Processing Workshop

Till Nagel, IUAV, 10/2008

„A man paints with his brain, not with his hands“

Michelangelo

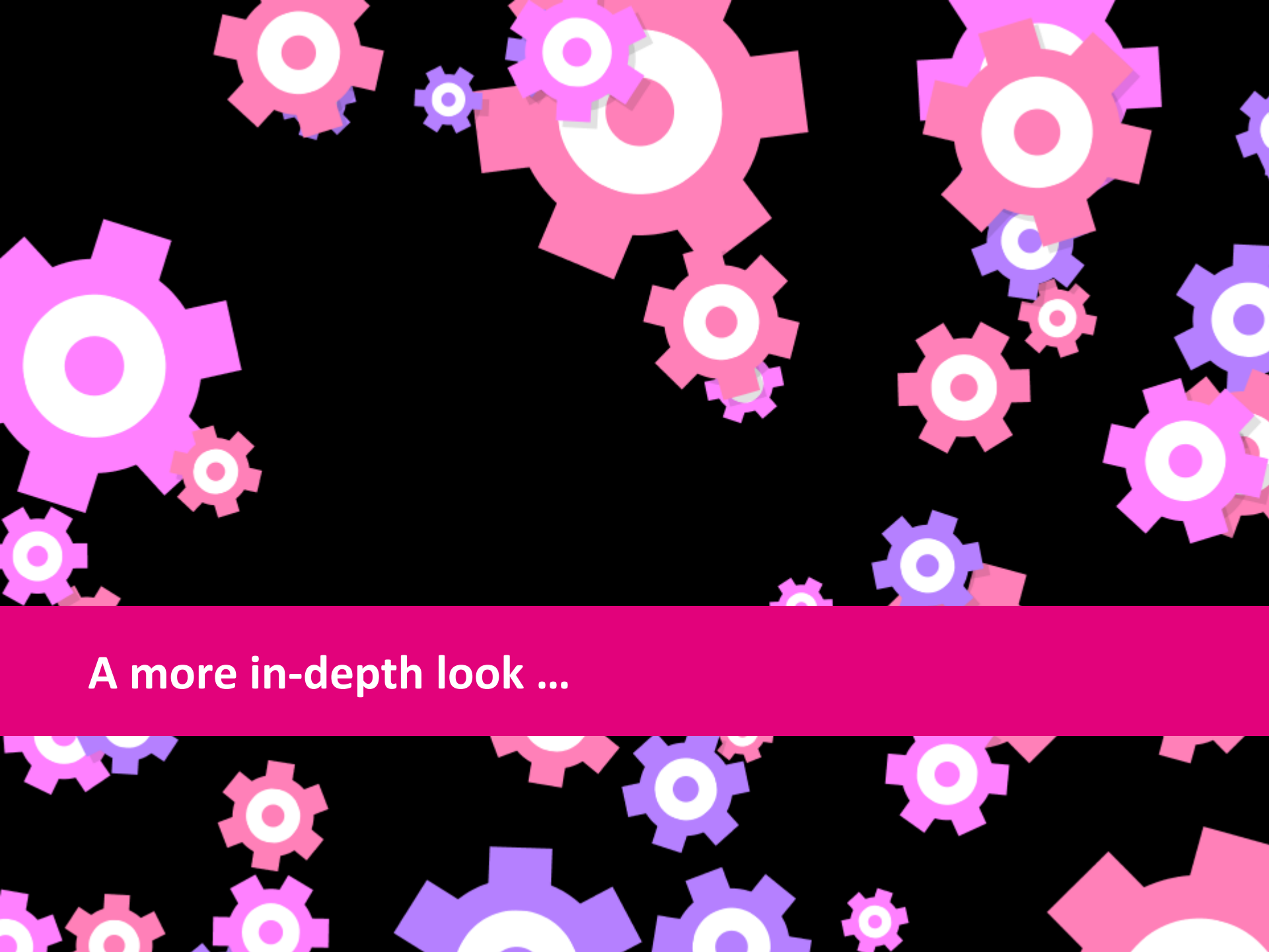
Processing Basics

```
size(200, 200);
background(255);
smooth();

// face
rect(30, 10, 140, 180);

// eyes
fill(0, 0, 255);
ellipse(70, 60, 20, 20);
ellipse(130, 60, 20, 20);

// nose
noFill();
ellipse(100, 100, 30, 30);
```

