

变量&动画

VARIABLES&ANIMATION

Week_02_1.1



Make Design Interact

变量 | VARIABLES

变量 | VARIABLES

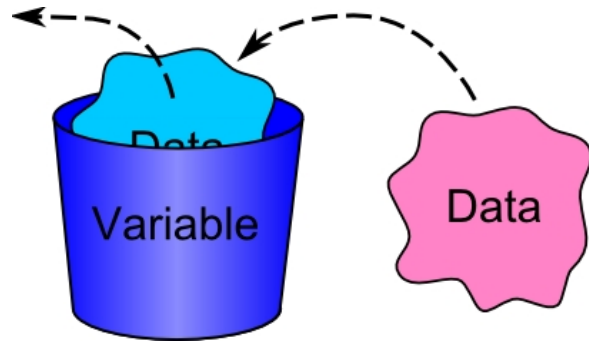
A variable is a symbolic name (an identifier) which contains data (a value).

In programming, variables are used to store information that may need to change over time.

Variables can represent all kinds of information, including: names of people or places, the size of something, lists of things, as well as more complex data structures.

变量是储存有信息 / 数据（值）的抽象概念或符号，在程序语言中，变量用于存储（不断在改变的）值。

变量可以存储各种信息，如地名、人名，尺寸，一个清单或者更复杂的数据结构。



基本变量类型 | PRIMITIVE VARIABLE TYPES

Primitive variables provide the basic building blocks of a programming language.

Processing's primitive variable types are:

基本变量提供了一个程序语言的基础，Processing的基本变量类型包括：

Boolean

Byte

Char

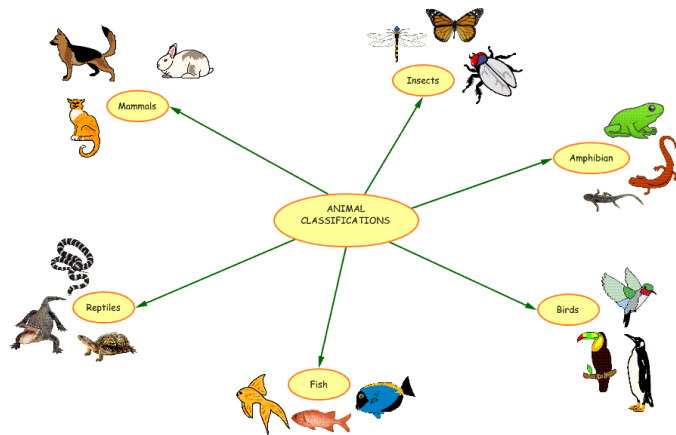
Color

Double

Float

Int

Long



Make Design Interact

变量声明，赋值&初始化

VARIABLE DECLARATION, ASSIGNMENT & INITIALIZATION

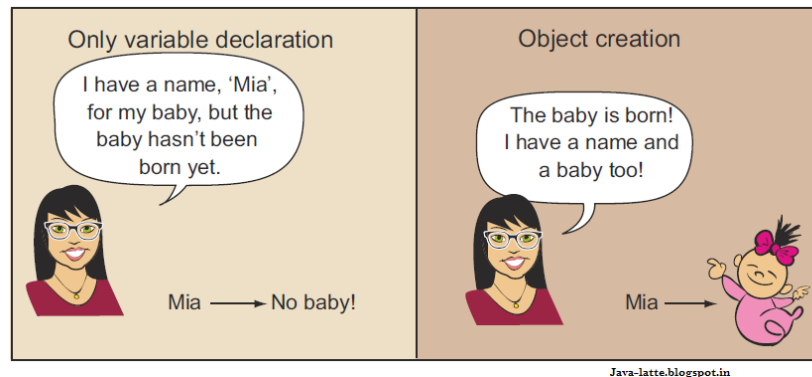
Before a variable can be used, it must first be declared in your program.

When declaring a variable in Processing you must define a data type and a name for the variable.

At the time of declaration you may set an initial value through a process called initialization, or you may set the value later through assignment.

