WEEK 14 – P5.JS

WEN-BIN JIAN

DEPARTMENT OF ELECTROPHYSICS, NATIONAL CHIAO TUNG UNIVERSITY

OUTLINE

BACKGROUND STORIES ABOUT P5.JS

- A JavaScript library that starts with the original goal of processing. https://p5js.org/
- p5.js has addon libraries for accessing html5 objects of text, input fields, video, webcam, sound, and serial port.
- The same goal for different languages are listed:
  - Processing p5.js
  - Processing.py
  - Processing for Android
  - Processing for Pi
- Processing
  - Developed by Ben Fry and Casey Reas in John Maeda's Aesthetics and Computation Group at MIT Media Lab since 1997.
  - It has grown up within a set of other institutes including UCLA, Carnegie Mellon, and Harvard.

Ref1:

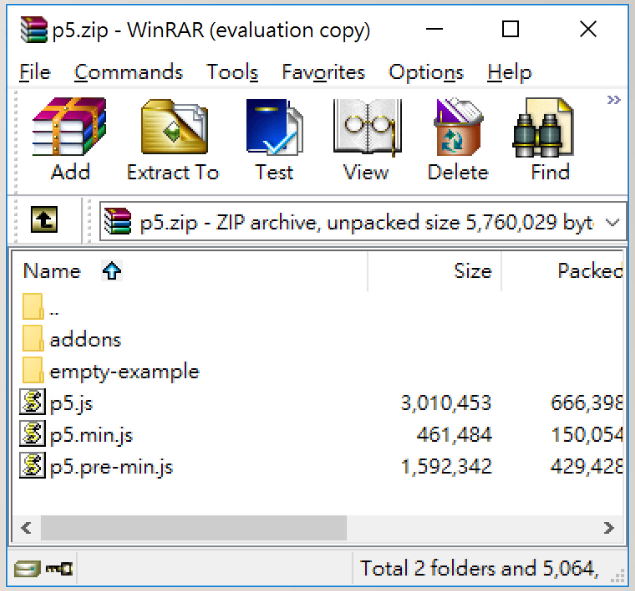
BACKGROUND STORIES ABOUT PROCESSING

- Processing relates software concepts to principles of visual form, motion, and interaction.
- Integrate programing language, development environment, and teaching methodology into a unified system
- Features:
  - Free to download and open source
  - Interactive programs with 2D, 3D or PDF output
  - OpenGL integration for accelerated 3D
  - For GNU/Linux, Mac OS X, and Windows
  - Over 100 libraries extend the core software
  - Well documented, with many books available
- Processing is a software for learning object-oriented programming.

Ref1:

INSTALL P5.JS

- Download p5.zip from https://p5js.org/download/support.html
- Files:
  - p5.js – full uncompressed version
  - p5.min.js – compressed version
  - addons\p5.dom.js & addons\p5.sound.js
- Web editor: https://editor.p5js.org/
- Put all the files in a folder like p5 together with your html files.
  - p5\p5.js
  - p5\p5.min.js
  - p5\addons\p5.dom.js & p5\addons\p5.sound.js



p5.zip - ZIP archive, unpacked size 5,760,029 byt

Name	Size	Packec
..		
addons		
empty-example		
p5.js	3,010,453	666,39ɛ
p5.min.js	461,484	150,054
p5.pre-min.js	1,592,342	429,42ɛ

Total 2 folders and 5,064,

USE THE WHOLE HTML AS A SKETCH

- In the HTMl file
  - <style> body{ padding:0; margin:0; } </style>
  - <script src="p5/p5.js"></script> <script src="sketch1.js"></script>
- In the sketch1.js file
  - function setup(){
  - createCanvas(800,600); // prepare a canvas with a size of 800 x 600 pixels
  - background(255); // fill the canvas with white color, used to clear screen
  - rect(0,0,799,599); // draw the boundary rectangle of the canvas}
  - function draw(){} // this is a redraw function that updates 60 times in a second

