attendance code:
4025

Intro to Processing + Agile Techniques

Workshop 2

Ruzanna Chitchyan, Jon Bird, Pete Bennett
TAs: Mitch Lui, Craig Barnfield, Kira Clements, Ollie Myers

Week	Date	**Lecture** Monday 11:00-12:00 *PHYS BLDG G44 FRANK*	**Workshop** Monday 13:00-15:00 *MVB 2.11 PC*	**Groupwork**
1	23/01/22	Introduction and Process [slides] [materials]	Teams, Waterfall Method and Project Brief [slides] [case study] [project brief]	Research games, create list on team repo. Install Processing
2	30/01/22	Agile Software Development [slides]	Intro to Processing, Agile Techniques	Decide on two game ideas
3	06/02/22	Requirements Engineering	Paper Prototyping, Requirements, Ideas Clinic	Collect requirements. Decide on final idea
4	13/02/22	Object Orientated Design	Classes Activity	Add requirements section to report
5	20/02/22	Implementation	Case Study, Spring Prep, Continuous Integration	Develop a working prototype over reading week!
6	28/02/22	READING WEEK	GAMES JAM	
7	06/03/22	Project Management	IN CLASS TEST (assessing lectures 1-4)	Define team roles
8	13/03/22	HCI - Qualitative	HCI Qualitative Task	Add qualitative assessment (of your choice) to report
9	20/03/22	HCI - Quantitative	HCI Quantitative Task	Add quantitative assessment (of your choice) to report
	27/03/22	EASTER week 1	SPRINT 1	
	03/04/22	EASTER week 2	SPRINT 2	
	10/04/22	EASTER week 3	SPRINT 3	
10	17/04/22	Software Engineering Extended	IN CLASS TEST (assessing lectures 5-9)	Develop Game
11	24/04/22	Coursework Feedback		Finish Report
12	01/05/22	Bank Holiday Monday (no class)	Demo Day Weds/Thurs (tbc)	Submit Report

Coursework

- Any questions about the coursework brief?
- Have you made contact with everyone in your team? Let us know now if not!
- Have you made a commit on your repo using Mitch’s git introduction from Software Tools?
    - Team Photo
    - List of games (inspiration and ideas)

Today’s Workshop

- Introduction to Processing (15mins)
- Develop an app (~ 80 mins)
  - Practice Pair Programming
  - Practice Kanban board use
- Try some (very light) evaluation (15mins)

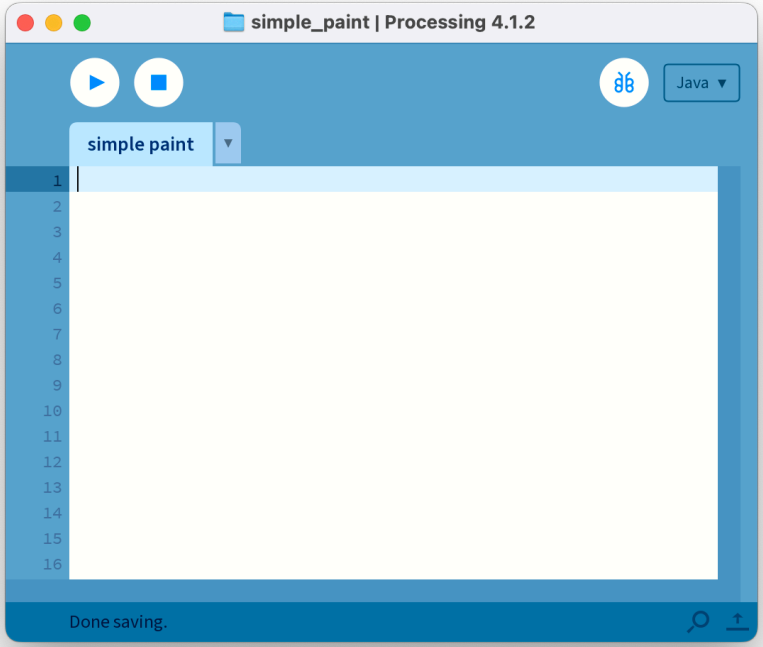
Simple Paint

- Today you are going to develop in pairs a **simple drawing tool** in Processing.
- Inspired by the classic Microsoft Paint, but… this is only a starting point, you will be adding any features you like.
- You will be swapping your paint app with another pair at the end to draw a **portrait**
- *…keep this use in mind before going off and making an abstract generative geometric paint program!*

