



Week 11 - Interaction II

Event Handlers & User Interface



User Input Handling via system variables

```
void draw() {  
    if (mousePressed) {  
        text("MOUSE", mouseX, mouseY);  
    }  
    ...  
}
```

User Input Handling via system variables

```
void draw() {  
  if (mousePressed) {  
    text("MOUSE", mouseX, mouseY);  
  }  
  ...  
}
```

System
variables
(aka built-in
variables)

User Input Handling via system variables

variables	Description
key	Value of the most recent key used on the keyboard
keyCode	So detecting special keys e.g. UP, DOWN, LEFT etc.
keyPressed	[boolean] true if a key is pressed, and false if no key pressed.

variables	Description
mouseX	Current Horizontal coordinate of the Mouse.
mouseY	Current Vertical coordinate of the Mouse.
pmouseX	Value of mouseX in last frame
pmouseY	Value of mouseY in last frame
mousePressed	[boolean] true if any mouse button is pressed, and false if no mouse button pressed.
mouseButton	Stores which button is pressed. =0 is no button pressed; OR = LEFT, RIGHT or CENTER

Input Event Handlers

Sketch's main
`draw()`

Input Event handlers are supplied by the users (names pre-defined), and they will be invoked only if the corresponding events occur. They are always invoked after `draw()`.

```
void draw() {  
  ...  
}
```

```
void mousePressed() {  
  ...  
}  
  
void mouseDragged() {  
  ...  
}  
  
void mouseReleased() {  
  ...  
}
```

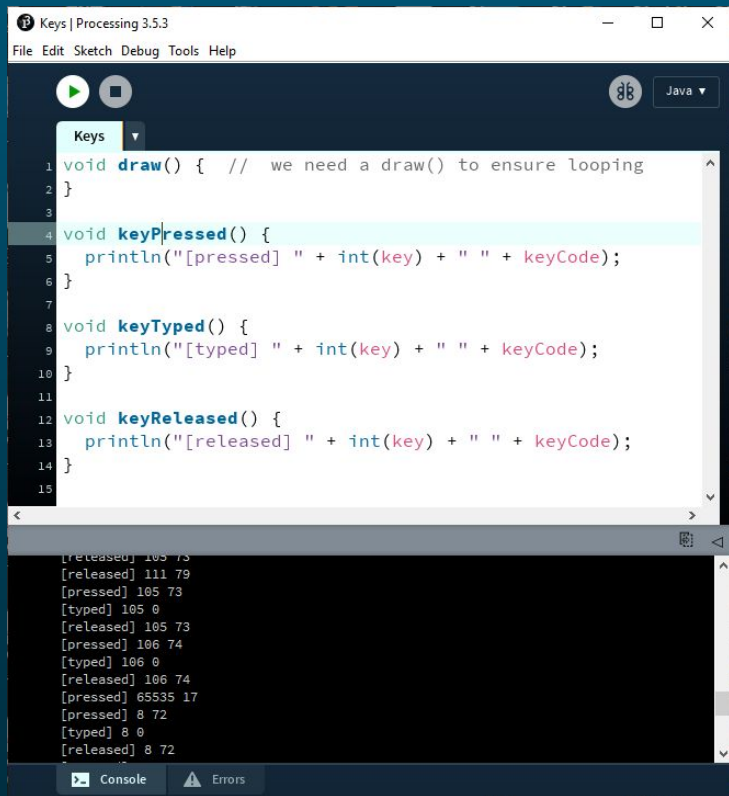


User Input Event Handlers

variables	Description
<u>keyPressed()</u>	called after a key is pressed. The actual key pressed is stored in the system variable key
<u>keyReleased()</u>	called after a key is released. The actual key released is stored in the system variable key
<u>keyTyped()</u>	called whenever a key is pressed but the action keys such as CTRL, Shift & ALT are ignored

variables	Description
<u>mouseClicked()</u>	called after a mouse button has been pressed and then released.
<u>mouseDragged()</u>	called whenever the mouse moves while a mouse button is pressed.
<u>mouseMoved()</u>	called whenever the mouse moves, and no mouse button is pressed.
<u>mousePressed()</u>	called once everytime a mouse button is pressed.
<u>mouseReleased()</u>	called once everytime a mouse button is released.
<u>mouseWheel()</u>	called whenever the mouse wheel moves.

