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 "$" and can container letters, digits, "_" and "$"

```
boolean bReady = true;
int i;
int j = 12;
float fSize = 10.0;
color _red = color(255,0,0);
String name123 = "Fred";
PImage 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

Type	Range	Default	Bytes
boolean	{ true, false }	false	?
byte	{ 0..255 }	0	1
int	{ -2,147,483,648 .. 2,147,483,647 }	0	4
long	{ -9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807 }	0	8
float	{ -3.40282347E+38 .. 3.40282347E+38 }	0.0	4
double	*much larger/smaller*	0.0	8
color	{ #00000000 .. #FFFFFFFF }	*black*	4
char	*a single character* 'a', 'b', …	'\u0000'	2

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 );
  }

}
```

What does this do?

An Aside: Handling Keyboard Events

```
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

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		€	
130	‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹
140	Œ		Ž			‘	’	“	”	•
150	–	—	˜	™	š	›	œ		ž	Ÿ
160		¡	¢	£	¤	¥	¦	§	¨	©
170	ª	«	¬	-	®	¯	°	±	²	³
180	´	µ	¶	·	¸	¹	º	»	¼	½
190	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç
200	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ
210	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û
220	Ü	Ý	Þ	ß	à	á	â	ã	ä	å
230	æ	ç	è	é	ê	ë	ì	í	î	ï
240	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù
250	ú	û	ü	ý	þ	ÿ				

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.

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

```
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() {}

void keyPressed() {
  switch(key)
  {
    case 'l':
    case 'L':
      println("Turning left");
      break;
    case 'r':
    case 'R':
      println("Turning right");
      break;
  }
}
```

What does this do?

```
int positionX = 250;
int positionY = 250;
int deltaX = 0;
int deltaY = 0;

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

void draw() {
  background(255);

  positionX = positionX + deltaX;
  positionY = positionY + deltaY;

  if (positionX < 0)
    positionX = 0;
  if (positionX > width)
    positionX = width;
  if (positionY < 0)
    positionY = 0;
  if (positionY > height)
    positionY = height;

  ellipse(positionX, positionY, 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

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

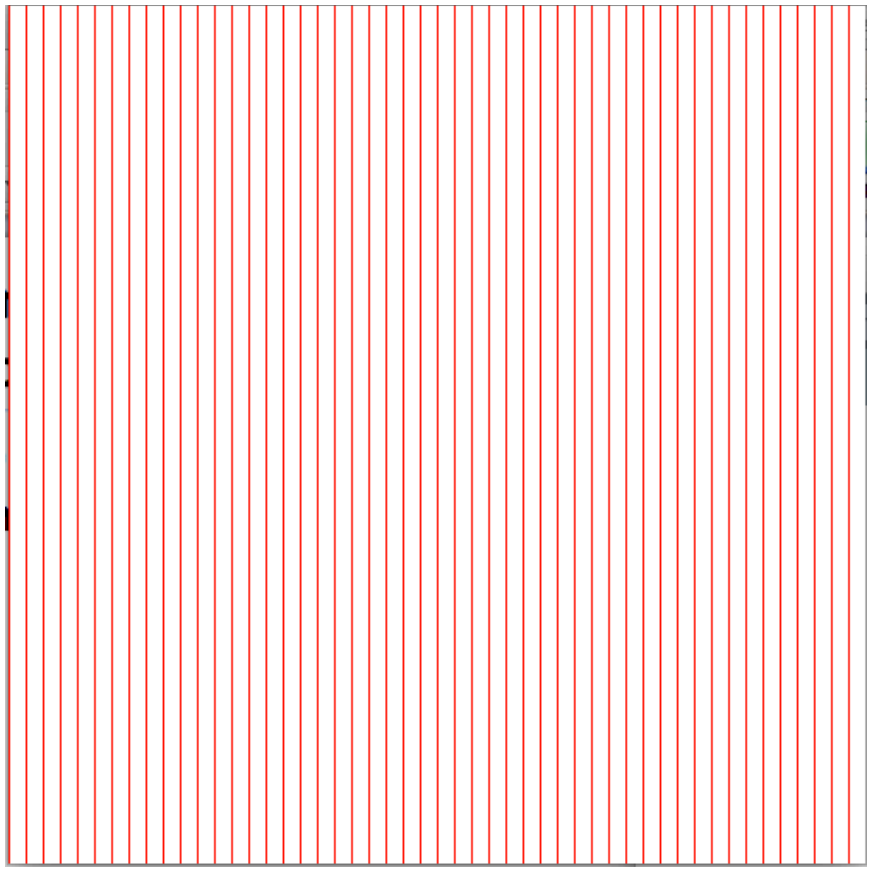
- Statements are repeatedly executed while the boolean expression evaluates to true;
- To break out of a while loop, call **break;**
- To stop execution of statements and start again, call **continue;**

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?

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

  float diameter = 500.0;
  while ( true ) {
       ellipse( 250, 250, diameter, diameter);
       diameter = diameter * 0.9;
       if (diameter <= 1.0 ) break;
  }
}

void draw() { }
```

An aside … Operators

`+`, `-`, `*`, `/` and …

`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)