IC_W1401.html

```
var x=0, y=0, vx=2, vy=4;
function setup(){
  createCanvas(800,600);
}
function draw(){
  rect(0,0,799,599);
  rect(x,y,50,50);
  x+=vx;
  y+=vy;
  if(x<=0 || x>=750) vx=-vx;
  if(y<=0 || y>=550) vy=-vy;
}
```

START UP A P5 SKETCH IN A CONTAINER(DIV)

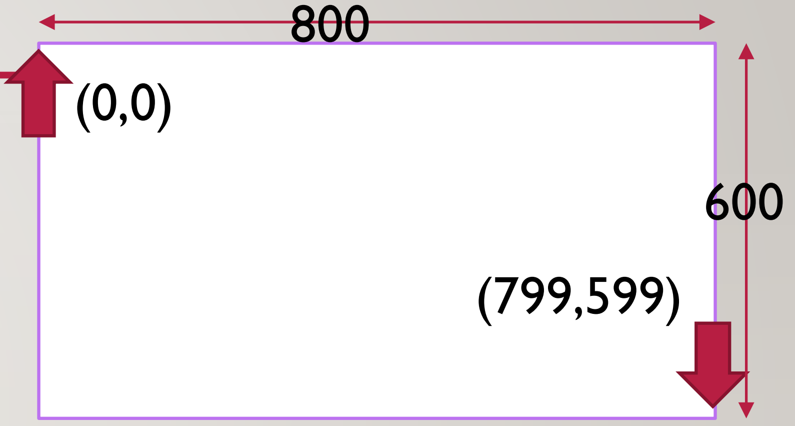
- p5 is an object that shows a canvas using for drawing 2D and 3D images, and for animation.
- Load the p5 library in the head of your html file:
  - <script src="p5/p5.js"></script>
- You can start to overwrite the functions of the p5 object.
- The first step is to rewrite the "setup" function.
- Use createCanvas(width, height) to create the element of p5 canvas.
- Put the html element inside a container such as a div element.

IC_W1402.html

```
<html>
<head>
<script src="p5/p5.js"></script>
</head>
<body>

<div id="p5_container"></div>

<script>
function setup()
{
    var cvs = createCanvas(640, 480);
    cvs.parent("p5_container");
    background(225, 225, 225);
}
</script>

</body>
</html>
```

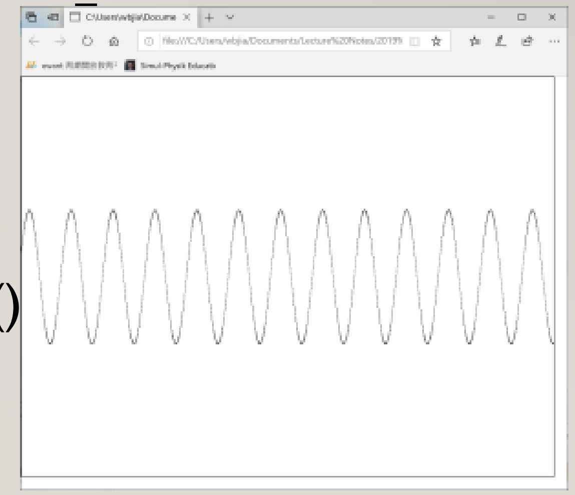
THE COORDINATE OF THE CANVAS

- createCanvas(800,600);
  - The size of the canvas is: width: 800 pixels, height 600 pixels.
  - One pixel is one physical point, a dot, in the picture.
  - The most upper left position is x=0 and y=0 (pixel).
  - The most lower right position is x=799 and y=599 (pixel).
- Color: either black white (BW) or full colors with red, green, and blue (RGB).
  - background(0) - set the full canvas to black color; background(255) set to white
  - stroke(BW) or stroke(R, G, B) to set the color of the drawing pen
- point(x,y) - draw a point in the color of the pen
- Usually you put background() and some drawing commands like point() in the draw() function



You clear the canvas and draw again on different position for making animation.

