Jukebox

simple images and sound, text, Booleans, conditionals, more functions

Variables vs. Functions

Variables

- Used to store *data* (i.e. some value)
- Needs to be declared and assigned before it can be used
- Reasons to use them:
 - Avoid repetition
 - Make code more readable
 - Allow for a change in values over time
 - Store values not known until the program runs

Functions

- Used to store code that does something
- Needs to be defined before it can be called/used
- Reasons to use them:
 - Avoid repetition
 - Make code more readable
 - Make code more portable
 - Allow for abstraction



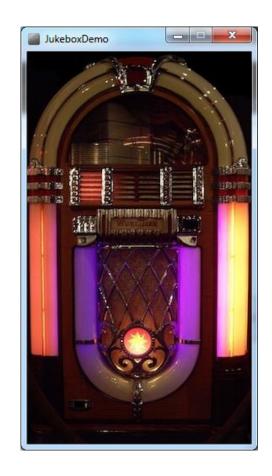
Jukebox!



Creating a Plan...

- 1. Load and draw an image of a jukebox, make window size same as image size
- 2. Load three songs
- 3. Draw three buttons labelled 1, 2, 3
- 4. Determine which button was clicked.
- 5. Play or stop song when an image is clicked.
 - Also stop any other songs playing when a new song plays.
- 6. Flash buttons when song is playing.

Load and draw an image of a jukebox, make window size same as image size



Load three songs using the minim library

Sketch > Import Library... > minim

Draw three buttons labelled 1, 2, 3



```
void drawButton(int x, int y, int buttonNumber)
{
  fill(240);
  rect(x, y, buttonWidth, buttonHeight);

fill(0);
  text(buttonNumber, x, y);
}
```

```
void drawButton(int x, int y, int buttonNumber)
{
  fill(240);
  rect(x, y, buttonWidth, buttonHeight);

fill(0);
  text(buttonNumber, x, y);
}
```

function becomes a new command

function (command) name

```
void drawButton (int x, int y, int buttonNumber)
{
  fill(240);
  rect(x, y, buttonWidth, buttonHeight);

fill(0);
  text(buttonNumber, x, y);
}
```

function (command) parameters

```
void drawButton (int x, int y, int buttonNumber)
{
  fill(240);
  rect(x, y, buttonWidth, buttonHeight);

fill(0);
  text(buttonNumber, x, y);
}
```

```
void drawButton(int x, int y, int buttonNumber)
{
  fill(240);
  rect(x, y, buttonWidth, buttonHeight);
  fill(0);
  text(buttonNumber, x, y);
}
```

parameters are variables that get assigned when the function is called, and can be used inside the function

function returns this type, where void means it returns nothing

```
void drawButton(int x, int y, int buttonNumber)
{
  fill(240);
  rect(x, y, buttonWidth, buttonHeight);

fill(0);
  text(buttonNumber, x, y);
}
```

```
void drawButton(int x, int y, int buttonNumber)
{
    fill(240);
    rect(x, y, buttonWidth, buttonHeight);
    fill(0);
    text(buttonNumber, x, y);
}
```

function body (what the command will do when you call it)

```
void draw()
{
    // Put the jukebox in the background
    image(backgroundImage, 0, 0);

    // Draw three buttons across the top...
    drawButton(leftButtonX, buttonY, 1);
    drawButton(middleButtonX, buttonY, 2);
    drawButton(rightButtonX, buttonY, 3);
}
```

```
void draw()
{
    // Put the jukebox in the background
    image(backgroundImage, 0, 0);

    // Draw three buttons across the top...
    drawButton(leftButtonX, buttonY, 1);
    drawButton(middleButtonX, buttonY, 2);
    drawButton(rightButtonX, buttonY, 3);
}
```

calling our function like it was a built in command

```
void draw()
{
    // Put the jukebox in the background
    image(backgroundImage, 0, 0);

    // Draw three buttons across the top...
    drawButton(leftButtonX, buttonY, 1);
    drawButton(middleButtonX, buttonY, 2);
    drawButton(rightButtonX, buttonY, 3);
}
```

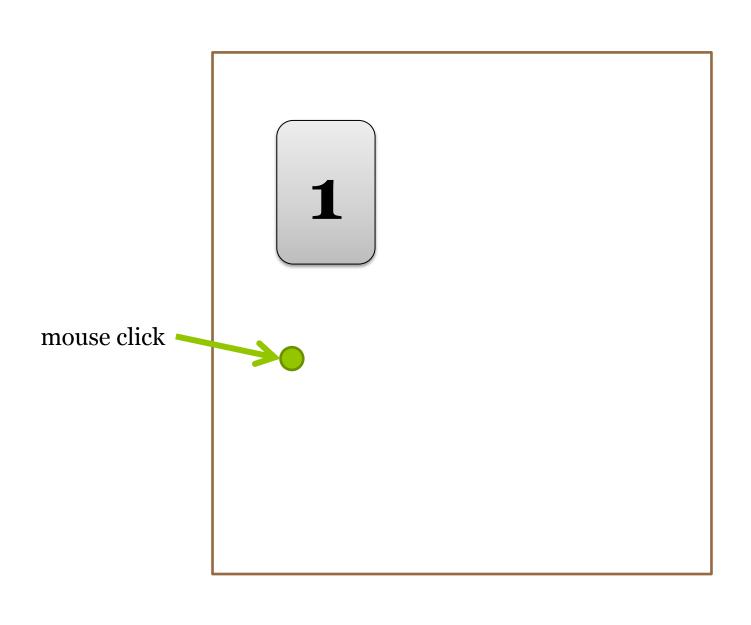
for this call to drawButton, the parameter x is assigned the value taken from leftButtonX

