Practical 6- Conditionals and Events

Objectives

On completion of this lab you should be able to code animated drawings using the following constructs:

- conditional statements
- relational operators
- logical operators
- variables
- mouse events
- key events

Conditional statements and boolean expressions

- In this step, we will implement the code examples 3.1 and 3.2 from your lectures.

Conditional Example 6.1

- Create a new Processing sketch in your workspace and call it **Example_6_1**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
{
  size(100, 100);
  noStroke();
  fill(0);
}

void draw()
{
  background(204);
  if (mouseX < 50)
  {
    rect(0, 0, 50, 100);
  }
  else
  {
    rect(50, 0, 50, 100);
  }
}
```

- Run your code. Does it work as you would expect?

Conditional Example 6.2

- Create a new Processing sketch in your workspace and call it **Example_6_2**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
 {
   size(100, 100);
   noStroke();
   fill(0);
}




void draw()
{
   background(204);
   if (mouseX < 33)
  {
      rect(0, 0, 33, 100);
   }
   else if (mouseX < 66)
  {
      rect(33, 0, 33, 100);
   }
   else
  {
       rect(66, 0, 33, 100);
   }
}
```

- Run your code. Does it work as you would expect?

Conditional statements and logical operators

- In this step, we will implement the code examples 3.3 and 3.4 from your lectures.

Conditional Example 6.3

- Create a new Processing sketch in your workspace and call it **Example_6_3**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
{
  size(100, 100);
  noStroke();
  fill(0);
}
```

```
void draw()
 {
  background(204);
  if ((mouseX > 40) && (mouseX < 80) &&  (mouseY > 20) && (mouseY < 80))
 {
     fill(255);
  }
else
 {
     fill(0);
  }
  rect(40, 20, 40, 60);
}
```

- Run your code. Does it work as you would expect?

Conditional Example 6.4

- Create a new Processing sketch in your workspace and call it **Example_6_4**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
{
  size(100, 100);
  noStroke();
  fill(0);
}

void draw()
 {
  background(204);
  if ((mouseX <= 50) && (mouseY <= 50))
  {
    rect(0, 0, 50, 50);              // upper-left
   }
else if ((mouseX <= 50) && (mouseY > 50))
 {
    rect(0, 50, 50, 50);             // lower-left
  }
else if ((mouseX > 50) && (mouseY <= 50))
{
    rect(50, 0, 50, 50);            // upper-right
  }
 else
 {
    rect(50, 50, 50, 50);            // lower-right
  }
}
```

- Run your code. Does it work as you would expect?

Mouse events

- In this step, we will implement the code examples 3.5 to 3.8 inclusive from your lectures.

Conditional Example 6.5

- Create a new Processing sketch in your workspace and call it **Example_6_5**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
{
  size(100,100);
}

void draw()
 {
   background(0);
   stroke(255);
   fill(128);
   if (mousePressed == true)  // or   if (mousePressed)
  {
      rect(45,45,34,34);
   }
   else
  {
      ellipse(45,45,34,34);
   }
}
```

- Run your code. Does it work as you would expect?

Conditional Example 6.6

- Create a new Processing sketch in your workspace and call it **Example_6_6**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
{
  size(100,100);
}
void draw()
 {
  background(204);
  if (mousePressed == true)
  {
```

```
      fill(255); // white
   }
 else
 {
      fill(0);     // black
   }
   rect(25, 25, 50, 50);
}
```

- Run your code. Does it work as you would expect?

Conditional Example 6.7

- Create a new Processing sketch in your workspace and call it **Example_6_7**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup() {
  size(100,100);
}
void draw()
{
    if (mousePressed == true)
    {
      if (mouseButton == LEFT)
      {
          fill(0);      // black
      }
      else if (mouseButton == RIGHT)
      {
          fill(255);    // white
      }
     else
      {
       fill(126);     // gray
       }
    rect(25, 25, 50, 50);
}
}
```

- Run your code. Does it work as you would expect?

Conditional Example 6.8

- Create a new Processing sketch in your workspace and call it **Example_6_8**.
- Enter the following code into your sketchbook (don't copy and paste...write the code out):

```
void setup()
{
  size(500,400);
  background(0);
}

void draw()
 {

  if (mousePressed)
  {
    background(0);
  }

  stroke(255);
  fill(45,45,45);
  ellipse(mouseX, mouseY, 100, 100);
}
```

- Run your code. Does it work as you would expect?

Exercises

- For each exercise listed below, open a new sketchbook.
- You may need to visit the Processing website for additional information.
- When you are finished all your exercises, zip all your exercises into one file and send them on Slack titled **Practical3**.

Exercise 1

- Create a display window of 500x400, with a black background.
- When the mouse is pressed, draw a red circle (100 pixel diameter) with a white outline. Note: when the mouse is pressed and dragged, multiple circles will be drawn. When the mouse is released and dragged, no circles will be drawn.

Exercise 2

- Create a display window of 500x400, with a black background.
- When the LEFT mouse button is pressed, draw a red circle (50 pixel diameter) with a white outline. Note: when this button is pressed and dragged, multiple circles will be drawn.
- When the RIGHT mouse button is pressed, draw a blue square (length of 50 pixels) with a white outline. Note: when this button is pressed and dragged, multiple squares will be drawn.
- When no button is pressed and the mouse is dragged, nothing is drawn.

Exercise 3

- Draw circles (diameter 100) as the mouse is moved (the background should not be cleared).
- When the mouse is pressed, reduce the size of the circle by 10 pixels.
- When the circle diameter reaches 50 pixels, don't reduce it anymore.

Exercise 4

- Draw circles (diameter 100) as the mouse is moved (the background should not be cleared).
- As each circle is drawn, its colour should vary from the previous one (use mathematical calculations on your RBG colours).
- Make sure you **boundary** test this exercise. What we mean by that is...test what happens when your colour reaches 0 or when it reaches 255. Does it reset itself and cycle through the colours again or does it get stuck in drawing only one colour?

Challenge Exercise 1

- Draw a continuously bouncing ball.
- For the moment, the xCoordiate should remain the same value i.e. the ball only bounces vertically, so it is only the yCoordiate that needs to change.

Challenge Exercise 2

- Visit the Processing website and read up on the following keyEvents:
    - keyPressed (boolean variable used to determine if a key was pressed).
    - key (which stores a single alphanumeric character i.e. the most recently pressed key).
- In a new sketch, draw a cirlce if the **C** key was pressed and a rectangle if any other key was pressed.
- Test your code...is the keyboard input case sensitive? Can you enter a lower case **c** to draw a circle?

Challenge Exercise 3

- In a new sketch, draw a vertical line that is the height of your display window.
- This vertical line should start in the left most position of your display window and move right, pixel by pixel, until it reaches the right hand side of your display window.
- Upon reaching the right hand side, the vertical line should reverse direction and return, pixel by pixel, to the left hand side of the display window.
- As your vertical line is continually traversing the display window, your grayscale background should be varying very slightly in colour.