IC_W1403.html



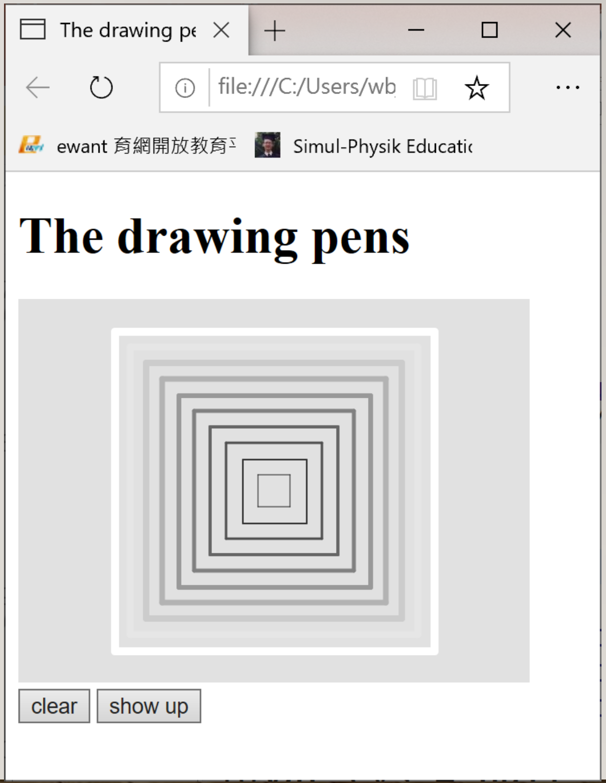
SOME MAIN FUNCTIONS NEED TO BE OVER-WRITTEN

- The p5 canvas is for animation. Similar to the p5 canvas and processing sketch, Arduino language also use similar rules for programming.
- For all these kinds of languages, you need to rewrite important functions:
- function preload() {}, prepare all images or calculations before you start your canvas
- function setup() {}, initialize your canvas
- function draw() {}, do animation by drawing canvas 60 times per second
- Other functions for controlling the animation:
  - remove(), noLoop(), loop(), push(), pop(), redraw()

CLEAR THE CANVAS AND SET THE DRAWING PEN

- clear() – clear all pixels, make canvas transparent, like background() & rect()
- **background(BW)** – use black/white level color to brush the canvas
- **background(R, G, B)** – clear the canvas and paint all with the RGB color
- **rect(x,y,w,h)** – if you set fill(255), the canvas will be painted in white color
- **A pen is used to draw point & boundary of shapes.**
- **noStroke()** – no drawing pens
- **strokeWeight(px)** – thickness of the drawing pen
- **stroke(BW), stroke(R, G, B)** – set the color of the drawing pen,

