```
mouseX, mouseY, pmouseY, pmouseY,
mousePressed, mouseButton, mouseReleased, mouseClicked, mouseMoved
mouseDragged, mouseWheel
----------------------------------------------------------------------
key, keyCode, keyPressed, keyReleased
----------------------------------------------------------------------
cursor(ARROW | CROSS | HAND | MOVE | TEXT | WAIT)
noCursor()
----------------------------------------------------------------------
void setup() {
  size(100, 100);
}
void draw() {
  background(204);
  if (mousePressed == true) {
    fill(255);  // White
  } else {
    fill(0);    // Black
  }
  rect(25, 25, 50, 50);
}
void setup() {
  size(100, 100);
}
----------------------------------------------------------------------
void draw() {
  if (mouseButton == LEFT) {
    fill(0);    // Black
  } else if (mouseButton == RIGHT) {
    fill(255);  // White
  } else {
    fill(126);  // Gray
  }
  rect(25, 25, 50, 50);
}
----------------------------------------------------------------------
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);
}
----------------------------------------------------------------------
int gray = 0;
void setup() {
  size(100, 100);
}
void draw() {
  background(gray);
}
void mousePressed() {
  gray += 20;
```

```
}
----------------------------------------------------------------------
void setup() {
  size(100, 100);
  strokeWeight(4);
}
void draw() {
  background(204);
  if (keyPressed == true) {  // If the key is pressed,
    line(20, 20, 80, 80);    // draw a line
  } else {                   // Otherwise,
    rect(40, 40, 20, 20);    // draw a rectangle
  }
}
----------------------------------------------------------------------
int x = 20;
void setup() {
  size(100, 100);
  strokeWeight(4);
}
void draw() {
  background(204);
  if (keyPressed == true) {  // If the key is pressed
    x++;                     // add 1 to x
  }
  line(x, 20, x-60, 80);
}
void setup() {
  size(100, 100);
  strokeWeight(4);
```

```
}
----------------------------------------------------------------------
void draw() {
  background(204);
  // If the 'A' key is pressed draw a line
  if ((keyPressed == true) && (key == 'A')) {
    line(50, 25, 50, 75);
  } else {  // Otherwise, draw an ellipse
    ellipse(50, 50, 50, 50);
  }
}
----------------------------------------------------------------------
void setup() {
  size(100, 100);
}
void draw() {
  background(204);
  line(10, 50, 90, 50);
  if (key == CODED) {
    if (keyCode == UP) {
      y = 20;
    } else if (keyCode == DOWN) {
      y = 50;
    }
  } else {
    y = 35;
  }
  rect(25, y, 50, 30);
----------------------------------------------------------------------
boolean drawT = false;
```

```
void setup() {
  size(100, 100);
  noStroke();
}
void draw() {
  background(204);
  if (drawT == true) {
    rect(20, 20, 60, 20);
    rect(39, 40, 22, 45);
  }
}
void keyPressed() {
  if ((key == 'T') || (key == 't')) {
    drawT = true;
  }
}
void keyReleased() {
  drawT = false;
}
----------------------------------------------------------------------
void setup() {
  size(100, 100);
  strokeWeight(7);
  noCursor();
}
void draw() {
  background(204);
  ellipse(mouseX, mouseY, 10, 10);
}
----------------------------------------------------------------------
```

```
void setup() {
  size(100, 100);
}
void draw() {
  background(204);
  if (mousePressed == true) {
    cursor(HAND);  // Draw cursor as hand
  } else {
    cursor(CROSS);
  }
  line(mouseX, 0, mouseX, height);
  line(0, mouseY, height, mouseY);
}
----------------------------------------------------------------------
PImage mouseCursor;
void setup() {
  size(640,480);
  mouseCursor = loadImage("MouseCursor.png");
}
void draw()
{
  if(mouseX < 100) {
    cursor(mouseCursor, 0, 0);
  } else {
    cursor(HAND);
  }
}
----------------------------------------------------------------------
// Free drawing
void setup()
```

```
{
size(500, 500);
background(0);
stroke(255);
strokeWeight(2);
smooth(); // Uses antialiasing techniques
}
void draw()
{
if (mousePressed)
line(pmouseX, pmouseY, mouseX, mouseY);
}
----------------------------------------------------------------------
// Free drawing
void setup()
{
size(500, 500);
background(0);
strokeWeight(2);
smooth(); // Uses antialiasing techniques
}
void draw()
{
if (mousePressed) {
if (mouseButton == LEFT) { // Draw
stroke(255);
strokeWeight(2);
}
else { // Erase
stroke(0);
strokeWeight(4);
}
line(mouseX, mouseY, pmouseX, pmouseY);
}
}
----------------------------------------------------------------------
// Free drawing
color colorStroke = color(255, 0, 0);
void setup()
{
size(500, 500);
background(0);
strokeWeight(2);
smooth(); // Uses antialiasing techniques
}
void draw()
{
if (mousePressed) {
if (mouseButton == LEFT) { // Draw
stroke(colorStroke);
strokeWeight(2);
}
else {
stroke(0); // Erase
strokeWeight(4);
}
line(mouseX, mouseY, pmouseX, pmouseY);
}
}
void keyPressed()
```

```
{
switch (key) {
case 'r':
case 'R':
colorStroke = color(255,0,0);
break;
case 'g':
case 'G':
colorStroke = color(0,255,0);
break;
case 'b':
case 'B':
colorStroke = color(0,0,255);
break;
}
}
-----------------------------------------------------------------------
/**
 * Rollover.
