Arrays

Day05_01

ARRAYS

ARRAYS

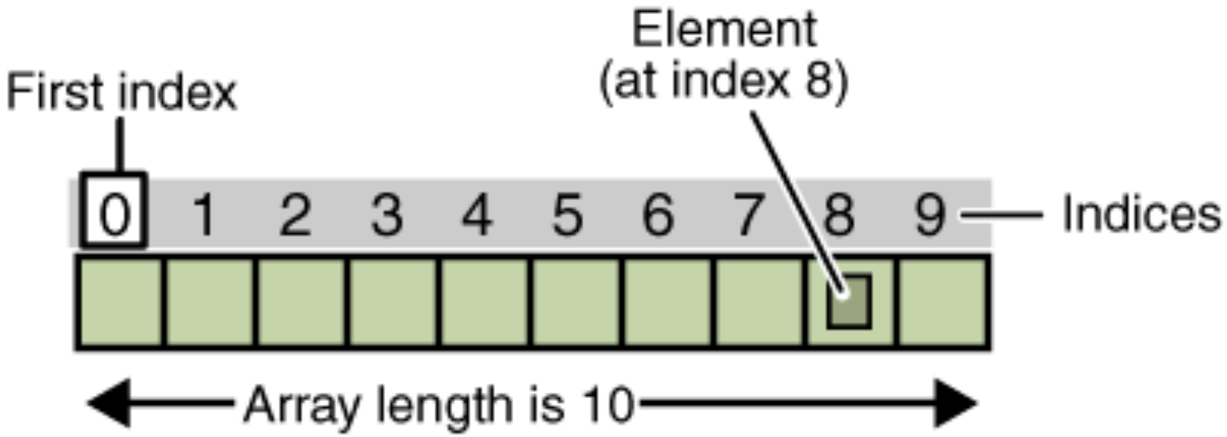
An array is a list of values.

Arrays can store any type of value, for example

booleans, chars or ints.

Each value in an array is identified by an index

number representing its position in the array.
The first index number in the array is 0.



Array指一列数值，可以是各种不同的数据类型，比如int, float,char,string.etc
Array中每个数值都有一个序号代表着该数值在整个Array中的位置，序号从0开始计数。

ARRAY DECLARATION & INITIALIZATION
声明&初始化

You have to declare and initialize an array.

It’s different than declaring and initializing a single variable.

Declared arrays without initialized values will result in null values. These often cause your program to crash.

Array在被使用之前需要声明和初始化，和声明和初始化一个Array和变量略有不同

A: Declaring and initializing an array in one line

B: Declaring array only and add value to later

```
int[] anArrayOfints;
anArrayOfints= new int[4];
anArrayOfints[0] = 5;
anArrayOfints[1] = 41;
anArrayOfints[2] = 33;
anArrayOfints[3] = 15;
```

this numbers is called the 'index'

ARRAY DECLARATION & INITIALIZATION EXAMPLE

声明&初始化示例

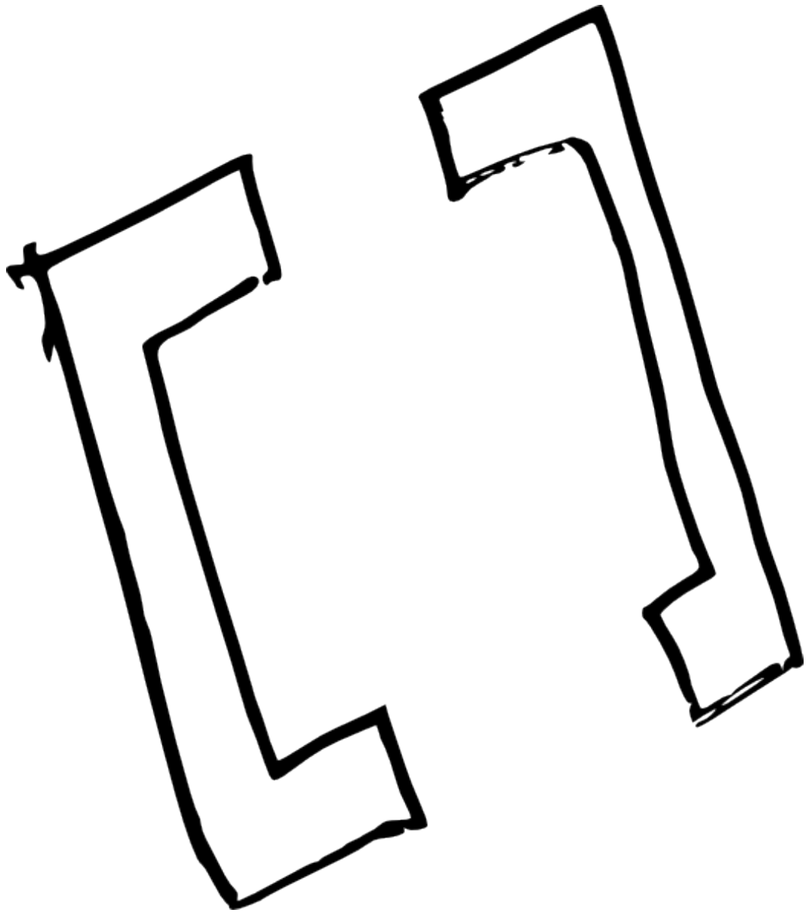
```
// declaring and initializing an array in one line
String[] states = {"AR", "MA", "NY"};
String state1 = “AR”;

// declaring an array only
int[] numbers = new int[5];
int number;

// adding values to an array later
numbers[0] = 88;
number = 88;
```