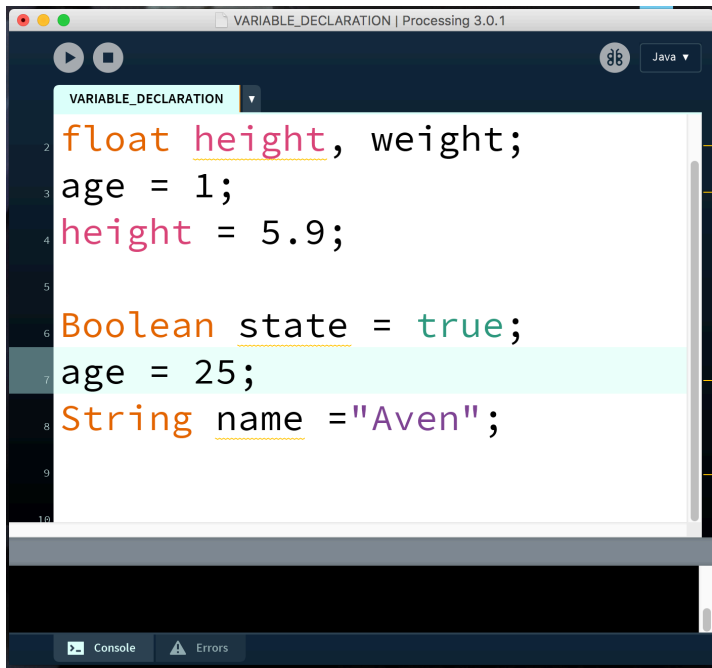
在一段程序中，变量必须在被声明之后才能被使用。而在processing中，声明变量要求同时定义变量的类型和变量名。

在声明变量时，你可以同时给变量赋以初始值或者也可以稍候完成赋值。



变量声明，赋值&初始化

VARIABLE DECLARATION, ASSIGNMENT & INITIALIZATION



```
VARIABLE_DECLARATION | Processing 3.0.1  
  
1  
2 float height, weight;  
3 age = 1;  
4 height = 5.9;  
5  
6 Boolean state = true;  
7 age = 25;  
8 String name = "Aven";  
9  
10  
Console Errors
```

```
int age;  
float height, weight;  
age = 1;  
height = 5.9;
```

```
Boolean state = true;  
age = 25;  
String name = "Aven";
```



作用域 | VARIABLE SCOPE

Where you declare your variable in your program has an effect on where within the program that variable is available, this is referred to as scope.

变量的有效区间位置和变量声明的位置相关，这个区间 / 位置称之为作用域。



Make Design Interact

环境变量 | ENVIRONMENT VARIABLES

Environment variables are variables that are already present within the program environment.

These variables do not need to be declared or initialized in order to use them.

Values such as the height and width of the screen or window, or the frame rate your program is running at, are available for your convenience.

环境变量指程序语言中已经存在的变量，它们不需要被声明活着初始化后再使用，比如 width, height, frameRate



PHOTO FROM WIKIPEDIA

WIDTH & HEIGHT

Two useful environment variables are width and height.

Width and height provide the value of the width and height of the sketch's window.

width: 窗口宽度

height: 窗口高度

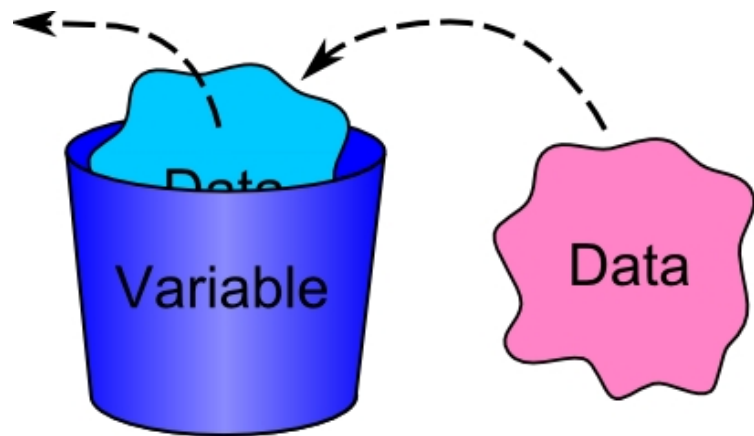


改变变量值 | VARYING VARIABLE VALUES

It is not only possible, but likely that you'll want to change the value of a variable at some point within your program.

Changing the value of a variable means that anywhere the variable identifier is used, the new value will be substituted.

程序中变量的值会经常改变（是代码作者经常改变变量的值），变量名保持不变，但是变量值却不断更新。



运算符 | OPERATORS

Operators allow you to assign, add, subtract, multiply, divide, compare, and perform other manipulations on values within your program.