IC_W1404.html



The drawing pens

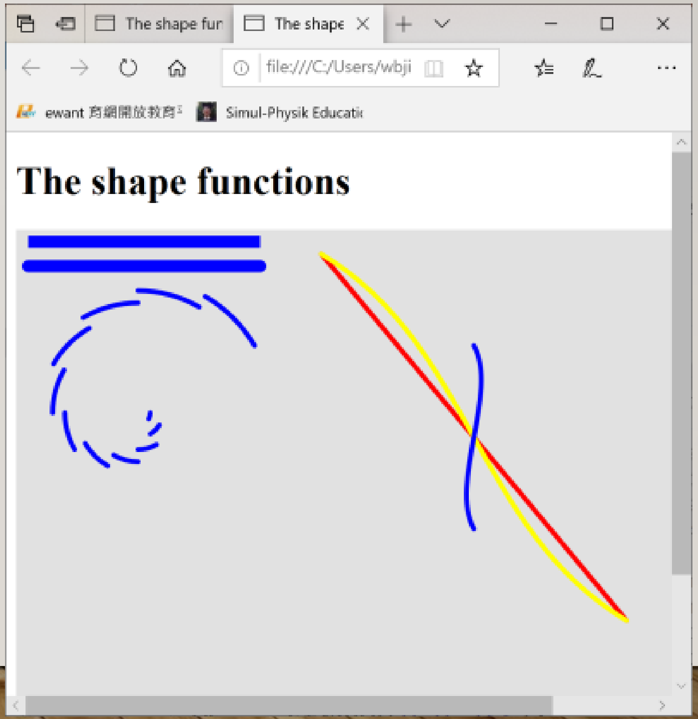
OUTLINE

1. p5.js & processing – background
2. Install p5.js & start a p5 canvas
3. Canvas attached to a container
4. Animation
5. Rewrite 3 main functions
6. Clear background & Se pen styles
7. **Draw lines/curves**
8. **Set brushe colors & draw shapes**
9. **Typography functions**
10. **Transform – Translate, Rotation, Scale**
11. **Shear, Matrix / Draw Images**
12. **Events & Animation**

THE SHAPE FUNCTIONS – LINES & CURVES

- strokeCap(SQUARE) – set the shape of the two ends of a line segment
- **point(x, y)**
- **line(x0, y0, x1, y1)**
- **arc(cntr_x, cntr_y, width, height, start angle, end angle)**
- bezier(x1, y1, x2, y2, x3, y3, x4, y4)
- curve(begin_ctrl_x, begin_ctrl_y, x1, y1, x2, y2, end_ctrl_x, end_ctrl_y)

IC_W1405.html

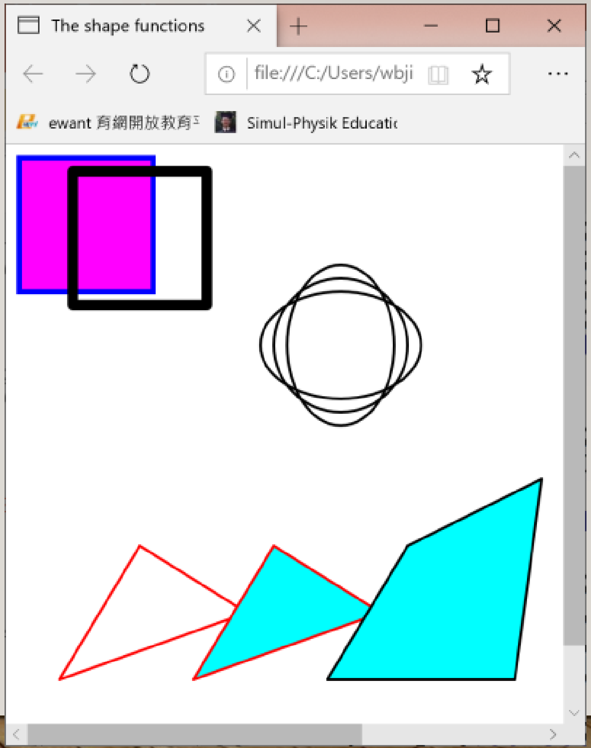


The shape functions

THE SHAPE FUNCTIONS – RECTANGLES, ELLIPSE, …

- **A brush is used to paint the bounded area in a specified color.**
- **noStroke()** – no boundary lines
- **noFill()** – no painting brushes, fill(BW), fill(R, G, B) – set the color of the painting brush
- **rect(x, y, w, h)**
- **ellipse(x, y, w, h)**
- **triangle(x1, y1, x2, y2, x3, y3)**
- **quad(x1, y1, x2, y2, x3, y3, x4, y4)**

