LCG: Informatica Applicata - C2

P5: Transformations, Input Response (mouse, slider, rollover), Random Data generation

Scuola del Design
Laurea Triennale in Communication Design

Aula B6.3.1, Politecnico di Milano
October 8th, 2024

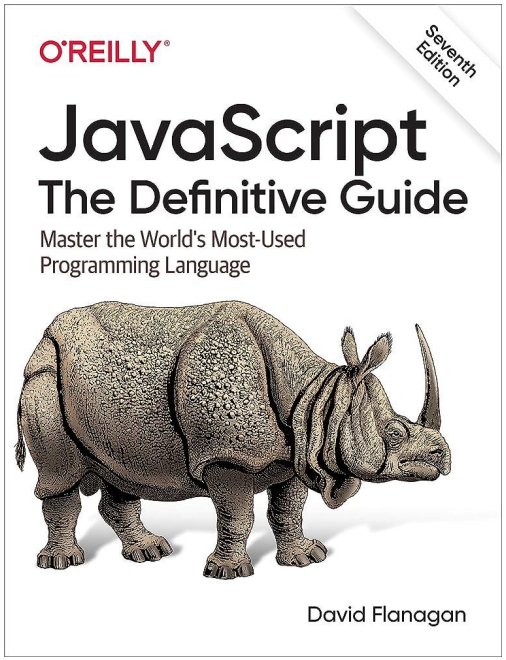
Davide Conficconi <davide.conficconi@polimi.it>
Alessandro Nazzari <alessandro.nazzari@polimi.it>

POLITECNICO
MILANO 1863

Important things: **Material**

https://bit.ly/lcg-c2-2024

https://webeep.polimi.it/course/view.php?id=307969



https://www.w3schools.com/js/default.asp

- https://git-scm.com/doc
  https://docs.github.com/en/get-started/using-git/about-git
- https://docs.github.com/en/pages/quickstart
- https://docs.github.com/en/pages/getting-started-with-github-p
  /creating-a-github-pages-site
- https://www.w3schools.com/git/git_intro.asp?remote=github
- https://git-scm.com/book/en/v2
- Github pages Docs
- https://docs.github.com/en/pages/getting-started-with-github-pages
  /creating-a-github-pages-site#next-steps
- Bootstrap reference
- From markdown to github pages

Recall: Last Time Objectives

p5.js

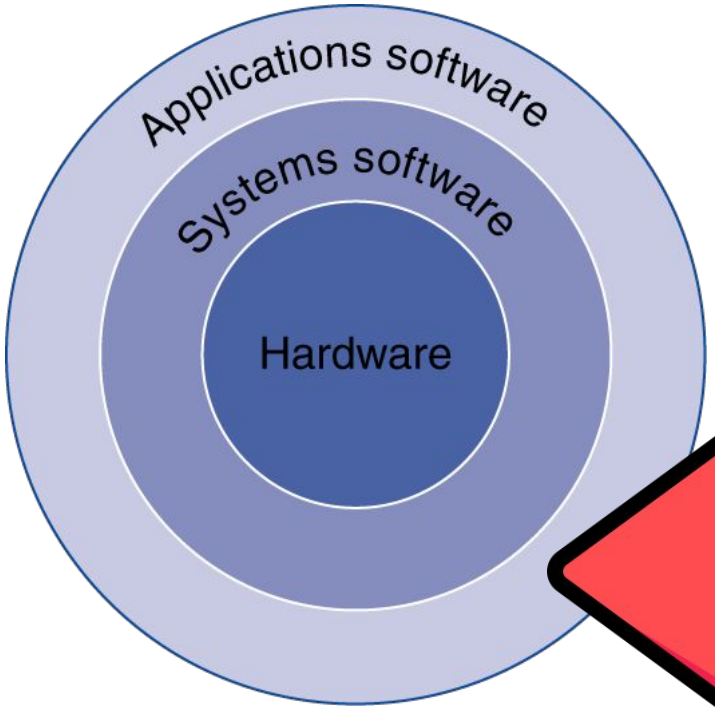
1. Recap + Challenge discussion
2. Refresh on Algorithms
3. Dev with P5: Live Editor and Local VScode
4. Draw with P5
5. Variables + JS arithmetic
6. Repetitions
7. Branches
8. Put everything live and check github!

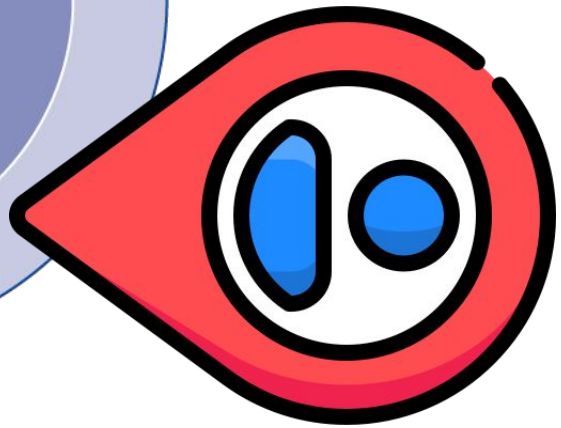
p5* https://editor.p5js.org/

GitHub Pages

git GitHub Desktop

Recall: Abstraction Stack of Reference





Recall: To Build our “web portfolio” of LCG

Indice di Informatica Applicata C2

Terremo traccia degli sviluppi tramite questo indice.

- Lezione 1
- Lezione 2
- Lezione 3
- Lezione 4
- Lezione 5
- Lezione 6

https://davideconficconi.github.io/indice-lcg2425/index.html

https://pages.github.com/

https://docs.github.com/en/pages/getting-started-with-github-pages/creating-a-github-pages-site#next-steps

Recall: Una definizione di Algoritmo

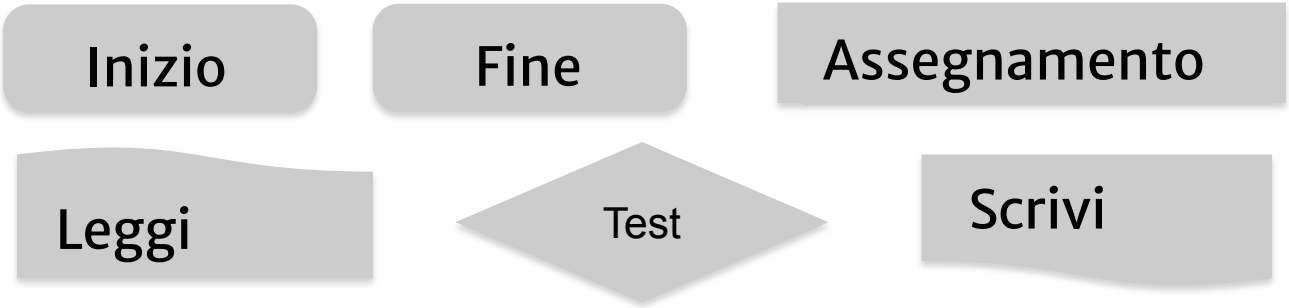
"Una sequenza di passi, definiti con precisione, che portano alla realizzazione di un compito."[1]

Come si specifica un algoritmo?

Utilizzando dello pseudocodice

Se A > B **allora** B = A **altrimenti** A = B

Utilizzando dei diagrammi di flusso (aka schemi a blocchi)



Recall:

What is p5.js?

https://hello.p5js.org/

p5.js is a **friendly** tool for
learning to code and make art.

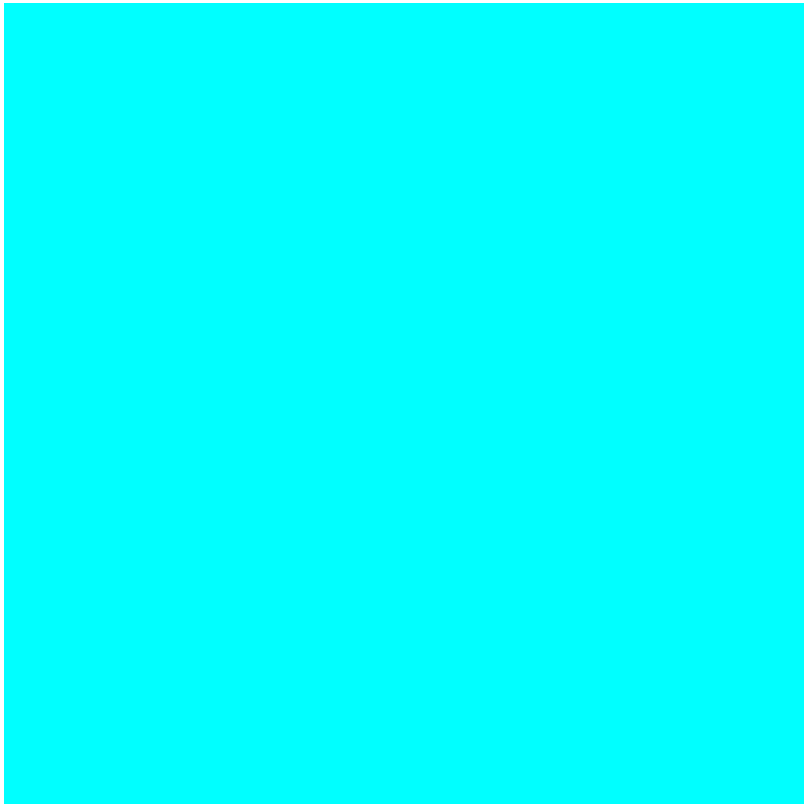
Recall: The Most Basic p5.js sketch

```
function setup() {
  createCanvas(400, 400);
}
function draw() {
  background(220);
}
```

setup() is called and runs one time. It can be used to set default values for your project.

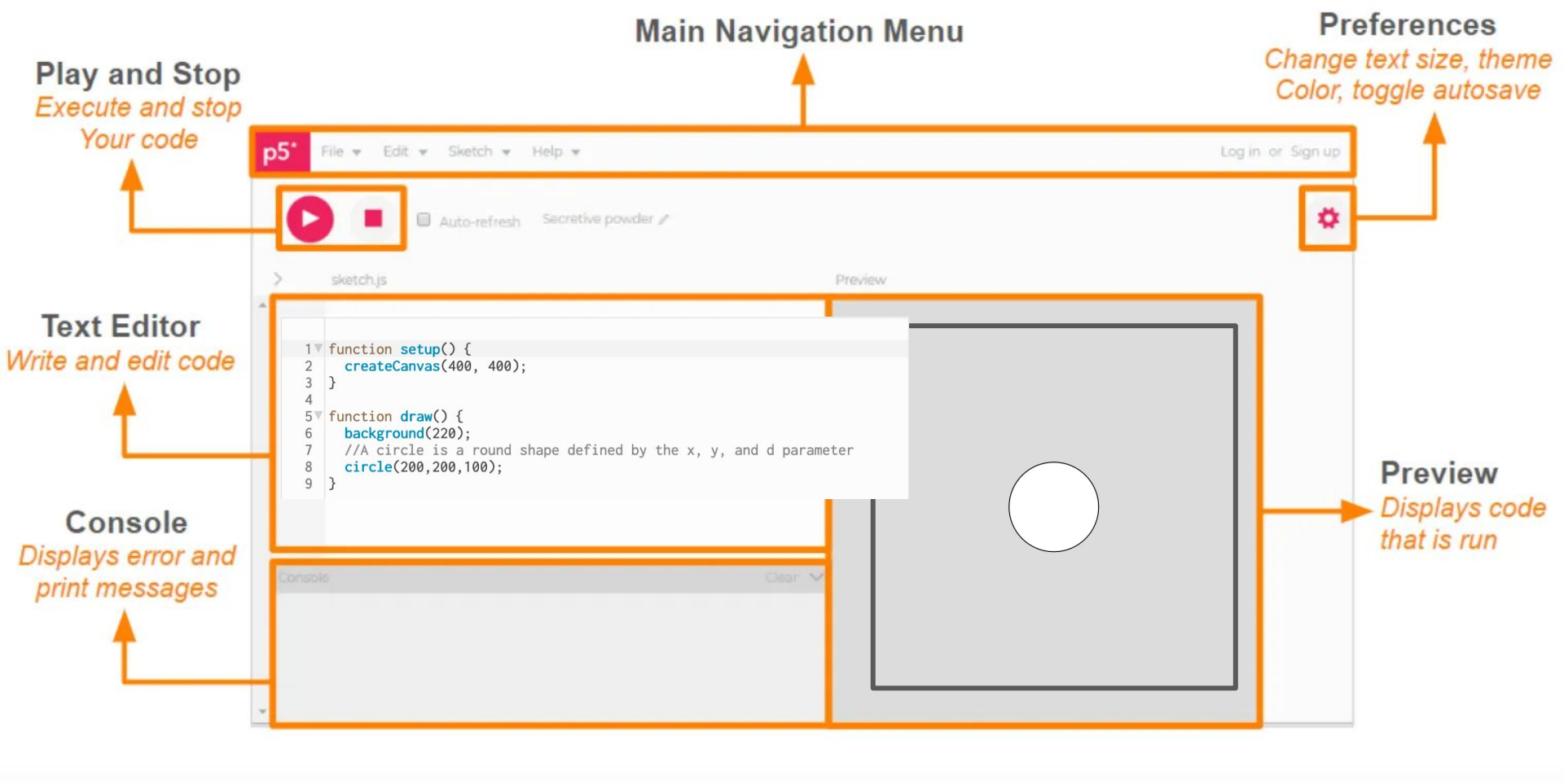
draw() is called directly after setup() and executes the lines of code inside its curly brackets 60 times per second until the program is stopped or the noLoop() function is called.

