

Introduction to Processing

Class #6: 10/24/2007

Taught by Ms. Madsen, Ms. Huhn & Ms. Yen

Course website: >> web.mit.edu/mish/www/processing

Processing website: >> www.processing.org

Email the teachers: mish@mit.edu, ahuhn@mit.edu, cyyen@mit.edu,

Here's how you can use `mousePressed()` to center your object:

```
int distanceFromTop = 0;
int distanceFromSide = 50;
color backgroundColorName = color(redValue, greenValue, blueValue);

void setup() {
    background(backgroundColorName);
    size(width, height);
    loop();
}

void draw(){
    background(backgroundColorName);
    ellipse(distanceFromSide, distanceFromTop, width, height);
    if (distanceFromTop > height) {
        distanceFromTop = 0;
    }
    distanceFromTop++;
}

void mousePressed(){
    distanceFromTop = mouseY;
    distanceFromSide = mouseX;
}
```

Here's how you can make a button.

Whatever you put in the "if" statement in mousePressed() will run whenever you press your button.

```
int distanceFromTop = 0;
int distanceFromSide = 50;
color backgroundColorName = color(redValue, greenValue, blueValue);
int buttonXStart = 0, buttonYStart = 0, buttonWidth = 20, buttonHeight=10;

void setup() {
  background(backgroundColorName);
  size(width, height);
  loop();
}

void draw(){
  background(backgroundColorName);
  color buttonColor = color(redValue, blueValue, greenValue);
  rect(buttonXStart, buttonYStart, buttonWidth, buttonHeight);

  ellipse(distanceFromSide, distanceFromTop, width, height);
  if (distanceFromTop > height) {
    distanceFromTop = 0;
  }
  distanceFromTop++;
}

void mousePressed(){
  distanceFromTop = mouseY;
  distanceFromSide = mouseX;
  if ( (mouseX > buttonXStart) && (mouseX < buttonXStart+buttonWidth)
  && (mouseY > buttonYStart) && (mouseY < buttonYStart+buttonHeight) ) {
    You can add whatever code you want here.
  }
}
```

Here are some choices for the code in the if statement. Or make up your own!

```
println("Button was clicked!");
```

```
buttonXStart++;
```

```
buttonYStart++;
```

```
buttonColor = color(redValue, blueValue, mouseX);
```

```
backgroundColorName = color(mouseX, mouseY, mouseX);
```

Making colors partially transparent:

You can make colors partially transparent! When you're using the *fill* command, you give a *percentage* (0-100) of full color. 0 is totally transparent; 100 is totally opaque (opposite of transparent); and 50 is halfway transparent.

Here are two ways of doing the same thing.

```
color newColor = color(0,0,255,80);    // this will be 20% transparent
fill(newColor);
```

```
color newColor = color(0,0,255);
fill(newColor, 80);                    // this will also be 20% transparent
```

Here are some ideas for other things you could do:

Maybe you want the background color to change every time you click the mouse. You can make another integer variable, *redLevel*, outside of any methods – near *distanceFromTop* – and set it to zero. Then, add the following code to *mousePressed()*:

```
redLevel++;
backgroundName = color(redLevel, greenValue, blueValue);
```

(Or, you could make a variable for a shape's transparency level and change THAT every time the mouse is pressed...)

What if you want to have another shape moving from bottom to top? You should make another integer variable outside of your methods, *distanceFromTop2*, and set that equal to the height of your window at the beginning. Then, add something like this to *draw()*:

```
rect(xStart, distanceFromTop2, width, height);
distanceFromTop2--;
```

You can add another if statement (like the one we added before) to make sure it doesn't go off the top of the screen. (Why don't we have to worry about it going off the bottom of the screen?)

```
if (distanceFromTop2 < 0) {
    distanceFromTop2 = height;
}
```