Processing Basics

```
void setup() {
  size(500, 500);
  smooth();
}

void draw() {
  background(200);
  rect(50, 50, mouseX - 50, mouseY - 50);
}
```

Example: RectResizeWithMouse


A more in-depth look …

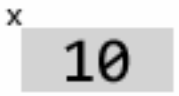
Variables

Variables

```
int x = 10;
```

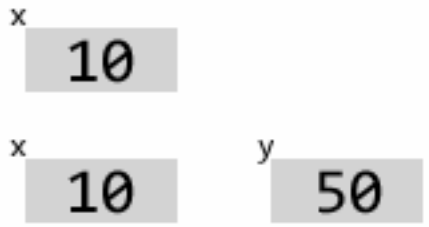
Variables

```
int x = 10;
```



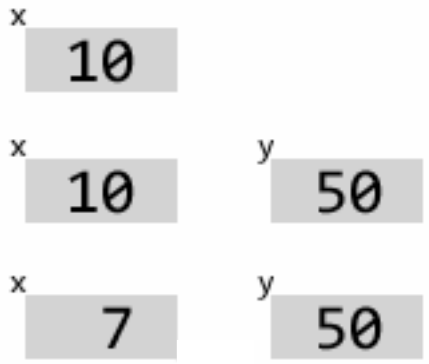
Variables

```
int x = 10;

int y = 50;
```



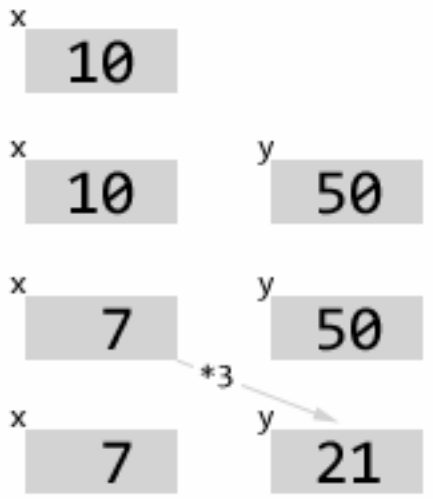
Variables

```
int x = 10;

int y = 50;

x = 2 + 5;
```



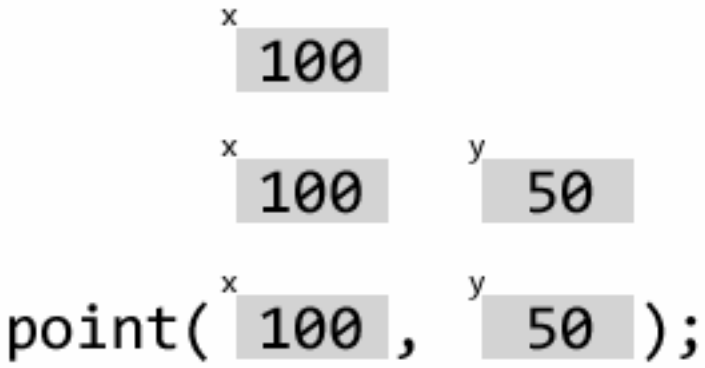
Variables

```
int x = 10;

int y = 50;

x = 2 + 5;

y = x * 3;
```



Variables in use

```
int x = 100;

int y = 50;

point(x, y);
```



Dynamic sketch

```
void setup() {
      size(200, 200);
}
void draw() {
      ellipse(mouseX, mouseY, 20, 20);
}
```

Continous variable update

```
int x = 10;
void setup() {
      size(200, 200);
}
void draw() {
      line(x, 0, x, 100);
      x = x + 2;
}
```

Continous variable update

```
line(x, 0, x, 100);

x = x + 2;
```

```
line(x, 0, x, 100);

x = x + 2;
```

```
line(x, 0, x, 100);

x = x + 2;
```

```
...
```

Continous variable update

```
line(x, 0, x, 100);
x = x + 2;
```

```
line(x, 0, x, 100);
x = x + 2;
```

```
line(x, 0, x, 100);
x = x + 2;
```

```
...
```

```
line( x 10 , 0, x 10 , 100);
```

Continous variable update

```
line(x, 0, x, 100);

x = x + 2;
```

```
line(x, 0, x, 100);

x = x + 2;
```

```
line(x, 0, x, 100);

x = x + 2;
```

```
...
```

```
line( x 10 , 0, x 10 , 100);
          +2
      x 12
```

Continous variable update

```
line(x, 0, x, 100);

x = x + 2;
```

```
line( x 10 , 0, x 10 , 100);
      +2
x 12
```

```
line(x, 0, x, 100);

x = x + 2;
```

```
line( x 12 , 0, x 12 , 100);
      +2
x 14
```

```
line(x, 0, x, 100);

x = x + 2;
```

...

