6. 数组

Contents

- 一维数组入门
  - 数组定义、特点、内存分配
  - 使用一维数组存储数据
  - for-each循环
- 一维数组的应用
  - 查询元素
  - 数组类型做形参
  - 查询最大值最小值
  - 添加元素或删除元素
  - 冒泡排序
  - Arrays工具类
  - 理解main（String args[]）
  - 可变参数
- 二维数组：
  - 二维数组含义、特点
  - 内存分配、
  - 举例线程组

本章技能点列表

技能点名称	难易程度	认知程度	重要程度
数组定义、特点、内存分配	中	理解	***
数组操作：使用数组存储元素并遍历	中	应用	**
for-each循环	易	应用	***
数组操作：查询数据	易	应用	***
数组类型做形参	中	理解	***
数组操作：查询最大值最小值	中	应用	**
数组操作：添加删除操作	中	应用	**

本章技能点列表

技能点名称	难易程度	认知程度	重要程度
冒泡排序	难	理解	***
数组工具类Arrays	中	了解	*
理解main方法	中	理解	**
可变参数	易	了解	*
二维数组	中	了解	*

创建数组（1）

- 数组是相同类型数据的有序集合.
  - 相同类型的若干个数据,按照一定先后次序排列组合而成。
  - 其中,每一个数据称作一个数组元素
  - 每个数组元素可以通过一个下标来访问它们.

- 数组特点:
  - 其长度是确定的。数组一旦被创建，它的大小就是不可以改变的。
  - 其元素必须是相同类型,不允许出现混合类型。
  - 数组中的元素可以是任何数据类型，包括基本类型和引用类型。

- 数组属引用类型
  - length, elements of the array

创建数组（1）

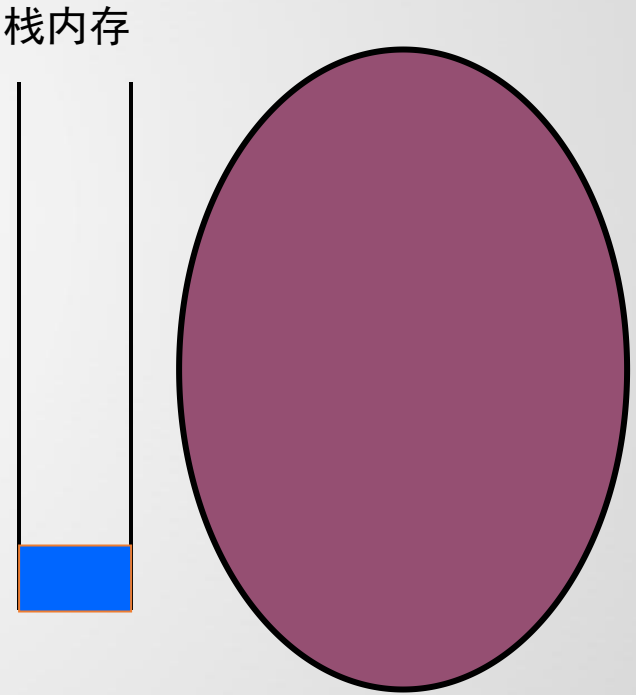
- 一维数组的声明方式有两种：
  - type[] arr_name;
  - type arr_name[];
- 例如：
  - int[] intArrays;   int intArrays[];
  - double[] doubleArrays;
  - Person[] pArrays;
  - String[] strArrays;

创建数组（1）

- Java中使用关键字new 创建数组对象
- 创建基本数据类型一维数组对象演示1

```
public class Test{
    public static void main(String args[]){
      int[] s = null;
      s = new int[10];
      for ( int i=0; i<10; i++ ) {
            s[i] =2*i+1;
            System.out.println(s[i]);
      }
    }
}
```

