Variables and Control Structures

CS 110
Eric Eaton and Paul Ruvolo
Assignment 2

• It should have a background generated using randomness and iteration (for or while loops) [covered today]

• Each time the user presses a key, the program should erase whatever is displayed and redraw the random background. The background should be different with each subsequent keypress. [covered today]

• As the user clicks on the sketch, it should draw an object at that location. Something about the physical shape of this object must change based on its location. [already covered]
Review: Variables

- A **name** to which data can be assigned
- A variable name is **declared** as a specific **data type**
- Names must begin with a letter, “_” or “$” and can contain letters, digits, “_” and “$”

```java
boolean bReady = true;
int i;
int j = 12;
float fSize = 10.0;
color _red = color(255,0,0);
String name123 = “Fred”;
PIImage img;
```
Review: Variable Uses

• Use a value throughout your program,
  – but allow it to be changed

• As temporary storage for an intermediate computed result

• ... etc
## Primitive Data Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
<th>Default</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>{ true, false }</td>
<td>false</td>
<td>?</td>
</tr>
<tr>
<td>byte</td>
<td>{ 0..255 }</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>int</td>
<td>{ -2,147,483,648 .. 2,147,483,647 }</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>long</td>
<td>{ -9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807 }</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>float</td>
<td>{ -3.40282347E+38 .. 3.40282347E+38 }</td>
<td>0.0</td>
<td>4</td>
</tr>
<tr>
<td>double</td>
<td>much larger/smaller</td>
<td>0.0</td>
<td>8</td>
</tr>
<tr>
<td>color</td>
<td>{ #00000000 .. #FFFFFFFF }</td>
<td>black</td>
<td>4</td>
</tr>
<tr>
<td>char</td>
<td>a single character 'a', 'b', ...</td>
<td>'\u0000'</td>
<td>2</td>
</tr>
</tbody>
</table>
Conditionals: if-else-if-statement

if ( boolean_expression_1 ) {
    statements;
} else if ( boolean_expression_2 ) {
    statements;
} else if ( boolean_expression_3 ) {
    statements;
} else {
    statements;
}
void setup() {
    size( 500, 500 );
    smooth();
}

void draw() {
    if ( mouseX > 100 )
    {
        background( 255, 0, 0 );
    } else if ( mouseX > 200 )
    {
        background( 0, 0, 255 );
    }
}
An Aside: Handling Keyboard Events

```java
void keyPressed() {
    // Called each time a key is pressed
}

void keyReleased() {
    // Called each time a key is released
}

void keyTyped() {
    // Called when a key is pressed
    // Called repeatedly if the key is held down
}
```
keyCode vs. key

key
  – A built-in variable that holds the character that was just typed at the keyboard

keyCode
  – A built-in variable that holds the code for the keyboard key that was touched

All built-in keyboard interaction functions …
  • Set keyCode to the integer that codes for the keyboard key
  • Set key to the character typed
  • All keyboard keys have a keyCode value
  • Not all have a key value (can you think of an example?)
### ASCII - American Standard Code for Information Interchange

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td></td>
<td>!</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
<td>/</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
</tr>
<tr>
<td>60</td>
<td>&lt;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>70</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td>O</td>
</tr>
<tr>
<td>80</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>90</td>
<td>Z</td>
<td>[</td>
<td>\</td>
<td>]</td>
<td>^</td>
<td>_</td>
<td>'</td>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>100</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
</tr>
<tr>
<td>110</td>
<td>n</td>
<td>o</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
</tr>
<tr>
<td>120</td>
<td>x</td>
<td>y</td>
<td>z</td>
<td>{</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>}</td>
<td>~</td>
</tr>
<tr>
<td>130</td>
<td>,</td>
<td>f</td>
<td>&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>%</td>
</tr>
<tr>
<td>140</td>
<td>Œ</td>
<td>Ž</td>
<td>Ÿ</td>
<td>'</td>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>•</td>
<td></td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>—</td>
<td>—</td>
<td>~</td>
<td>™</td>
<td>Š</td>
<td>Ž</td>
<td>Ÿ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>i</td>
<td>c</td>
<td>£</td>
<td>€</td>
<td>¥</td>
<td>€</td>
<td>§</td>
<td>&quot;</td>
<td>©</td>
<td></td>
</tr>
<tr>
<td>170</td>
<td>à</td>
<td>a</td>
<td>s</td>
<td>±</td>
<td>²</td>
<td>²</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>ñ</td>
<td>ñ</td>
<td>ñ</td>
<td>ñ</td>
<td>ñ</td>
<td>ñ</td>
<td>ñ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>190</td>
<td>%</td>
<td>¾</td>
<td>À</td>
<td>Á</td>
<td>Â</td>
<td>Á</td>
<td>À</td>
<td>Æ</td>
<td>Ç</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
</tr>
<tr>
<td>210</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
</tr>
<tr>
<td>220</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
</tr>
<tr>
<td>230</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
</tr>
<tr>
<td>240</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
</tr>
<tr>
<td>250</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
<td>Õ</td>
</tr>
</tbody>
</table>
List of Key Codes