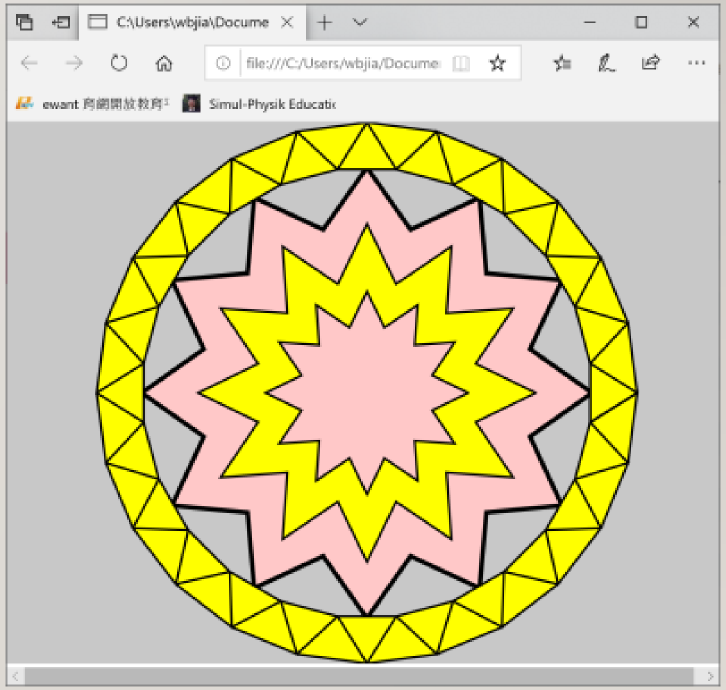
IC_W1406.html



THE SHAPE FUNCTIONS – USER DEFINED SHAPES

- strokeJoin(ROUND) – set the intersection area of the joined lines, used for **beginShape() and vertex()** commands
- beginShape(); vertex()…, endShape(CLOSE);
- beginShape(TRIANGLE_STRIP); beginShape(QUAD_STRIP);
- beginShape(); … beginContour(); … endContour(); … endShape(CLOSE);