Recall: Can we change the background color?



Let's have a look at https://p5js.org/reference/

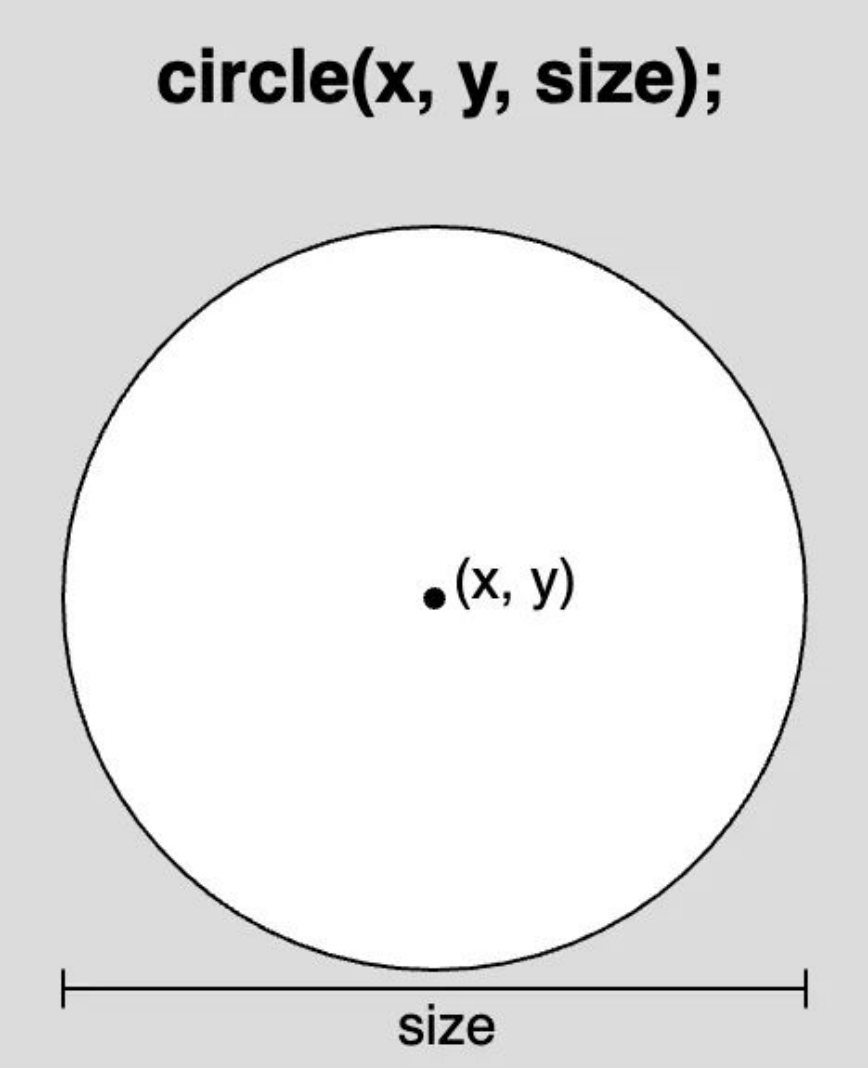
Recall: Web Editor Interface+Drawing circle

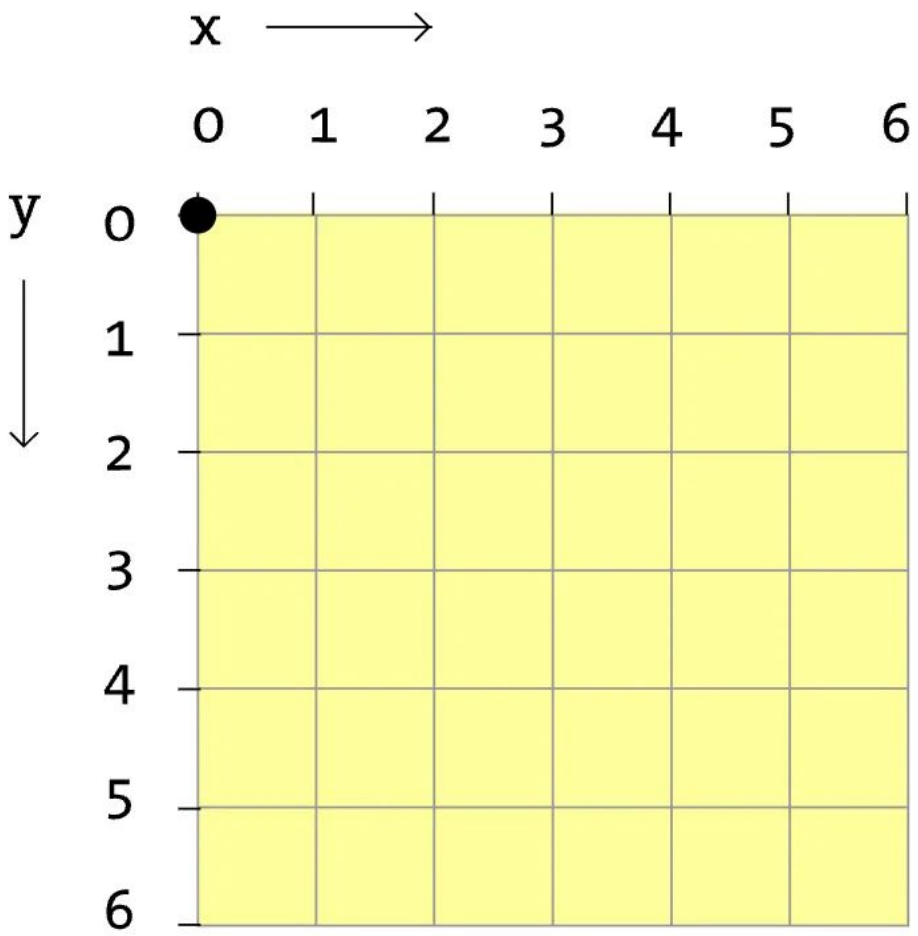


Recall: Draw a Full Moon in the top right corner

https://p5js.org/tutorials/get-started/

Recall: Draw a Full Moon in the top right corner



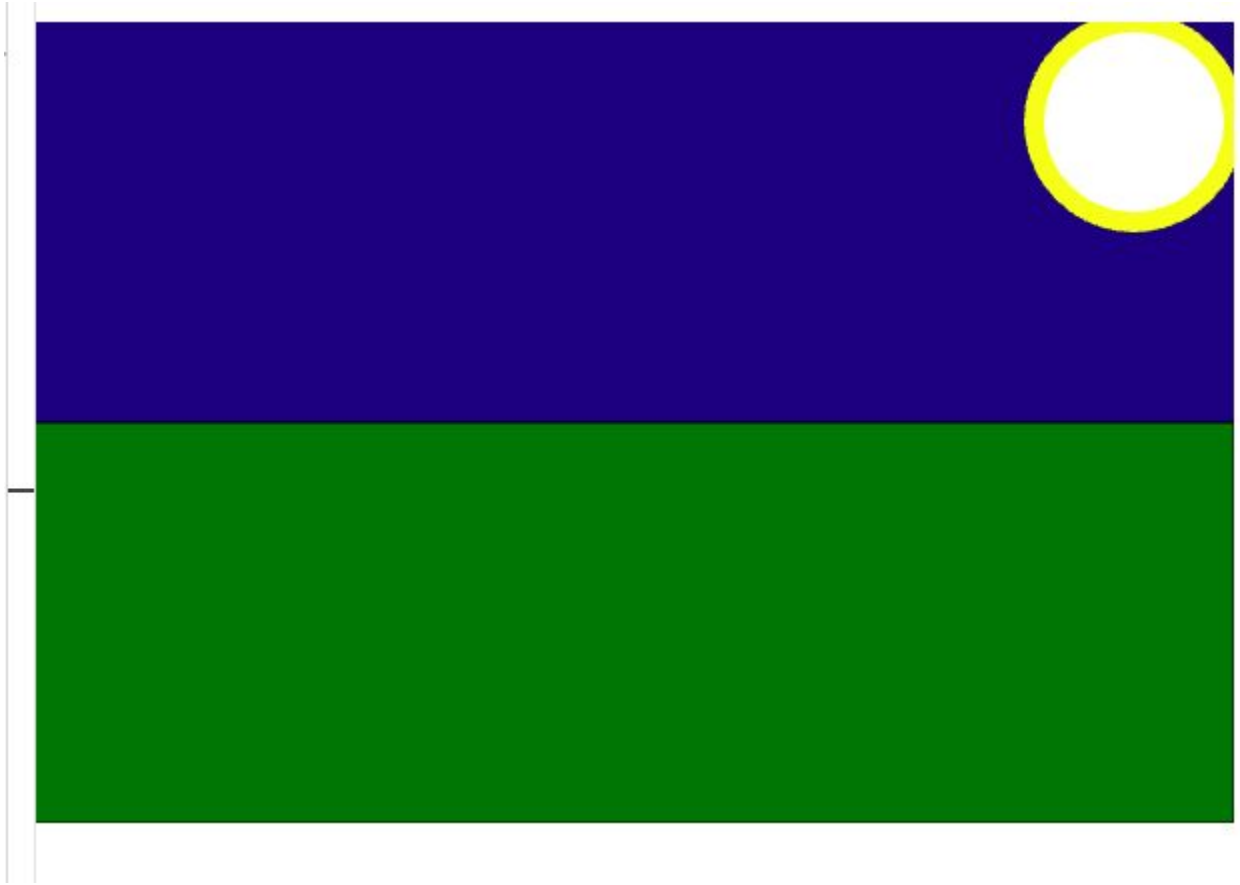


Learn more about the HTML canvas coordinate system and shapes on this p5.js reference page.

Visit the p5.js reference page for `circle()` to learn more.

