(Inter)Active Scripts

Creative Coding & Generative Art in Processing 2
Ira Greenberg, Dianna Xu, Deepak Kumar

Slides revised by Michael Goldwasser

Static Program Structure

```
// Create and set canvas
size(width, height);
smooth();
background(color);

// Draw something
…
// Draw something else
…
// etc.
```

(Inter)Active Program Structure

From this point forward, most Processing programs we write will have the following structure:

```
<Declare variables>

void setup() {

   <initial canvas set up goes here>

} // setup()


void draw() {

   <drawing stuff goes here>

} // draw()
```


Program Structure: Active Mode

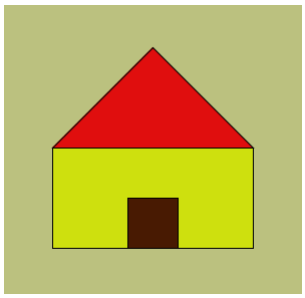
Most Processing programs we will write will have the following structure:

```
// Draw a simple house
void setup() {
  // Create and set canvas
  size(300, 300);
  smooth();
  background(187, 193, 127);
} // setup()

void draw() {
  // wall
  fill(206, 224, 14);
  rect(50, 150, 200, 100);

  // Draw Door
  fill(72, 26, 2);
  rect(125, 200, 50, 50);

  // Draw roof
  fill(224, 14, 14);
  triangle(50, 150, 150, 50, 250, 150);
} // draw()
```



Processing: Dynamic Sketches

```
// Draw a simple house
void setup() {
  // Create and set canvas
  size(300, 300);
  smooth();
  background(187, 193, 127);
} // setup()

void draw() {
  // wall
  fill(206, 224, 14);
  rect(50, 150, 200, 100);

  // Draw Door
  fill(72, 26, 2);
  rect(125, 200, 50, 50);

  // Draw roof
  fill(224, 14, 14);
  triangle(50, 150, 150, 50, 250, 150);
} // draw()
```

Code Block:
{
  …
  …
}

Processing: Dynamic Sketches

```
// Draw a simple house
void setup() {
  // Create and set canvas
  size(300, 300);
  smooth();
  background(187, 193, 127);
} // setup()

void draw() {
  // wall
  fill(206, 224, 14);
  rect(50, 150, 200, 100);

  // Draw Door
  fill(72, 26, 2);
  rect(125, 200, 50, 50);

  // Draw roof
  fill(224, 14, 14);
  triangle(50, 150, 150, 50, 250, 150);
} // draw()
```

setup() block:

Commands here are executed once each time a sketch is played.

draw() block:

Commands here are repeated ~60 times/sec.

Processing: Dynamic Sketches

```
// Draw a simple house
void setup() {
  // Create and set canvas
  size(300, 300);
  smooth();
  background(187, 193, 127);
} // setup()

void draw() {
  // Wall
  fill(206, 224, 14);
  rect(50, 150, 200, 100);

  // Draw Door
  fill(72, 26, 2);
  rect(125, 200, 50, 50);

  // Draw roof
  fill(224, 14, 14);
  triangle(50, 150, 150, 50, 250, 150);
} // draw()
```

But…

What are these???

For now…
Necessary syntax

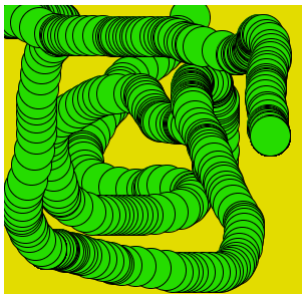
More later…

Something More Interesting…

```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);
color color3 = color(0);

void setup() {
  // create and set canvas
  size(300, 300);
  smooth();
  background(color1);
  stroke(color3);
  fill(color2);
} // setup()

void draw() {
  ellipse(mouseX, mouseY, 40, 40);
} // draw()
```



Predefined variables: pmouseX, pmouseY

```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);
color color3 = color(0);

void setup() {
  // create and set canvas
  size(300, 300);
  smooth();
  background(color1);
  stroke(color2);
  strokeWeight(5);
} // setup()


void draw() {
  line(pmouseX, pmouseY, mouseX, mouseY);
} // draw()
```

Events: More Interactivity

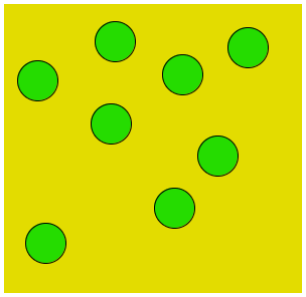
```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);
color color3 = color(0);

void setup() {
  // create and set canvas
  size(300, 300);
  smooth();
  background(color1);
  stroke(color3);
  fill(color2);
} // setup()


void draw() {
  // nothing here, but is required
} // draw()

void mousePressed() {
  ellipse(mouseX, mouseY, 40, 40);
} // mousePressed()
```

Circles are drawn
ONLY when mouse is pressed.