Example 1 - Keys



```
Keys | Processing 3.5.3
File Edit Sketch Debug Tools Help

void draw() { // we need a draw() to ensure looping
}

void keyPressed() {
  println("[pressed] " + int(key) + " " + keyCode);
}

void keyTyped() {
  println("[typed] " + int(key) + " " + keyCode);
}

void keyReleased() {
  println("[released] " + int(key) + " " + keyCode);
}

[released] 105 73
[released] 111 79
[pressed] 105 73
[typed] 105 0
[released] 105 73
[pressed] 106 74
[typed] 106 0
[released] 106 74
[pressed] 65535 17
[pressed] 8 72
[typed] 8 0
[released] 8 72
```

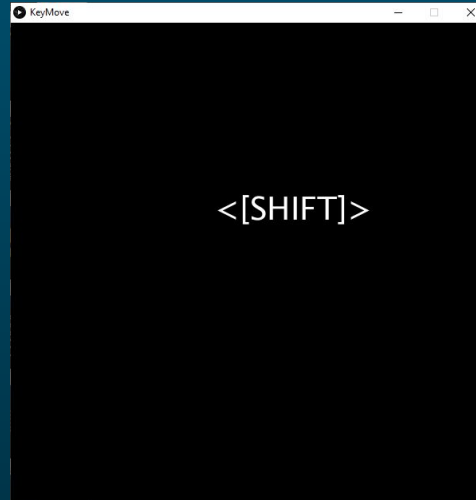
A utility alike example which visualizes how Processing calls key-related event handlers. This example is also useful for checking KeyCode of the keys that one desires to handle.

Example 2 - KeyMove



```
KeyMove | Processing 3.5.3
File Edit Sketch Debug Tools Help

KeyMove Key
1 Key ctrlKey;
2 Key altKey;
3 Key shiftKey;
4
5 void setup() {
6   size(600,600);
7   textAlign(CENTER, CENTER);
8   textSize(40);
9   fill(255);
10  shiftKey = new Key(16, "[SHIFT]");
11  ctrlKey = new Key(17, "[CTRL]");
12  altKey = new Key(18, "[ALT]");
13 }
14
15 void draw() { // we need a draw() to ensure looping
16   background(0);
17   String msg = shiftKey.display() + ctrlKey.display() + altKey.display();
18   text(msg, mouseX, mouseY);
19 }
20
21 void keyPressed() {
22   ctrlKey.checkPressed();
23   shiftKey.checkPressed();
24   altKey.checkPressed();
25 }
26
27 void keyReleased() {
28   ctrlKey.checkReleased();
29   shiftKey.checkReleased();
30   altKey.checkReleased();
31 }
32
33 Console Errors
```



A utility alike example which visualizes how Processing calls key-related event handlers. This example is also useful for checking KeyCode of the keys that one desires to handle.

Simple GUI Widget - Button

Graphical User Interface (GUI) widgets are visible elements which interact with users in an application. They offer visual feedbacks to reflect their current states.

Button states



Normal
(Not in focus)



Focused
(Mouse over)



Pressed
(Mouse pressed)