ARRAY ACCESS OPERATOR｜操作符

The array access operator, [ ] (square braces), allows you to get a value stored within a specific element of an array.

[ ] Array特殊操作符， Array名＋ ［序列号］可以访问指定位置上的元素

不能访问未被初始化赋值的元素，会导致程序崩溃



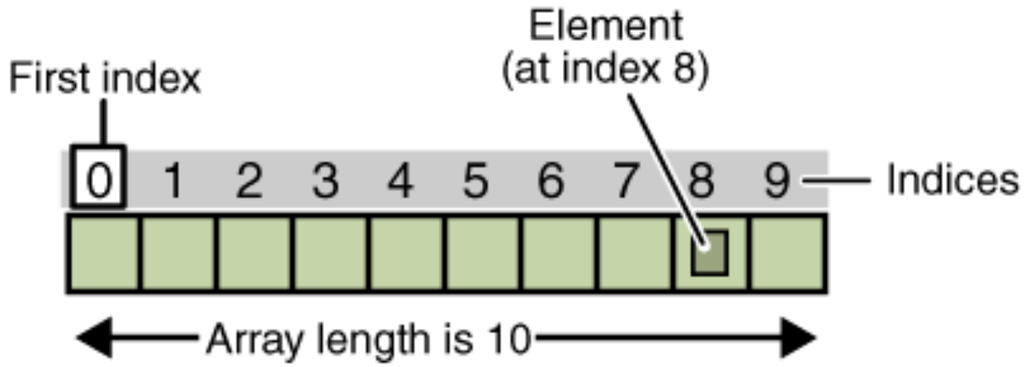
ARRAY ACCESS OPERATOR EXAMPLE丨操作符示例

```
int numbers = new int[5];
numbers[0]= 33;
numbers[3]= 77;


int n = numbers[3];
```

ARRAY LENGTH & RESIZING｜长度&尺寸调整

The array length field stores how many values an array contains.Arrays have an initial length, but the value of length will change as you add and remove elements from the array. Using the length field makes it easy to loop over all of the elements in an array. The last index number of an array is the array length minus one.

Array的长度是固定的，指列表中元素个数，注意长度为10的array，序列号将从0-9.最后一位元素的序列号将是array.length -1
Array具有初始长度，当你增加和减少元素时长度会被改变,使用循环时需要注意array的长度

