  
  
    titleLength = [];  
  
    for (var i = 0; i < datarows.length; i++)  
    {  
        titleLength.push(datarows[i].get("Title").length);  
    }  
  
    minTitleLength = min(titleLength);  
    maxTitleLength = max(titleLength);  
}
```

In drawPublication:

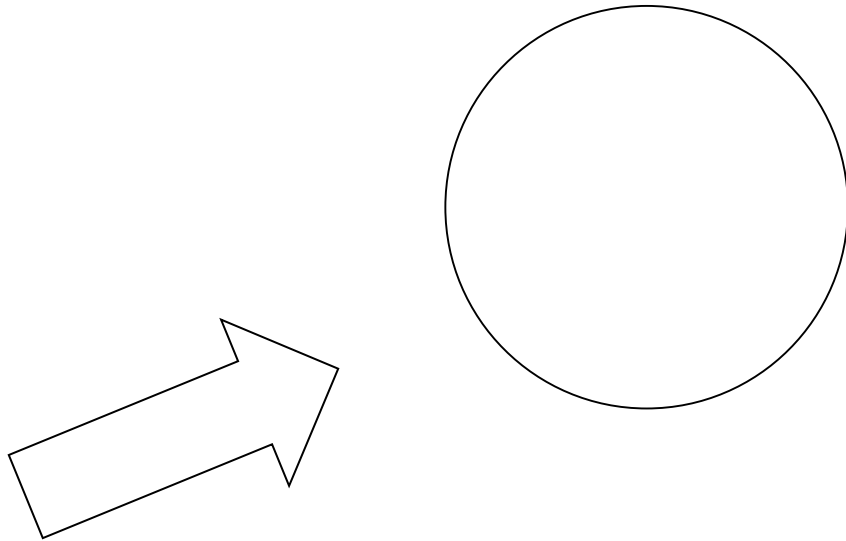
```
//draw line to represent the length of the title of each paper  
drawingAngle = 60;  
lineLength = map(titleLength[pubIndex],minTitleLength,maxTitleLength,3,size); //we want the line to a  
var linestartx = x + size * 0.5 * cos(drawingAngle);  
var linestarty = y + size * 0.5 * sin(drawingAngle);  
drawLineAtAngle(linestartx,linestarty,lineLength,drawingAngle);
```

HOW CAN WE GET ACTUAL DETAILS?

Lets add some interaction...

First we need to test if a mouse is over a publication

How can we do that?



```
function testMouseOver(x, y, circleRadius){  
    //we test if the mouse is within the radius of the circle  
    if(dist(mouseX,mouseY,x,y) < circleRadius){  
        return true;  
    }  
    return false;  
}
```

In drawPublication:

```
function drawPublication(x,y,size,label,pubIndex){  
  size = max(textWidth("InfoVis"),textWidth("VAST"),textWidth("SciVis))+ 5;  
  
  var mouseOver = testMouseOver(x,y,size * 0.5);  
  
  //this will draw the representation of one publication  
  stroke (255);  
  
  if(mouseOver){  
    fill(110);  
  }  
  else{  
    fill(210);  
  }  
  
  ellipse(x,y,size);
```

YOU:

To practice more:

- Show a text label on mouse-over that shows the title of the publication
- Color the circle based on the conference
- Add additional lines that represent data