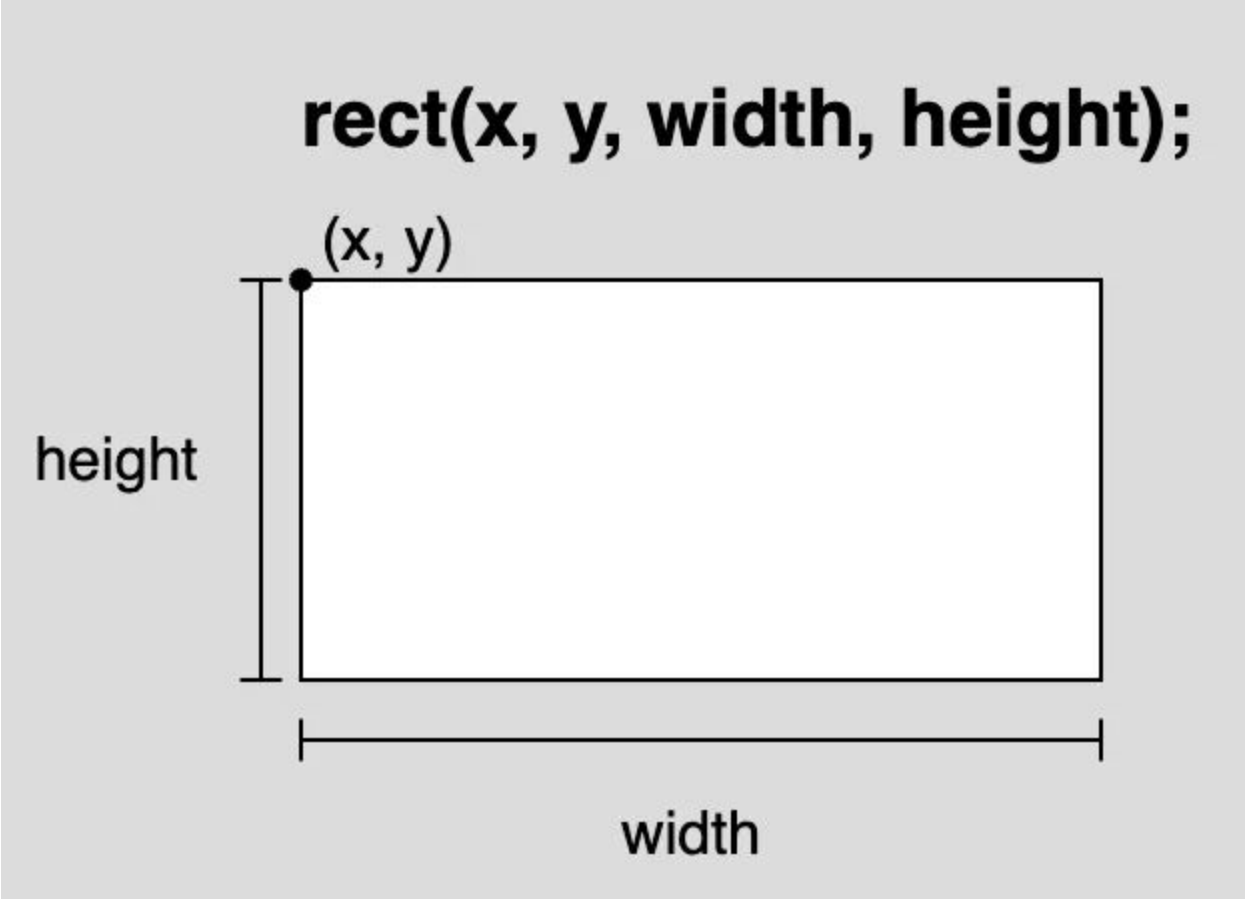
https://p5js.org/tutorials/get-started/

Recall: Let's add the grass



https://p5js.org/tutorials/get-started/

Recall: Let's add the grass



Recall: This is only the beginning…

Recall: This is only the beginning…

https://p5js.org/tutorials/get-started/

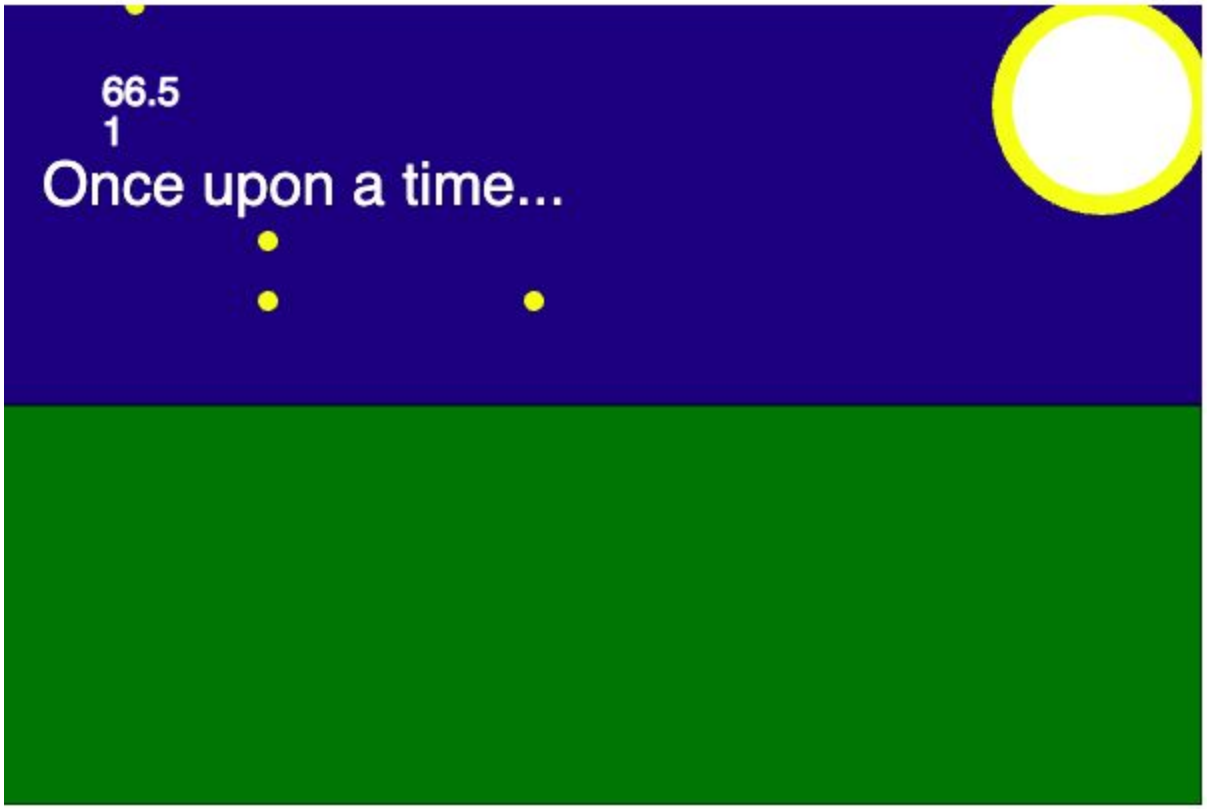
Recall: JavaScript: Datatypes

L'insieme dei tipi di dato in Javascript si dividono in **primitivi** e **oggetti** (vedremo quest'ultimi più avanti)

I tipi primitivi principali sono:

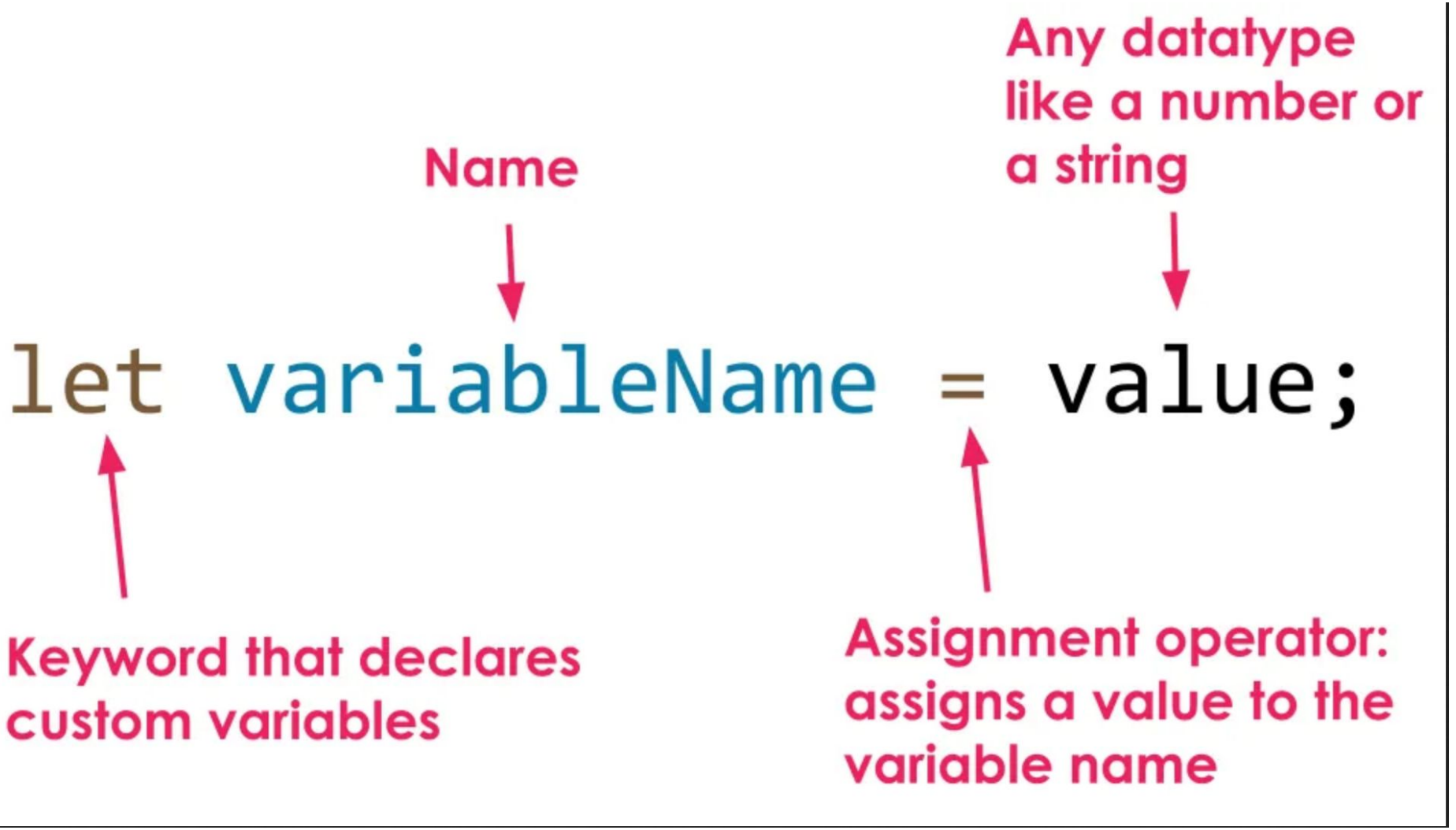
Nome	**Descrizione**	**Esempio**
Boolean	Due possibili stati: vero o falso	*true*
Number	Interi e numeri con la virgola	3.14
String	Sequenze di caratteri	"Informatica applicata"
Null	Un tipo speciale che indica l'assenza di un valore	*null*
Undefined	Un tipo speciale che indica l'assenza di una variable	

Recall: If we use variables

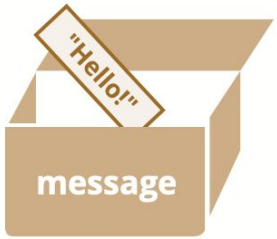


https://p5js.org/tutorials/get-started/

Recall: JS Variables

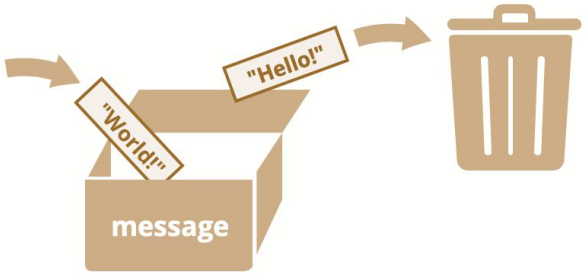


Recall: JS Variables and Constants



Variables are named storage for data
– Like in any other language…
– They are initialized, read and written
– Declaring twice triggers an error
– Case matters, symbol $ and _ are legal

```
let message = ”Hello!”;
console.log(message);
```



Constants use `const` instead of `let`

```
let message = ”Hello!”;
message = ”World!”;
console.log(message);
```

Recall: JS: Operatori Matematici

Binari

Simbolo	Descrizione
+	Somma
-	Differenza
*	Moltiplicazione
/	Divisione
%	Resto della divisione intera
**	Elevamento a potenza

Unari

Simbolo	Descrizione
++a	Pre-Incremento
a++	Post-Incremento
--a	Pre-Decremento
a--	Post-Decremento

Gli operatori binari possono anche essere utilizzati come assegnamento:

`a = a + 5` può essere scritto come `a += 5`

L'operatore di **divisione /** calcola:

`13 / 5` è uguale `2.6`

L'operatore **Modulo** calcola il **resto** della divisione:

`13 % 5` è uguale `3`

Recall: JS: Operatori di Confronto

Simbolo	Descrizione
==	Uguale
!=	Diverso
===	Strettamente Uguale *
!==	Strettamente Diverso *
> e >=	Maggiore e Maggiore Uguale
< e <=	Minore e Minore Uguale

* L'operatore **strettamente** uguale verifica che anche il **tipo di dato** sia equivalente, senza eseguire il casting.

In modo analogo funziona lo strettamente diverso.

Esempio:

- 5 == 5 e 5 === 5 sono veri
- 5 == "5" è vero
- 5 != "5" è falso
- 5 === "5" è falso
- 5 !== "5" è vero

When comparing **different types**, everything is **converted** to **Number**
– This leads to funny consequences… make sure what type you are comparing with what else!
– The value undefined cannot be compared to anything, yields false

Recall: JS: Operatori Logici

Simbolo	Nome	Descrizione
&&	and	Entrambi gli operandi devono essere veri
||	or	Almeno uno degli operandi deve essere vero
!	not	Inverte il valore dell'operando

A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

(somma logica)

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

(prodotto logico)

A	not A
0	1
1	0

(negazione)

Precedenza: l'operatore "not" precede l'operatore "and", che a sua volta precede l'operatore "or"

A and not B or B and C = (A and (not B)) or (B and C)

Recall: Tabella di verità di un'espressione logica

A and B or not C

A B C	**X** = A and B	**Y** = not C	**X** or **Y**
0 0 0	0 and 0 = 0	not 0 = 1	0 or 1 **= 1**
0 0 1	0 and 0 = 0	not 1 = 0	0 or 0 **= 0**
0 1 0	0 and 1 = 0	not 0 = 1	0 or 1 **= 1**
0 1 1	0 and 1 = 0	not 1 = 0	0 or 0 **= 0**
1 0 0	1 and 0 = 0	not 0 = 1	0 or 1 **= 1**
1 0 1	1 and 0 = 0	not 1 = 0	0 or 0 **= 0**
1 1 0	1 and 1 = 1	not 0 = 1	1 or 1 **= 1**
1 1 1	1 and 1 = 1	not 1 = 0	1 or 0 **= 1**

Recall: JS Variables Scope


```
let totale = 0;

function compute_area(base, height) {
    let area;
    area = base * height;
    totale += area;
    return area;
}

compute_area(5,5);
console.log(totale); // Stampa 25
```