image

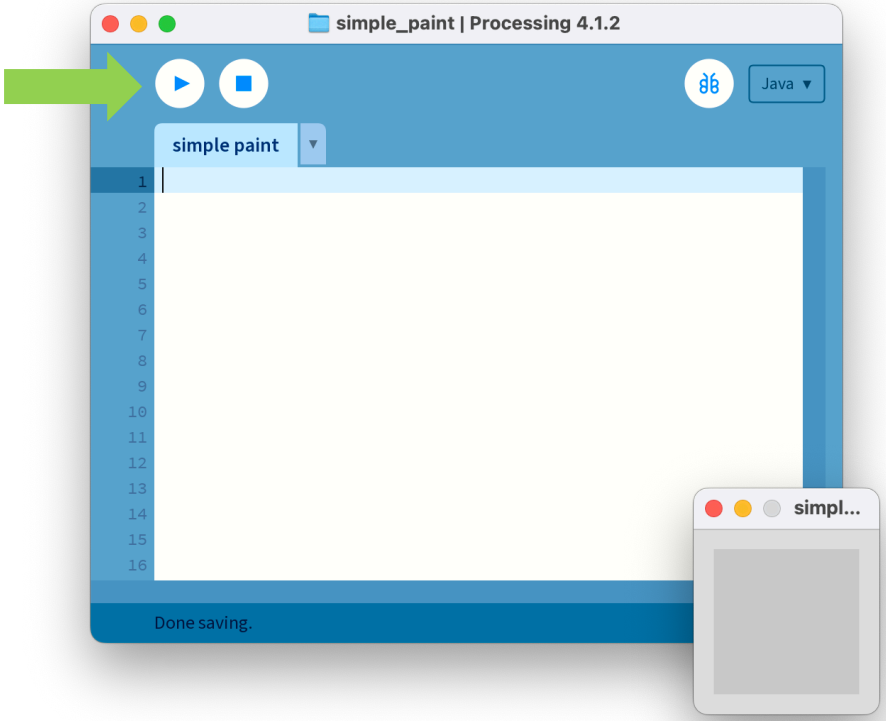
Live Demo

- Follow along with this quick demo.
- This will start you all off with a very bare bones Minimum Viable Product (MVP)
- If you miss anything, the steps I'll show you now are in the following slides.



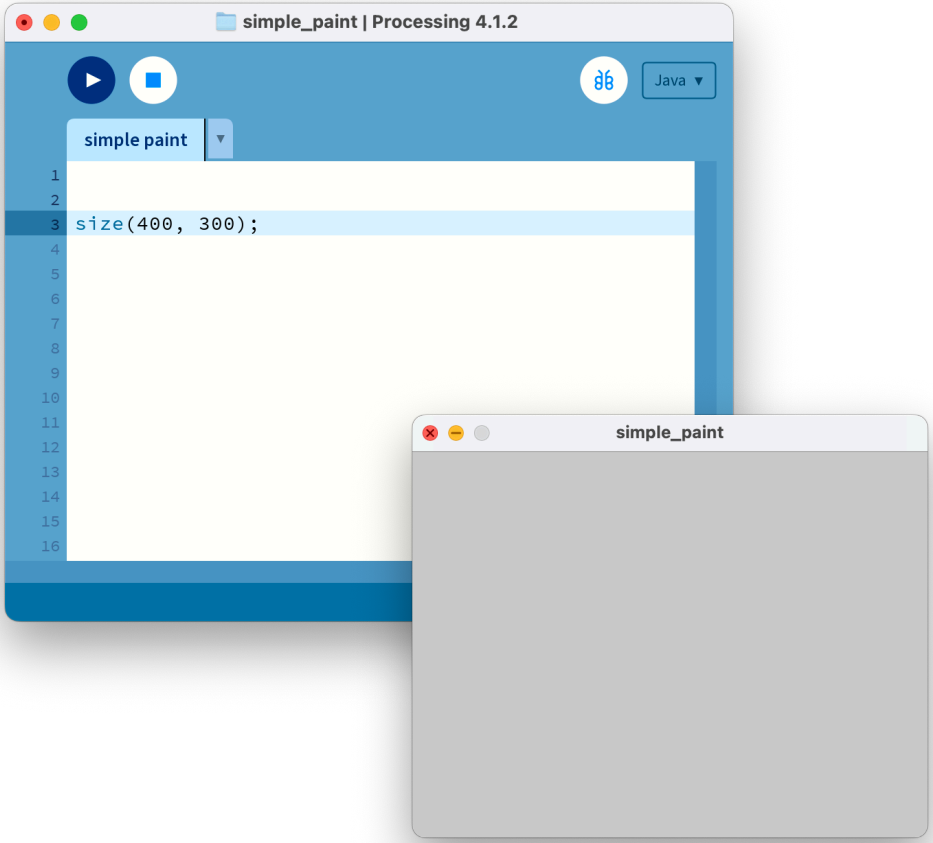
Run + Stop

- Use the start and stop buttons to start and stop your sketch from running.
- If you run and empty sketch, it will still work!