处内存状态

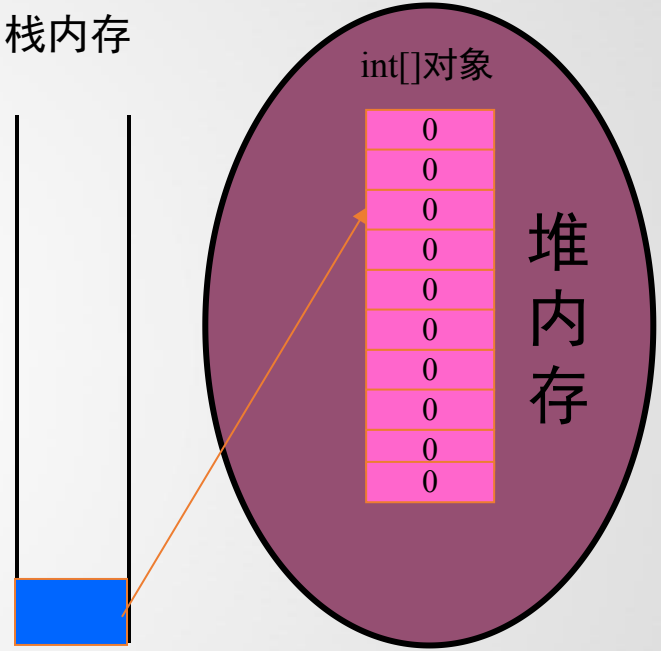


创建数组（2）

- 创建基本数据类型一维数组对象演示2

```
public class Test{
    public static void main(String args[]){
        int[] s = null;
        s = new int[10];
        for ( int i=0; i<10; i++ ) {
            s[i] =2*i+1;
            System.out.println(s[i]);
        }
    }
}
```

处内存状态 main s

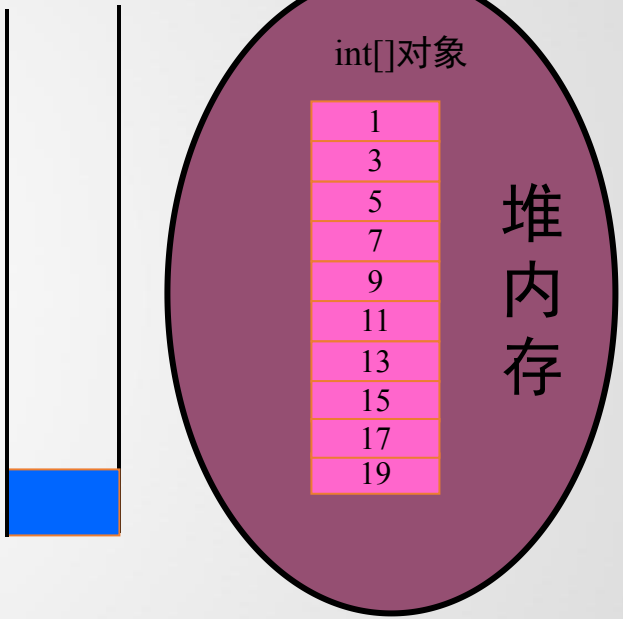


创建数组（3）

- 创建基本数据类型一维数组对象演示3

```
public class Test{
    public static void main(String args[]){
        int[] s = null;
        s = new int[10];
        for ( int i=0; i<10; i++ ) {
            s[i] =2*i+1;
            System.out.println(s[i]);
        }
    }
}
```

处内存状态 main s



数组初始化

- 动态初始化
  - 数组定义与为数组元素分配空间并赋值的操作分开进行。

```
int  a[] = null;
a  = new int[3];
a[0] = 3;
a[1] = 9;
a[2] = 8;
```

数组初始化

- 静态初始化：
  - 除了用new关键字来产生数组以外,还可以直接在定义数组的同时就为数组元素分配空间并赋值。
    - 格式: 类型 [] 数组名 = {元素1[, 元素2 ……]};
    - int [] a = {1, 2, 3, 4, 5};

```
public class Test {
    public static void main(String args[]) {
        int [] a = { 3, 5, 7 };
    }
}
```

数组初始化

- 数组是引用类型，它的元素相当于类的实例变量，因此数组一经分配空间，其中的每个元素也被按照实例变量同样的方式被隐式初始化

```
public class ArrayTest3 {
    public static void main(String args[]) {
         int a[] = new int[2];
        boolean [] b = new boolean[2];
        String[] s = new String[2];
        for(int i = 0; i < 2; i++)
                System.out.println(a[i]);
        for(int i = 0; i < 2; i++)
                System.out.println(b[i]);
        for(int i = 0; i < 2; i++)
                System.out.println(s[i]);
    }
}
```