运算符用以赋值，加减乘除，比较以及其他变量相关操作。

>

Greater Than

<

Less Than

>=

Greater Than or Equal To

<=

Less Than or Equal To

!=

Not Equal To

!

Not

&&

And

||

Or

数学运算符 | MATH OPERATORS

These math operators allow you to perform mathematical operations on numbers:

- + (addition), ++ (increment) & += (add assign)
- - (minus), -- (decrement) & -= (subtract assign)
- * (multiply) & *= (multiply assign)
- / (divide) & /= (divide assign)
- % (modulo)



随机 | RANDOM() FUNCTION

You can easily generate random numbers within your program by using the random() function.

The value returned by random() is a floating point number, however it is easy to convert this value to a whole number.

随机函数，需要注意的是random () 的返回值数据类型是float，但是可以比较容易地把float类型转化成int (integer)



PHOTO FROM WIKIPEDIA

随机函数示例 | RANDOM() EXAMPLE

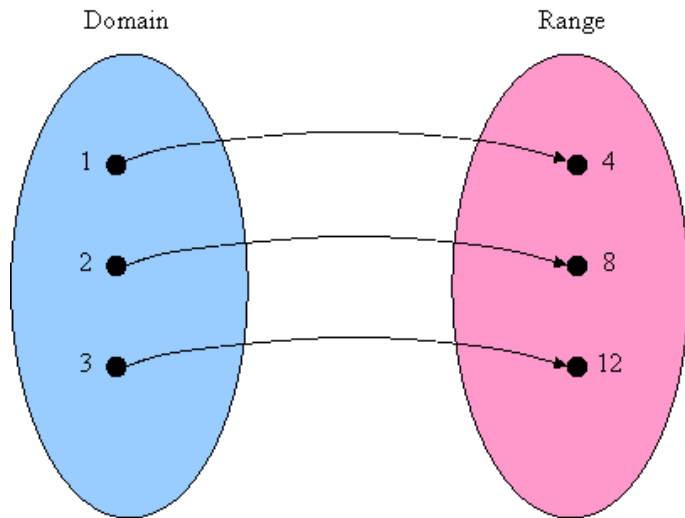
```
float    r  = random(1, 100);  
int r     = int(random(1,100));
```

MAP() FUNCTION

Sometimes the value of your variable is not within a range that is directly useful to you.

You can remap a numeric value from one range to another.

如果变量的区间并不能被直接使用，可以用Map（）function 将该变量map到新的区间



MAP() EXAMPLE

```
float m = map(v, 0, 100, 0, width);
```


动画 | ANIMATION

动画 | ANIMATION

Animation is a technique that involves quickly showing a series of drawings or images so as to create the illusion of movement or life.

动画由一帧帧定格图像组成。



THE ILLUSION OF LIFE

Ollie Johnston and Frank Thomas were two Disney animators who, in their 1981 book, *The Illusion of life*, formalized the techniques of Disney's animation team into 12 principles.

本书第三张：“12项法则”被独立出版

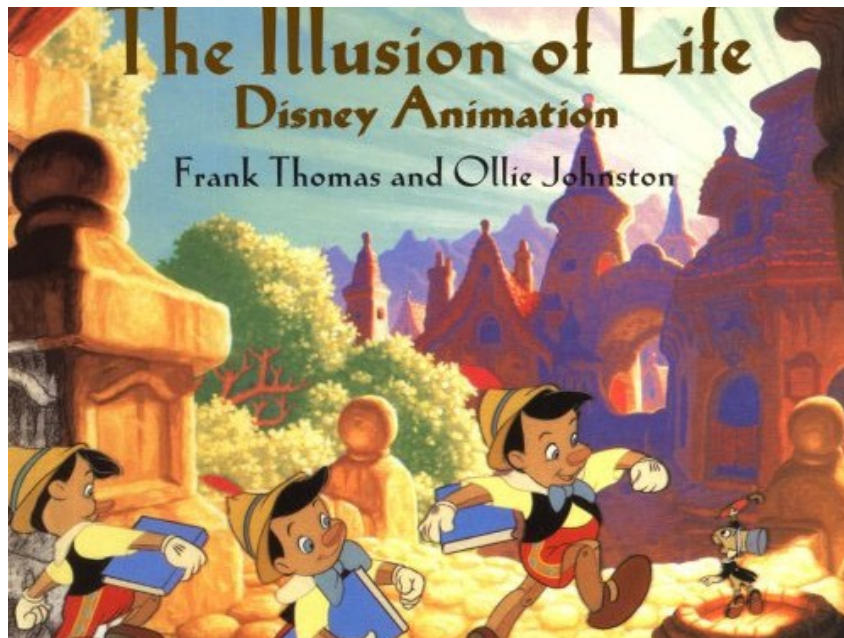


PHOTO FROM WIKIPEDIA

12 PRINCIPLES OF ANIMATION

The 12 principles of animation are:

- Squash & Stretch
- Anticipation
- Staging
- Straight Ahead or Pose to Pose
- Follow Through & Overlapping Action
- Arcs
- Slow In & Slow Out
- Secondary Action
- Timing
- Exaggeration
- Solid Drawing
- Appeal

12项法则

- 1.1 挤压与伸展 (Squash and stretch)
- 1.2 预期动作 (Anticipation)
- 1.3 演出方式 (Staging)
- 1.4 接续动作与关键动作 (Straight ahead action and pose to pose)
- 1.5 跟随动作与重叠动作 (Follow through and overlapping action)
- 1.6 渐快与渐慢 (Slow in and slow out)
- 1.7 弧形 (Arcs)
- 1.8 附属动作 (Secondary action)
- 1.9 时间控制 (Timing)
- 1.10 夸张 (Exaggeration)
- 1.11 纯熟的手绘技巧 (Solid drawing)
- 1.12 吸引力 (Appeal)

Processing 动画 | ANIMATION IN PROCESSING

Animation can be achieved in Processing by using a variety of different techniques, including the use of variables and functions that facilitate the movement and transformation of shapes.

Processing动画需要使用到variable, function来完成动作和形状改变

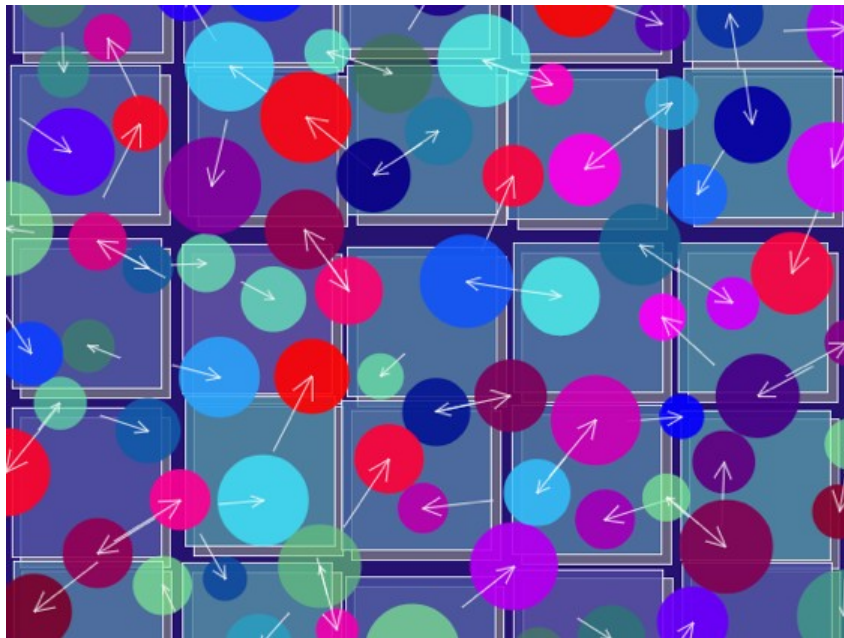


PHOTO BY CHRISTINE WILKS

Processing两种模式 | STATIC & ACTIVE MODES

Processing has two modes, static and active. In static mode your program will run once from beginning to end and then stop. By contrast, active mode allows you to create programs that run indefinitely over time as well as take into account user Input.

Static: 静态的；指在processing中只运行一次的程序部分

Active: 活跃的，动态的，积极的；指在processing中无限循环重复的程序部分



结构 | STRUCTURAL FUNCTIONS

Static mode and active mode is established through the use of two structural functions, `setup()` and `draw()`.