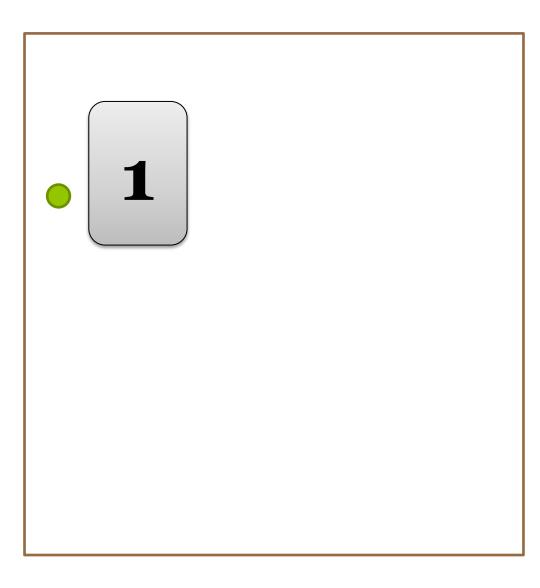
Determine which button was clicked.

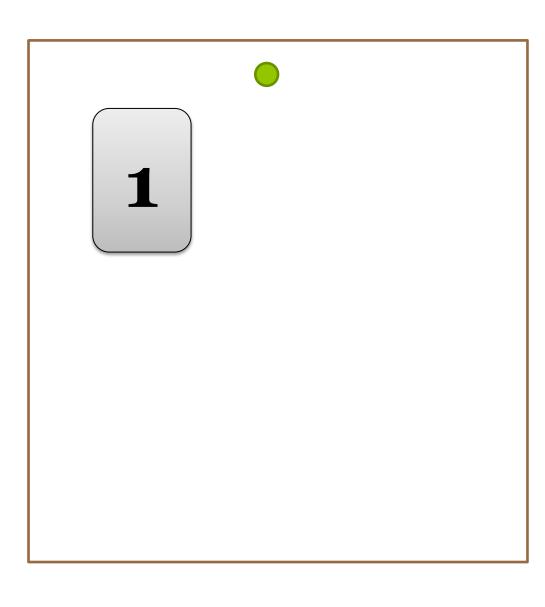
2 3

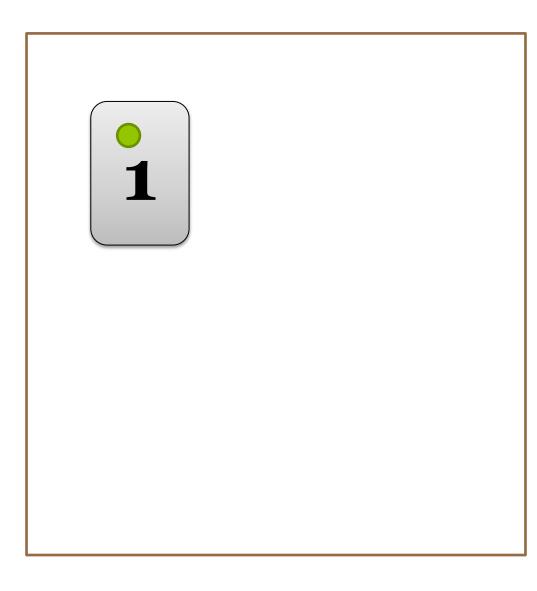
1 2 3

Can we use a switch statement?









If Statements

Does the expression evaluate to true?

YES:

Run the code in the body.

NO:

Skip past the if statement.



It is raining



Wear a raincoat



Grade is at least 50



Pass the class

...examples...

and

Everything has to be true

and

true and true = true true and false and true = false false and false = false

or

Only one thing has to be true

or

true or true = true true or false or true = true false or false = false

not

Flips a Boolean value

not

not true = false not false = true

not

not $(\mathbf{a} \text{ and } \mathbf{b}) = (\text{not } \mathbf{a}) \text{ or } (\text{not } \mathbf{b})$ not $(\mathbf{a} \text{ or } \mathbf{b}) = (\text{not } \mathbf{a}) \text{ and } (\text{not } \mathbf{b})$



if not married and not engaged and like her:

should put a ring on it

if not (married or engaged) and like her:

should put a ring on it

I am buying a movie ticket *and* I am a student



I will get a discount on the price iff

My percentage is at least 77 and my percentage is at most 79



My grade is B+

if

The battery is dead *or* there is no gas



The car will not start



(mouseX >= leftButtonX (buttonWidth/2)
and
(mouseX <= leftButtonX +
(buttonWidth/2)</pre>



Toggle song and button flashing for left button

Exercise:

What will the result of this expression be?

```
(true and not
  (false or
      (true and not false)))
```

Logical Operators in Processing

and: &&

or:

```
not:!

equals: ==
less than: <
less than or equal: <=
greater than: >
greater than or equal: >=
```

...example...