Example 3 - Buttons

```
Buttons | Processing 3.5.3
File Edit Sketch Debug Tools Help

Buttons Button
Button b1, b2;

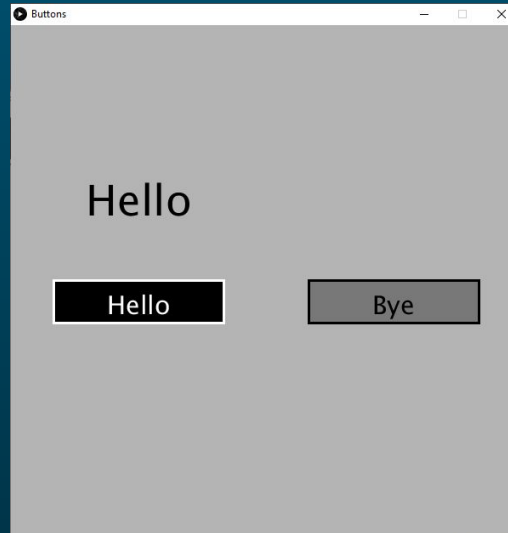
void setup() {
  size(600, 600);
  textAlign(CENTER, CENTER);
  b1 = new Button("Hello", 50, 300, 200, 50);
  b2 = new Button("Bye", 350, 300, 200, 50);
}

void draw() {
  background(180);
  if (b1.pressed) {
    textSize(50);
    fill(0);
    text(b1.label, 150, 200);
  }
  if (b2.pressed) {
    textSize(50);
    fill(0);
    text(b2.label, 450, 200);
  }
  b1.display();
  b2.display();
}

void mousePressed() {
  b1.userMousePressed();
  b2.userMousePressed();
}

void mouseReleased() {
  b1.userMouseReleased();
  b2.userMouseReleased();
}

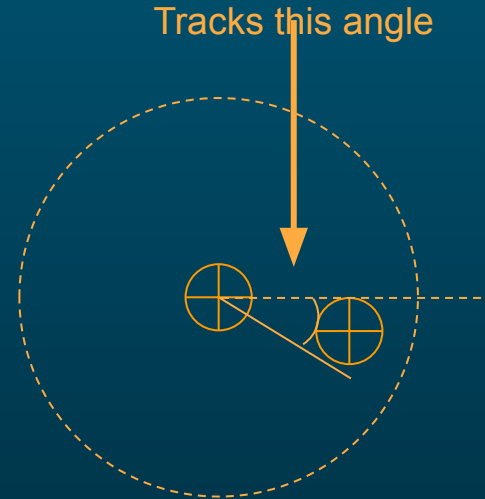
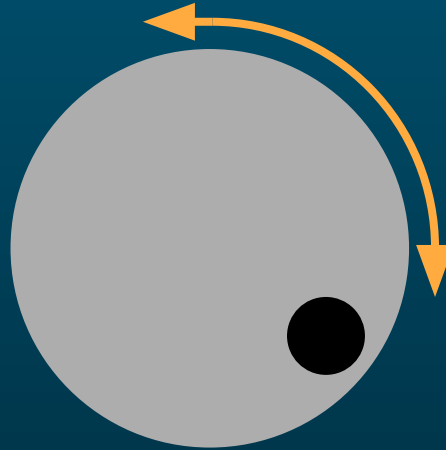
void mouseMoved() {
  b1.userMouseMoved();
  b2.userMouseMoved();
}
```



A simple example of how a simple GUI **Button** widget class looks like.

Custom GUI Widget - Dial

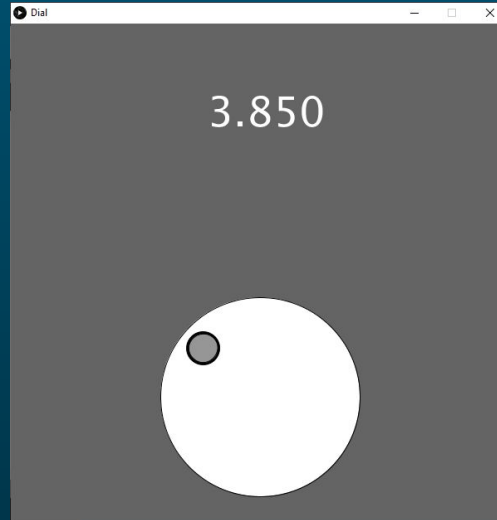
Dial - is a GUI widget which can be used to get 'value' changes from the user. Unlike slider, Dial has no lower or upper bound of values. It can be used as for example the jog control for video playback.



Example 4 - Dial

```
Dial | Processing 3.5.3
File Edit Sketch Debug Tools Help

Dial DialButton ▾
1 DialButton dial;
2
3 float V = 0;
4 //
5 void setup() {
6   size(600, 600);
7   textAlign(CENTER, CENTER);
8   textSize(50);
9   dial = new DialButton(width*0.5, height*0.75, 120, 20);
10 }
11
12 void draw() {
13   background(100);
14   dial.display();
15   fill(255);
16   text(V, width/2, 100);
17 }
18
19 void mouseDragged() {
20   if (dial.userMouseDragged()) {
21     V += dial.delta();
22   }
23   // println(jog.value(), jog.delta());
24 }
25
26 Console Errors
```



A simple example showing how a simple GUI **Dial** widget class looks like.

