Introduction to Processing Contd..

Mithun Paul, CS345, Fall 2017

Resources

- Processing web site:
  http://www.processing.org/

- Reference: http://www.processing.org/reference/index.html

SETUP and DRAW

```
void setup()
{
      size(400, 400);
      stroke(255);
      background(192, 64, 0);
}
void draw()
{
      line(150, 25, mouseX, mouseY);
}
```

- The **setup()** block runs once, and the draw() block runs repeatedly.
- As such, setup() can be used for any initialization; in this case, setting the screen size, making the background orange, and setting the stroke color to white.
- The **draw()** block is used to handle animation. The size() function must always be the first line inside setup().

Things To Remember

The (0, 0) coordinate is the upper left-hand corner of the display window.

X Axis: From top Left Corner: Left to Right ----->
Y Axis: From top Left Corner: Top to Bottom

```
rect(10, 100, 50, 50);
rect(100, 10, 30, 30);
```

Interactive events

-

Active Sketch

- Most programs will employ active mode, which use the setup() and draw() blocks.

- More advanced mouse handling can also be introduced; for instance, the mousePressed() function will be called whenever the mouse is pressed.

mousePressed() function

- Called whenever the mouse is pressed
- Example

```
void setup() {
    size(400, 400);
    stroke(255);
}
void draw() {
    line(150, 25, mouseX, mouseY);
}
void mousePressed() {
    background(192, 64, 0);
}
```

Class Exercise 2

1. Draw a white circle with diameter 100 pixels in the center of a red window the size of the display.
2. Modify your sketch so that the circle changes color when you press the mouse.

More Tools For you

Variables

- variables provide a way to save information within your sketch and use it to control the position, size, shape, etc of what you are drawing
- variables have a data type, a name and a value
- valid data types are:
  - int — for storing integers (whole numbers)
  - float — for storing floating point (real) numbers
  - boolean — for storing true or false values
  - char — for storing single characters
  - String — for storing multiple (strings of) characters
- example:

```
int x1 = 10;
int y1 = 10;
int x2 = 20;
int y2 = 20;
line( x1, y1, x2, y2 );
```

- *Global: Define before setup*

Looping

- loops are used for doing things repeatedly
- there are two basic types of loops:
    - for loops
    - while loops
- loops are handy for animation, because you typically want to display things repeatedly when you are doing animation
- looping is a type of:
    - repetition (required element of imperative programming)
    - iteration (same thing as repetition)

for loops

- for loops repeat things for a fixed number of times
- syntax:

```
for ( init; test; update ) {
    statements
}
```

- example:

```
int x = 10;
int y1 = 10;
int y2 = 20;
for ( int i=0; i<10; i++ ) {
    line( x, y1, x, y2 );
    x = x + 10;
}
```

while loops

- while loops repeat things as long as a condition holds true
- syntax:

```
while ( expression ) {
    statements
}
```

- example:

```
int x = 10;
int y1 = 30;
int y2 = 40;
while ( x < width ) {
line( x, y1, x, y2 );
x = x + 10;
}
```

Standard Processing Program

1. Setup any variables or classes you are going to use.
2. Use setup() function to specify things to do once, when the sketch first opens
3. Use draw() function to specify things to do repeatedly
    - use frameRate() function to specify how often things should be repeated in draw();
    - default frame-rate is 60 (60 frames per second)
    - NOTE: call to frameRate() should be done inside setup() function
4. Declare and event-listeners that you are going to use.
5. Declare any custom made functions you are going to use.
6. Declare any classes that you are going to use.

The button class

example:gui _skeleton

Declaring buttons

```
//Define Buttons before Setup

Button readFileButton;

Button restartButton;

Button nextButton;

Button highlightButton;

Button quitButton;

Button partyButton;
```

INITIALIZING BUTTONS

```
void setup() {

  size(800, 500);

  smooth();

  textSize(16);



  //Create Clickable Buttons

  restartButton = new Button("Restart", 15, 450, 115, 35);

  readFileButton = new Button("Read File", 145, 450, 115, 35);

} //END setup
```

Call draw buttons function from the button class

```
void drawButtons() {
  restartButton.drawButton();
  readFileButton.drawButton();
  highlightButton.drawButton();
  nextButton.drawButton();
  quitButton.drawButton();
  partyButton.drawButton();
}
```

Call it inside draw()

```
void draw() {
  smooth();
  fill(256,256,256);
  rect(0, 0, 799, 399);
  drawButtons();
}
} //END draw
```

KEYPRESSED()

```
void keyPressed() {
  if (key=='s')
    showpoly= !showpoly ;
  if (key=='m')
    showminmax = !showminmax ;
  if (key=='c')
    gs();
  if (key=='z') loop() ;
   if (key=='p') gspatially();
      if (key=='2') sort2();
}
```

More keypressed()

```
if (keyCode == BACKSPACE) {

 } else if (keyCode == DELETE) {

 } else if (keyCode == ENTER) {

 } else if (keyCode != SHIFT && keyCode != CONTROL && keyCode != ALT) {

   userText = userText + key;

 }
```