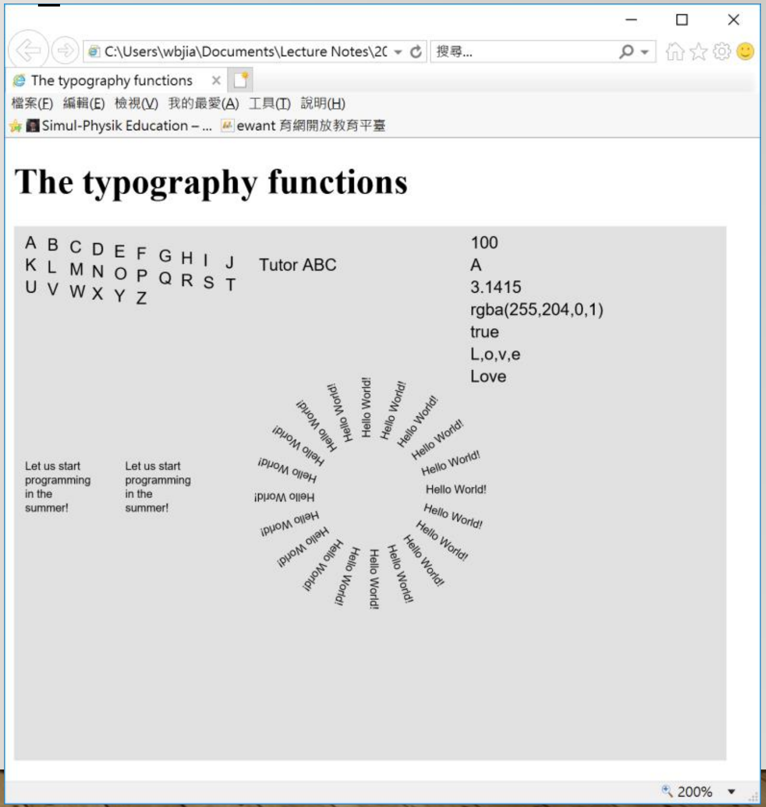
IC_W1407.html



TYPOGRAPHY FUNCTIONS

- **text(str, x, y)** or text(str, x0, y0, x1, y1), write a string on x, y or in a box confined in x0, y0, x1, y1 – here fill(R, G, B) is the color of the string
- **textFont**('Georgia'); textFont("font_name")
- textWidth('A') – return the width of the char
- **textSize(n)** – set the size of the font used
- **textStyle(ITALIC)** – NORMAL, ITALIC or BOLD
- **textAlign**(horizAlign, [vertAlign]) – horizAlign (LEFT, CENTER, or RIGHT) and vertAlign (TOP, BOTTOM, CENTER, or BASELINE)
- textLeading(leading) or textLeading() – set or read the spacing, in pixels, between lines of text

IC_W1408.html


C:\Users\wbjia\Documents\Lecture Notes\2(
搜尋...
The typography functions
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
Simul-Physik Education – ...
ewant 育網開放教育平臺
200%

The typography functions

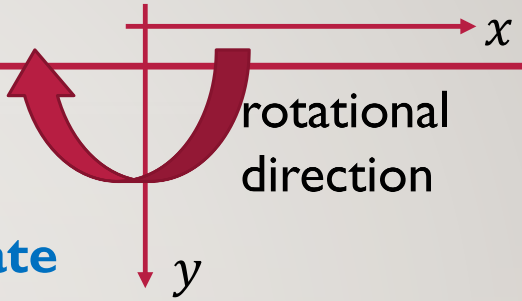
```
A B C D E F G H I J    Tutor ABC
K L M N O P Q R S T
U V W X Y Z
```

```
100
A
3.1415
rgba(255,204,0,1)
true
L,o,v,e
Love
```

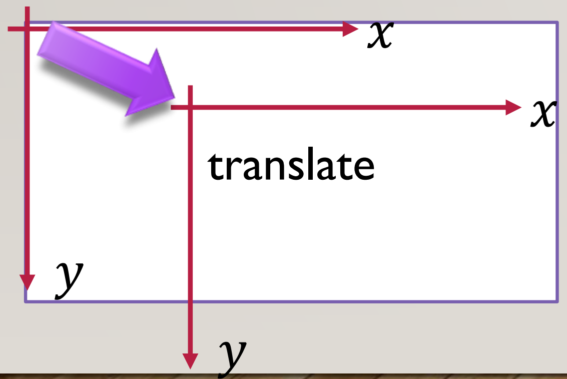
```
Let us start      Let us start
programming       programming
in the            in the
summer!           summer!
```

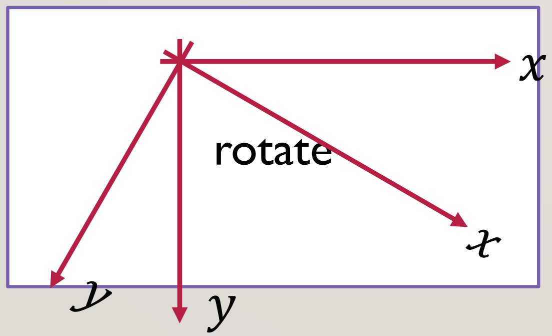
Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World! Hello World!