Example 5 - VideoJog

```
VideoJog | Processing 3.5.3
File Edit Sketch Debug Tools Help

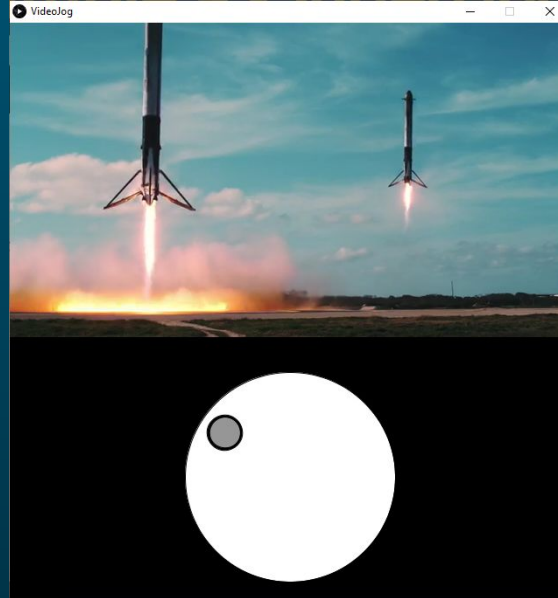
VideoJog DialButton
import processing.video.*;
Movie mov; // Single Global 'mov'
DialButton jog;
float movLength, now;
boolean jogged = false;

//
void setup() {
  size(640, 660);
  jog = new DialButton(width*0.5, height - 140, 120, 20);
  mov = new Movie(this, "falcon.mp4");
  mov.loop();
  movLength = mov.duration();
}

void draw() {
  background(0);
  if (jogged) {
    float now = map(jog.value(), 0, 360, 0, movLength);
    mov.jump(now);
  }
  image(mov, 0, 0);
  jog.display();
  jogged = false;
}

void mouseDragged() {
  jog.userMouseDragged();
  jogged = true;
}

void movieEvent(Movie m) {
  m.read();
}
```



A simple example showing how a GUI **Dial** is used as a jog control for video playback.

Using ArrayList

Regular Array in Processing is static which means the number of members stored in an array cannot be changed dynamically (unlike the array in P5js). **ArrayList** in Processing allows dynamic growth.

Constructors:

```
ArrayList<Type> () ;
```

```
ArrayList<Type> (initialSize) ;
```

```
ArrayList<Slide> ppt;
```

```
ppt = new ArrayList<Slide>() ;
```

```
ppt.add(slideA) ;
```

```
println(ppt.size()) ;
```

Example 6 - PresentApp



```
PresentApp | Processing 3.5.3
File Edit Sketch Debug Tools Help

PresentApp Deck Slide ▾

17 int projector_width = 1366;
18 int projector_height = 768;
19 int content_width = 1366;
20 int content_height = 768;
21 float global_scale;
22 float w_scale, h_scale;
23
24 //
25 void setup() {
26
27   size(1366, 768);
28
29   w_scale = float(projector_width) / content_width;
30   h_scale = float(projector_height) / content_height;
31   if (w_scale < h_scale) global_scale = h_scale;
32   else global_scale = w_scale;
33
34   mov = new VLCJVideo(this);
35   video_prefix = sketchPath() + "\\data\\movies\\";
36
37   txtFont = createFont("Arial Rounded MT Bold", 32 * global_scale);
38   textFont(txtFont);
39
40   footer = "SM2715 - Creative Coding L02, Mike Wong";
41
42   deck.addSlide(new Slide("CC 12\nPresentApp", TEXT_MODE_1, "images\\dummy_01.jpg", 0, 0));
43   deck.addSlide(new Slide("Dummy 02", IMAGE_MODE, "images\\dummy_02.jpg", 0, 0));
44   deck.addSlide(new Slide("Falcon (Dummy Video)", VIDEO_MODE,
45     video_prefix + "falcon.mp4", 1280, 720 ));
46
47   deck.addSlide(new Slide("Dummy 03", IMAGE_MODE, "images\\dummy_03.jpg", 0, 0));
48   deck.addSlide(new Slide("Fire 360 (Dummy Video)", VIDEO_MODE,
49     video_prefix + "fire_360.mp4", 1280, 720 ));
50
51   deck.update();
52
53 }
54
55 void draw() {
56   background(0);
57   fill(200);
58   deck.display();
59 }
```

Console

```
[h204 @ 00000001f56dad] Error splitting the input into NAL units.
Tag[10]: new: No audio tracks
[00000001f56dad770] main vout display error: failed to set on top
[00000001f56dad770] main vout display error: failed to set on top
```



A simple SlideShow app.