More Mouse Interaction

- mouseX and mouseY
  - indicate (x, y) location of mouse pointer
- **mouseClicked**()
  - handles behavior when user clicks mouse button (press and release)
- **mouseMoved**()
  - handles behavior when user moves mouse (moves it without pressing button)
- **mouseDragged**()
  - handles behavior when user drags mouse (moves it with button pressed)
- **mouseButton**
  - indicates which button was pressed, on a multi-button mouse (on a Mac, use Cntl-click for left mouse button, Alt-click for middle mouse button and Apple-click for right mouse button)

Example 1 (mouse location)

```
void setup() {
    size( 200, 200 );
}

void draw() {
    background( #cccccc );

    fill( #000099 );
    rect( mouseX, mouseY, 20, 20 );
}
```

Example 2 (mouseMoved)

```
void setup() {
	size( 200, 200 );
}
void draw() {
	background( #cccccc );
	fill( #990000 );
	rect( mouseX, mouseY, 20, 20 );
}
void mouseMoved() {
	fill( #000099 );
	rect( mouseX, mouseY, 20, 20 );
}
/* how does this behave differently from the mouse location example? */
```

Example 3 (mouseDragged)

```
void setup() {
        size( 200, 200 );
}
void draw() {
        background( #cccccc );
        fill( #990000 );
        rect( mouseX, mouseY, 20, 20 );
}
void mouseMoved() {
        fill( #000099 );
        rect( mouseX, mouseY, 20, 20 );
}
void mouseDragged() {
        fill( #009900 );
        rect( mouseX, mouseY, 20, 20 );
}
/* how does this behave differently from the previous two examples? */
```

Example #4 (mouseClicked)

```
int r = 0;
int g = 0;
int b = 0;
void setup() {
        size( 200, 200 );
}
void draw() {
        background( #ffffff );
        fill( r, g, b );
        rect( 50, 50, 20, 20 );
}
void mouseClicked() {
    r = r + 51;
    if ( r > 255 ) {
        r = 0;
        g = g + 51;
        if ( g > 255 ) {
            g = 0;
            b = b + 51;
                if ( b > 255 ) {
                    b = 0;
                }
```

Example #5 (mouseButton)

```
void setup() {
      size( 200, 200 );
}
void draw() {
      background( #cccccc );
      rect( mouseX, mouseY, 20, 20 );
}
void mousePressed() {
      if ( mouseButton == LEFT ) {
      fill( #990000 );
      }
      else if ( mouseButton == CENTER ) {
                fill( #009900 );
      }
      else if ( mouseButton == RIGHT ) {   // Ctrl-click on mac
                fill( #000099 );
      }
}
```

Mouse Pressed vs Mouse Clicked

Press: press

Click: Press and release

https://processing.org/reference/mouseClicked_.html

https://processing.org/reference/mousePressed_.html

Read button from mouse press

```
void mousePressed() {
 // user presses "Restart"
 if (restartButton.mouseOver()) {
   restart();
 }
 // user presses "Read File" or "Read New File"
 else if (readFileButton.mouseOver()) {
 }
 // user presses "Quit"
```

Class Exercise 2

1. Create a canvas
2. Create a button in it.
3. On click of the button show an alert box which says “You Just Clicked Me”

Java Functions

```
void mousePressed() {
  // user presses "Restart"
  if (restartButton.mouseOver()) {
      javax.swing.JOptionPane.showMessageDialog(null,
  "restart Button Pressed ");
   //restart();
  }
```

FILE OPERATIONS

Reading from a file: Create Reader function

HTTPS://PROCESSING.ORG/REFERENCE/CREATEREADER_.HTML

```
void readSegments(String FILENAME) {

  String str= null;

  BufferedReader read;

  int i = 0;

  try {

   read = createReader(FILENAME);

   while((str = read.readLine()) != null){

    //first line stores n number of segments

    if (i == 0) {

} //END readSegments
```

WRITING TO A FILE :createWriter FUNCTION

HTTPS://PROCESSING.ORG/REFERENCE/CREATEWRITER_.HTML

```
void writeList(SegmentList A, String FILENAME) {
  PrintWriter output = createWriter(FILENAME);
 for(int i = 0; i < A.size(); i++) {
    output.println(A.get(i).toString());
  }
  output.flush();
  output.close();
} //END writeList
```

The data folder

- all input must be located in the sketch's data directory"
- Error: "The file "movie.mov" is missing or inaccessible, make sure the URL is valid or that the file has been added to your sketch and is readable.
- Example: lower envelopes

HIGHLIGHTING

```
void drawSegment(boolean isHighlighted, boolean isMergeSeg, boolean red, boolean blue){

    if (isHighlighted){

      strokeWeight(4);

    }

    else if (isMergeSeg){

      strokeWeight(2);

    }

    else {

      strokeWeight(1);

    }
```

CLASSES/TABS

Modular OOP like Code

Example: lower envelope

Class Exercise 2

1. Create a canvas
2. Create a button in it.
3. On click of the button read a file which contains ten input points (x1,y1).
4. Draw red circles at each of these x,y coordinate locations

Thank You