# 第1章 react的前世今生

react.js作为新一代的web框架必定给web开发带来一种新的体验，那在react.js出现之前web开发是一个什么样的现状呢？在react.js出现之后又将把web开发带向什么样的一个方向呢？（以上内容属于章和节之间的过渡段，要求至少3行，说明本章要学习的内容，为什么要学习本章，读者能从本章中学到什么等等）（下面是说明本章的知识点，让读者知道最终学习完后能干什么）

本章主要涉及到的知识点有：

* react出现之前的开发现状：了解是什么样的历史状况下出现的react。
* web应用的出现：web应用的出现促进了react的诞生。
* react的诞生：react闪亮登场web开发领域。

注意：本章内容不包含动态数组。（注意技巧样式的应用。）

## 1.1 刀耕火种的时代

本节首先介绍web1.0的开发模式，理解过去开发模式的缺点和痛楚之后。才能理解为什么react要这么设计，react的设计模式解决了什么样的问题。（节和小节之间的过渡段，至少2行）

### 1.1.1 网络的诞生

1969年美国国防部高级研究计划署（ARPA）建立ARPANet。ARPANet最初只包括四个站点，即加州大学洛杉矶分校UCLA、加州大学圣巴巴拉分校UCSB、犹他大学Utah和斯坦福研究所SRI。

1970年美国夏威夷大学的诺曼.阿勃拉姆逊研制成ALOHANet，这是早期著名互联网之一。ARPANet开始采用由加州大学洛杉矶分校的斯蒂夫.克洛克设计的网络控制协议NCP(NetworkControlProtocol)。

1971年ARPANet发展到15个站点，23台主机。新接入的站点包括哈佛大学、斯坦福大学、林肯实验室、麻省理工学院、卡内基.梅隆大学、美国航空航天局的Ames研究中心等。1972年互联网工作组（INWG）宣告成立。其目的在于建立互联网通讯协议，主席是斯坦福的温登.泽夫。BBN公司（由Bolt、Beranek和Newman合伙建立的位于波士顿的一家咨询公司）的雷.汤姆林森（ReyTomlinson）发明电子邮件，在互联网上很快就流行起来。

1973年ARPANet扩展成国际互联网。第一批接入的有英国和挪威的计算机。

1974年ARPA的鲍勃.凯恩和斯坦福的温登.泽夫合作，提出TCP/IP协议。

1975年由于ARPANet已由试验性互联发展为实用型网络，其运行管理由ARPA移交给国防通信局DCA。世界上第一台微机Altair8008诞生于新墨西哥州的MITS公司。微机时代来临，比尔.盖茨和保罗.艾伦借Altair起家，建立了如今声名显赫的微软公司。

1976年AT&T的贝尔实验室推出UUCP（Unix-to-UnixCopy），一年后开始随UNIX捆绑销售。

1977年威斯康辛大学建成THEORYNet。这个互联网用UUCP向100多用户提供E-mail服务。

1978年美国国防部决定以TCP/IP协议的第4版作为其数据通信网络的标准。互联网通讯协议标准化的实施极大地推动了互联网的发展。

1982年TCP/IP加入UNIX内核中，商业电子邮件服务在美国25个城市开始启动。

1983年ARPANet分离出MILNet，DCA把ARPANet各站点的通讯协议全部转为TCP/IP，这是全球Internet正式诞生的标志。

欧洲建成科学和研究网EARN（European Academic and Research Network）。

托姆.詹宁斯（Tom Jennings）开发成功Fidonet，主要连接MS-DOS的个人电脑（Fido是美国人对心爱的哈巴狗的通称，其名称来源于此）。

1984年日本建成JUNet（Japan Unix Network）。

作家吉布森（Gibson）在他的一篇小说《精神漫游者》中首次提出Cyberspace这个术语。

1985年美国科学家基金会（NSF）建立NSFNet。

1986年NSFNet成为Internet主干网（56Kbps）。

Internet完成取代了ARPA Net。

1987年连接在Internet上的主机数突破10，000台。

NSF与IBM、Merit网络公司、MCI等公司签约，把NSFNet主干网的传输速率从56kbps提高到1.54Mbps。

1989年Internet主干网升为T1速率（1.54Mbps）。

最早的也是最著名的Internet服务提供商之一——Compuserve成立。

欧洲核子研究中心（CERN）的物理学家蒂姆·贝纳斯一李（Tim Berners-Lee）研制成World Wide Web，推出世界上第一个所见即所得的超文本浏览器/编辑器。