Local variable that exists only within the function scope

Functions can access also **global** variables

Recall: JS Variables Scope cont'd

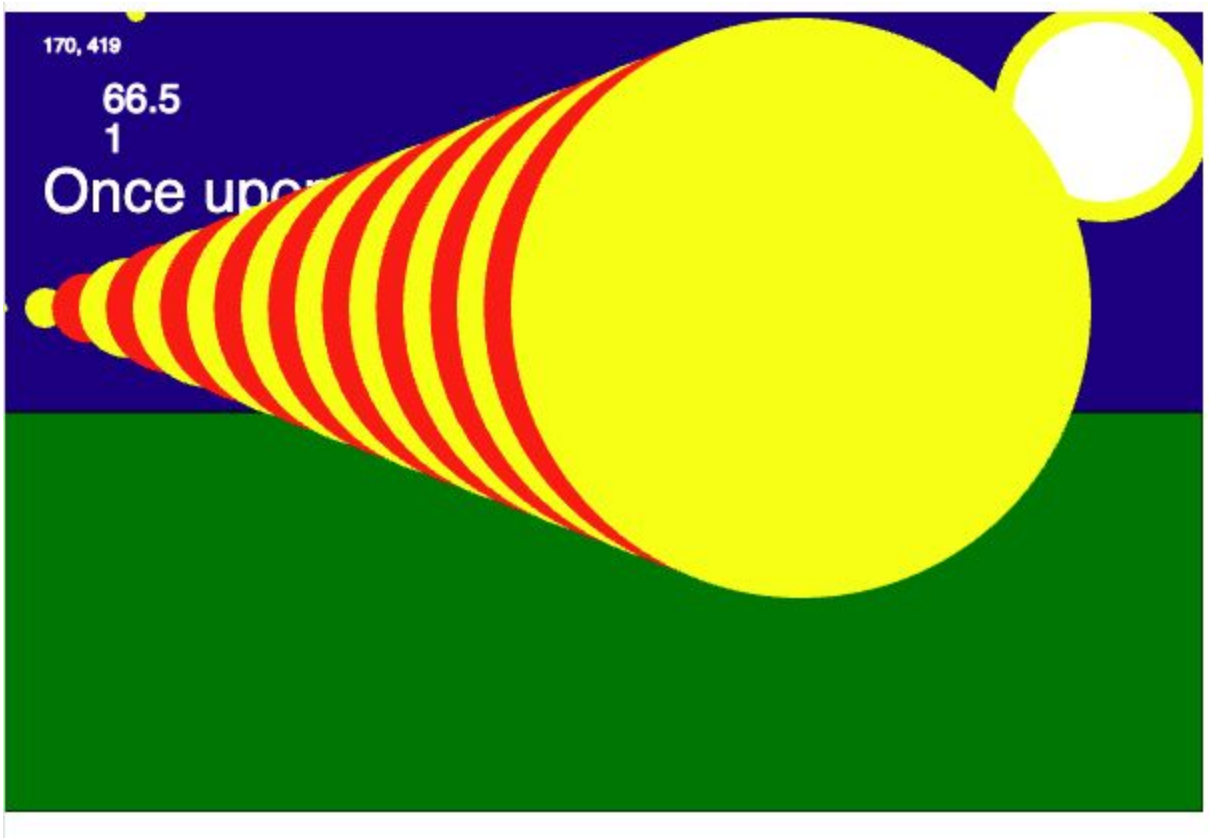

```
let totale = 0;

function compute_area(base, height) {
    let area;
    area = base * height;
    let totale=0;
    totale += area;
    console.log(totale); // Stampa 25
    return area;
}

compute_area(5,5);
console.log(totale); // Stampa 0
```

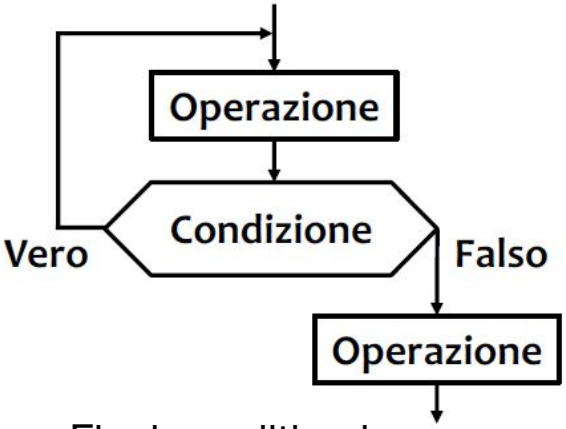
WARNING!!!
If defined a local variable with the same name, then global variable not accessible anymore

If we want to do conditional things (a.k.a. branches)

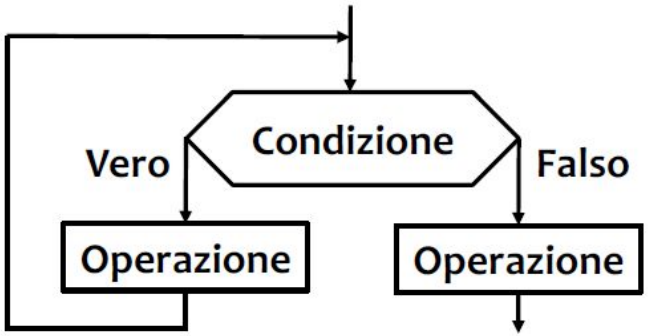


https://p5js.org/tutorials/get-started/

Recall: JS: Loops



Final condition loop:
Body executed at least once
→ **`do ... while`**



Initial condition loop:
Body executed zero or more
→ **`while, for`**

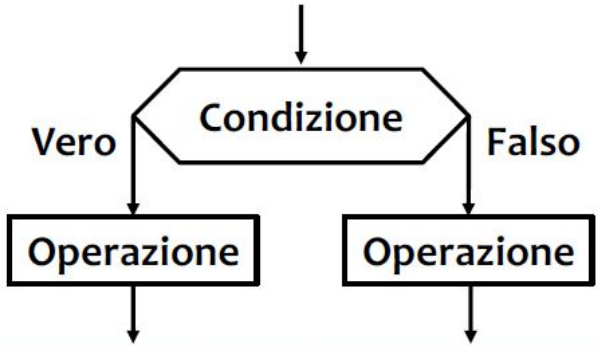
Loops may be boken with **`break`**, or skip the current iteration with **`continue`**

```
for (let i=0; i<10; i++) {
    console.log(i);
    if (i==8) break;
}
```

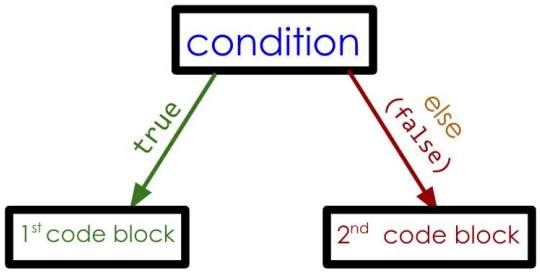
Effectively executes only nine iterations!

Recall: JS: Branching

Branching with if works the usual way
– Logical operators too!
– The and (or) operator evaluates to the first false (true) value, or the last one
• Consider the type conversion rules
– 0, “”, null, undefined, and NaN all become false
– Everything else becomes true
• Also switch is available..



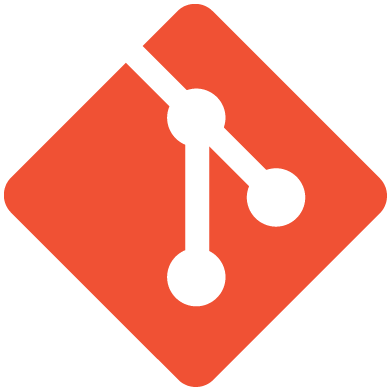

```
if (date.getMonth()==0) {
    console.log("January");
} else if (date.getMonth()==1) {
    console.log("February");
} else {
    console.log("March and later..");
}
```



Recall:

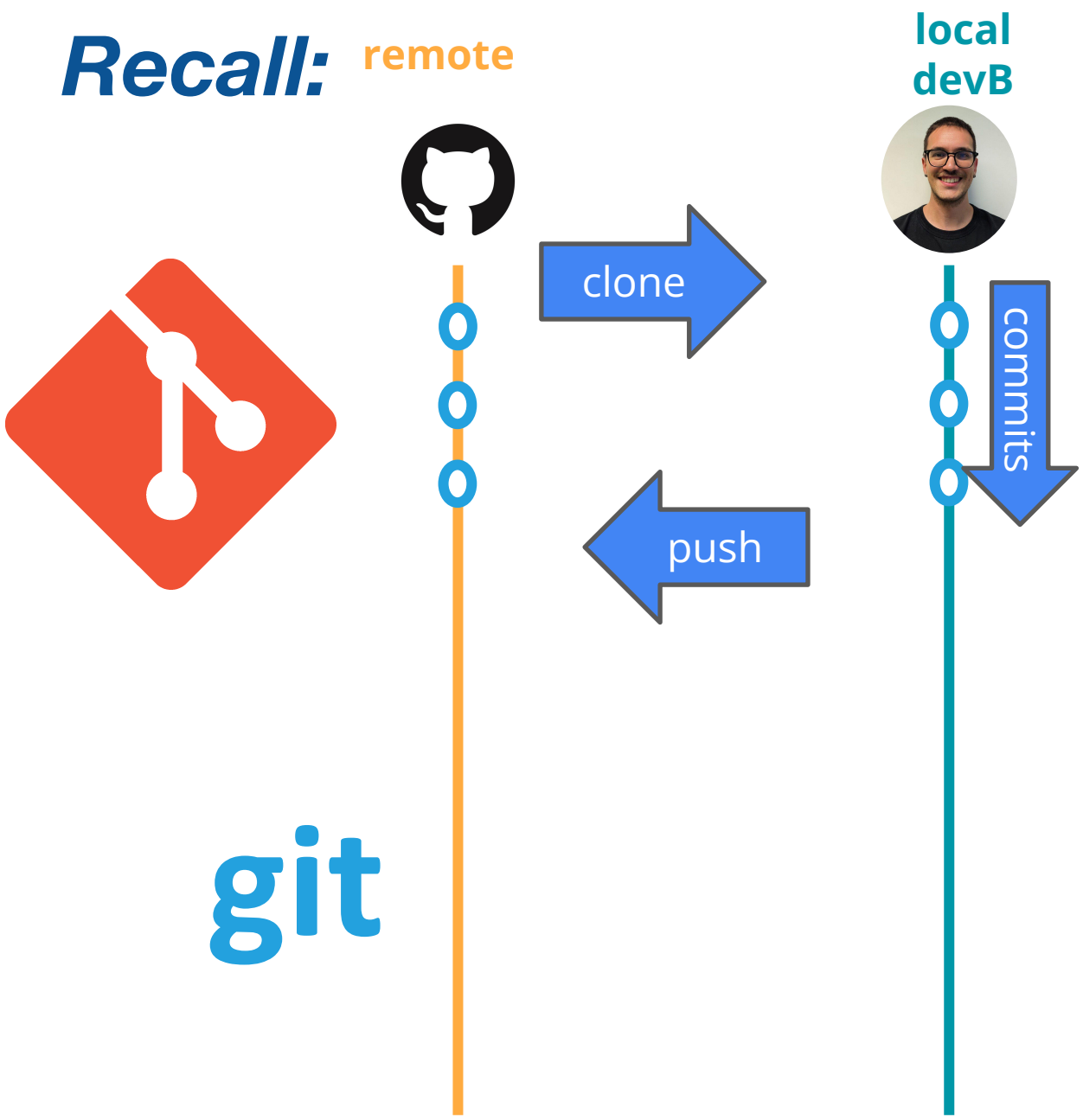
remote

local
devB

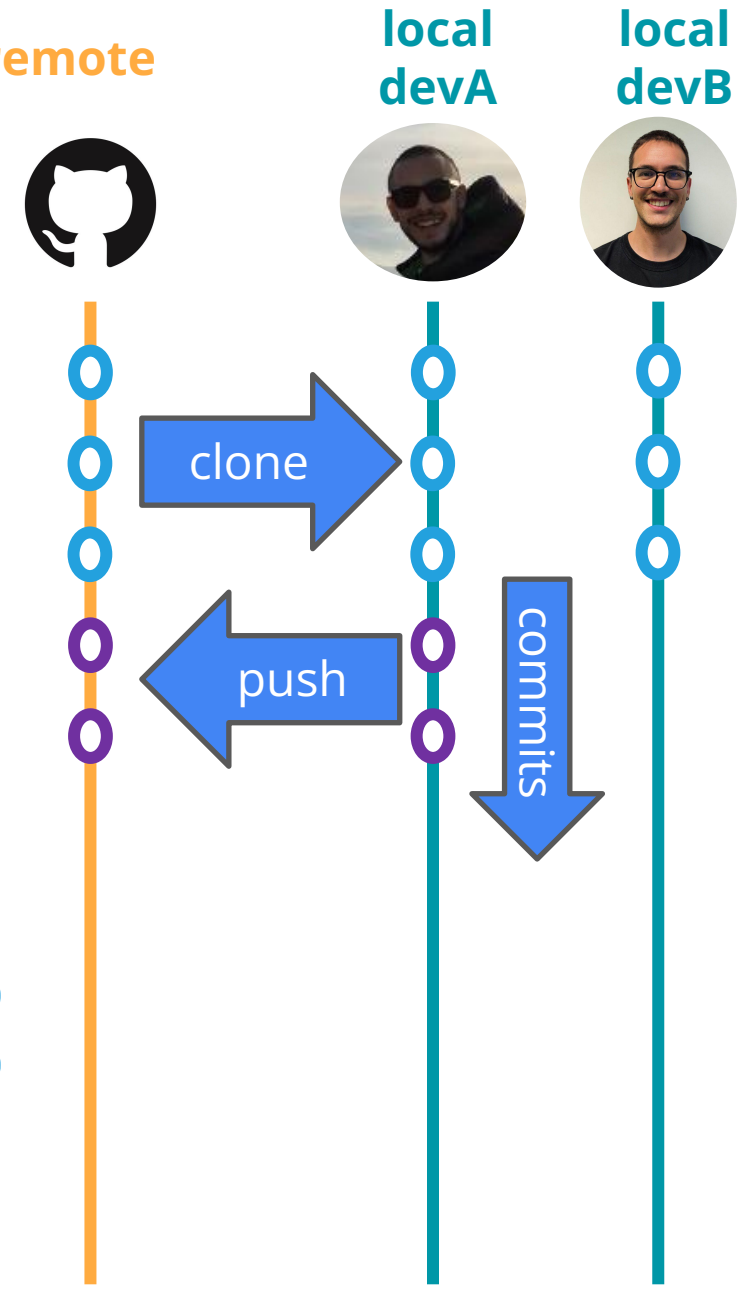


git

Recall:

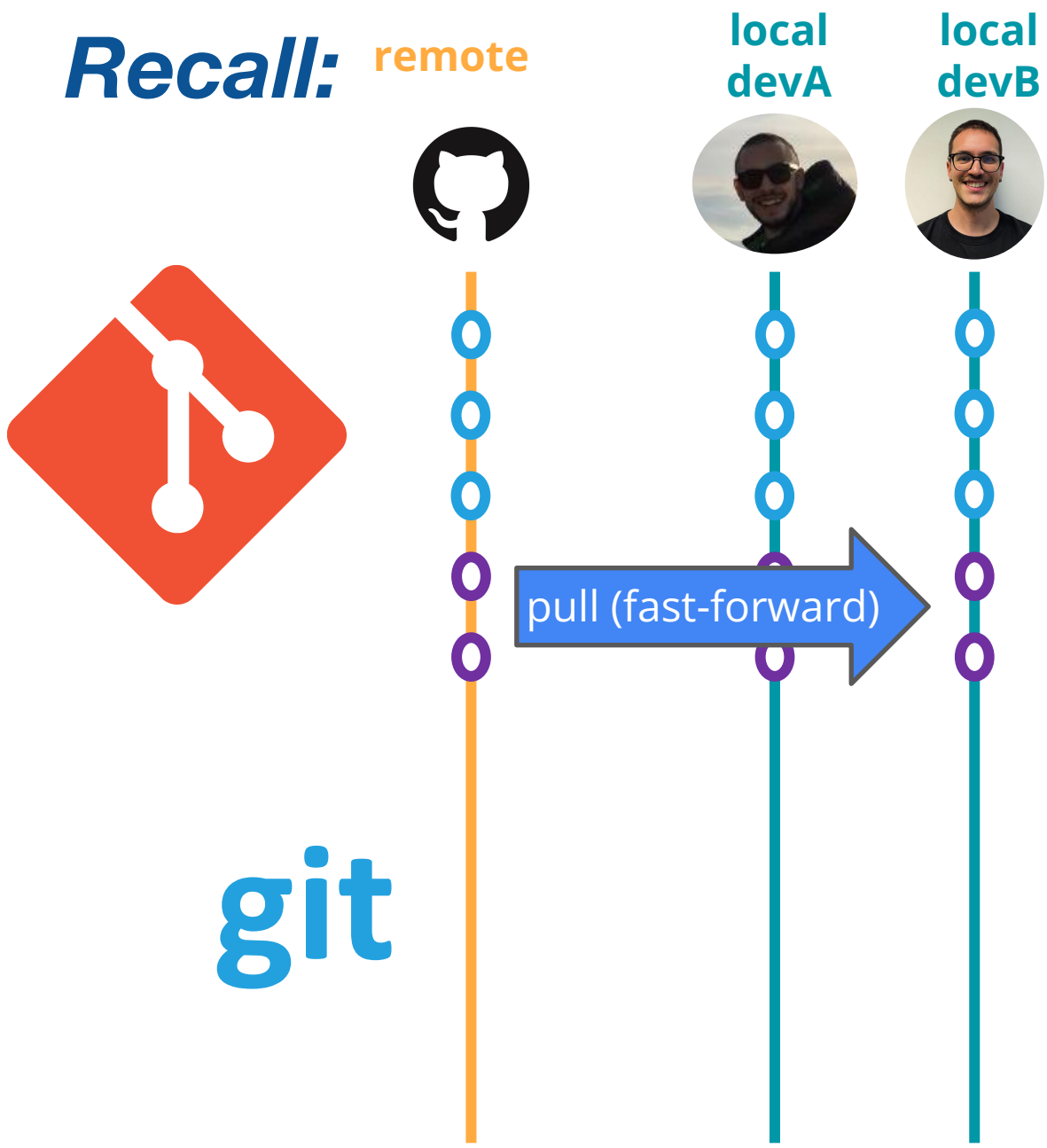


Recall:

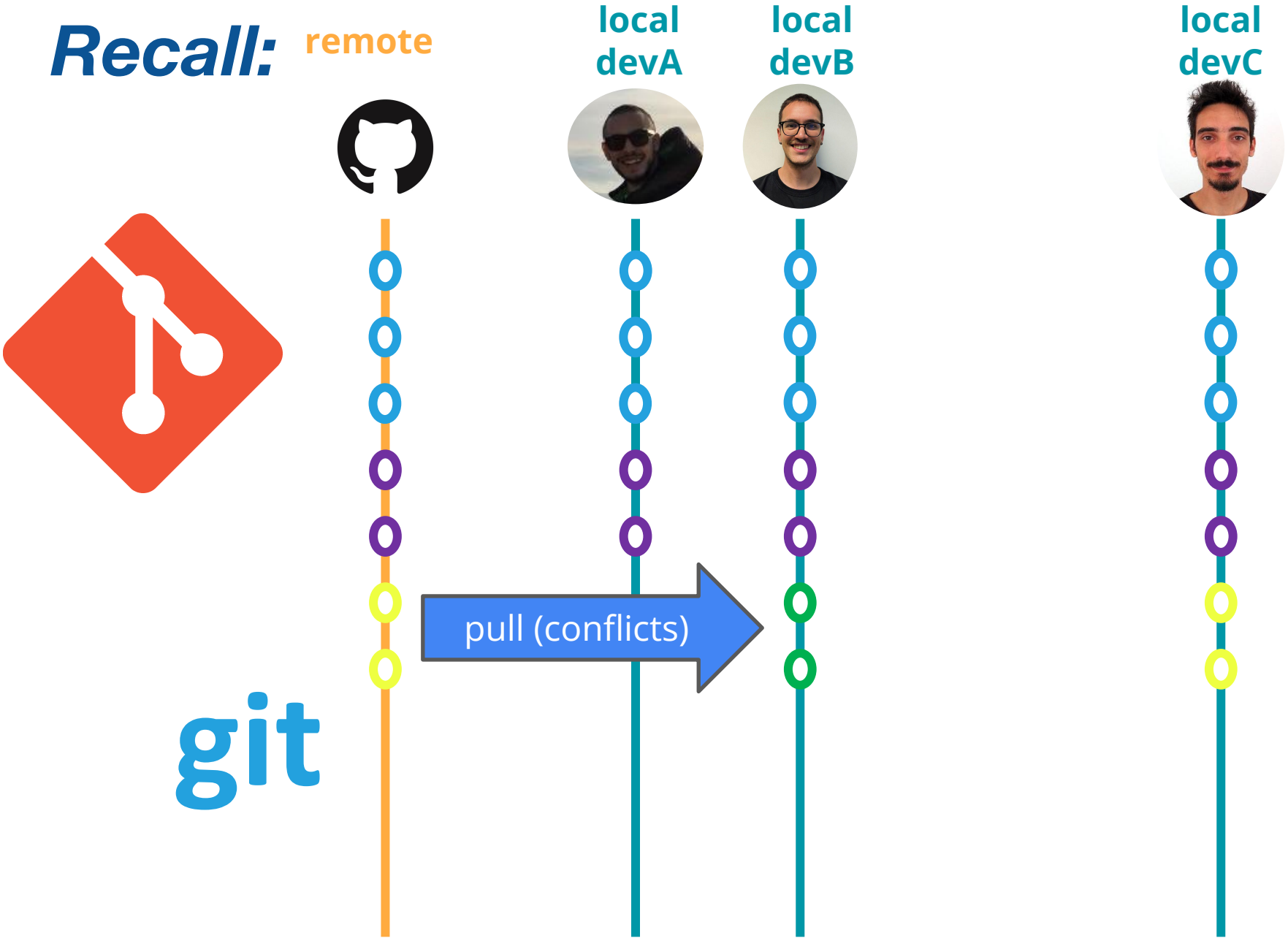


git

Recall:



Recall:



Recall:

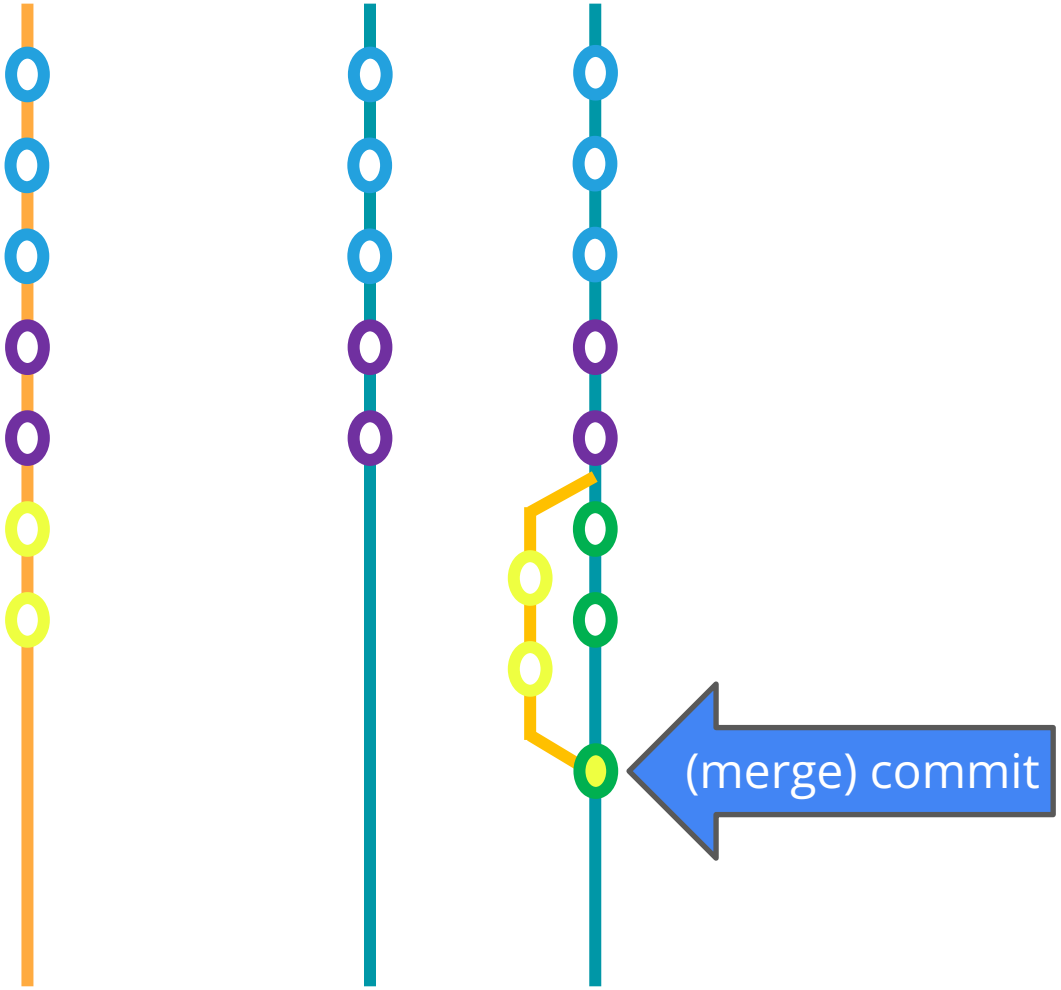
git

remote

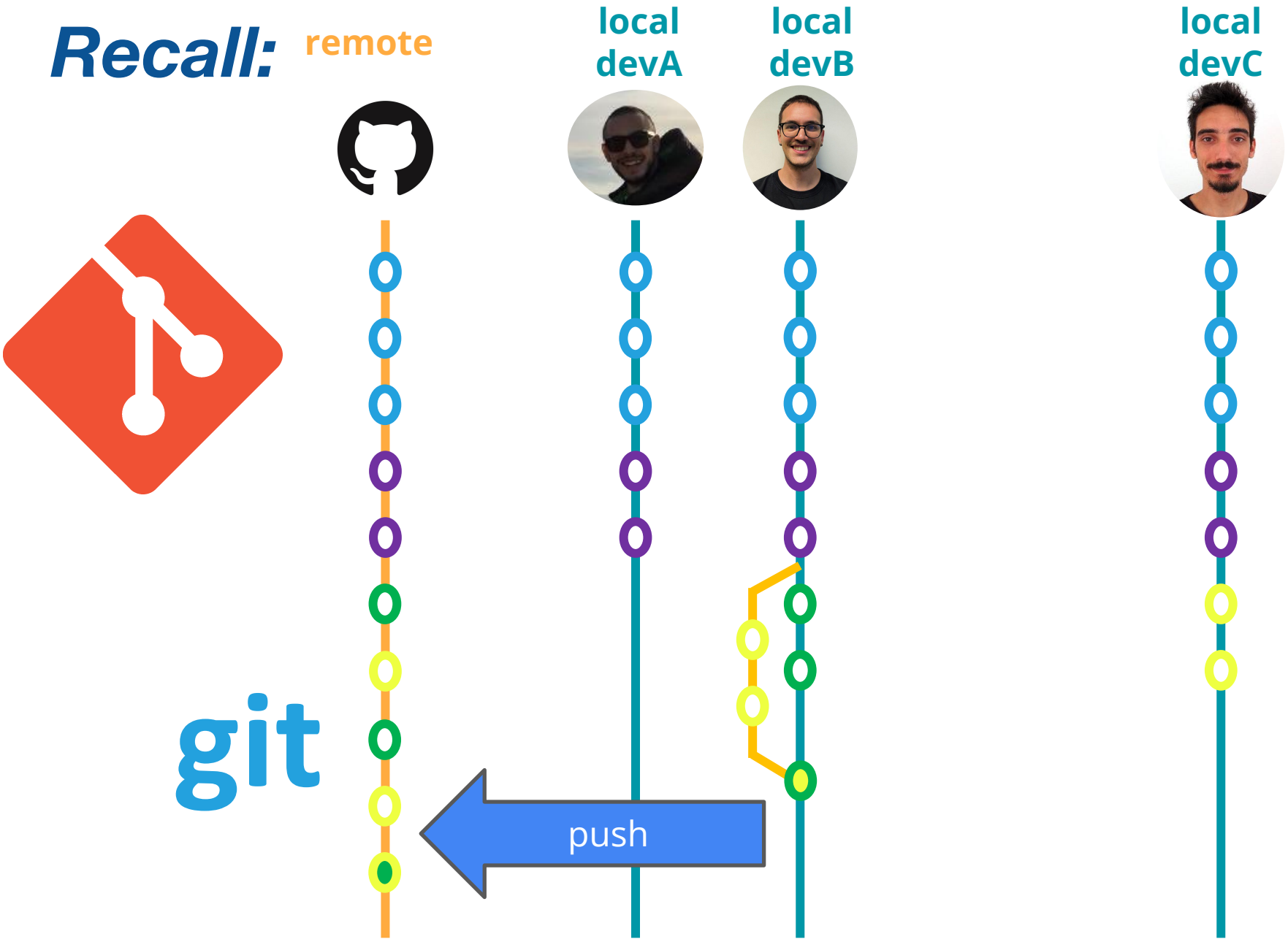
local
devA

local
devB

local
devC



Recall:

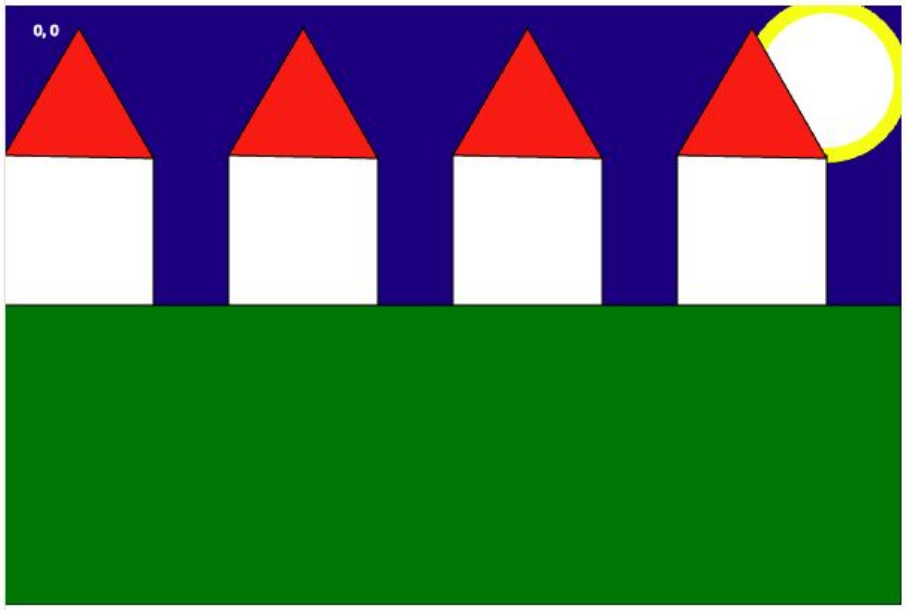


Recall: Lecture 2 Challenge

Prepare a drawing of something you would like to draw.
Example below.

USE VARIABLES, LOOPS and Arithmetics and whatsoever :D
(add the link to the tab:
https://docs.google.com/spreadsheets/d/1Ykz1MleWlT3YBjkBwBtLP5mb76Tm7z604qrcklUBCwU/edit?usp=drive_link)

Deadline: Monday 7th October 12:00



https://pixabay.com/photos/candle-light-the-background-fire-7562931/

Tempo dell'Appello!

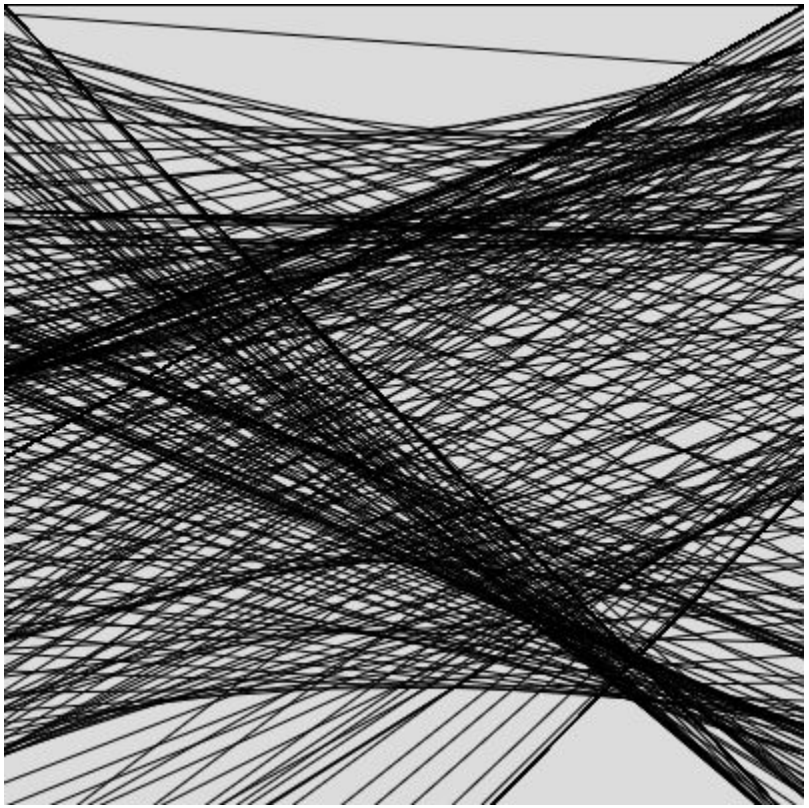
Image by Daud Siburian from Pixabay

Today Objectives

p5.js

1. ~~Recap + Challenge discussion~~
2. Response
3. Transformations
4. Random
5. Motion
6. Put everything live!

Follow the ~~SUN~~ mouse!

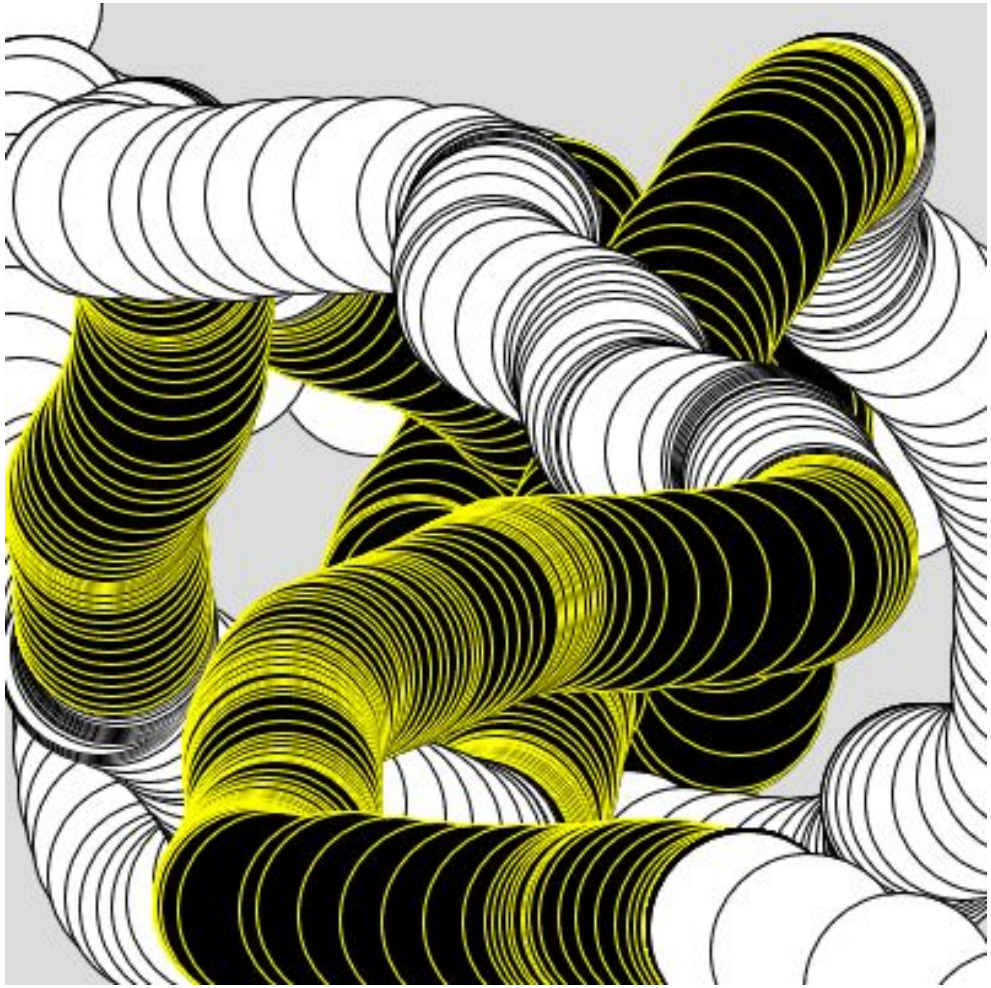


Follow the ~~SUN~~ mouse!

```
let canvasXMax=400;
let canvasYMax=400;
function setup() {
 createCanvas(canvasXMax, canvasYMax);
 background(220);
}

function draw() {
 //mouseX keeps track of the mouse's position relative to the top-left
 //corner of the canvas. and so mouseY
 line(0, mouseX, canvasYMax, mouseY);
}
```

~~DON'T~~ Try this at Home

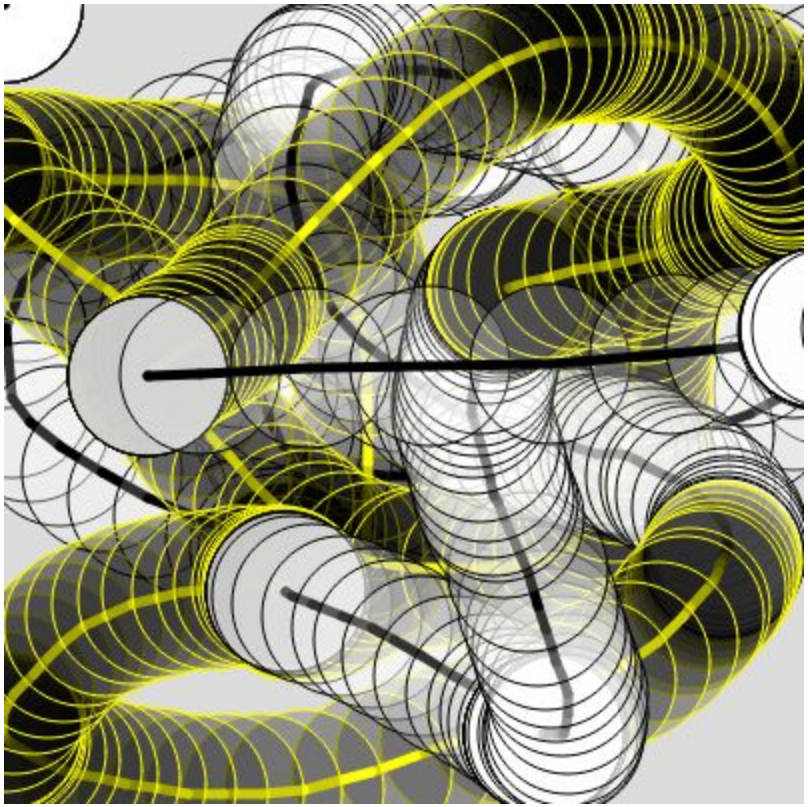


~~DON'T~~ Try this at Home

```
function setup() {
   createCanvas(400, 400);
   background(255);
 }
  function draw() {
   circle(mouseX, mouseY, 80);
 }
```


https://p5js.org/tutorials/setting-up-your-environment/

Follow the ~~SUN~~ mouse!