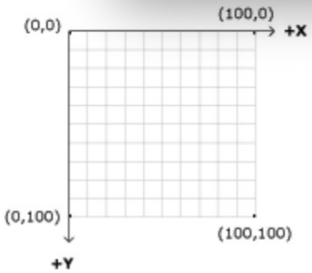
Canvas Size

- First task is to make the canvas a bit larger with the size function:
  - size(*width*, *height*);
  - size(400, 300);
- the size is set in pixels
- Don’t forget the semi-colon



Draw a Circle

- Now draw a circle using the ellipse function that takes four arguments:
  - **ellipse**(*x, y, width*, *height*);
  - **ellipse**(200, 150, 20, 20);
- To draw a rectangle use:
  - **rect**(*x, y, width, height);*
- Coordinate system: **top left is 0, 0**

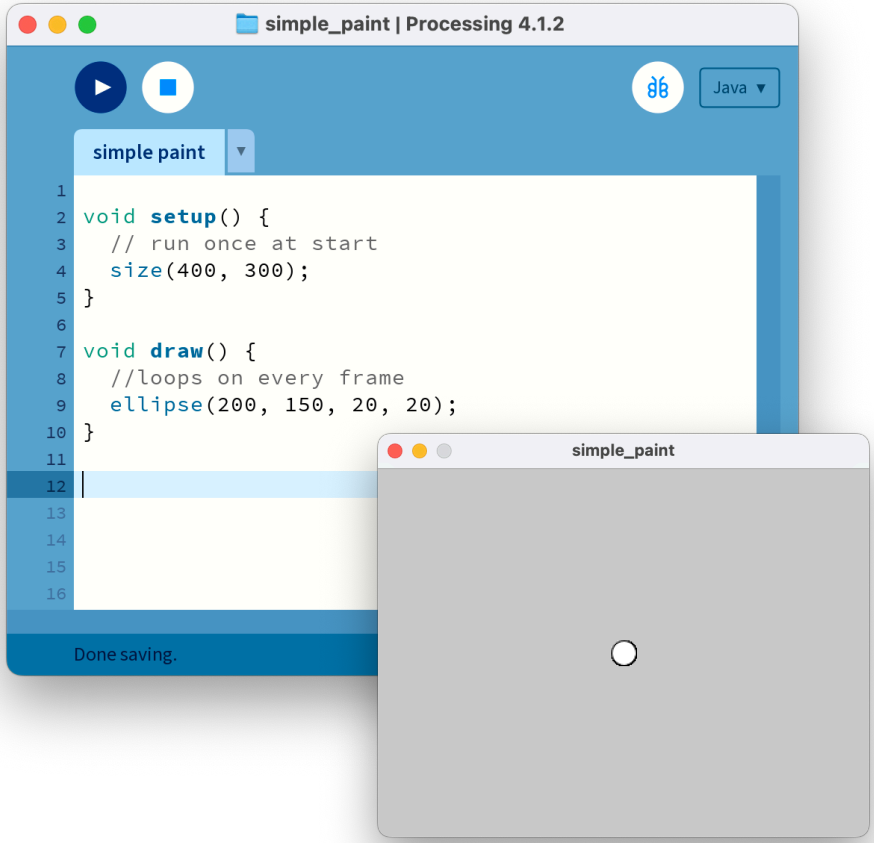


setup & draw

- The last sketch ran through only once then stops. This can work for some non-interactive applications, but we want to be able to animate! So we need to use **setup** and **draw** functions:

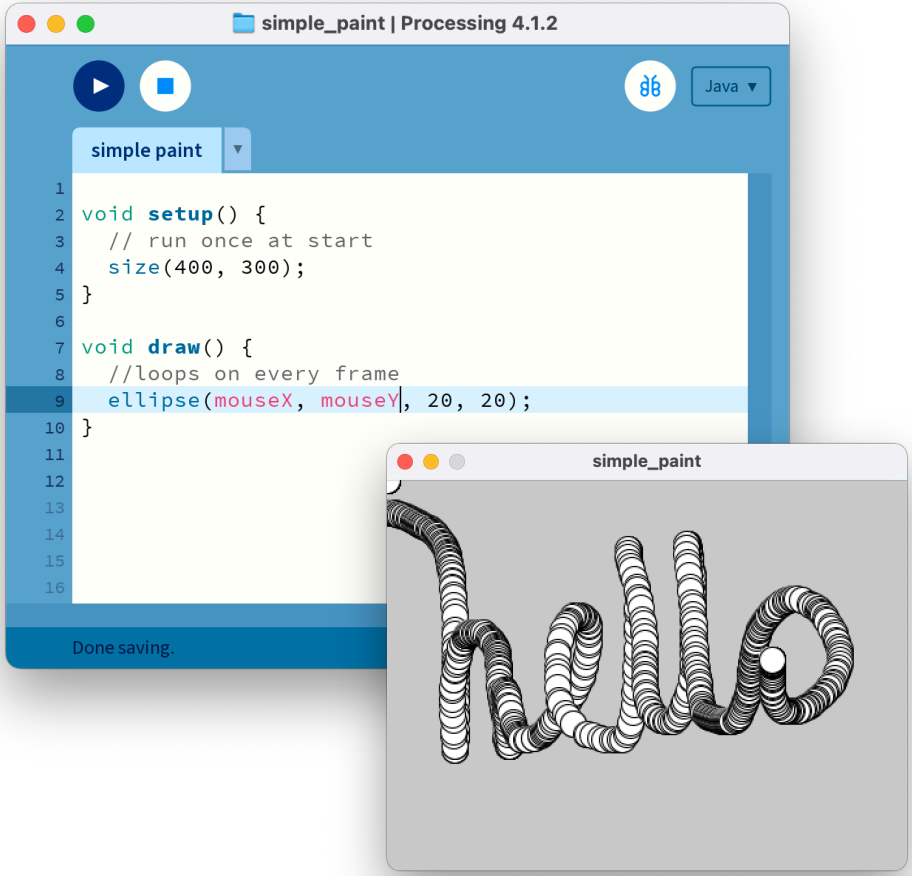
- void **setup**() {
  // runs once at the start
  }

- void **draw**() {
  // runs every frame in a loop
  }



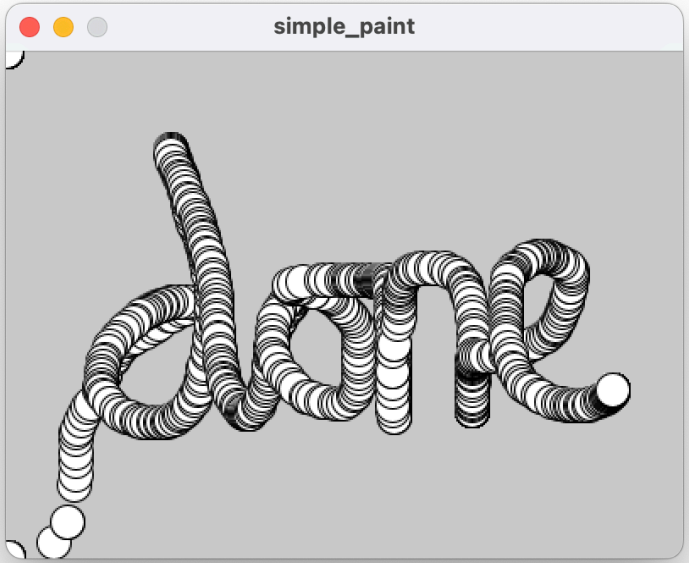
mouseX & mouseY

- Processing has some handy global variables predefined that you can access
  - **mouseX** returns the current mouse x pos
  - **mouseY** returns the current mouse y pos
- The IDE highlights the variable pink to show that it's a predefined variable.
- Other useful ones include **width**, **height** of the sketch window. And previous mouse positions **pmouseX** and **pmouseY**.



MVP achieved!