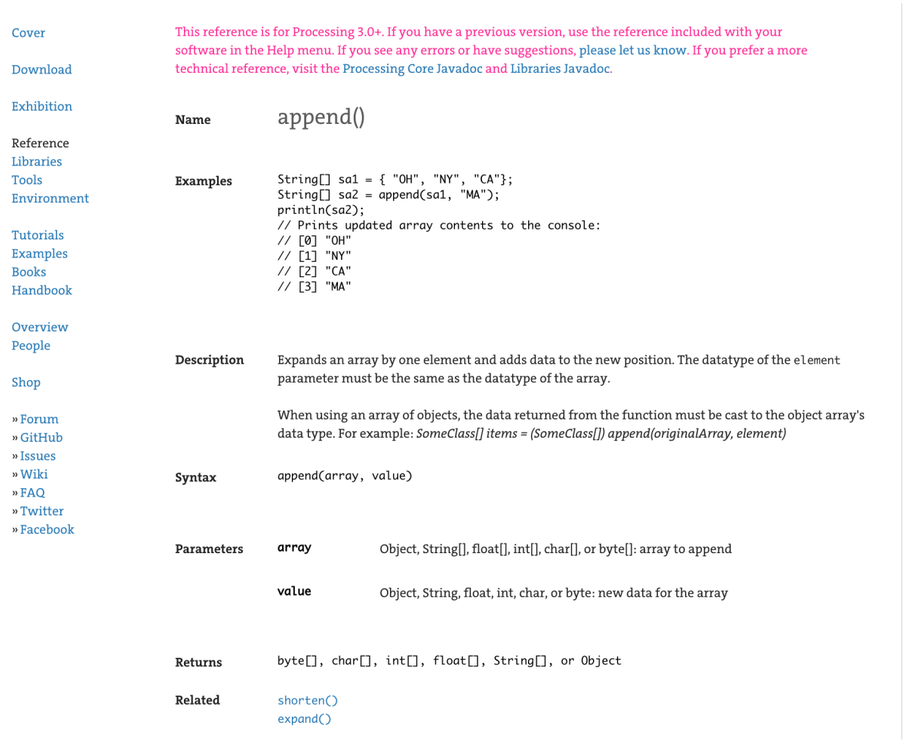


ARRAY FUNCTIONS

Arrays have several built-in methods that facilitate various array activities, for example adding and removing elements.

These functions allow you to adjust the length as well as arrange and extract values.

Array相关function可以添加和减少元素，可以改变array长度，提取元素，重新排序



Cover

Download

Exhibition

Reference
Libraries
Tools
Environment

Tutorials
Examples
Books
Handbook

Overview
People

Shop

» Forum
» GitHub
» Issues
» Wiki
» FAQ
» Twitter
» Facebook

This reference is for Processing 3.0+. If you have a previous version, use the reference included with your software in the Help menu. If you see any errors or have suggestions, please let us know. If you prefer a more technical reference, visit the Processing Core Javadoc and Libraries Javadoc.

Name append()

Examples

```
String[] sa1 = { "OH", "NY", "CA"};
String[] sa2 = append(sa1, "MA");
println(sa2);
// Prints updated array contents to the console:
// [0] "OH"
// [1] "NY"
// [2] "CA"
// [3] "MA"
```

Description Expands an array by one element and adds data to the new position. The datatype of the `element` parameter must be the same as the datatype of the array.

When using an array of objects, the data returned from the function must be cast to the object array's data type. For example: *SomeClass[] items = (SomeClass[]) append(originalArray, element)*

Syntax `append(array, value)`

Parameters

`array` Object, String[], float[], int[], char[], or byte[]: array to append

`value` Object, String, float, int, char, or byte: new data for the array

Returns `byte[], char[], int[], float[], String[], or Object`

Related
`shorten()`
`expand()`

ARRAY METHODS

- append() adds a value to the end of the array.
- arrayCopy() makes a copy of an array.
- concat() combines two arrays.
- expand() increases the length of the array.
- reverse() reverses the order of the array.
- shorten() decreases the length of the array by one.
- sort() will arrange an array of values numerically or alphabetically.
- splice() inserts a value or list of values into an array.
- subset() allows you to extract a range of values from an array.

ARRAY USAGE EXAMPLE

```
String[] states = {"OH", "NY", "CA"};
states = append(states, "NY");
int l = states.length;
states = splice(states, "KY", 1);
states = shorten(states);
println(states[1]);
```

