Processing Documentation

- Lectures and Projects

QILEI GE
ZHDK IAD
1st Semester
18 - 28.09.13

DAY 1 - 17.09.13
DAY 2 - 18.09.13

Subjects:

- Processing background introduction
- Startup and drawing shapes
- Variables, Functions and Conditions
- Project: Create your own drawing-tool

Processing is a relatively user-friendly programming system. The result is rapidly shown. The application could be variable, for example, a quick sketch of the idea or further extension of the idea.

I always start with setup. At least to create a window where I will generate images on.
size () defines the size and measure of the window.
background() sets up the background colors by defining RGB parameters and transparency.
Calling a rectangular with certain degree of transparency allows the ghostly effects of previous draw.

There are commands to draw shapes (line, ellipse, rect, triangle...). In these commands, the parameters provide the flexibility such as the positions, size and etc. For example, using mouseX or mouseY in parameters could make shapes follow the mouse move: line(mouseX, mouseY, pmouseX, pmouseY)). beginShape() and endShape() could draw more complicated shapes. Other command for example sin() realizes other movement: ellipse(mouseX,mouseY,sin(ani)*100, sin(ani)*100). There are more in the library to be discovered.

Variables hold the values. The ones defined at the very top are global variables, which are used throughout the program. One example is an animation of an expanding circle. At the beginning of the program, a variable float i is setup. Within draw() function, I can give i value initial value 0 and i+=1 always adds 1. By this way, a non-stop expanding circle is drawn.

In some situation, we would like to stop the movement, for example the expansion of the circle. Here, conditions allow us to set up the limits and control the movements. For example, drawing a circle only when left mouse button is pressed and increasing the size; otherwise diameter of the circle returns to 0: if (mousePressed == true && i<=100) {ellipse(mouseX, mouseY, i, i);} else i=0;. Switch() is another condition type.

A function is a collection of commands. Function like draw() is run constantly called. But function like mousePressed() is only called when mouse is pressed. We can also create own functions. For example to create a general tree, called tree(). tree() runs when this function is called. By defining parameters of a function, such as tree(), we could have variant trees.

We started to create a drawing tool. Through this exercise, I practised on calling functions by certain buttons clicked or mouse clicked. The colors of strokes can be switched by button 1,2,3,4. 1,2,3 are RGB respectively, but 4 is a random color. Left clicked is to draw but right click is to remove. Keep pressing on button "r" calls out a constant change of colors. Button "l" calls out straight lines connecting point (0,0) and random point in the window. Button "d" is to delete all and come back to white background.

Link: http://blogs.iad.zhdk.ch/codingspace/2013/09/20/processing-day1-drawing-tool-17-09-13/

DAY 3 - 19.09.13

Subjects:

- Random
- Loop (for(), while())
- Project: Create a forest

Command random() could generate unexpected combinations. However, by setting up the parameters, the result could be limited in a desired range.

Today we also learned about loops, such as "for" and "while". One important trick to remember is add noLoop() in Setup. Otherwise, there will be constant running images.
for(int i=0; i <= 100; i++){statement}
int x=0; while(x <= 100){statement; x++}

The project is to create a forest.

For my first forest, I searched a tree image from the internet. After editing in PS, I loaded it into Processing. First step is to define the variables with datatype PImage. Command loadImage() images and imageMode() sets the cooridnate system 0 point. Another goal of the first forest is to apply scales according to position y, therefore, I defined scale(y/height).

For my second forest, I created a function tree1(). Whenever I want to draw a tree, I could call this function. I wanted to have variable colours of green leaves and brown tree trunks, so I use the random function to alter the fill.

For my third forest, I wanted to use the graphic tree I have created and also apply the scale. Plus, I would like to make the tree blinking except the tree trunk. In order to realise this, I will need tree positions and scales for all trees. I have to use arrays and assign position X and Y with random value and also the scale. The positions and scales are saved in the memory. Using pushMatrix and popMatrix, trees are generated according to the positions and scales.

Link: http://blogs.iad.zhdk.ch/codingspace/2013/09/20/processing-day3-loop-and-creating-a-forest-18-09-13/

DAY 4 - 20.09.13

Subjects:

- Coordinate System
- Recursive Function
- Project: based on the tree example, create other forms or objects with recursive function.

By moving or changing the Coordinate System (Cartisian), we could realise animation in a much easier way. Translate, rotate and scale are the basic three types of the transformation. pushMatrix() and popMatrix() are used to switch between current coordinate system and previous system.