- You all now have the starting point of a very basic paint program.
    - Try painting the person next to you.
- Your job for the remainder of the workshop is to work in pairs using pair programming and a kanban board to improve on this.
- You will be testing your creation out on another person (with no instructions!) to make a short life drawing portrait at the end of the workshop. But… here are some starting points:



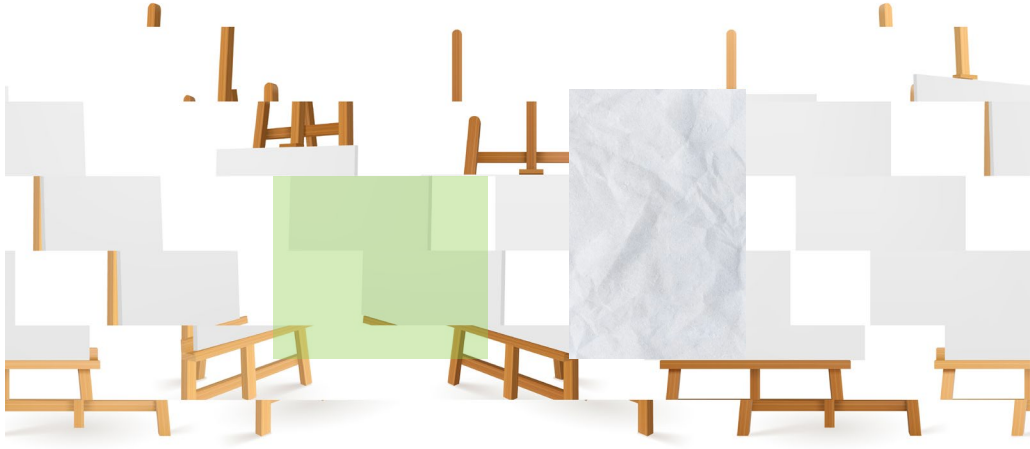
Changing Colour

- Colours values are between 0-255
- **fill**(*red*, *green blue*)
  - Changes the fill colour for the next drawing command.
- **stroke**(*red*, *green*, *blue*)
  - Change the colour for the outline of the next shape

image

Canvas / Background

- Change the background colour of the sketch by calling background(*r,g,b*)
- Try loading an image as a background using image() … note, do this in setup, rather than the draw loop!
- Explore using paper, canvas textures, a sketchbook or use a photograph (say of sky + clouds for a cloud drawing app)



image

Random

- Try randomising some elements with:

  random(*max*)
  random(*min, max*)

- For instance filling a shape with a random colour:

  - **fill**(**random**(255, **random**(255), **random**(255));

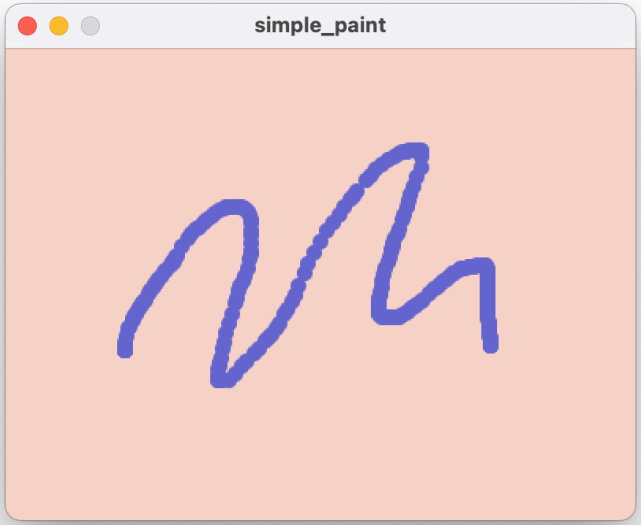
image

simple_paint | Processing 4.1.2

Java

simple paint

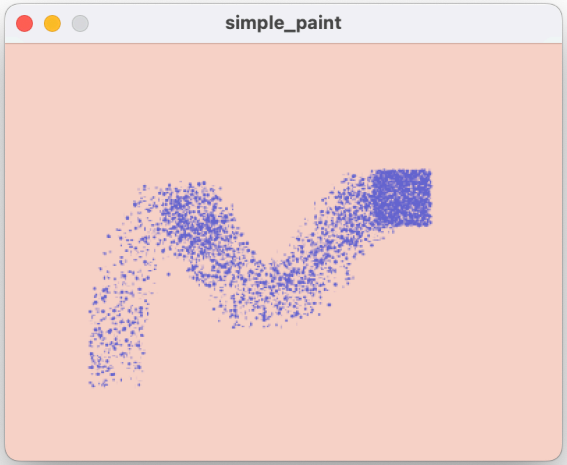
```
1
2  void setup() {
3    // run once at start
4    size(400, 300);
5    background(240, 210, 200); // light pink
6  }
7
8  void draw() {
9    //loops on every frame
10   if (mousePressed) {
11     noStroke();
12     fill(100, 100, 200); // light blue
13       ellipse(mouseX, mouseY, 10, 10);
14   }
15 }
16
17
18
```



simple_paint | Processing 4.1.2

simple paint

```
1
2  void setup() {
3    // run once at start
4    size(400, 300);
5    background(240, 210, 200); // light pink
6  }
7
8  void draw() {
9    //loops on every frame
10   if (mousePressed) {
11     noStroke();
12     fill(100, 100, 200); // light blue
13     for (int n = 0; n < 10; n++) {
14       ellipse(mouseX+random(-20, 20), mouseY+random(-20, 20), random(3), random(3));
15     }
16   }
17 }
18
19
```