MULTI-DIMENSION ARRAYS｜多纬度

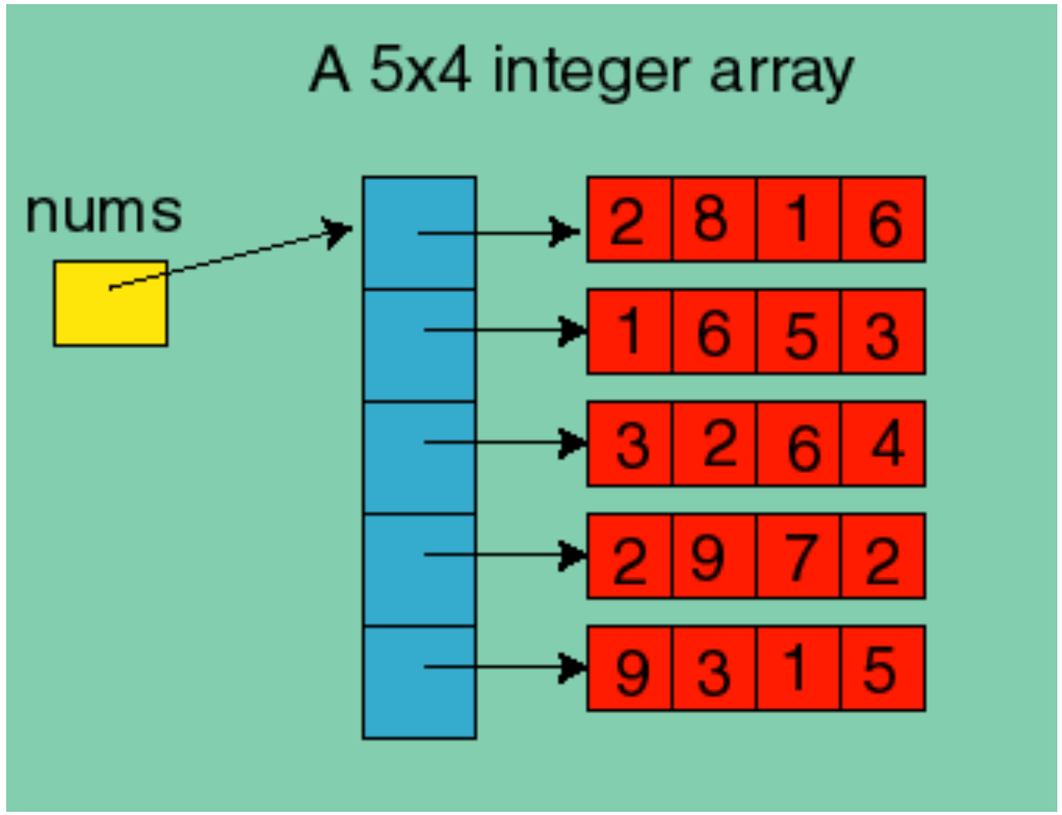
Arrays can also store data in more than one dimension!

A 2D array is essentially a list of two 1D arrays but kept in one array. The syntax is as follows:

多维度array，2D的array指一列由array作为元素的array

语法如下

```
int[ ][ ] i = { {50, 0} , {100, 20} };
println(i[0][0]); // prints 50
println(i[1][1]); // prints 20
```



2D ARRAY EXAMPLE｜示例

```
int[][] i = { {50, 0} , {100, 20} };

println(i[0][0]); // prints 50
println(i[1][1]); // prints 20
```

ARRAYS AND LOOPS｜循环

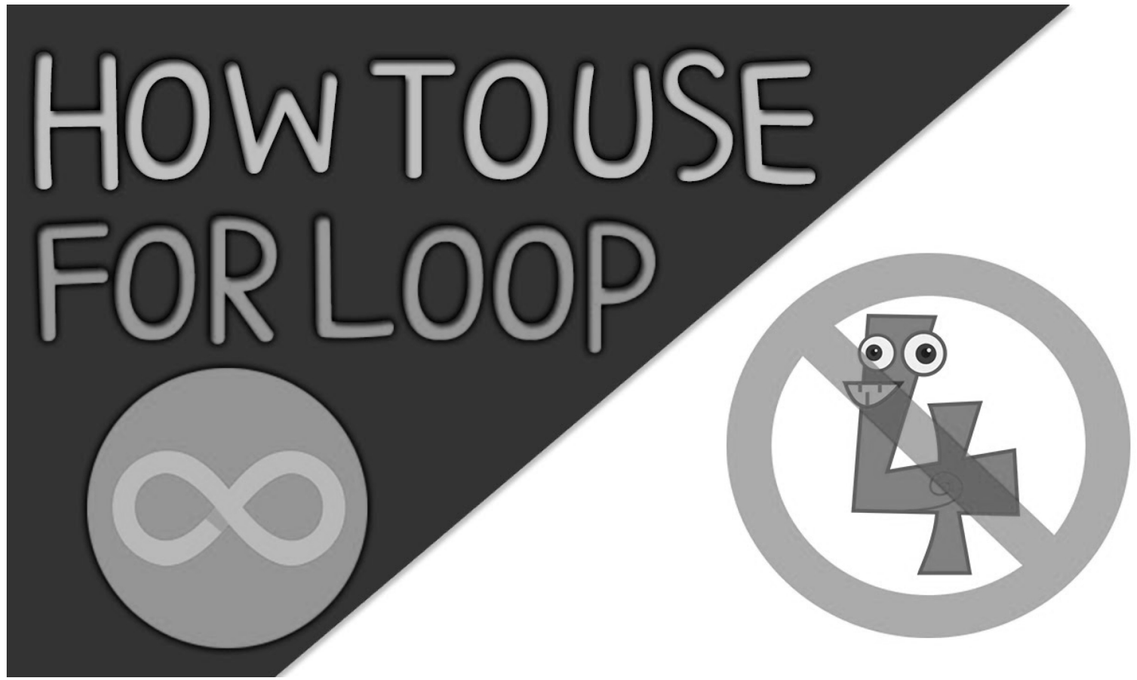
Combining arrays with loops give you the ability to create all sorts of information.