Something More Interesting…

```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);
color color3 = color(0);

void setup() {
  // create and set canvas
  size(300, 300);
  smooth();
  background(color1);
  stroke(color3);
  fill(color2);
} // setup()


void draw() {
  ellipse(mouseX, mouseY, 40, 40);
} // draw()
```

What happens when…

You move the `background(…)` command to draw()?

Controlling Frame Rate

`frameRate(N);`
Changes frame rate to N times/ second

```
<Declare variables>

void setup() {
   …
   frameRate(30);
} // setup()


void draw() {
   <drawing stuff goes here>
} // draw()
```

`noLoop()`
Controls the use of frame rate.

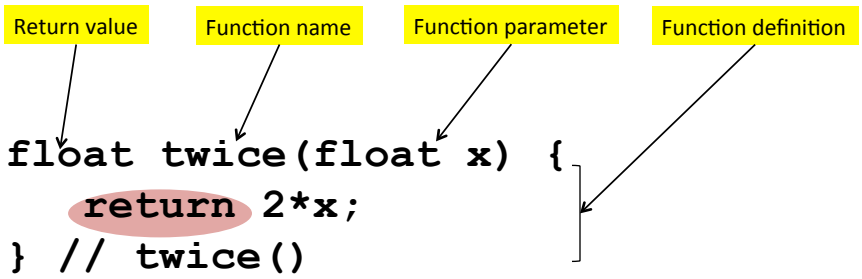
```
<Declare variables>

void setup() {
   …
   noLoop();
} // setup()


void draw() {
   <drawing stuff goes here>
} // draw()
```

Processing: Defining Functions

$y = twice(x) = 2x$



Processing: Defining Functions

Syntax:

```
returnType functionName(parameters) {
   …
   return expression;
}
```

Example:

```
float twice(float x) {
   return 2*x;
} // twice()
```

Use:

```
y = twice(5);
```

Defining Functions: void

Use void as *returnType* when no value is returned.

Syntax:

```
returnType functionName(parameters) {
   …
   return expression;
}
```

Example:

```
void square(float x, float y, float side) {
   rectMode(CORNER);
   rect(x, y, side, side);
} // square()
```

Use:

```
square(50, 50, 100); // draws a 100x100 square at 50, 50
```

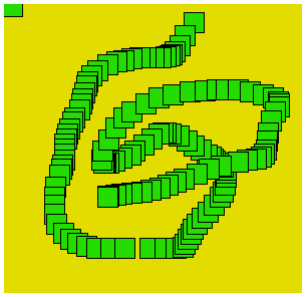

Program Structure: Functions

```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);

void setup() {
  // create and set up canvas
  size(300, 300);
  smooth();
  background(color1);
} // setup()


void draw() {
  fill(color2);
  square(mouseX, mouseY, 20);
} // draw()

void square(float x, float y, float side) {
  rectMode(CORNER);
  rect(x, y, side, side);
} // square()
```



Variables & Scope

```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);

void setup() {
  // create and set up canvas
  size(300, 300);
  smooth();
  background(color1);
} // setup()


void draw() {
  fill(color2);
  square(mouseX, mouseY, 20);
} // draw()

void square(float x, float y, float side) {
  rectMode(CORNER);
  rect(x, y, side, side);
} // square()
```

Global Variables

Either pre-defined Or defined at top

Are visible everywhere In the program

Variables & Scope

```
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);

void setup() {
  // create and set up canvas
  size(300, 300);
  smooth();
  background(color1);
} // setup()


void draw() {
  fill(color2);
  square(mouseX, mouseY, 20);
} // draw()

void square(float x, float y, float side) {
  rectMode(CORNER);
  rect(x, y, side, side);
} // square()
```

Local Variables

Either parameters Or defined inside blocks

Are visible ONLY in the block After they are defined

Processing: Pre-defined Math Functions

- **Calculation**
  abs(), ceil(), constrain(), dist(), exp(), floor(), lerp()
  log(), mag(), map(), max(), min(), norm(), pow()
  round(), sq(), sqrt()
- **Trigonometry**
  acos(), asin(), atan(), atan2(), cos(), degrees(),
  radians(), sin(), tan()
- **Random**
  noise(), noiseDetail(), noiseSeed(), random(),
  randomGaussian(), randomSeed()

Example: Using random()

```
void setup() {// Create and set canvas
  size(300, 300);
  smooth();
  background(255);
} // setup()

void draw() {
  stroke(0);
  rect(random(width),
       random(height),
       random(5, 20),
       random(5, 20));
} // draw();
```