➢ 输出结果：

```
0
0
false
false
null
null
```

数组的界限

- 定义并用运算符new为之分配空间后，才可以引用数组中的每个元素；
- 数组元素的引用方式：arrayName[index]
  - index为数组元素下标，可以是整型常量或整型表达式。如a[3] , b[i] , c[6*i];
  - 数组元素下标从0开始；长度为n的数组合法下标取值范围： 0 ~ n-1；
- 每个数组都有一个属性length指明它的长度，例如：a.length 指明数组a的长度(元素个数)；
  - 数组的长度: 数组名.length
- 起点和终点
  - 起点: 数组名[0]
  - 终点: 数组名[length-1]

```
int[] i = {4, 56, 78, 9, 34};
i.length → 5
i[0] → 4
i[length-1]=i[4]→34
i[a] 若a>4 则???
```

课堂练习（20分钟）

- 编写一应用程序实现下述功能：创建一基本(primitive)数据类型的数组并输出各数组元素的值。例如：
  - char[] s;
  - s = new char[26];
  - for (int i=0; i<26; i++) {
    - s[i] = (char) (’A’ + i);
    - System.out.println(s[i]);
    - // System.out.println(“s[” + i + “]=” + s[i]);
  - }
- 编写一应用程序练习数组对象的两种初始化方式，并输出各元素的值。
- 编写程序，练习使用数组类型对象的length属性，测试并体会数组元素的默认初始化机制；

二维数组

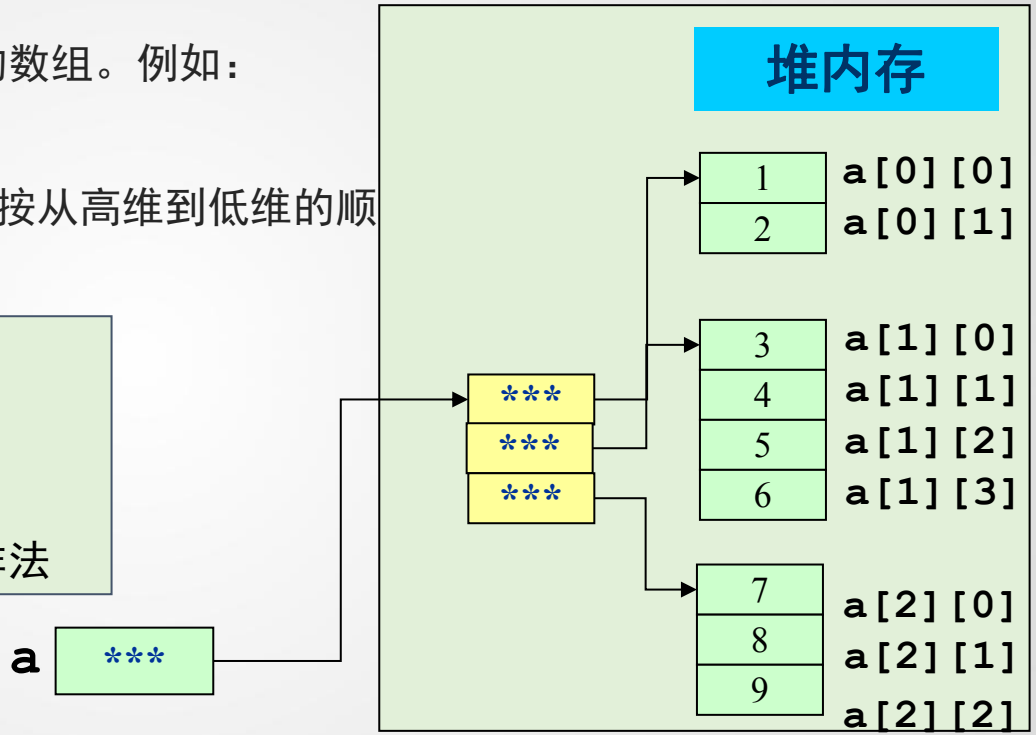
- 二维数组举例：
  - int [][] a = {{1,2},{3,4,0,9},{5,6,7}};
  - Java中多维数组不必须是规则矩阵形式

i \ j	j = 0	j = 1	j = 2	j = 3
i = 0	1	2		
i = 1	3	4	0	9
i = 2	5	6	7	

二维数组

- 二维数组可以看成以数组为元素的数组。例如：
  - int [][] a= {{1,2},{3,4,5,6},{7,8,9}};
- Java中多维数组的声明和初始化应按从高维到低维的顺
- 例如：

```
int [][]  a= new int[3][];
a[0] = new int[2];
a[1] = new int[4];
a[2] = new int[3];
int t1[][] = new int[][4]; //非法
```