* Roll over the colored squares in the center of the image
 * to change the color of the outside rectangle.
 */

int rectX, rectY;      // Position of square button
int circleX, circleY;  // Position of circle button
int rectSize = 90;     // Diameter of rect
int circleSize = 93;   // Diameter of circle

color rectColor;
color circleColor;
color baseColor;

boolean rectOver = false;
boolean circleOver = false;

void setup() {
  size(640, 360);
  rectColor = color(0);
  circleColor = color(255);
  baseColor = color(102);
  circleX = width/2+circleSize/2+10;
  circleY = height/2;
  rectX = width/2-rectSize-10;
  rectY = height/2-rectSize/2;
  ellipseMode(CENTER);
}

void draw() {
  update(mouseX, mouseY);

  noStroke();
  if (rectOver) {
    background(rectColor);
  } else if (circleOver) {
    background(circleColor);
  } else {
    background(baseColor);
  }

  stroke(255);
  fill(rectColor);
  rect(rectX, rectY, rectSize, rectSize);
  stroke(0);
  fill(circleColor);
  ellipse(circleX, circleY, circleSize, circleSize);
}

void update(int x, int y) {
  if( overCircle(circleX, circleY, circleSize) ) {
    circleOver = true;
    rectOver = false;
  } else if ( overRect(rectX, rectY, rectSize, rectSize) ) {
    rectOver = true;
    circleOver = false;
  } else {
    circleOver = rectOver = false;
  }
}

boolean overRect(int x, int y, int width, int height) {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}

boolean overCircle(int x, int y, int diameter) {
  float disX = x - mouseX;
  float disY = y - mouseY;
  if(sqrt(sq(disX) + sq(disY)) < diameter/2 ) {
    return true;
  } else {
    return false;
  }
}
----------------------------------------------------------------------
/**
 * Button.
* Click on one of the colored squares in the
 * center of the image to change the color of
 * the background.
 */

int rectX, rectY;      // Position of square button
int circleX, circleY;  // Position of circle button
int rectSize = 90;     // Diameter of rect
int circleSize = 93;   // Diameter of circle
color rectColor, circleColor, baseColor;
color rectHighlight, circleHighlight;
color currentColor;
boolean rectOver = false;
boolean circleOver = false;

void setup() {
  size(640, 360);
  rectColor = color(0);
  rectHighlight = color(51);
  circleColor = color(255);
  circleHighlight = color(204);
  baseColor = color(102);
  currentColor = baseColor;
  circleX = width/2+circleSize/2+10;
  circleY = height/2;
  rectX = width/2-rectSize-10;
  rectY = height/2-rectSize/2;
  ellipseMode(CENTER);
}

void draw() {
  update(mouseX, mouseY);
  background(currentColor);

  if (rectOver) {
    fill(rectHighlight);
  } else {
    fill(rectColor);
  }
  stroke(255);
  rect(rectX, rectY, rectSize, rectSize);

  if (circleOver) {
    fill(circleHighlight);
  } else {
    fill(circleColor);
  }
  stroke(0);
  ellipse(circleX, circleY, circleSize, circleSize);
}
```

```
void update(int x, int y) {
  if ( overCircle(circleX, circleY, circleSize) ) {
    circleOver = true;
    rectOver = false;
  } else if ( overRect(rectX, rectY, rectSize, rectSize) ) {
    rectOver = true;
    circleOver = false;
  } else {
    circleOver = rectOver = false;
  }
}

void mousePressed() {
  if (circleOver) {
    currentColor = circleColor;
  }
  if (rectOver) {
    currentColor = rectColor;
  }
}

boolean overRect(int x, int y, int width, int height)  {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}
```

```
boolean overCircle(int x, int y, int diameter) {
  float disX = x - mouseX;
  float disY = y - mouseY;
  if (sqrt(sq(disX) + sq(disY)) < diameter/2 ) {
    return true;
  } else {
    return false;
  }
}
----------------------------------------------------------------------
size(100, 100, P3D);
noFill();
ortho(-width/2, width/2, -height/2, height/2); // Same as ortho()
translate(width/2, height/2, 0);
rotateX(-PI/6);
rotateY(PI/3);
box(45);
----------------------------------------------------------------------
size(100, 100, P3D);
noFill();
background(204);
frustum(-10, 0, 0, 10, 10, 200);
rotateY(PI/6);
box(45);
----------------------------------------------------------------------
// Re-creates the default perspective
size(100, 100, P3D);
noFill();
float fov = PI/3.0;
float cameraZ = (height/2.0) / tan(fov/2.0);
perspective(fov, float(width)/float(height),
            cameraZ/10.0, cameraZ*10.0);
translate(50, 50, 0);
rotateX(-PI/6);
rotateY(PI/3);
box(45);

----------------------------------------------------------------------
void setup() {
  size(640, 360, P3D);
  fill(204);
}

void draw() {
  lights();
  background(0);

  // Change height of the camera with mouseY
  camera(30.0, mouseY, 220.0, // eyeX, eyeY, eyeZ
         0.0, 0.0, 0.0, // centerX, centerY, centerZ
         0.0, 1.0, 0.0); // upX, upY, upZ

  noStroke();
  box(90);
  stroke(255);
  line(-100, 0, 0, 100, 0, 0);
  line(0, -100, 0, 0, 100, 0);
  line(0, 0, -100, 0, 0, 100);
}
```

```
----------------------------------------------------------------------
size(100, 100, P3D);
noFill();
background(204);
camera(70.0, 35.0, 120.0, 50.0, 50.0, 0.0,
       0.0, 1.0, 0.0);
translate(50, 50, 0);
rotateX(-PI/6);
rotateY(PI/3);
box(45);
----------------------------------------------------------------------
/**
 * Rollover.
