The Imaginary Institute
www.imaginary-institute.com
2Dcourse@imaginary-institute.com

# 2D ANIMATION & INTERACTION COURSE
# WEEK 3: INTERACTION

## WEEK 3: INTERACTION

Week 3 is about building up our programming skills and getting started on writing interactive animations. Now that you have solid control of the fundamentals, we'll start moving much faster to expand our expressive power and control.

This week is where we start to pick up steam. We'll cover a lot of vital material, so make sure you give yourself the time to understand and internalize it. Try getting in the habit of writing lots of tiny little toy programs to try out a new technique or idea. Then mess around with it, exploring what you can do with it and what ideas it stimulates.

As usual, many of the videos have Processing sketches (or programs) associated with them. These sketches fall into two categories:

**Example sketches** are meant for you to look at now, because they illustrate the points we're talking about. These sketches appear in a green section, like this.

**Support sketches** are programs that I wrote for use in the videos. They frequently use advanced techniques that we haven't covered yet. These sketches appear in a gray section, like this.

## GROUP 1: OVERVIEW

We start by looking at this week's context, so you have an overview of where we're going and how the pieces all fit together. Then we'll write a very little game. We'll do it pretty quickly and a lot of the steps will be new, but by the end of this week you'll understand everything we've done.

**W3 G1 V1 Week 3 Overview** *(6m31s)*

We start by looking at this week's context, so you have an overview of where we're going and how the pieces all fit together. We'll see a bunch of new ideas: variables, functions, if statements, and responding to the keyboard.

W3 G1 V2a **A MiniGame Part 1** *(9m10s)*
W3 G1 V2b **A MiniGame Part 2** *(7m9s)*
W3 G1 V2c **A MiniGame Part 3** *(7m42s)*

To show what you can do with this week's material, we'll write a little game together. I'll go through it all rather quickly. By the end of this week everything here will be familiar to you.

**MiniGame.pde**

A ball moves from the left of the screen to the right. Press the spacebar to instantly stop its horizontal motion and cause it to drop down. If it lands inside the gap at the bottom, you score big points!

## GROUP 2: VARIABLES

We'll expand our control of variables in two ways. First, we'll see how to create and use color variables so we don't have to type in numbers every time we change colors. Then we'll look at a number of universally-used shorthands for doing basic numerical operations with variables.

W3 G2 V1a **Color Variables Part 1** *(9m11s)*
W3 G2 V1b **Color Variables Part 2** *(9m50s)*

How to create color variables, and how to get the color components back out of them.

W3 G2 V2a **Using Color Variables Part 1** *(7m18s)*
W3 G2 V2b **Using Color Variables Part 2** *(7m49s)*

The proper technique for changing and combining color variables.

**AddColors.pde**

How to add two color variables.

**DoubleColor.pde**

How to make a color variables twice as bright.

**NestedEllipsesStart.pde**

The hard way to make an ellipse appear to have a hole in it.

**NestedEllipsesFinal.pde**

The easier way to make an ellipse appear to have a hole in it. This approach also makes it much easier to change the colors in the drawing.

W3 G2 V3a **Shorthand Arithmetic Part 1** *(12m36s)*
W3 G2 V3b **Shorthand Arithmetic Part 2** *(4m54s)*

We look at some universally-used shortcuts for doing common arithmetic with numerical variables. Once you've got these under your fingertips you'll use them all the time.

## GROUP 3: FUNCTIONS

We often want to package up bits of code so that we can repeat them in several places without having to retype everything. The mechanism of a *function* lets us do that. Even better, we can tell the function that some of its variables (called *arguments* or *parameters*) should take on particular values each time the function runs. This gives us an enormous amount of expressive power, since the same steps repeated with different values can give us dramatically different results.

W3 G3 V1a **Functions Part 1** *(9m45s)*
W3 G3 V1b **Functions Part 2** *(11m55s)*

How to create our own functions.

**Stacks.pde**

We use a function to draw a stack of three ellipses. By calling the function multiple times, we can easily draw lots of stacks of different colors.

**textAnimator.pde**

Animated text to show how to go from a line of English that describes a function to the Processing version.

W3 G3 V2a **Using Functions Part 1** *(7m29s)*
W3 G3 V2b **Using Functions Part 2** *(9m53s)*

How to use the functions we create.

**scoring.pde**

Using a function to find the score for a very simple game. In this sketch, there's no interaction yet.

**Stacks.pde**

We use a function to draw a stack of three ellipses. By calling the function multiple times, we can easily draw lots of stacks of different colors.

W3 G3 V3a **Locals and Globals Part 1** *(10m11s)*
W3 G3 V3b **Locals and Globals Part 2** *(10m2s)*

Variables can be *local* to a function, meaning they only have a value to lines of code inside that function, or they can be *global* to the whole program, and accessible everywhere.

**FruitCount.pde**

We deliberately create a local variable with the same name as a global variable. This almost always leads to confusion and misbehaving programs.

**MovingEllipse.pde**

We see how to use global variables to remember the position of a circle from one frame to the next. This is a key technique for writing interactive programs.