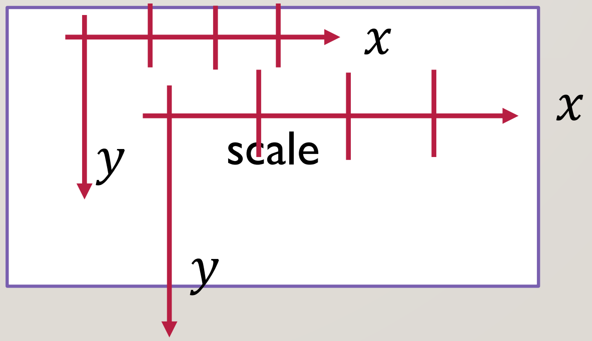
TRANSFORM – TRANSLATE, ROTATE & SCALE



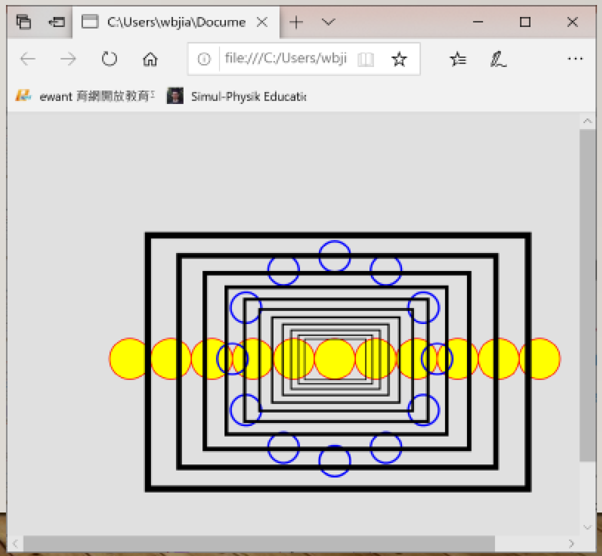
- **push() – store the coordinate information**
- **pop() – pop out & go back to the stored coordinate**
- **translate(x, y)**, move origin to (x, y)
- **rotate(angle)**, rotate along the z axis pointing to the screen
- **scale(n)**, scale up for n > 1.0 & scale down for 1.0 > n > 0





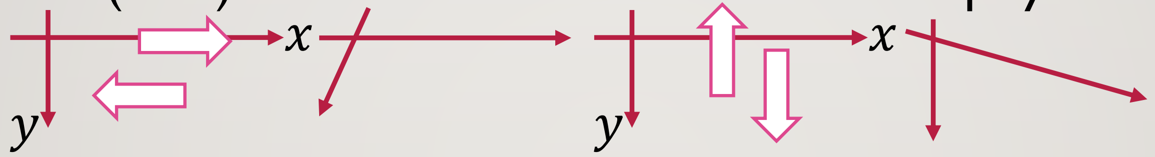


IC_W1409.html



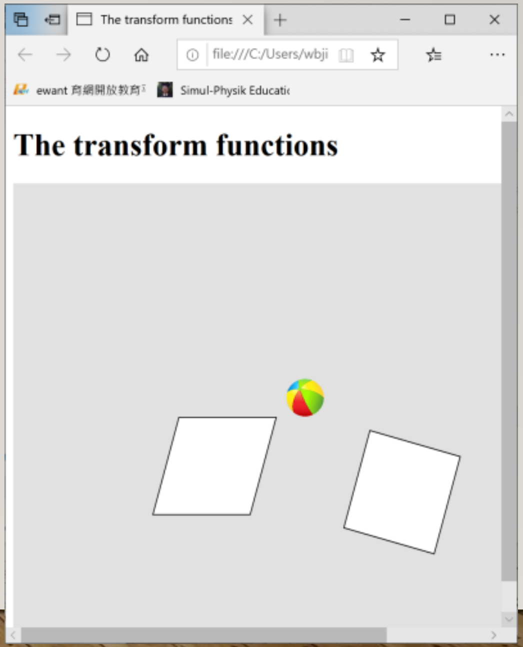
TRANSFORM – SHEAR, MATRIX DRAW IMAGES

- fullscreen(true)/fullscreen(false) // set the canvas to full screen display or not
- shearX(), shearY()



- applyMatrix(a, b, c, d, e, f) $\begin{bmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{bmatrix}$ $\quad x' = ax + cy + e$, $\; y' = bx + dy + f$
- resetMatrix()
- var img = loadImage("file_name"); // load image, img.width & img.height
- image(img, x, y, width, height); // draw a jpg image at x, y
- imageMode(modes) – either CORNER, CORNERS, or CENTER