Continous variable update

```
line(x, 0, x, 100);
x = x + 2;
```

```
line( x 10 , 0, x 10 , 100);
      +2
x 12
```

```
line(x, 0, x, 100);
x = x + 2;
```

```
line( x 12 , 0, x 12 , 100);
      +2
x 14
```

```
line(x, 0, x, 100);
x = x + 2;
```

```
line( x 14 , 0, x 14 , 100);
      +2
x 16
```

```
...
```

Example: VarLoopChange

Multiple use of variables

```
void setup() {
      size(200, 200);
}
void draw() {
      fill(mouseX, mouseY, 0);
      ellipse(mouseX, mouseY, 20, 20);
}
```

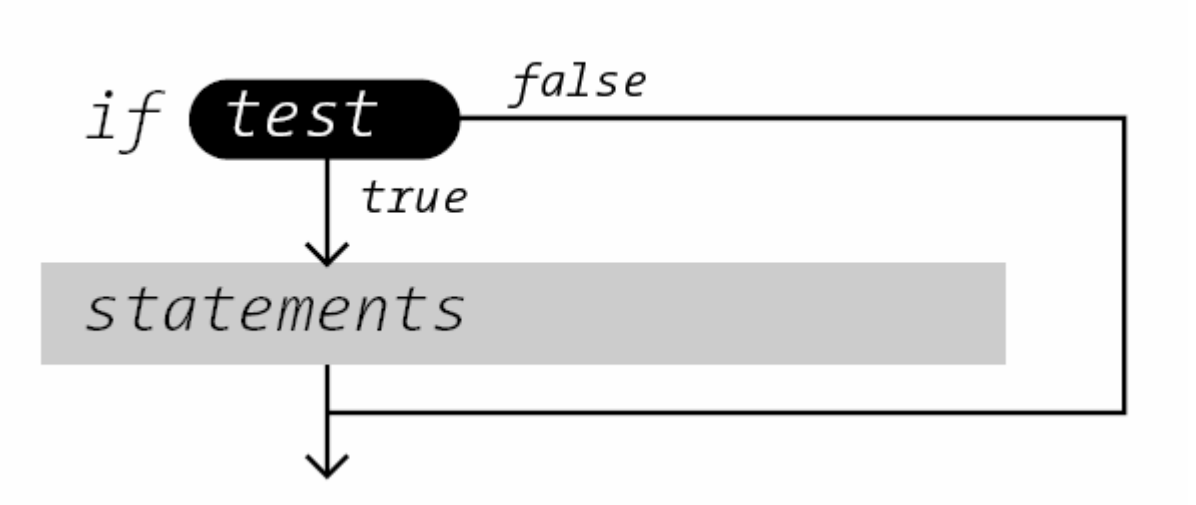
Example: MultiUse

Homework discussion

Conditionals

Conditionals

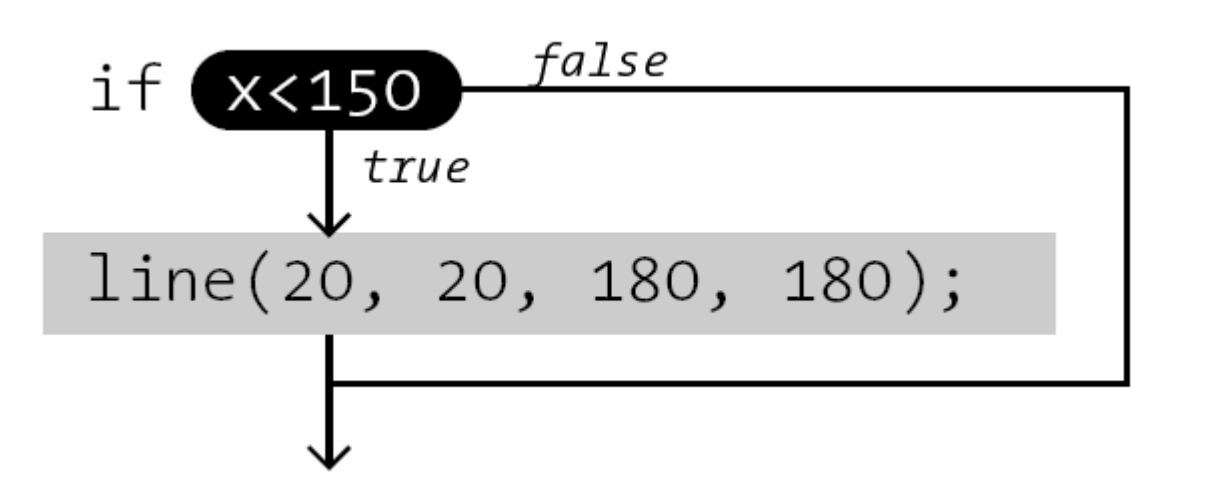