• Check processing API: 
  http://processing.org/reference/keyCode.html

• Built-in key code variables:
  – UP, DOWN, LEFT, RIGHT

• Others can be found at: 
  http://docs.oracle.com/javase/6/docs/api/java/awt/event/KeyEvent.html

• List of numerical key codes: 
  http://home-1.worldonline.nl/~bmc88/java/sbook/021.html
Conditionals: switch-statement

- Works like a if-else statement.
- Convenient for large numbers of value tests.

```java
switch( expression ) {
  case label1:         // label1 equals expression
    statements;
    break;
  case label2:         // label2 equals expression
    statements;
    break;
  default:             // Nothing matches
    statements;
}
```
Conditionals: switch-statement

- Works like an if-else statement.
- Convenient for large numbers of value tests.

```java
switch( expression ) {
    case label1:         // label1 equals expression
        statements;
        break;
    case label2:         // label2 equals expression
        statements;
        break;
    default:             // Nothing matches
        statements;
}
```

Any idea what this does?
void setup() {
    size(500, 500);
    smooth();
}

void draw() {}
int posX = 250;
int posY = 250;
int deltaX = 0;
int deltaY = 0;

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

void draw() {
    background(255);

    posX = posX + deltaX;
    posY = posY + deltaY;

    if (posX < 0)     posX = 0;
    if (posX > width) posYX = width;
    if (posY < 0)     posY = 0;
    if (posY > height) posY = height;

    ellipse(posX, posY, 50, 50);
}

void keyPressed() {
    switch (keyCode) {
    case LEFT:
        deltaX = -2;
        deltaY = 0;
        break;
    case RIGHT:
        deltaX = 2;
        deltaY = 0;
        break;
    case UP:
        deltaY = -2;
        deltaX = 0;
        break;
    case DOWN:
        deltaY = 2;
        deltaX = 0;
        break;
    }
}
Introduction to Loops

• What is a loop? Executing the same code over and over again.
• We are already using loops, you just might not know it.
• How would I write a program to draw many random lines?
Introduction to Loops

• What if I only want to draw 200 lines and then stop?
Another Program

• What if we don’t want to wait for the lines to show up? How can I modify the program to do that?
We Need Something More Flexible: Iteration

Repetition of a program block
• Iterate when a block of code is to repeated multiple times.

Options
• The while-loop
• The for-loop
Iteration: while-loop

```java
while ( boolean_expression ) {
    statements;
    // continue;
    // break;
}
```

- Statements are repeatedly executed while the boolean expression evaluates to true;
- To break out of a while loop, call `break`;
- To stop execution of statements and start again, call `continue`;
200 Random Lines

size(500,500);
background(255);
int i = 0;
while (i < 200) {
    stroke(random(0,255),random(0,255),random(0,255));
    line(random(0,width),random(0,height),random(0,width),random(0,height));
    i = i+1;
}
Doing something different in each “iteration” of the loop

• How would I write code to generate the following image in processing?
void setup() {
    size(500, 500);
    smooth();

    float diameter = 500.0;
    while (diameter > 1.0) {
        ellipse(250, 250, diameter, diameter);
        diameter = diameter * 0.9;
    }
}

void draw() { }

What does this do?
An aside ... Operators

+, −, *, / and ...

```plaintext
i++;    equivalent to    i = i + 1;
i += 2; equivalent to    i = i + 2;
i--;    equivalent to    i = i - 1;
i -= 3; equivalent to    i = i - 3;
i *= 2; equivalent to    i = i * 2;
i /= 4; equivalent to    i = i / 4;
```

i % 3; the remainder after i is divided by 3 (modulo)