Within the exercise "Smiley Face", the first change I made was to let the small face rotate around it's own center. By pushMatrix, the original coordinate system was shifted to the center of the window where the big face is. With popMatrix, the coordinate system went back to the initial position (0,0). The second pushMatrix defined the position of small face. Based on this understanding, if we want to make the small face rotate around the big face, small face and big face should share the same coordinate system. Therefore, the small face should be included within the pushMatrix() and popMatrix() of the big face. If I define the position for small face as (100,100), the small face will rotate with a radius of 100 around the center point. If I want the small face to rotate around its own center, I need to use pushMatrix and popMatrix again to shift the initial point from the center point to the position of small face. map() is used to convert a value from one range to another. For example, the same mouse position could generate variable positions of circles, but all related to the mouse position.

Recursive function is a function calling itself again, therefore it is important to define the termination point. Defining a variable such as count is for the number of branches generated at each point. The depth could be defined by the parameter of the recursive function. In this case, the stopping mechanism relies on the count and the depth variables.

For the project, I programmed two situations: one is an animation with a changing river; the other one is the baloons. The animation has four elements: sun, rain, river and oasis. In this exercise, I combined the elements we have learned such as recursive, changing cooridate system and loop.

The balloons are the transformation from recursive function. The challenge I would like to solve is to store the whole image at the memory and move coordinate system to make the balloons fly.

Link: http://blogs.iad.zhdk.ch/codingspace/2013/09/24/processing-day-4-20-09-13/

DAY 5 - 24.09.13

Subjects:

- Array and Array List
- SVG vs. images
- Vectors and Animation
- Project: Line animation

Array is a sery of variables with the same type. An array is static, so that the length of an array is defined and not be able to change. An array list is a dynamic array. It means that the length could be changed. When one of the "drawers" is taken away, the following "drawers" will move forward to fill up the spot.

Define array list, such as ArrayList<Integer> List. Commands such as add and remove could change the size of array list. The format is such as List.add() or List.remove(). They add or remove values. However, List.get() gets the value from certain position. For example, List.get(10) will get the value on the 10th position. Since the array list is dynamic, removing or adding a value will change the sequence of following values. Therefore, command get() will get different result before and after the array size changes.

We also talked about the difference between SVG vs. JPG vs. PNG. SVG is better for resolution while zooming, because SVG is a shape file. Therefore, when loading SVG file, I should use PShape instead of PImage. JPG and PNG are both image file but JPG lose information when compressing and PNG keeps the original image information. When load JPG and PNG, I should use PImage.

PVector is a vector class, which could be 2 or 3 dimensional. For example vec1.x refers to the x dimention of vec1. A vector name followed by "." refers to the sub dimension of this vector. Further, a vector name followed by ".add()" means adding onto the vector. vec1.add(vec2) means vec1 = vec1+vec2. Or we could also set up a new vector, such as vecnew=PVector.add(vec1+vec2). Vector length is vec1.mag(); angle between 2 vectors is degrees(PVector.angleBetween(vec1, vec2)). Vector normalize is to normalize the vector to length 1.

The project requires to draw a line between twice mouse click. A circle is drawn on the line and moving back and forth on the line.

For dawing the line, two boolean variables should be set up. With the first mouse click, first variable is turned on true. With the second click, second variable is turned on true. When both are true, the line in between is drawn. the circle is drawn too. The direction of the movement is defined by subtract start position from end position. The travel speed is defined by direction multiply with time but should add on start position. A direction switch is defined: when current time is higher than total animation time, the direction should be opposite; when current time is lower than 0, the direction should be opposite as well.

DAY 6 - 8
25 - 27.09.13

Subjects:

- Class
- BreakOut game initial codes
- End Project: Base on initial codes, create my own game

Classes are the most important programming contructs of Java. They are a model of an object. As a blueprint, a class defines the attributes of an object. It could include variables and functions. Classes could be extended. All the attributes are herited by the extension object, but any of them could be overwritten or new attributes could be added.

In the example of BallKlass, a ballklass is defined, which includes the position, shape and moving directions. In such way, it is easier to make the codes in a logical order. After ball klass is defined, the question is to design the interaction part. The reason for the balls to move is the point of mouse position. Therefore, in the interaction part, the direction is defined as the subtraction of current position from mouse position. The actions of mouse pressed, mouse dragged and mouse released give different orders.

As exercise, I did two changes to the bouncing ball game. First, instead of drawing only for ball1, I would like to draw lines for both ball1 and ball2. Second scenario is to draw line 1 to ball1, and with second click, draw line 2 to ball 2. And repeat the sequence.

For the end project, I took the theme of getting more candies on the plate. The original mechanism was not changed, however I extended to capture the candies. The candies' "Life" is extended from 1 life to 3 lives. First life is when they are all displayed in rolls in the window; second is when they are hit by the ball, they will fall downwards; third is when some of them drop within a range of the plate and they stay there.

Link: http://blogs.iad.zhdk.ch/codingspace/2013/09/30/endproject/