```
if (test) {
      statements
}
```



From: Casey Reas, Ben Fry, Processing, MIT Press, 2007

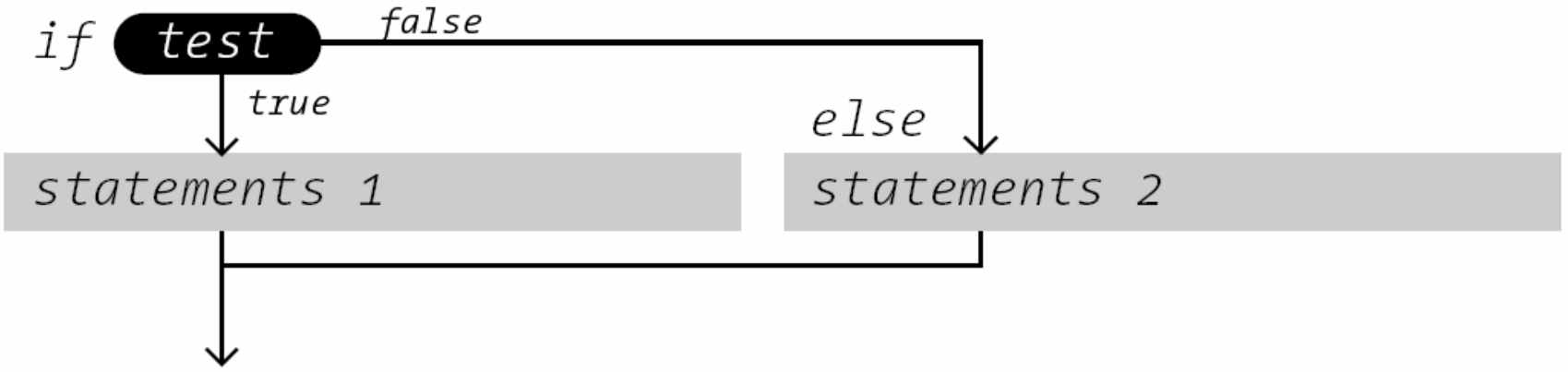
Conditionals

```
if (x < 150) {
      line(20, 20, 180, 180);
}
```



Conditionals

```
if (test) {
      statements 1
}
else {
      statements 2
}
```



Conditionals

```
if (x < 150) {
      line(20, 20, 180, 180);
}
else {
      ellipse(50, 50, 30, 30);
}
```

Example: If1

Conditionals

Conditionals control the program flow.

Each condition can be either true or false.

It checks if a condition is true.

If condition is true, the inner statements are executed.