* Roll over the colored squares in the center of the image
 * to change the color of the outside rectangle.
 */

int rectX, rectY;      // Position of square button
int circleX, circleY;  // Position of circle button
int rectSize = 90;     // Diameter of rect
int circleSize = 93;   // Diameter of circle

color rectColor;
color circleColor;
color baseColor;

boolean rectOver = false;
boolean circleOver = false;

void setup() {
  size(640, 360);
  rectColor = color(0);
  circleColor = color(255);
  baseColor = color(102);
  circleX = width/2+circleSize/2+10;
  circleY = height/2;
  rectX = width/2-rectSize-10;
  rectY = height/2-rectSize/2;
  ellipseMode(CENTER);
}

void draw() {
  update(mouseX, mouseY);

  noStroke();
  if (rectOver) {
    background(rectColor);
  } else if (circleOver) {
    background(circleColor);
  } else {
    background(baseColor);
  }

  stroke(255);
  fill(rectColor);
  rect(rectX, rectY, rectSize, rectSize);
  stroke(0);
  fill(circleColor);
  ellipse(circleX, circleY, circleSize, circleSize);
}
```

```
void update(int x, int y) {
  if( overCircle(circleX, circleY, circleSize) ) {
    circleOver = true;
    rectOver = false;
  } else if ( overRect(rectX, rectY, rectSize, rectSize) ) {
    rectOver = true;
    circleOver = false;
  } else {
    circleOver = rectOver = false;
  }
}

boolean overRect(int x, int y, int width, int height) {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}

boolean overCircle(int x, int y, int diameter) {
  float disX = x - mouseX;
  float disY = y - mouseY;
  if(sqrt(sq(disX) + sq(disY)) < diameter/2 ) {
    return true;
  } else {
    return false;
  }
```

```
}
----------------------------------------------------------------------
/**
 * Button.
* Click on one of the colored squares in the
 * center of the image to change the color of
 * the background.
 */

int rectX, rectY;      // Position of square button
int circleX, circleY;  // Position of circle button
int rectSize = 90;     // Diameter of rect
int circleSize = 93;   // Diameter of circle
color rectColor, circleColor, baseColor;
color rectHighlight, circleHighlight;
color currentColor;
boolean rectOver = false;
boolean circleOver = false;

void setup() {
  size(640, 360);
  rectColor = color(0);
  rectHighlight = color(51);
  circleColor = color(255);
  circleHighlight = color(204);
  baseColor = color(102);
  currentColor = baseColor;
  circleX = width/2+circleSize/2+10;
  circleY = height/2;
  rectX = width/2-rectSize-10;
  rectY = height/2-rectSize/2;
  ellipseMode(CENTER);
}

void draw() {
  update(mouseX, mouseY);
  background(currentColor);

  if (rectOver) {
    fill(rectHighlight);
  } else {
    fill(rectColor);
  }
  stroke(255);
  rect(rectX, rectY, rectSize, rectSize);

  if (circleOver) {
    fill(circleHighlight);
  } else {
    fill(circleColor);
  }
  stroke(0);
  ellipse(circleX, circleY, circleSize, circleSize);
}

void update(int x, int y) {
  if ( overCircle(circleX, circleY, circleSize) ) {
    circleOver = true;
    rectOver = false;
  } else if ( overRect(rectX, rectY, rectSize, rectSize) ) {
    rectOver = true;
    circleOver = false;
  } else {
    circleOver = rectOver = false;
  }
}

void mousePressed() {
  if (circleOver) {
    currentColor = circleColor;
  }
  if (rectOver) {
    currentColor = rectColor;
  }
}

boolean overRect(int x, int y, int width, int height)  {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}

boolean overCircle(int x, int y, int diameter) {
  float disX = x - mouseX;
  float disY = y - mouseY;
  if (sqrt(sq(disX) + sq(disY)) < diameter/2 ) {
    return true;
  } else {
    return false;
  }
}
--------------------------------------------------------------------
```