Spraypaint – could be better, perhaps use **cos**() and **sin**() to create a circular area rather than a square?

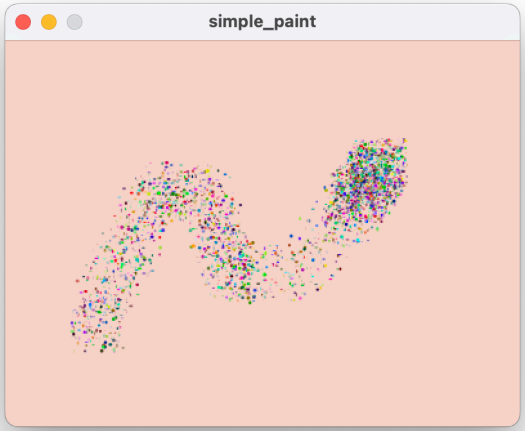
simple_paint | Processing 4.1.2

Java

simple paint

```
1
2  void setup() {
3    // run once at start
4    size(400, 300);
5    background(240, 210, 200); // light pink
6  }
7
8  void draw() {
9    //loops on every frame
10   if (mousePressed) {
11     noStroke();
12     for (int n = 0; n < 10; n++) {
13       fill(random(250), random(250), random(250)); // RANDOM COLOUR DROPS
14       ellipse(mouseX+random(-20, 20), mouseY+random(-20, 20), random(3), random(3));
15     }
16   }
17 }
18
19
```

Done saving.

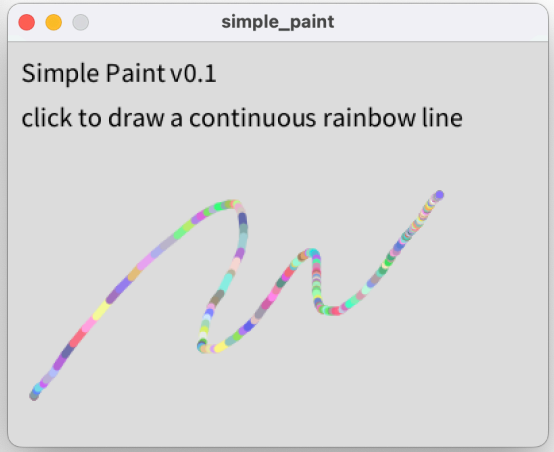


Spraypaint – move the **fill**() command and change the colour for every drop? Try passing two arguments to **random**(*min*, *max*) for a more realistic palette

simple_paint | Processing 4.1.2

simple paint

```
1
2  void setup() {
3    // run once at start
4    size(400, 300);
5    background(220); // light grey (single argument is greyscale)
6    textSize(20);
7    fill(0); // black
8    text("Simple Paint v0.1 \nclick to draw a continuous rainbow line", 10, 30);
9  }
10
11 void draw() {
12   //loops on every frame
13   if (mousePressed) {
14    strokeWeight(6);
15    stroke(random(100, 255), random(100, 255), random(100, 255));
16    line(pmouseX, pmouseY, mouseX, mouseY);
17   }
18 }
19
20
21
```



Documentation + **Commenting** - make sure that a new user can work out how to use your app, and please remember to use comments, either // or /**multiline*/ so that your code is legible

attendance code:
4025

Challenges

These are potential challenges that you could start to fill your Kanban board up with:

- Add a way to change brush colour
- Change the brush size
- Add key commands **keyPressed**()
- Save an image with save("example.jpg")
- Multiple brush types, select with a key
- Use an image() as a brush, rotate randomly to vary the stroke
- Add an eraser. Can this be on right click?
- Add a way to reset/clear the whole canvas
- Auto-scribbler, automatically nudge the pen around the drawing position
- Symmetry – draw a second point on the opposite side of the canvas
- Create smooth lines by drawing between last and current mouse position:
  line(*p*mouseX, pmouseY, mouseX, mouseY)
- Add opacity to your brush by passing a fourth parameter (0 = transparent, 255 = opaque):
  fill(red, green, blue, alpha)
- Create lined or grid paper using a for loop.
- Change the **strokeWeight**() of your line based on the speed of mouse movement. Use pmouseX and pmouseY along with mouseX and mouseY to determine the distance moved

- **NOTE**: please add **documentation** – someone's going to be using your system without you being able to explain it. Use **println**() to send text to console or even better **text**() to write to screen.