For example, adding individual colors to a grid of rectangles.

Drawing and remembering the trails of a mouse.

Or keeping an X and Y location of items.

通常情况下，array和循环一起使用，赋值或者调用



ARRAYS AND LOOPS丨循环示例

```
int[] arrayX;
int[] arrayY;

void setup() {
  arrayX = new int[10];
  arrayY = new int[10];
  for (int i = 0; i < 10; i++) {
    arrayX[i] = int(random(0, width));
    arrayY[i] = int(random(0, height));
  }
}
void draw() {
  for (int i = 0; i < 10; i++) {
    ellipse(arrayX[i], arrayY[i], 20, 20);
  }
}
```

ARRAYS IN-CLASS

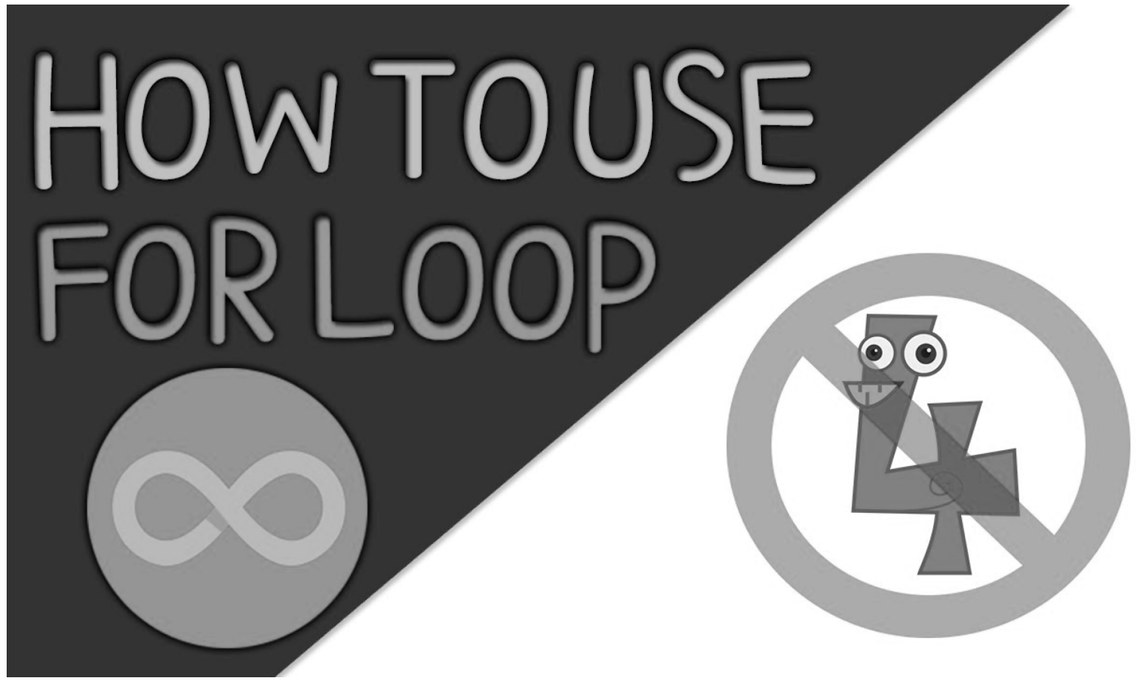
RANDOM NUMBER GENERATOR

Create a one-dimensional array of int values.

Fill it with random numbers.

Print them all on the console.

创造一个一维array，并赋值以随机数，然后在console中打印出来



ARRAYS AND LOOPS

Combine an array with a loop to draw multiple items on the screen.

You can use the array to hold an x and a y location, to hold different color variations, or to hold the size of primitive objects.