W3 G3 V4 **Useful Globals** *(7m39s)*

Processing maintains a variety of global variables for us. Here we look at six of the most important ones.

## GROUP 4: IF STATEMENTS

Every time you run an interactive program, you can get a different result. If it's a game, the user might score a goal immediately, or never. If it's a drawing program, the user might draw a snowman, a cupcake, or an abstract field of red blobs. Each time through it's different. A central mechanism for giving your program this kind of flexibility is the *if statement*.

W3 G4 V1 **If Statement Overview** *(3m12s)*

A look at if statements and what they can do for us.

W3 G4 V2 **Booleans** *(6m30s)*

A *boolean* is a data type that can hold only two values: the pre-defined key words `true` or `false`. These booleans are used in tests of all kinds, including if statements.

**ShowBools.pde**

Print out boolean variables, see how to pass them to functions, and how to return them from functions.

W3 G4 V3a **If Statements Part 1** *(10m54s)*
W3 G4 V3b **If Statements Part 2** *(9m52s)*

Using logical tests, we can build if statements that choose whether to execute either one set of statements, or another.

**backgroundColor.pde**

Increase the redness of the background over time.

**textAnimator.pde**

Animate a line of text from an English statement into an if statement in

Processing.

W3 G4 V4a **Logical Tests Part 1** *(7m41s)*
W3 G4 V4b **Logical Tests Part 2** *(6m6s)*

We look at two ways to combine two Booleans: whether both are true (called the AND test) or either is true (called the OR test).

**MouseInBoxesAnd.pde**

We use the mouse to explore logical tests with if statements.

**testDemoWithMouse.pde**

Another example of using if statements to control interactive behavior.

W3 G4 V5a **Using Logical Tests Part 1** *(12m11s)*
W3 G4 V5b **Using Logical Tests Part 2** *(10m54s)*

Using AND and OR tests to determine if a point is inside a given box, if it's inside a given circle, or if it's inside a more complicated shape.

W3 G4 V6a **Combining If Statements Part 1** *(8m26s)*
W3 G4 V6b **Combining If Statements Part 2** *(11m34s)*

Using the *else* clause in an if statement lets us nest together multiple statements to make a compact and easily-understood chain of tests for handling complicated situations.

## GROUP 5: KEYBOARD

Here we see how to write code that responds when the user has pressed a key on the keyboard, and take different actions depending on which key it was. We also see how to respond to when a key is released, and how to handle a few special but common situations.

W3 G5 V1 **char** *(4m10s)*

The *char* data type holds a single character.

W3 G5 V2a **Keyboard Part 1** *(10m37s)*
W3 G5 V2b **Keyboard Part 2** *(11m33s)*

How to respond to typical keyboard events, like a key going down or being released.

**KeyboardDemo.pde**

Demonstrates the code that lets us use the keyboard in the most common and important way: changing the value of a global variable.

**keyExplorer.pde**

Show when different keyboard functions are called in response to the user's actions, and what arguments they receive.

W3 G5 V3 **Keyboard AutoRepeat** *(9m4s)*

Many computers automatically repeat a key if it's held down. Here's how to keep that from becoming a problem.

**autorepeat.pde**

Demonstration of how to avoid problems due to auto-repeat.

W3 G5 V4 **Keyboard Extras** *(7m14s)*

Responding to non-printing keyboard keys, like the shift, control, or arrow keys, requires some special treatment.

**KeyboardDemo.pde**

Demonstrates the code that lets us use the keyboard in the most common and important way: changing the value of a global variable.

**keyExplorer.pde**

Show when different keyboard functions are called in response to the user's input, and what arguments they receive.

## GROUP 6: EXAMPLES

This week we've covered a lot of material. Here we put the pieces together in two examples.

W3 G6 V1a **A Bouncing Ball Part 1** *(9m46s)*
W3 G6 V1b **A Bouncing Ball Part 2** *(10m39s)*
W3 G6 V1c **A Bouncing Ball Part 3** *(4m23s)*

We write a program that draws a little circle that moves in a straight line across the window until it hits one of the four edges, where it "bounces" like a billiard ball. This technique is a common building-block for many different kinds of animation.

**BouncingBall.pde**

The basic technique for a bouncing ball.

W3 G6 V2a **A Little Game Part 1** *(9m52s)*
W3 G6 V2b **A Little Game Part 2** *(10m41s)*

We write a little game of (not much) skill. Pressing the space bar the right moment will cause a "dart" to land in a goal zone. You can also speed up or slow down the game to control the difficulty.

**PmanGame.pde**
The little dart-shooting game.

## GROUP 7: RECAP AND HOMEWORK

We survey what we've seen this week, and then you get to put it into action in your homework.

W3 G7 V1 **Week 3 Recap** *(4m48s)*

A recap of everything we've seen this week. Remember that there's a PDF that summarizes this information

W3 G7 V2 **Week 3 Homework** *(7m8s)*

Homework! Also summarized in the PDF handout.