Going Further

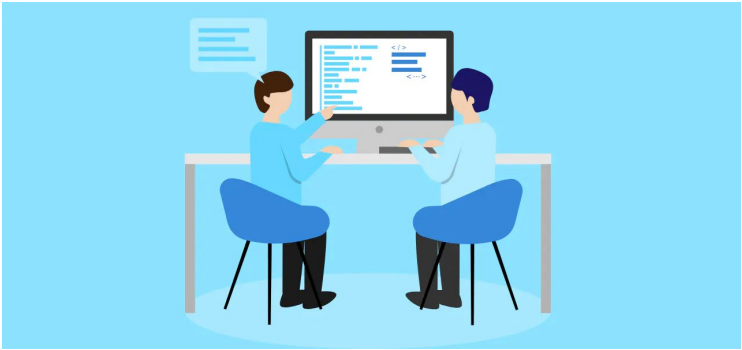
- Many more functions documented in the Processing Reference:
  - https://processing.org/reference
- Look through the reference and see what functions could be interesting to try out
- More info on the Processing environment here:
  - https://processing.org/environment/

`mouseButton`	Shows which mouse button is pressed
`mouseClicked()`	Called once after a mouse button has been pressed and then released
`mouseDragged()`	Called once every time the mouse moves and a mouse button is pressed
`mouseMoved()`	Called every time the mouse moves and a mouse button is not pressed
`mousePressed`	Variable storing if a mouse button is pressed
`mousePressed()`	Called once after every time a mouse button is pressed
`mouseReleased()`	Called every time a mouse button is released
`mouseWheel()`	The code within the `mouseWheel()` event function is run when the mouse wheel is moved
`mouseX`	The system variable that always contains the current horizontal coordinate of the mouse
`mouseY`	The system variable that always contains the current vertical coordinate of the mouse
`pmouseX`	The system variable that always contains the horizontal position of the mouse in the frame previous to the current frame
`pmouseY`	The system variable that always contains the vertical position of the mouse in the frame previous to the current frame

Pair Programming + Kanban

~80 mins

- Partner up. Swap roles regularly.
- Use a Kanban board to plan and track which features you will be working on.
    - *Not Started - > In Progress -> Done*
- Start simple! Keep features small.
- Consider including a ‘shelved’ or ‘parked’ column to put features in that aren’t working out (given 90min timescale). Don’t get stuck!



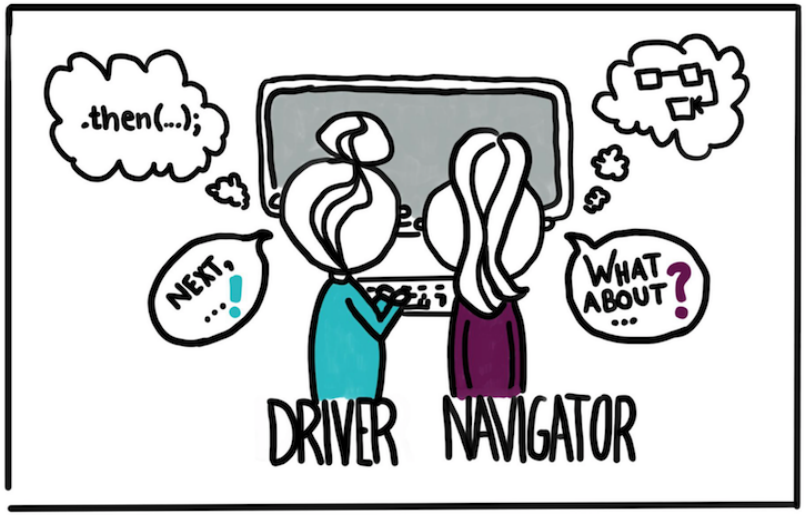
Pair Programming Roles

Driver (Helm)

- The person typing
- Focused on the short-term goal
- Places longer-term goals on backburner
- Talks through what they are doing