二维数组初始化

- Declare, create and initiate in the same time：
  - int intA[][] = {{1,2},{2,3},{3,4,5}};
  - int intB[3][2] = {{1,2},{2,3},{4,5}};//非法
- Declare, create and initiate separately：
  - int a[][] = new int[3][5];
  - int b[][] = new int[3][] ;
  - b[0] = new int[2];
  - b[1] = new int[3];
  - b[2] = new int[5];

课堂练习(15分钟)

- 编写一应用程序实现下述功能：创建一基本(primitive)数据类型的二维数组并输出各数组元素的值。例如：

```
……
int a[][] = {{1,2},{2,3,4,5},{5,6,7}};
for(int i=0;i<a.length;i++) {
    for(int j=0;j<a[i].length;j++) {
        System.out.println(intArray1[i][j]);
    }
}
```

数组的拷贝

- 使用java.lang.System类的静态方法
  - public static void arraycopy
    - (Object src,int srcPos,Object dest,
      - int destPos,int length)
- 可以用于数组src从第srcPos项元素开始的length个元素拷贝到目标数组从destPos项开始的length个位置。
- 如果源数据数目超过目标数组边界会抛出 IndexOutOfBoundsException 异常。

数组的拷贝举例

```
public class ArrayTest7 {
  public static void main(String args[]) {
    String[] s = {"Mircosoft","IBM","Sun","Oracle","Apple"};
    String[] sBak = new String[6];
    System.arraycopy(s,0,sBak,0,s.length);
       for(int i=0;i<sBak.length;i++){
              System.out.print(sBak[i]+" ");
       }
       System.out.println();
    int[][] intArray = {{1,2},{1,2,3},{3,4}};
    int[][] intArrayBak = new int[3][];
    System.arraycopy(intArray,0,intArrayBak,0,intArray.length);
    intArrayBak[2][1] = 100;
    for(int i = 0;i<intArray.length;i++){
        for(int j =0;j<intArray[i].length;j++){
            System.out.print(intArray[i][j]+"  ");
        }
        System.out.println();
    }
  }
}
```

命令行参数

- JAVA应用程序的主方法(程序的入口)
  - public static void main (String args[]) {…}
  - public static void main (String[] args) {…}
- 命令行参数
  - 在启动Java应用程序时可以一次性地向应用程序中传递0~多个参数----命令行参数
  - 命令行参数使用格式：
  - java ClassName lisa "bily" "Mr Brown“
  - 由参数args接收
  - 空格将参数分开
  - 若参数包含空格，用双引号引起来

命令行参数用法举例

```
public class Test {
    public static void main(String[] args) {
            for ( int i = 0; i < args.length; i++ ) {
                   System.out.println("args[" + i + "] = " + args[i]);
            }
    }
}
```

//运行程序

java Test lisa "bily" "Mr Brown"

