Better Program Design

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

Slides revised by Michael Goldwasser

Variables: Naming Values

• Values
  42, 3.14159, 2013, “Hi, my name is Joe!” , true, false, etc.
  
  – Numbers
    • Integers
      int meaningOfLife = 42;
      int year = 2013;
    • Floating point numbers
      float pi = 3.14159;
  
  – Strings
    String greeting = “Hi, my name is Joe!”;
  
  – Boolean
    boolean keyPressed = true;
Variables: Naming Values

Variables have a Type

• Values
  42, 3.14159, 2013, “Hi, my name is Joe!”, true, false, etc.
  – Numbers
    • Integers
      int meaningOfLife = 42;
      int year = 2013;
    • Floating point numbers
      float pi = 3.14159;
  – Strings
    String greeting = “Hi, my name is Joe!“;
  – Boolean
    boolean keyPressed = true;

Variables: Naming Values

Variables have a Name

• Values
  42, 3.14159, 2013, “Hi, my name is Joe!”, true, false, etc.
  – Numbers
    • Integers
      int meaningOfLife = 42;
      int year = 2013;
    • Floating point numbers
      float pi = 3.14159;
  – Strings
    String greeting = “Hi, my name is Joe!“;
  – Boolean
    boolean keyPressed = true;
Variables: Naming Rules & Conventions

- Names begin with a letter, an underscore (_), or a dollar sign ($)
  Examples: weight, _meaningOfLife, $value

- Names may include numbers, but only after the initial character
  Examples: value1, score5, $bestFriends

- No spaces are permitted in names
  Examples: value1, dollar_sign

- Processing Conventions
  - Names begin with a lowercase letter
    Example: meaningOfLife, highestScore
  - Constants are written in all caps
    Example: DAYS_IN_WEEK, PI

Variables: Declarations & Initialization

- Declaring variables

  ```java
  int meaningOfLife;
  int year;
  float pi;
  String greeting;
  boolean keyPressed;
  ```

- Initializing values in declarations

  ```java
  int meaningOfLife = 42;
  int year = 2013;
  float pi = 3.14159;
  String greeting = "Hi, my name is Joe!";
  boolean keyPressed = true;
  ```
The color type

• Processing has a type called color

```java
color firebrick = color(178, 34, 34);
color chartreuse = color(127, 255, 0);
color fuchsia = color(255, 0, 255);

fill(firebrick);
rect(50, 100, 75, 125);
```

Expressions: Doing Arithmetic

• Assignment statement

  `<variable> = <expression>;`

  Examples:

  ```java
  meaningOfLife = 42;
  area = length * height;
  perc = statePop/totalPop*100.0;
  ```

• Operators

  + (addition)
  - (subtraction)
  * (multiplication)
  / (division)
  % (modulus)

  Example:

  ```java
  mouth_x = ( (leftIris_x + irisDiam)/2 + eyeWidth )/4;
  ```
Arithmetic with int and float values

int x = 42; \hspace{1em} \textit{vs} \hspace{1em} \text{int x = 42.0;}

float x = 42.0 \hspace{1em} \textit{vs} \hspace{1em} \text{float x = 42;}

float x = 7/2; \hspace{1em} \textit{vs} \hspace{1em} \text{float x = 7.0/2.0;}

- Type of variable is important and determines the value that can be assigned to it.
- Result of division depends upon operands

\begin{itemize}
\item \text{int/int} \hspace{1em} \text{yields an integer result}
\item \text{float/int} \hspace{1em} \text{yields a float result}
\item \text{int/float} \hspace{1em} \text{yields a float result}
\item \text{float/float} \hspace{1em} \text{yields a float result}
\end{itemize}
Using Variables

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

// 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);

A Better House Sketch

// Draw a simple house
int houseX = 50;                 // bottom left corner of house
int houseY = 250;                // overall width and height of house
int houseWidth = 200;
int houseHeight = 200;
int wallHeight = houseHeight/2;   // height of wall is 1/2 of house height
int roofHeight = houseHeight/2;
int doorHeight = houseHeight/4;
int doorWidth = houseWidth/4;