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
{
    // play song if button was clicked
}</pre>
```

if statement

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
{
    // play song if button was clicked
}</pre>
```

Expression that results in a Boolean value (true or false)

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
{
    // play song if button was clicked
}</pre>
```

"and" – allows for more complex logic

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
{
    // play song if button was clicked
}</pre>
```

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
{
    // play song if button was clicked
}</pre>
```

body of if statement: what to do when the Boolean expression is true

If-Else Statements

Does the expression evaluate to true?

YES:

Run the code in the body.

NO:

Run the code in the else.

Else-If Statements

Does the expression evaluate to true?

YES:

Run the code in the body.

NO:

Check the next expression.

Does the expression evaluate to true?

YES:

Run the code in the else-if body.

NO:

Run the code in the else, or done.

...example...

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))</pre>
   // Left button clicked
else if (mouseX >= middleButtonX - (buttonWidth/2) &&
         mouseX <= middleButtonX + (buttonWidth/2))</pre>
   // Middle button clicked
else if (mouseX >= rightButtonX - (buttonWidth/2) &&
         mouseX <= rightButtonX + (buttonWidth/2))</pre>
   // Right button clicked
```

```
if
   (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
   // Left
           If this expression is true...
else if (mouseX >= middleButtonX - (buttonWidth/2) &&
         mouseX <= middleButtonX + (buttonWidth/2))</pre>
   // Middle button clicked
else if (mouseX >= rightButtonX - (buttonWidth/2) &&
         mouseX <= rightButtonX + (buttonWidth/2))</pre>
   // Right button clicked
```

```
(mouseX >= leftButtonX - (buttonWidth/2)
    mouseX <= leftButtonX + (buttonWidth/2))
   // Left button clicked
else if
                                       itonWidth/2) &&
        (m)
         ... then run the body, and tonWidth/2))
                   we're done.
   // Middre purrou cricked
else if (mouseX >= rightButtonX - (buttonWidth/2) &&
         mouseX <= rightButtonX + (buttonWidth/2))</pre>
   // Right button clicked
```

```
(mouseX >= leftButtonX - (buttonWidth/2) &&
if
    mouseX <= leftButtonX + (buttonWidth/2))
                 But if this is false...
   // Left b
else if (mouseX >= middleButtonX - (buttonWidth/2) &&
         mouseX <= middleButtonX + (buttonWidth/2))</pre>
   // Middle button clicked
else if (mouseX >= rightButtonX - (buttonWidth/2) &&
         mouseX <= rightButtonX + (buttonWidth/2))</pre>
   // Right button clicked
```

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
   mouseX <= leftButtonX + (buttonWidth/2))</pre>
  // Left button clicked
else if (mouseX >= middleButtonX - (buttonWidth/2) &&
        mouseX <= middleButtonX + (buttonWidth/2))</pre>
                        ...check the next
  // Middle button
                           expression.
mouseX <= rightButtonX + (buttonWidth/2))</pre>
  // Right button clicked
```

```
if (mouseX >= leftButtonX - (buttonWidth/2) &&
    mouseX <= leftButtonX + (buttonWidth/2))
   // Left button clicked
else if (mouseX >= middleButtonX - (buttonWidth/2) &&
         mouseX <= middleButtonX + (buttonWidth/2))</pre>
   // Middle button clicked
else if (mouseX >= rightButtonX - (buttonWidth/2) &&
         mouseX <= rightButtonX + (buttonWidth/2))</pre>
   // Right button clicked
                                 Either zero or one
                                 if/else-if/else body
```

will run

Poll Everywhere Question

What is the output?

```
int x = 5;
if (x < 15)
   if (x < 8)
      println("one");
   else if (true)
      println("two")
   else
      println("three");
else
   println("four");
```

Text 37607

548436: one
548437: two
548438: three
548477: four
548439: (more than one of the above)
548440: (none of the above)

Step 5

Play or stop song when an image is clicked.

Also stop any other songs playing when a new song plays.

Step 6

Flash buttons when song is playing.

```
if (buttonNumber != songPlaying ||
   buttonCounter < timeBetweenButtonChange)
{
   // button drawing code here
}</pre>
```

```
if (buttonNumber != songPlaying ||
    buttonCounter < timeBetweenButtonChange)
{
    // button drawing code here
}</pre>
```

check whether to draw button or hide when flashing

```
if (buttonNumber != songPlaying ||
   buttonCounter < timeBetweenButtonChange)
{
   // button dra
   always true
   when button's
   song is not
      playing</pre>
```

```
if (buttonNumber != songPlaying ||
  buttonCounter < timeBetweenButtonChange)

{
  // button dra only tested when
  this button's
  song is playing</pre>
```