Conditionals

```
int a = 10;
int b = 20;
if (a > 10) {
      line(10, 10, 100, 10);
}
if (b >= 20) {
      line(10, 20, 100, 20);
}
```

Conditionals

```
int a = 10;
int b = 20;
if (a >= 10) {
      line(10, 10, 100, 10);
      b = b + 1;
}
if (b > 20) {
      line(10, 20, 100, 20);
}
```

Comparison operators

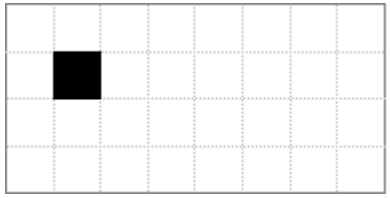
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
==	equal to
!=	not equal to

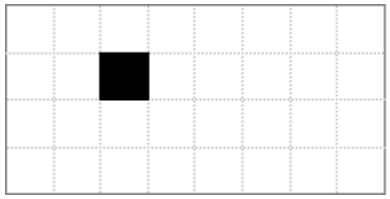
Simple patterns

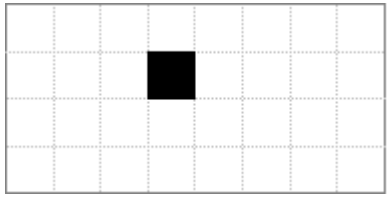
Movement

```
int x = 1;
int y = 1;

void setup() {
  size(8, 3);
}

void draw() {
  point(x, y);
}
```

Movement

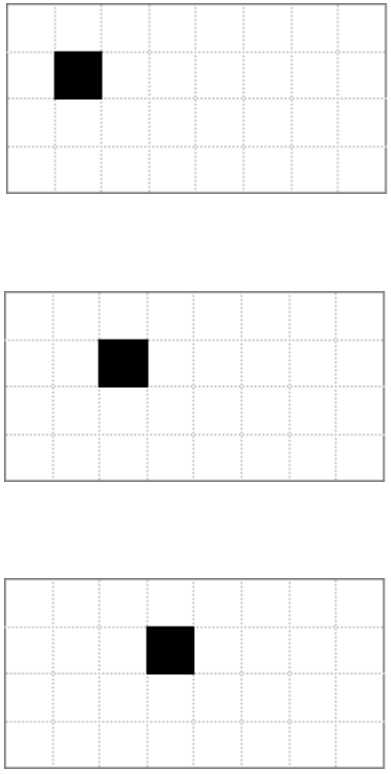






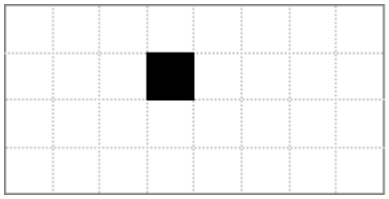
```
int x = 1;
int y = 1;

void setup() {
  size(8, 3);
}

void draw() {
  point(x, y);
  x = ??
}
```

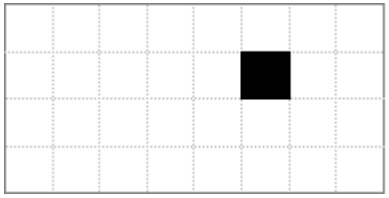
Movement



```
int x = 1;
int y = 1;

void setup() {
  size(8, 3);
}

void draw() {
  point(x, y);
  x = x + 1;
}
```

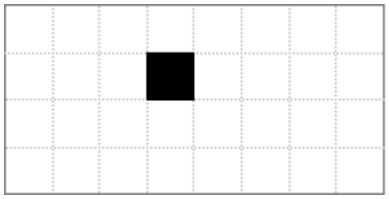
Movement

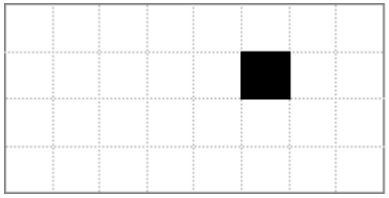




```
int x = 1;
int y = 1;

void setup() {
  size(8, 3);
}

void draw() {
  point(x, y);
  x = x + 2;
}
```