IC_W1410.html

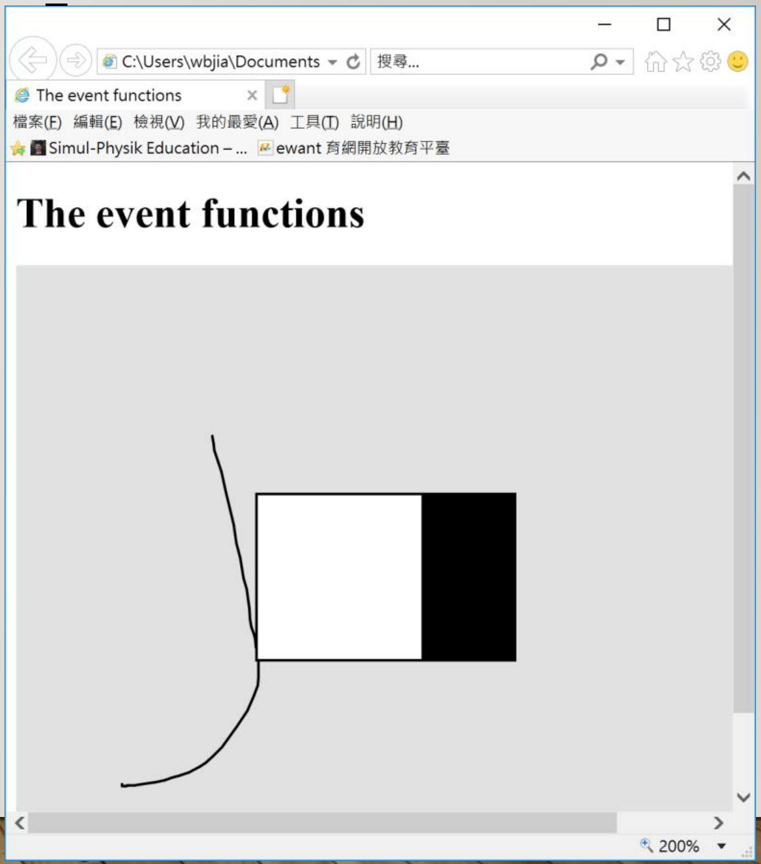


The transform functions

EVENTS

- function **mouseClicked()** {}
  - **mouseDragged()**, mousePressed(), mouseReleased(), doubleClicked(), **mouseWheel()**
- function touchStarted(), touchMoved(), touchEnded()
- function **keyPressed()**
  - keyReleased(), keyTyped(), keyIsDown()
- function deviceMoved(), deviceTurned(), deviceShaken()
- setMoveThreshold(), setShakeThreshold(), …
- Attributes:
  - mouseX, mouseY, mouseIsPressed, keyIsPressed, acceleration, …

IC_W1411.html


C:\Users\wbjia\Documents
搜尋...
The event functions
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
Simul-Physik Education – ...
ewant 育網開放教育平臺
200%

The event functions

ANIMATION

- The draw function will be called every one sixtieth second.
- Rewrite the draw function for animation: function draw(){…}
- Redraw the canvas so you can see the moving feature.
- **frameRate(n)** – set the number of frames per second
- frameCount – the number of frames in animation, deltaTime – time difference previous and current frame
- **saveFrame**(filename, extension, period, framerate)
- **saveCanvas**(selCanvas, filename, ext)

IC_W1412.html

```
function draw(){
	background(0);
	var x = 20;
	while(x > 0){
		x = x - 3 * Math.random();
		fill(255*Math.random(), 255*Math.random(), 
255*Math.random());
		rect(width * Math.random(), height * 
Math.random(), 20, 20);
	}
}
```

EXERCISE

1. Now you can write a game of projectile motion. Prepare some images or drawing on the p5 canvas and a fire button on html window. The direction of the projectile is changing from 30 to 90 degree. The player can press the button to fire the projectile to the right. There will be some gifts on the right. When the gifts are hit by the projectile, the player can win something that is shown on the p5 canvas.