// Create and set canvas
size(300, 300);
smooth();
background(187, 193, 127);

// wall
fill(206, 224, 14);
rect(houseX, houseY - wallHeight, 
houseWidth, wallHeight);

// Draw Door
fill(72, 26, 2);
rect(houseX + houseWidth/2 - doorWidth/2, houseY-doorHeight, 
doorWidth, doorHeight);

// Draw roof
fill(224, 14, 14);
triangle(houseX, houseY - wallHeight, 
houseX+houseWidth/2, houseY-houseHeight, 
houseX+houseWidth, houseY-wallHeight);
A Better House Sketch

```java
// Draw a simple house
int houseX = 50; // bottom left corner of house
int houseY = 250;
int houseHeight = 100; // overall width and height of house
int houseWidth = 100;

int wallHeight = houseHeight/2; // height of wall is 1/2 of house height
int roofHeight = houseHeight/2;
int doorHeight = houseHeight/4;
int doorWidth = houseWidth/4;

// Create and set canvas
size(500, 500);
smooth();
background(187, 193, 127);

// wall
fill(206, 224, 14);
rect(houseX, houseY - wallHeight,
    houseWidth, wallHeight);

// Draw Door
fill(72, 26, 2);
rect(houseX + houseWidth/2 - doorWidth/2,
    houseY - doorHeight,
    doorWidth, doorHeight);

// Draw roof
fill(224, 14, 14);
triangle(houseX, houseY - wallHeight,
    houseX + houseWidth/2, houseY - houseHeight,
    houseX + houseWidth, houseY - wallHeight);
```

Processing: Predefined Variables

- **width, height**
The width & height of the canvas used in the sketch

- **PI, HALF_PI, TWO_PI**
  For different values of π. Note that
  
  \[
  \text{HALF_PI} = \frac{\text{PI}}{2} \\
  \text{TWO_PI} = 2 \times \text{PI}
  \]

- **displayWidth, displayHeight**
The width and height of the monitor being used. This is useful in running fullscreen sketches using:

  ```java
  size(displayWidth, displayHeight);
  ```

- **mouseX, mouseY**
The current mouse location in sketch (...coming soon!)
Additional Bells and Whistles

Basic Shapes: Arcs

• What is an arc?
Basic Shapes: Arcs

\[ \text{arc}(x, y, \text{width}, \text{height}, \text{startAngle}, \text{endAngle}); \]

- degrees vs radians

\[ \text{noFill();} \]
\[ \text{stroke}(255, 0, 0); \]
\[ \text{arc}(200, 200, 150, 150, 0, \pi); \]

\[ \text{fill}(255, 255, 0); \]
\[ \text{stroke}(255, 0, 0); \]
\[ \text{arc}(200, 200, 150, 150, 0, \pi); \]
Basic Shapes: Arcs

```javascript
start = 30 degs
end = 120 degs
```

```javascript
start = 2 degs
end = 130 degs
```

```javascript
start = 15 degs
end = 160 degs
```

```javascript
start = 18 degs
end = 190 degs
```

```javascript
start = 21 degs
end = 200 degs
```

```javascript
start = 24 degs
end = 340 degs
```

```javascript
start = 27 degs
end = 350 degs
```

```javascript
start = 30 degs
end = 360 degs
```

```javascript
start = 33 degs
end = 370 degs
```

```javascript
start = 36 degs
end = 390 degs
```

Basic Shapes: Quadrilaterals

```javascript
quad(x1, y1, x2, y2, x3, y3, x4, y4);
```

```javascript
fill(123, 17, 80);
quad(60, 60, 180, 60, 180, 180, 60, 180);
```
Drawing Text

text(string, x, y);
Draws string with bottom left corner at x, y

textSize(fontSize);
Can be used to specify font size

fill(…) used to specify color

textAlign(…) used to control vertical, horizontal alignment
(see Processing reference)