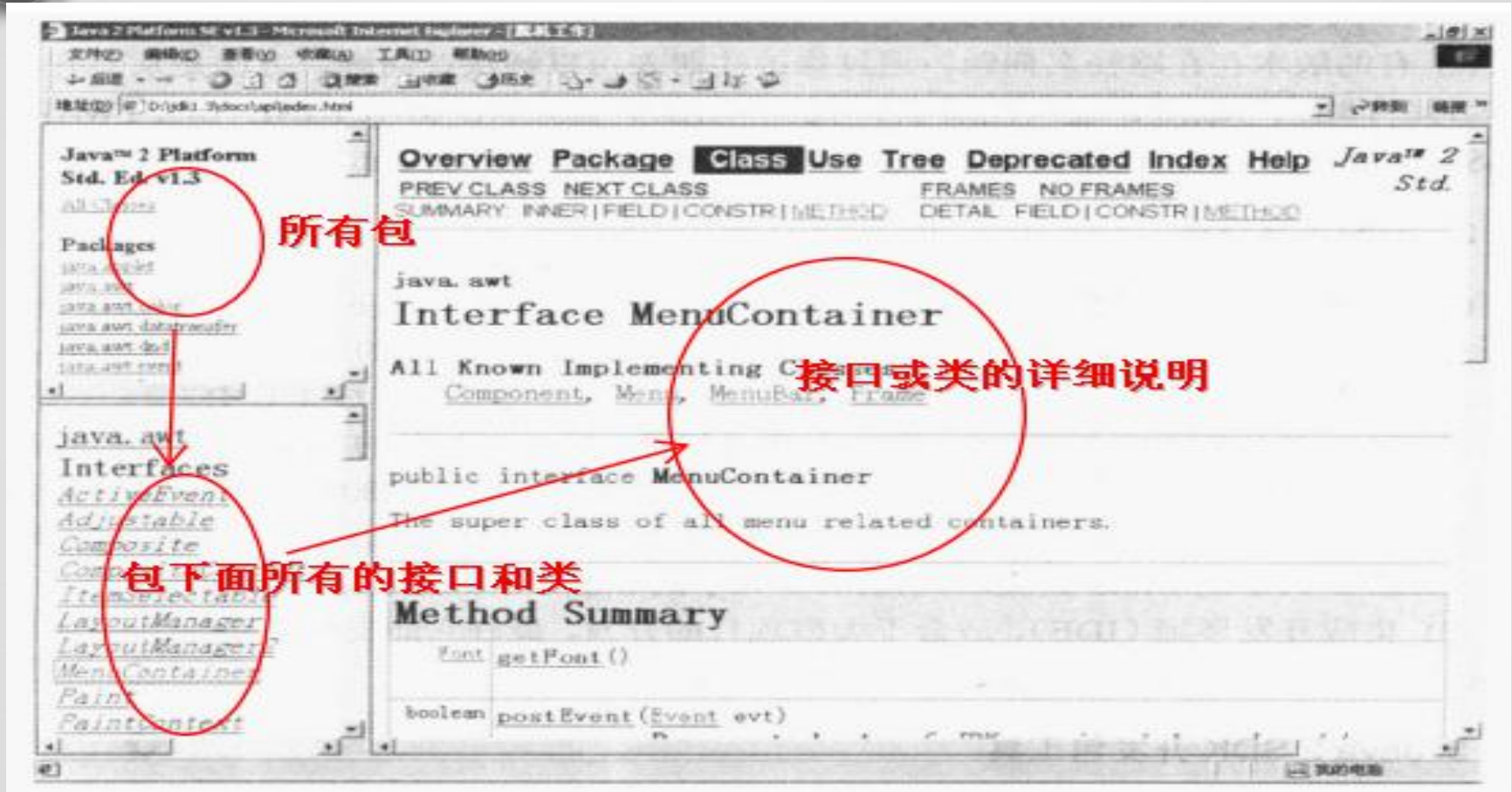
//输出结果：

args[0] = lisa
args[1] = bily
args[2] = Mr Brown

Java.uitl.Arrays

- 该类提供了关于数组操作的API.
  - 打印数组----toString方法。
  - 比较两个数组是否相同----equals方法。
  - 数组排序----sort方法。
  - 数组查找----binarySearch 方法

API文档如何查看 A


Java™ 2 Platform
Std. Ed. v1.3
Packages
所有包
Overview Package Class Use Tree Deprecated Index Help
PREV CLASS NEXT CLASS
FRAMES NO FRAMES
java.awt
Interface MenuContainer
接口或类的详细说明
java.awt
Interfaces
包下面所有的接口和类
public interface MenuContainer
The super class of all menu related containers.
Method Summary
Font getFont()
boolean postEvent(Event evt)

总结

- 一维数组入门
  - 数组的特点：长度固定，连续空间，存储同一种类型数据
  - 数组内存分配图
  - for-each循环：简单、主要用于遍历操作
- 一维数组的应用
  - 数组优缺点
    - 优点：按照索引查询效率高
    - 缺点：添加删除元素效率低；按照内容查询效率低（无序）
  - 冒泡排序：基本的排序算法，理解排序规则，实现并完善排序代码
  - 数组类型做形参
- 二维数组：
  - 实质是每个元素是一维数组的一维数组；二维数组内存分配图