Static 和active模式是通过`setup ()` 和 `draw ()` function实现的

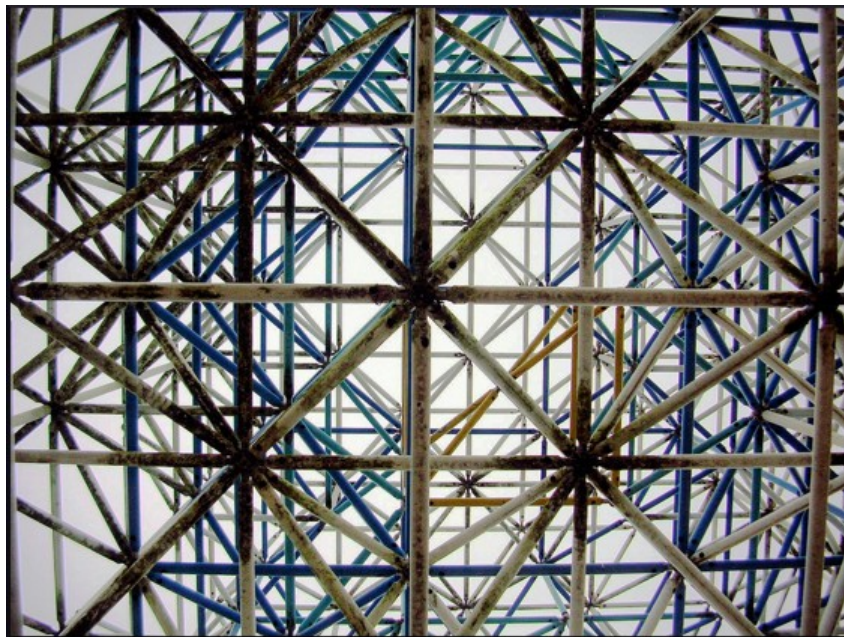


PHOTO BY P MEDVED

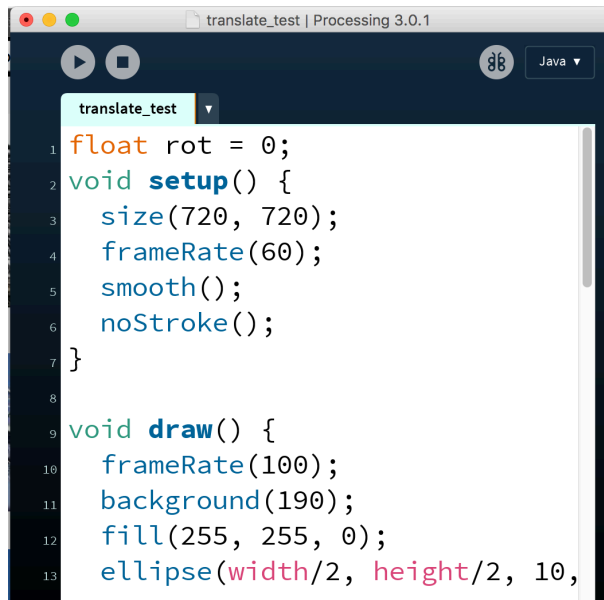
SETUP() & DRAW() FUNCTIONS

Code contained within `setup()` is executed once when the sketch first begins.

`setup()` is commonly used to define values for generally static properties such as window size and background color.

Code contained within `draw()` is repeatedly executed until your sketch is stopped.

Setup () function里面的代码只执行一次，所以setup function常用以定义静态的参数如窗口尺寸和图板背景色（可动态变化）而draw () function中个代码则是循环执行（直到程序被终止）



```
translate_test | Processing 3.0.1
float rot = 0;
void setup() {
  size(720, 720);
  frameRate(60);
  smooth();
  noStroke();
}

void draw() {
  frameRate(100);
  background(190);
  fill(255, 255, 0);
  ellipse(width/2, height/2, 10,
```

SETUP() & DRAW() FUNCTIONS 示例

```
void setup() {  
    size(500, 500);  
}  
void draw() {  
    background(0);  
    ellipse(10, 10, 50, 50);  
}
```

帧率 | FRAMERATE

Framerate is the frequency with which individual frames (drawings or images) within a sequence of frames is shown.

In Processing you can set the framerate by using the `frameRate()` function.

You can also get the framerate by using the framerate environment variable.