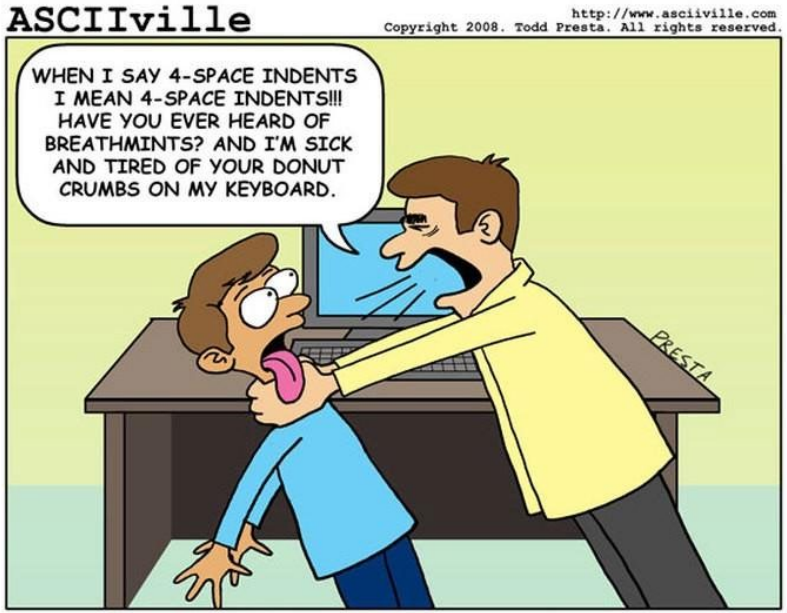
Navigator (Tactician)

- Observes the drivers work
- Reviews code in real-time
- Notes suggestions
- Scans horizon for longer-term issues



Pair Programming Tips

- **Distractions**. Don't check your phone/mail. Stay focused. Build in more individual time if you need it.
- **Micro-management.** Stick to higher level comments, and avoid saying things such as "now type..."
- **Impatience.** Don't jump right in when the driver makes a typo. They may have seen it and just haven't gone back to correct. Avoid breaking their flow.
- **Keyboard hogging.** Make sure to stick to a rotation schedule and avoid sticking to one role.



The dark side of pair programming.

User Testing

15 mins

- Find another pair and swap over.
- **Without any instruction**, use their paint app to draw a portrait of your pair programming partner (take turns)
- Show the creators of the app how you used it, and your artistic results!
- What worked? What didn't? Did you enjoy it? What would you improve? What was similar with your own? How was the resulting portrait?

image

Week	Date	**Lecture** Monday 11:00-12:00 *PHYS BLDG G44 FRANK*	**Workshop** Monday 13:00-15:00 *MVB 2.11 PC*	**Groupwork**
1	23/01/22	Introduction and Process [slides] [materials]	Teams, Waterfall Method and Project Brief [slides] [case study] [project brief]	Research games, create list on team repo. Install Processing
2	30/01/22	Agile Software Development [slides]	Intro to Processing, Agile Techniques	Decide on two game ideas
3	06/02/22	Requirements Engineering	Paper Prototyping, Requirements, Ideas Clinic	Collect requirements. Decide on final idea
4	13/02/22	Object Orientated Design	Classes Activity	Add requirements section to report
5	20/02/22	Implementation	Case Study, Spring Prep, Continuous Integration	Develop a working prototype over reading week!
6	28/02/22	READING WEEK	GAMES JAM	
7	06/03/22	Project Management	IN CLASS TEST (assessing lectures 1-4)	Define team roles
8	13/03/22	HCI - Qualitative	HCI Qualitative Task	Add qualitative assessment (of your choice) to report
9	20/03/22	HCI - Quantitative	HCI Quantitative Task	Add quantitative assessment (of your choice) to report
	27/03/22	EASTER week 1	SPRINT 1	
	03/04/22	EASTER week 2	SPRINT 2	
	10/04/22	EASTER week 3	SPRINT 3	
10	17/04/22	Software Engineering Extended	IN CLASS TEST (assessing lectures 5-9)	Develop Game
11	24/04/22	Coursework Feedback		Finish Report
12	01/05/22	Bank Holiday Monday (no class)	Demo Day Weds/Thurs (tbc)	Submit Report

homework / groupwork

- Make sure you can make a commit. Then take and upload a team photo!
- Decide on **TWO IDEAS** to bring along to next weeks workshop.
- **Add these two ideas to your repo**! One paragraph each. Images welcome!