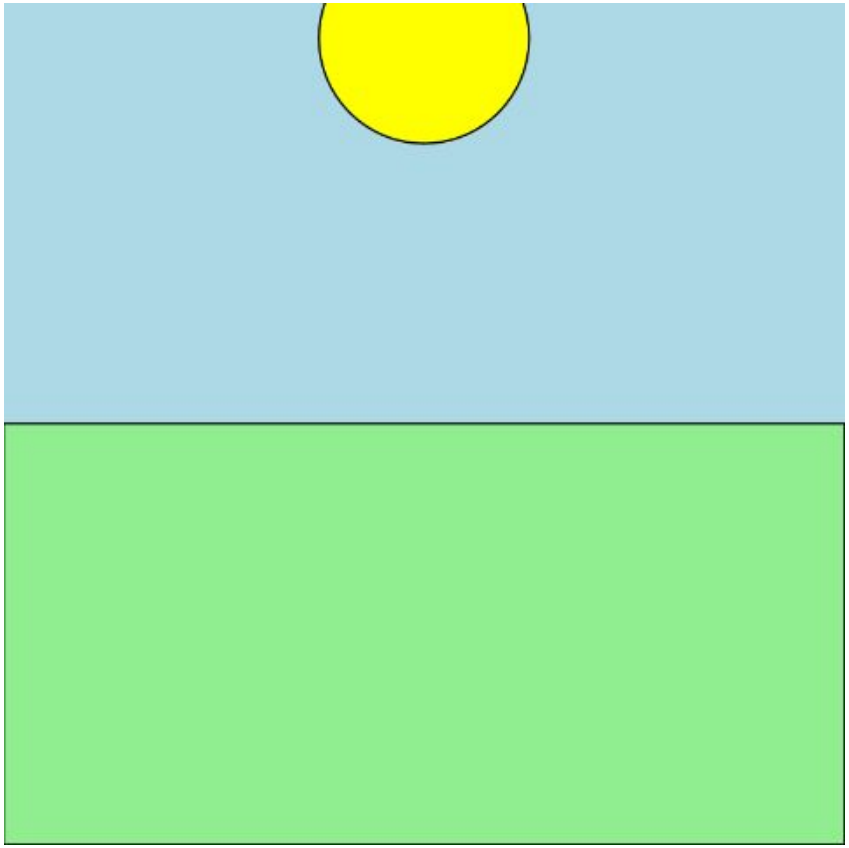


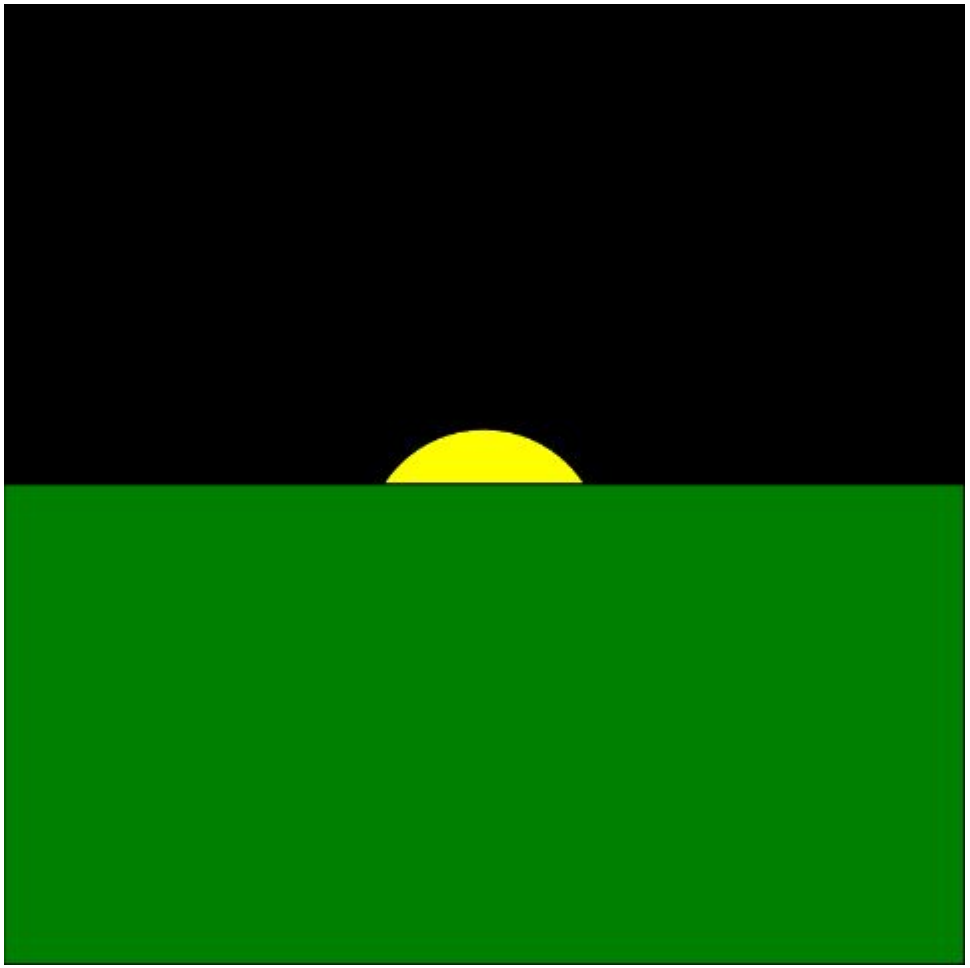
Follow the ~~SUN~~ mouse!

```
let canvasXMax=400; let canvasYMax=400;
function setup() {
 createCanvas(canvasXMax, canvasYMax);
 background(220);
}

function draw() {
 strokeWeight(1);
 circle(mouseX, mouseY, 80);
 if (mouseIsPressed === true) {
   fill(0, 50);
   stroke("yellow");
 } else {
   fill(255, 50);
   stroke("black"); }
 strokeWeight(5);
 line(mouseX, mouseY, pmouseX, pmouseY); }
```

Follow the ~~SUN~~ mouse (p5 editor)





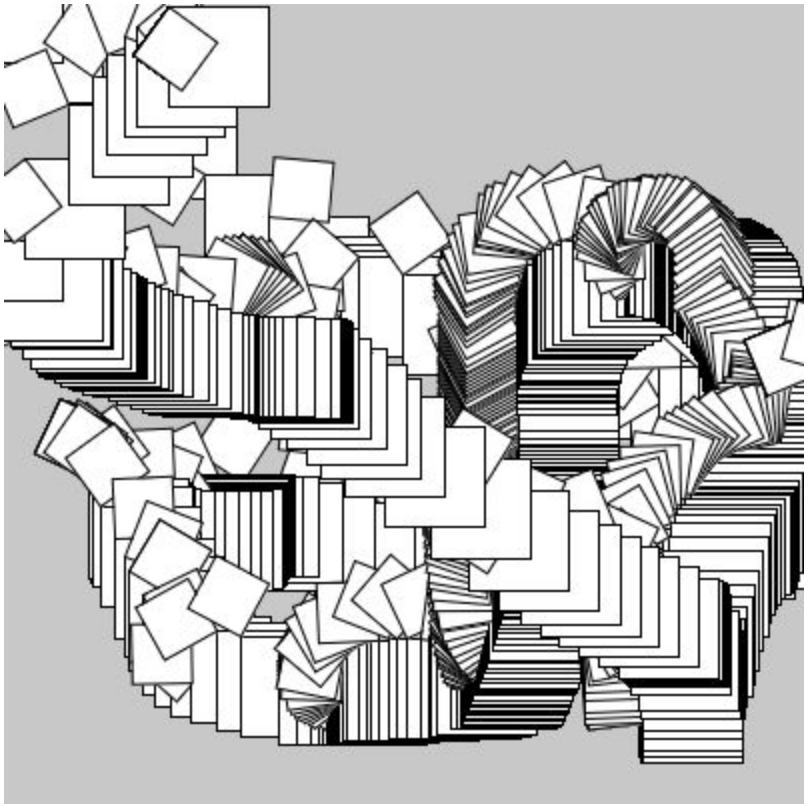
https://editor.p5js.org/gbenedis@gmail.com/sketches/nNVmHVf5m/

More on the reference!

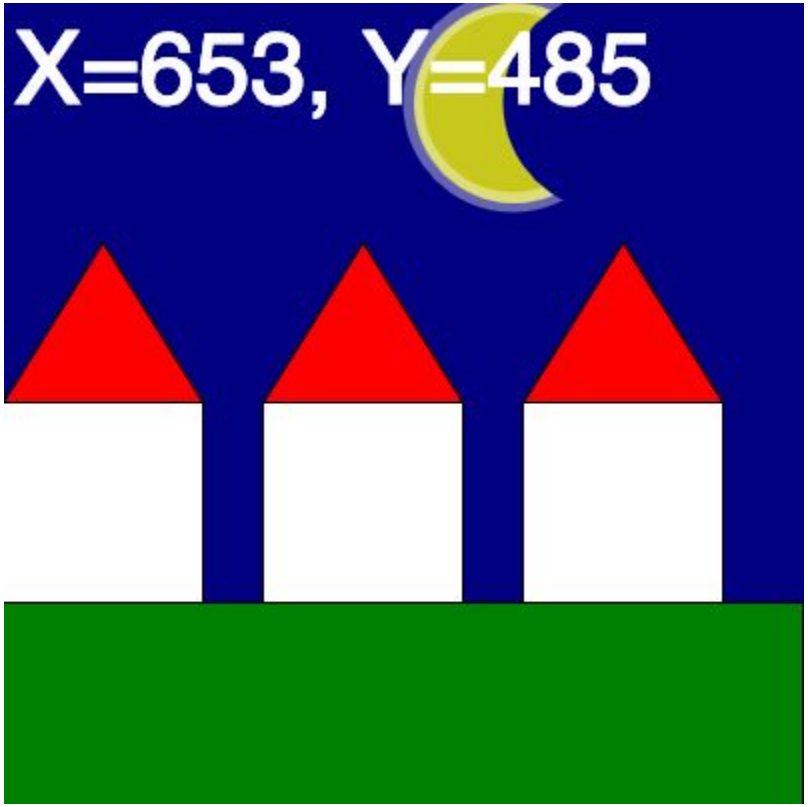
https://p5js.org/reference/#Events

- **Acceleration**
- **Keyboard**
- **Mouse**
- **Touch**

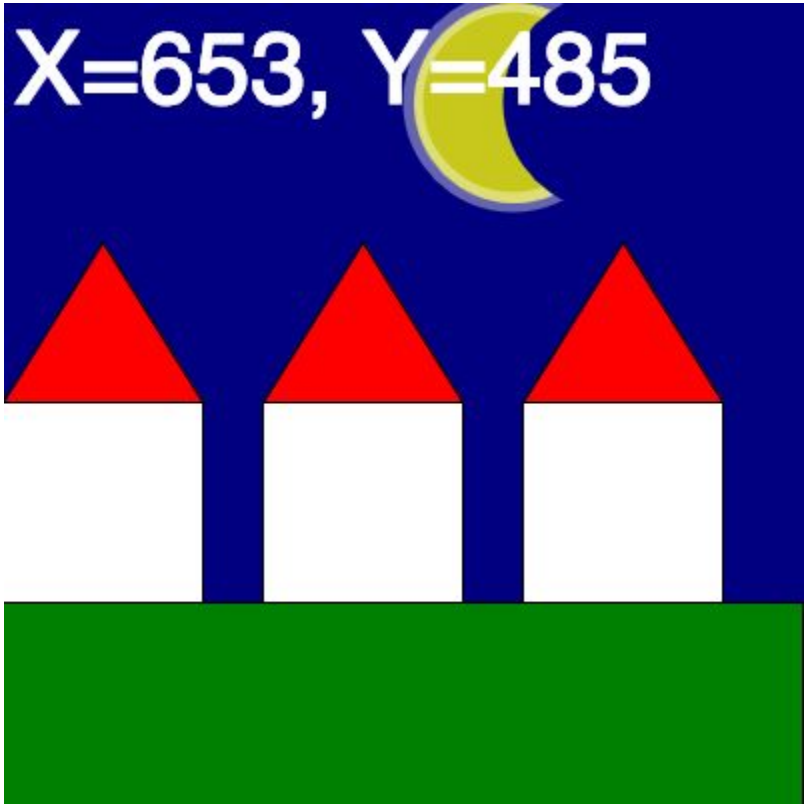
Can we do some Geometry?

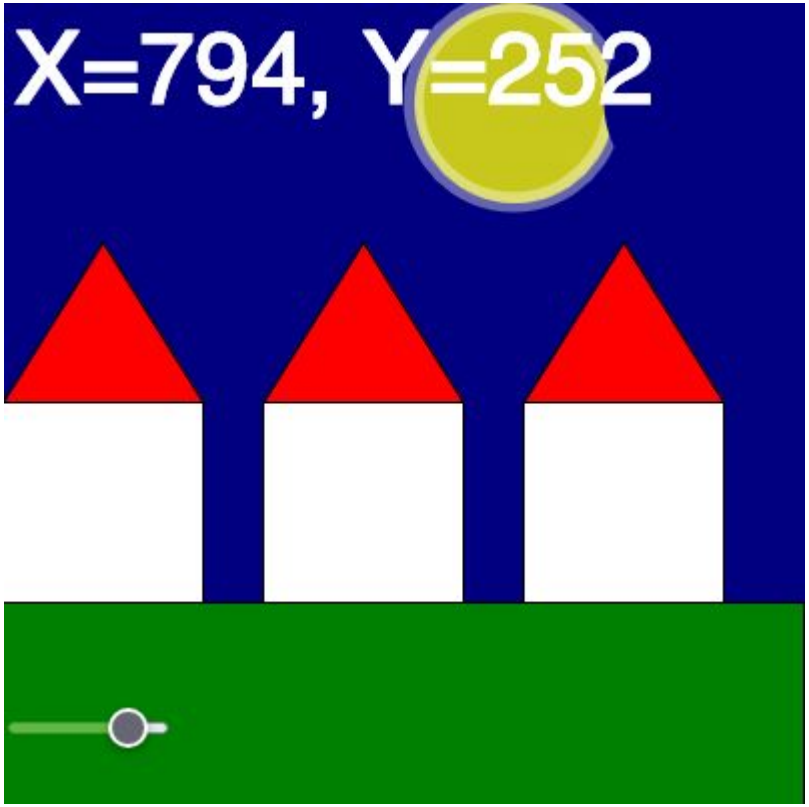


From the simple houses….