帧率指动画每秒刷新的帧数

Processing中可以用`frameRate()` function 改变动画帧率，而`frameRate`是一个环境变量



FRAMERATE EXAMPLE

```
void setup() {  
  size(500, 500);  
  frameRate(60);  
}  
void draw() {  
  println(frameRate);  
}
```

XY坐标系 | X & Y COORDINATES

A shape can be moved by changing the x or y coordinates of the shape over time. This is done by associating the x and y coordinates with a numeric variable that increments or decrements within the draw() function.

图形的移动是通过x, y坐标值变换实现的, 在draw () function中, x值递增则物体右移, y值递增则物体下移, 反之则反。



X & Y COORDINATES EXAMPLE

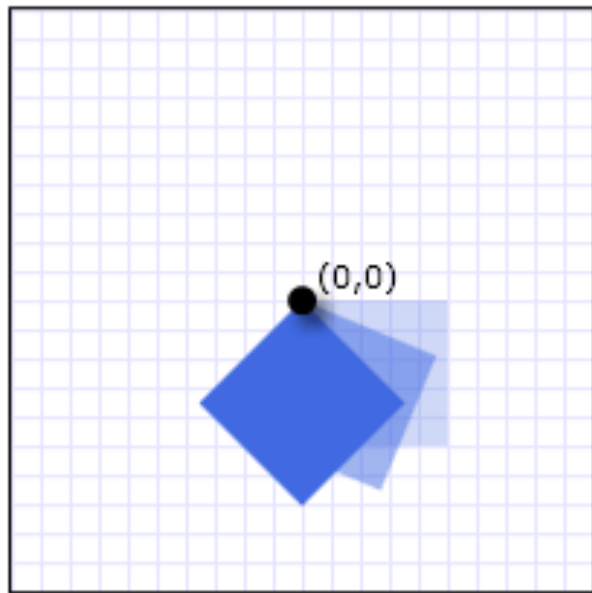
```
int  x  =  0;
int  y  =  0;
int  w  =  50;
void setup() {
    size(500, 500);
}
void draw() {
    background(0);
    rect(x, y, w, w);
    x++;
    y += 5;
}
```

变换 | TRANSFORM FUNCTIONS

Transform functions allow you to modify the size and proportions of a drawn shape.

Transformations are cumulative and apply to everything that is drawn after the transform function is called.

该系列的function用以变换图形的尺寸和其他属性，代码是一行行执行，transform function将对该行代码后全部代码产生影响。



旋转 | ROTATE FUNCTIONS



The rotate functions rotate a shape the amount specified by the angle parameter around the x, y or z axis, specified in radians. The rotate functions:

Rotate function 以指定点（或轴）为圆心（或中心轴）旋转图形指定角度（使用弧度制）：

SHEAR FUNCTIONS

The shear functions shear a shape around the X or Y axis the amount specified by the angle parameter, specified in radians.

The shear functions:

图解:

- `shearX()`
- `shearY()`



PHOTO BY CHSKYLAB

缩放 | SCALE FUNCTION

The scale() function scales (increases or decreases) the size of a shape by expanding and contracting vertices. Objects always scale from their relative origin to the coordinate system. Scaled values are specified as decimal percentages.

