Practical 5 - Using Methods

Objectives

- On completion of this lab you should be able to use methods to handle mouse events and also be able to write your own methods.

Understanding mouse event methods

- In this step, you will work on the processing mouse examples from your lectures.

Coding the setup() method

- Create a new Processing sketch in your workspace and call it **Practical5_mouseMethods**.
- Enter the following code into your sketchbook (don't cut and paste it):

```
void setup()
{
   size(400, 400);
   background(0);
   textAlign(CENTER);
   textSize(24);
   fill(255);
   text("mouse has done nothing", width/2, height/2);
}
```

- Run the code so that you understand exactly what it does.
- You have used a new method called **text**. Go to the processing website and read about this method in the API reference area.

Coding the draw( ) method

- Mouse and keyboard events only work when a program has draw( ).
- Without draw( ), the code is only run once and then stops listening for events.
- To your open sketch, add the draw( ) method to it. The body of the method should be empty.

Coding the mouseMoved( ) method

- Add a mouseMoved method that will change the background to a red colour.
- Print the following text to the display window (not the console):

  "mouse was moved"

- Run your code. Does it work as you would expect when you move the mouse?
- Note: the code for this method is here:

```
void mouseMoved( )
{
   background(150, 10, 70);
   text("mouse was moved", width/2, height/2);
}
```

Coding the mouseDragged( ) method

- Add a mouseDragged method that will change the background to a blue colour.
- Print the following text to the display window (not the console):

  "mouse was dragged"

- Run your code. Does it work as you would expect when you drag the mouse?
- Note: the code for this method is here:

```
void mouseDragged( )
{
   background(10, 70, 100);
   text("mouse was dragged", width/2, height/2);
}
```

Coding the mouseReleased( ) method

- Add a mouseReleased method that will change the background to a purple colour.
- Print the following text to the display window (not the console):

  "mouse was released"

- Run your code. Does it work as you would expect when you release the mouse?
- Note: the code for this method is here:

```
void mouseReleased()
{
   background(100, 0, 100);
   text("mouse was released", width/2, height/2);
}
```

Coding the mousePressed( ) method

- Add a mousePressed method that will change the background to a green colour.
- Print the following text to the display window (not the console) when the left button was pressed:

  "mouse was pressed and it was the left button"

- Print the following text to the display window (not the console) when the right button was pressed:

  "mouse was pressed and it was the right button"

- Run your code. Does it work as you would expect when you press the left and then the right mouse buttons?
- Note: the code for this method is here:

```
void mousePressed()
{
   background(100, 100, 0);
   text("mouse was pressed", width/2, height/2);
   if ( mouseButton == LEFT)
   {
      text("and it was the left button", width/2, height/2 + 40);
   }
  else  if (mouseButton == RIGHT)
   {
      text("and it was the right button", width/2, height/2 + 40);
   }
}
```

Mouse event methods

- The code examples 3.5 to 3.8 inclusive from your lectures use the mouse system variables e.g. mousePressed.
- In this step, you will re-write the code to use mouse event methods instead e.g. void mousePressed( ).

Example 3.5

- Create a new Processing sketch in your workspace and call it **Practical05_Exercise_3_5_reworked**.
- Cut and paste the following code into your sketchbook:

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

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

- Run the code so that you understand exactly what it does.
- Rework the code so that it no longer tests the **mousePressed** variable but uses the **void mousePressed( )** method instead.
- Run your code. Does it work as you would expect?

Example 3.6

- Create a new Processing sketch in your workspace and call it **Practical05_Exercise_3_6_reworked**.
- Cut and paste the following code into your sketchbook:

```
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 the code so that you understand exactly what it does.
- Rework the code so that it no longer tests the **mousePressed** variable but uses the **void mousePressed( )** method instead.
- Run your code. Does it work as you would expect?

Example 3.7

- Create a new Processing sketch in your workspace and call it **Practical05_Exercise_3_7_reworked**.
- Cut and paste the following code into your sketchbook:

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

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

- Run the code so that you understand exactly what it does.
- Rework the code so that it no longer tests the mouse system variable but uses the mouse event methods instead.
- Run your code. Does it work as you would expect?

Example 3.8

- Create a new Processing sketch in your workspace and call it **Practical05_Exercise_3_8_reworked**.
- Cut and paste the following code into your sketchbook:

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

void draw() {

    if (mousePressed == true)
    {
      background(0);
     }

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

- Run the code so that you understand exactly what it does.
- Rework the code so that it no longer tests the mouse system variable but uses the mouse event methods instead.
- Run your code. Does it work as you would expect?

Writing your own methods

Learning Outcomes

- On completion of this lab you should:
    - Be familiar with Input and Output to Methods

NOTE

At the end of the draw method for the following questions please call the noLoop method.

This will ensure your draw only executes once.

The line is noLoop();

Question 1

Write a program that uses a method called **add()** to calculate the sum of 2 numbers.

- In **draw()** create 2 numbers.
- The numbers are to store it 2 float variable called **num1** and **num2**.
- The numbers entered (use random method, research how to do this) are then passed into a method called **add( )**.
- The method **add( )** calculates the sum of the numbers passed into it from **draw()** by adding the 2 numbers together and then prints the answer to the sketchbook (use text method).
- Nothing is returned from **add( )** to **draw( )**.

Question 2

- Write a program to enter 3 decimal numbers (use random method) and store these 3 numbers in 3 variables.
- The 3 numbers should be passed into the method **add()**, where the method should add the 3 numbers and print the answer to the sketchbook.

Question 3

The following program uses a method called **square( )** to calculate the square of a number.

- In **draw( )** use random method enter a number.
- This number is to be stored in a float variable called **num**.
- The number created is then passed into a method called **square( )**.
- The **square( )** method calculates the square of the number passed into it from **draw()** by multiplying the number by itself and then prints the answer to the screen.
- Nothing is returned from **square( )** to **draw( )**.

Question 4

- Add to question 3 and add a new method called **cube( )** which calculates the cube of the entered number passed from **draw( )** and then prints the answer to the screen.

Question 5

- Write a program that will enter in their age (use random method). Pass this value into a method called age( ).
- Write the method age( ) to calculate whether or not they are over 18.
- If they are over 18 then age( ) should print a message to the sketchbook telling them they can drink, otherwise print a message telling them they are too young to drink.

Question 6

- Write a program which uses a method to calculate the area of a circle.
- Enter a value for the radius of a circle (use random method).
- Pass this value into a method called area( ).
- This method should calculate the area of a circle using the formula area = 3.1416 * r2 and print the result to the sketchbook from inside that method.

Question 7

- Rewrite question 6 to do the following:
- The method area( ) should calculate the area of a circle using the formula area = 3.1416 * r2 and **return** the answer to draw( ) where it should then be printed to the sketchbook.

Question 8

- Write a program which tries to find the larger of two numbers.
- The values 10 and 30 should be stored in 2 variables.
- Pass these values into a method largestNum( )

- largestNum() should calculate which of the 2 numbers is largest and **return** this value to draw() where it should be printed to the sketchbook.

Question 9

- Rewrite **Question 8** to add another method:
- the 3 values are 10, 20, 5.
- passes the 3 values into the method **largestNum()**
- **largestNum()** calculates which is the larger of the 3 numbers, and returns this value to draw() where it should be printed to the sketchbook.

Question 10

- Rewrite **Question 9** so the information is printed to the sketchbook inside the methods.