From the simple houses….To the growing moon





More on the reference!

https://p5js.org/reference/#Transform

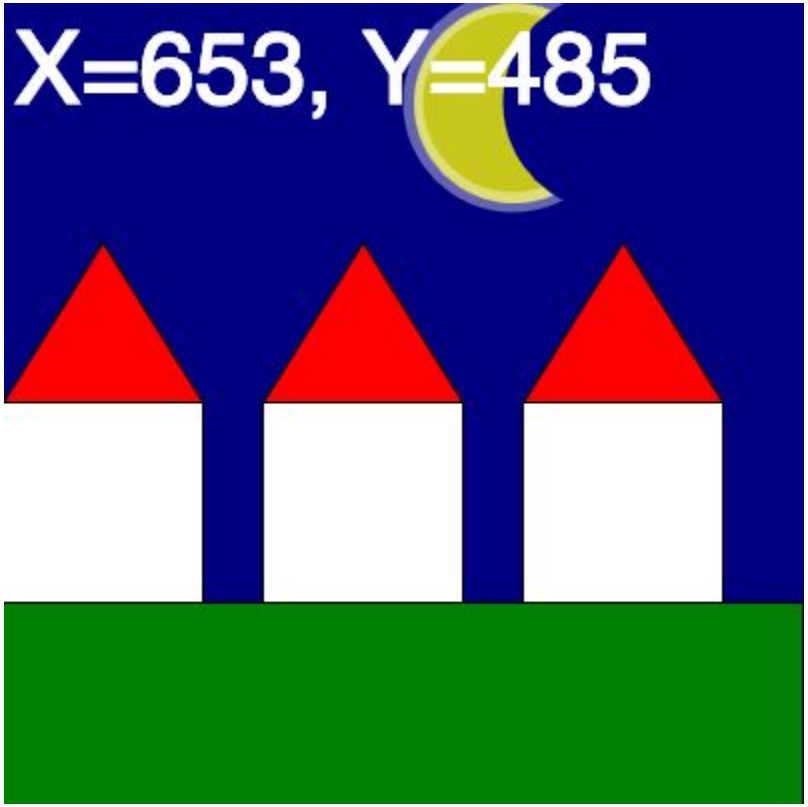
https://p5js.org/examples/transformation-translate/

https://p5js.org/examples/transformation-rotate/

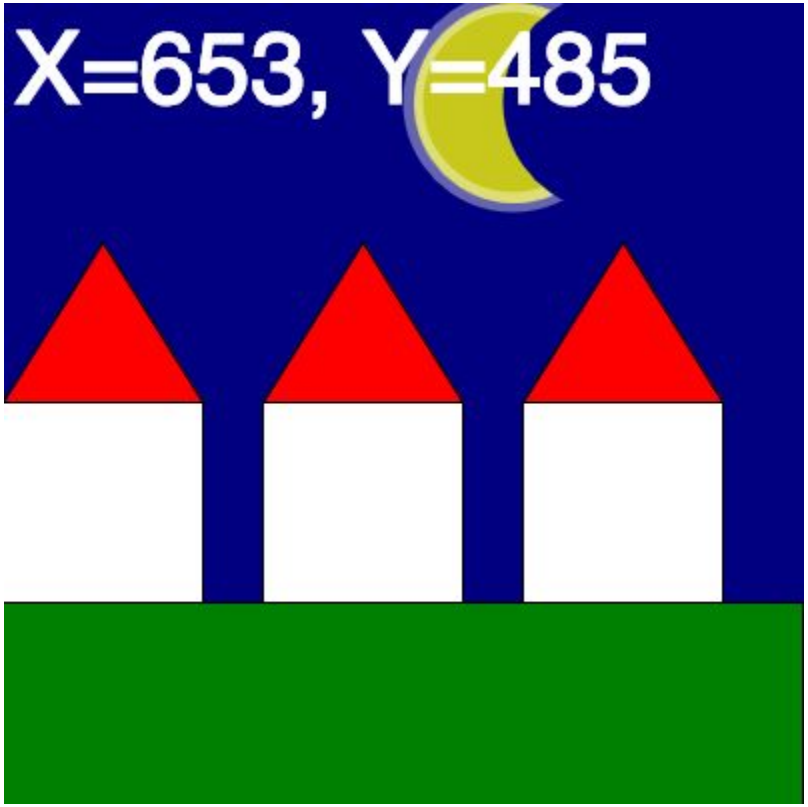
https://p5js.org/examples/transformation-scale/

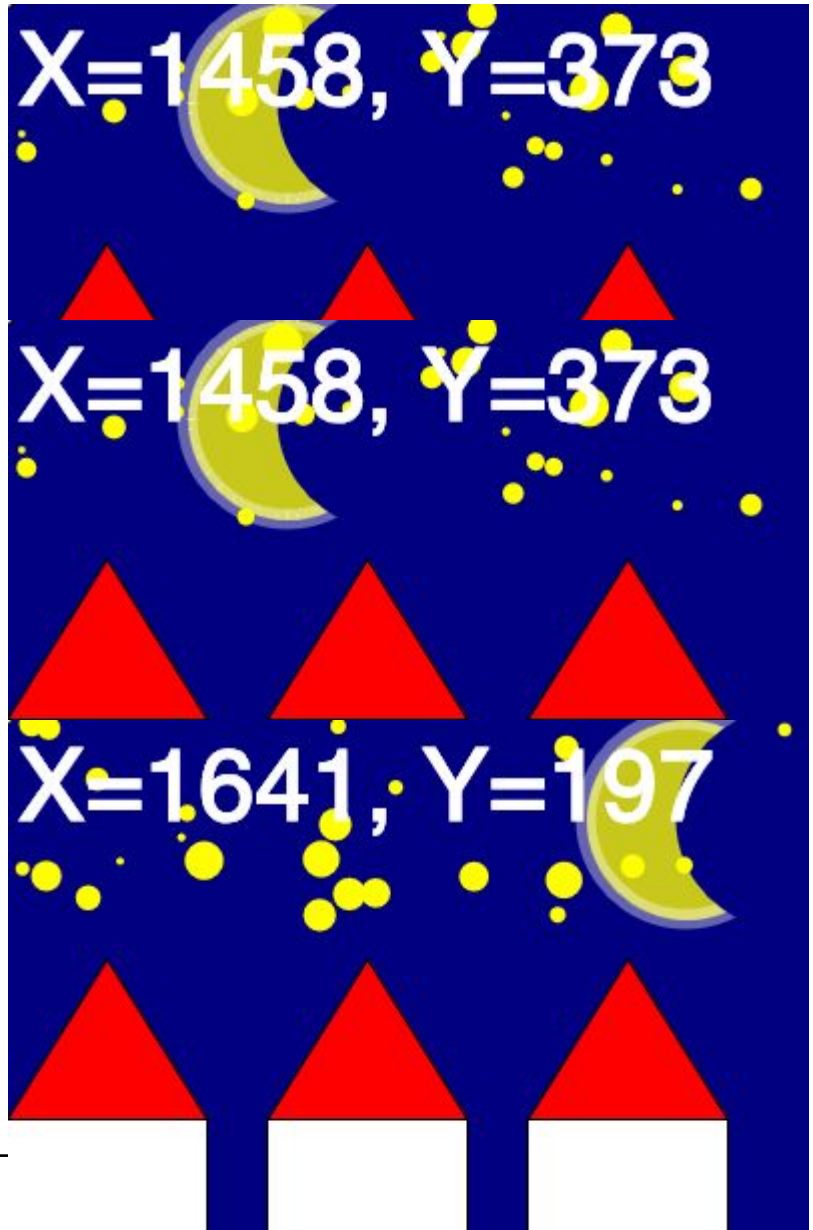
https://editor.p5js.org/DavideConficconi/sketches/v2k2k0w1q

From the simple houses….



From the simple houses…. To the random Stars!





Moving Moon…

```
let moonXpos=255;
let moonYpos=50;
//... ->within draw()
 background("navy");
 //change the framerate of the drawing
 frameRate(15); //set frame rate to 15
 //moon
 circle(moonXpos, 50, 100)

 //overlappin circle
 circle(moonXpos+50, 50,100);
 moonXpos = frameCount%canvasXMax;
 moonYpos = frameCount%canvasYMax;
```

- `frameCount` is a built-in variable that saves the number of times `draw()` runs. This value continues to increase as long as your program is running.

See this example to view the values stored in `frameCount`.

Random Stars

```
let xStar=0;
let yStar=0;
let strokeWeightStars=3;
for(let numStars=0; numStars<25; numStars++){
  stroke("yellow");
  strokeWeight(strokeWeightStars);
  point(xStar, yStar);
  xStar=random(0,width);
  yStar=random(0,height-y);
  strokeWeightStars=random(3,20);
}
```

`width` is a built-in variable that stores the width of the canvas defined in `createCanvas()`. We can see that in this example, the `width` is 400 and the `height` is 400.

https://p5js.org/reference/p5/random/

Today Objectives

p5.js

1. ~~Recap + Challenge discussion~~
2. ~~Response~~
3. ~~Transformations~~
4. ~~Random~~
5. ~~Motion~~
6. Put everything live!

Let’s put everything on github!

Image by Mariusz from Pixabay

Today Objectives

p5.js

1. ~~Recap + Challenge discussion~~
2. ~~Response~~
3. ~~Transformations~~
4. ~~Random~~
5. ~~Motion~~
6. ~~Put everything live!~~

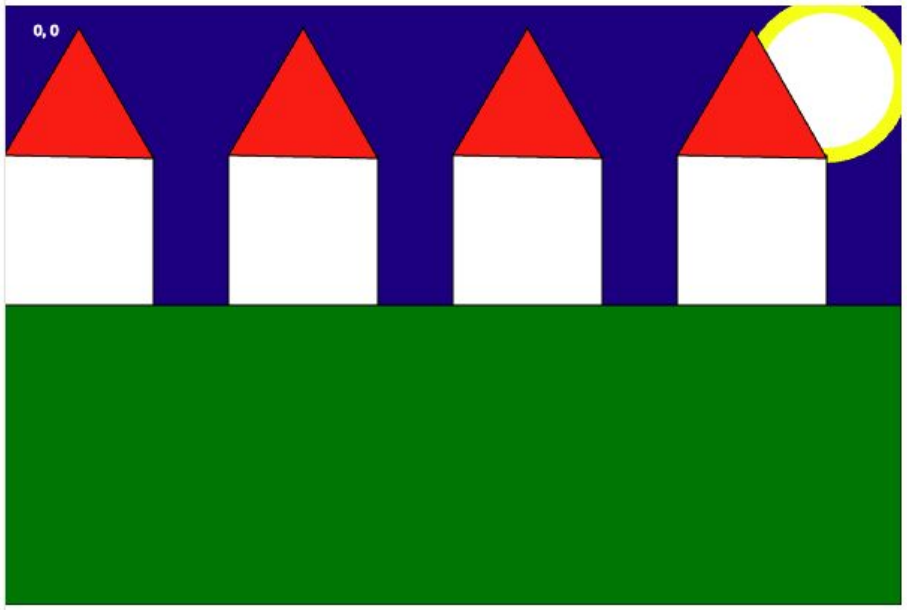
Lecture 3 Challenge

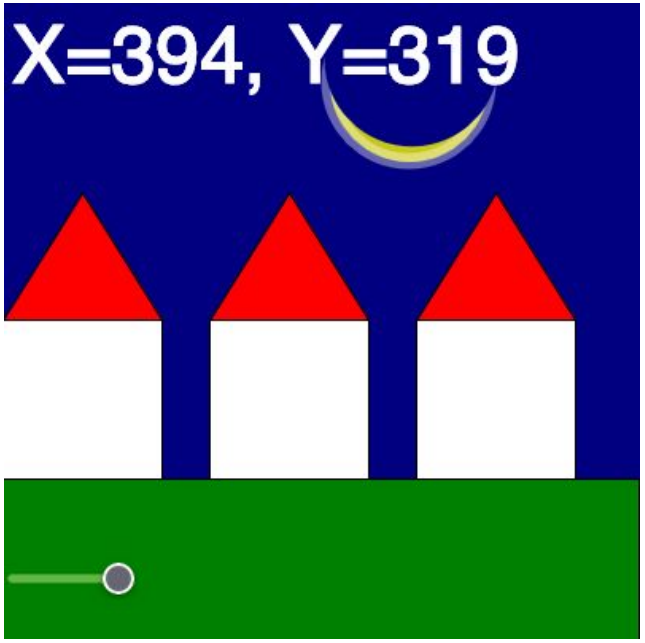
Now let's animate the drawing you created before (or start from scratch if you prefer)

Use: input-driven transformations and random data generation (Be careful of the ASSIGNMENT)

(add the link to the NEW tab: https://docs.google.com/spreadsheets/d/1Ykz1MleWIT3YBjkBwBtLP5mb76Tm7z604qrcklUBCwU/edit?usp=drive_link)

Deadline: Monday 28th October 12:00





Thank you for your attention

Davide Conficconi <davide.conficconi@polimi.it>
Alessandro Nazzari <alessandro.nazzari@polimi.it>

Acknowledgements

Everything already cited in the slides
Part of this material comes from:

- LCG- IA 22-23; 23/24 edition and their corresponding Acks, especially I. Di Dio Lavore
- https://thecodingtrain.com
- https://p5js.org/tutorials/
- The book on the right :)