缩放：以指定点为原点按指定比例缩放

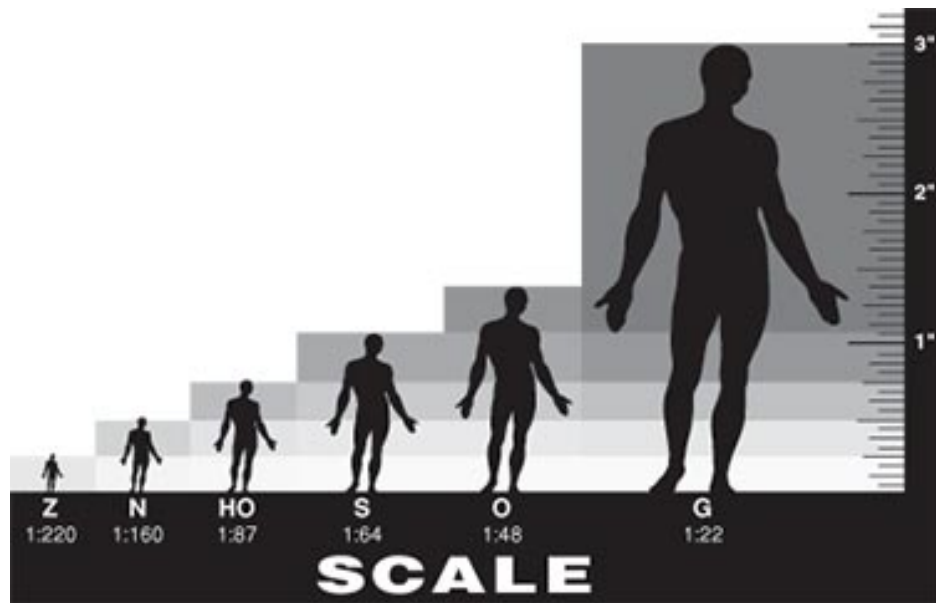


PHOTO BY JD HANCOCK

TRANSFORM FUNCTIONS EXAMPLE

```
rect(20, 20, 50, 50);  
scale(0.5);  
rect(20, 20, 50, 50);
```

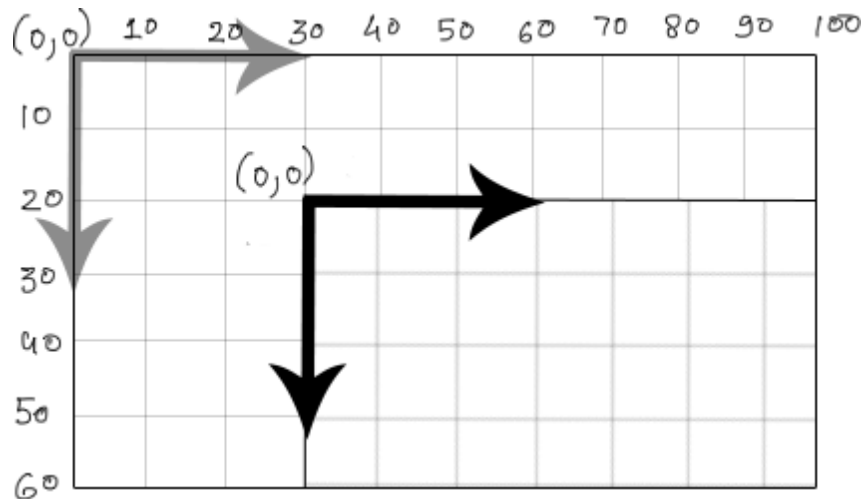
TRANSLATE FUNCTION

The `translate()` function specifies an amount to displace shapes within the display window.

Basically, it converts from one coordinate system to another.

转换：

移动 / 转换坐标系原点



TRANSLATE FUNCTIONS EXAMPLE

```
rect(0, 0, 50, 50);  
translate(width/2, height/2);  
rotate(PI/3.0);  
rect(0, 0, 50, 50);
```

MATRIX FUNCTIONS

The `pushMatrix()` function saves the current coordinate system to the stack and `popMatrix()` restores the prior coordinate system.

`pushMatrix()` and `popMatrix()` are used in conjunction with the other transformation functions and help to control the scope of transformations.

`pushMatrix ()` 保留储存当前坐标, `popMatrix ()` 恢复之前储存的坐标
组合使用时, 可以局部建立独立的相对坐标系, 并在绘图完成后恢复原坐标系



MATRIX FUNCTIONS EXAMPLE

```
fill(255);  
rect(0, 0, 50, 50);
```

```
pushMatrix();  
translate(30, 20);  
fill(0);  
rect(0, 0, 50, 50);  
popMatrix();
```

```
fill(100);  
rect(15, 10, 50, 50);
```

LIBRARIES

Libraries can be used to expand the capabilities of both Processing and Arduino.

For example, Processing libraries can be used to manipulate images, audio, and video, as well as for communication.

Libraries may be created by the Processing or Arduino teams, or they may be contributed by independent developers.



PHOTO BY SAMANTHA MARX

INSTALLING LIBRARIES

In Processing many Libraries can be installed by selecting Sketch / Import Library... / Add Library... from the menu and searching for a library.

In Arduino libraries can be installed by selecting Sketch / Import Library... / Add Library... from the menu and selecting the .zip file of the library. Libraries can also be downloaded and manually installed by copying the library's files into the libraries folder.



ANI ANIMATION LIBRARY

Ani is a library for creating animations and transitions in Processing.

Ani helps you to move things around on the screen or to animate any numeric variable.

Ani can be installed from within Processing or downloaded from here:

- looksgood.de/libraries/Ani/