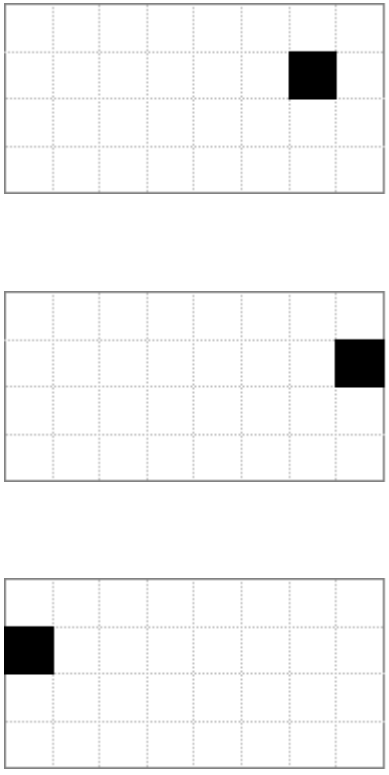
Movement: Velocity





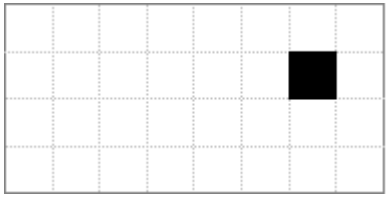
```
int x = 1;
int y = 1;
int v = 2;

void setup() {
  size(8, 3);
}

void draw() {
  point(x, y);
  x = x + v;
}
```

Movement: Collision detection



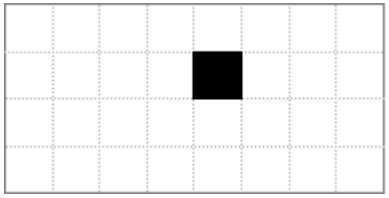
```
int x = 1;
int y = 1;
int v = 1;

// setup

void draw() {
  point(x, y);
  x = x + v;
  if (x > width) {
    x = 0;
  }
}
```

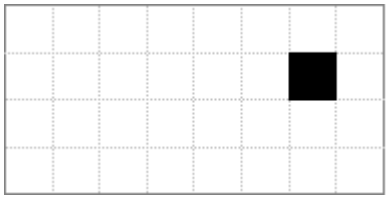
Movement: Collision detection

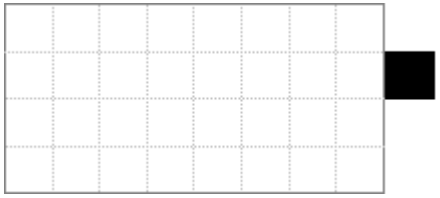


```
int x = 1;
int y = 1;
int v = 1;

// setup

void draw() {
  point(x, y);
  x = x + v;
  if (x == width) {
    x = 0;
  }
}
```

Movement: Collision detection

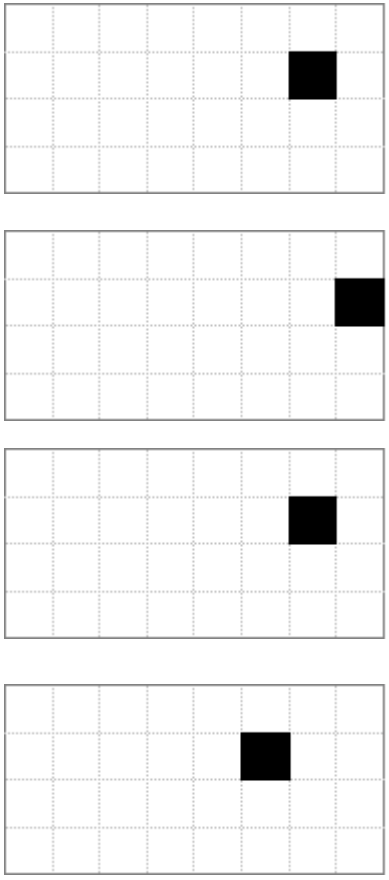






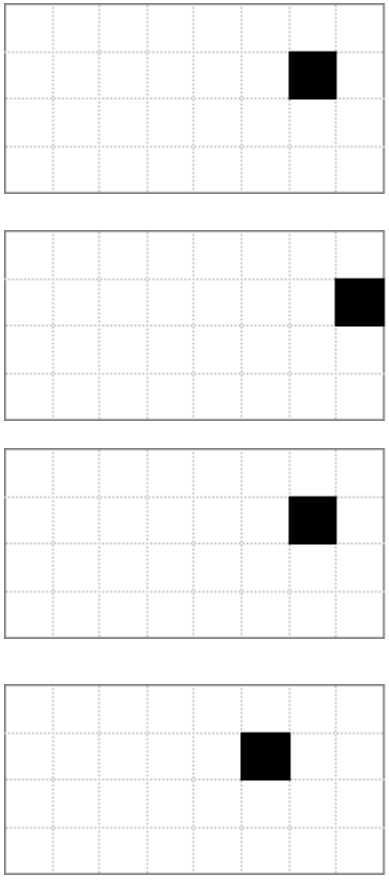
```
int x = 1;
int y = 1;
int v = 2;

// setup

void draw() {
  point(x, y);
  x = x + v;
  if (x == width) {
    x = 0;
  }
}
```

Movement: Collision detection



```
int x = 1;
int y = 1;
int v = 1;

// setup

void draw() {
  point(x, y);
  x = x + v;
  if (x >= width) {
    ??
  }
}
```

Movement: Collision detection



```
int x = 1;
int y = 1;
int v = 1;

// setup

void draw() {
  point(x, y);
  x = x + v;
  if (x >= width) {
    v = -1;
  }
}
```

Example: Movement1

Simple pattern creation: An example

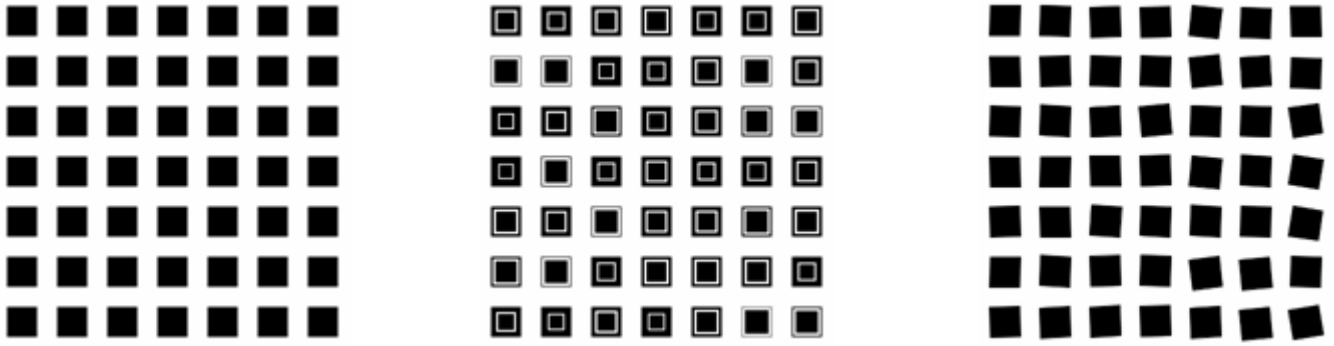
Example: Pattern1

Exercises

E6: Copy the sketch Pattern_Template and modify it to create different patterns.

Variation: Make it interactive so that the results depends on mouse movements.

Variation: Use `random()` to randomize the graphic.



Boolean variables

```
boolean upper = mouseY < height/2;
if (upper) {
      fill(255, 0, 0);
}
else {
      fill(0, 255, 0);
}
ellipse(mouseX, mouseY, 10, 10);
```

Example: IfElseShowcase

Conditionals & Interaction

```
if (mousePressed) {
    point(100, 200);
}
```

Conditionals & Interaction

```
if (mousePressed) {
     x = x + 1;
}
ellipse(x, y, 10, 10);
```

Combined conditions

```
if (mouseX > 50) {
     fill(255, 0, 0);
}
rect(50, 0, 100, height);
```

Example: If2

Combined conditions

```
if (mouseX > 50 && mouseX < 150) {
     fill(255, 0, 0);
}
rect(50, 0, 100, height);
```

Combined conditions

```
if (mouseX > 50 && mouseX < 150) {
     fill(255, 0, 0);
}
else {
     fill(255);
}
rect(50, 0, 100, height);
```

Logical operators

&&	and
||	or
!	not

Exercises

E8: Create a simple drawing program. A visual element should be drawn at the mouse position if the user has pressed a mouse button.

Variation: Use `mouseButton` to draw different shapes dependent on which mouse button the user has pressed.

E9: Draw an ellipse which increases its size as long as the mouseButton is pressed.

Variaton: Make other visual variables dependent on the size (strokeWeight, colour, transparency…)

Exercises

E10: Create a button with two states: When the mouse is over and when it is out, again.

Variation: Implement mouse click, too.

Variation: Use this button to activate a behaviour.

Conditionals & Interaction

```
boolean active = false;
if (mousePressed) {
     active = true;
}
if (active) {
     ellipse(x, y, 10, 10);
}
```

Assignment

A1: Create three buttons. Each of it should trigger some action.

- As group of 2 students
- Make sketches (on paper) to discuss about your idea
- Thursday: brief presentation