### 1.1.2 Javascript的诞生

JavaScript因为互联网而生，紧随着浏览器的出现而问世。回顾它的历史，就要从浏览器的历史讲起。

1990年底，欧洲核能研究组织（CERN）科学家Tim Berners-Lee，在全世界最大的电脑网络——互联网的基础上，发明了万维网（World Wide Web），从此可以在网上浏览网页文件。最早的网页只能在操作系统的终端里浏览，也就是说只能使用命令行操作，网页都是在字符窗口中显示，这当然非常不方便。

1992年底，美国国家超级电脑应用中心（NCSA）开始开发一个独立的浏览器，叫做Mosaic。这是人类历史上第一个浏览器，从此网页可以在图形界面的窗口浏览。

1994年10月，NCSA的一个主要程序员Marc Andreessen联合风险投资家Jim Clark，成立了Mosaic通信公司（Mosaic Communications），不久后改名为Netscape。这家公司的方向，就是在Mosaic的基础上，开发面向普通用户的新一代的浏览器Netscape Navigator。

1994年12月，Navigator发布了1.0版，市场份额一举超过90%。

Netscape公司很快发现，Navigator浏览器需要一种可以嵌入网页的脚本语言，用来控制浏览器行为。当时，网速很慢而且上网费很贵，有些操作不宜在服务器端完成。比如，如果用户忘记填写“用户名”，就点了“发送”按钮，到服务器再发现这一点就有点太晚了，最好能在用户发出数据之前，就告诉用户“请填写用户名”。这就需要在网页中嵌入小程序，让浏览器检查每一栏是否都填写了。

管理层对这种浏览器脚本语言的设想是：功能不需要太强，语法较为简单，容易学习和部署。那一年，正逢Sun公司的Java语言问世，市场推广活动非常成功。Netscape公司决定与Sun公司合作，浏览器支持嵌入Java小程序（后来称为Java applet）。但是，浏览器脚本语言是否就选用Java，则存在争论。后来，还是决定不使用Java，因为网页小程序不需要Java这么“重”的语法。但是，同时也决定脚本语言的语法要接近Java，并且可以支持Java程序。这些设想直接排除了使用现存语言，比如Perl、Python和TCL。

1995年，Netscape公司雇佣了程序员Brendan Eich开发这种网页脚本语言。Brendan Eich有很强的函数式编程背景，希望以Scheme语言（函数式语言鼻祖LISP语言的一种方言）为蓝本，实现这种新语言。

1995年5月，Brendan Eich只用了10天，就设计完成了这种语言的第一版。它是一个大杂烩，语法有多个来源：

基本语法：借鉴C语言和Java语言。

数据结构：借鉴Java语言，包括将值分成原始值和对象两大类。

函数的用法：借鉴Scheme语言和Awk语言，将函数当作第一等公民，并引入闭包。

原型继承模型：借鉴Self语言（Smalltalk的一种变种）。

正则表达式：借鉴Perl语言。

字符串和数组处理：借鉴Python语言。

为了保持简单，这种脚本语言缺少一些关键的功能，比如块级作用域、模块、子类型（subtyping）等等，但是可以利用现有功能找出解决办法。这种功能的不足，直接导致了后来JavaScript的一个显著特点：对于其他语言，你需要学习语言的各种功能，而对于JavaScript，你常常需要学习各种解决问题的模式。而且由于来源多样，从一开始就注定，JavaScript的编程风格是函数式编程和面向对象编程的一种混合体。

Netscape公司的这种浏览器脚本语言，最初名字叫做Mocha，1995年9月改为LiveScript。12月，Netscape公司与Sun公司（Java语言的发明者和所有者）达成协议，后者允许将这种语言叫做JavaScript。这样一来，Netscape公司可以借助Java语言的声势，而Sun公司则将自己的影响力扩展到了浏览器。

之所以起这个名字，并不是因为JavaScript本身与Java语言有多么深的关系（事实上，两者关系并不深），而是因为Netscape公司已经决定，使用Java语言开发网络应用程序，JavaScript可以像胶水一样，将各个部分连接起来。当然，后来的历史是Java语言的浏览器插件失败了，JavaScript反而发扬光大。

1995年12月4日，Netscape公司与Sun公司联合发布了JavaScript语言。

1996年3月，Navigator 2.0浏览器正式内置了JavaScript脚本语言。

## 1.2 web应用的出现

随着网络建设的完善和javascript语言本身不断的健壮，web页面从一开始满足文档的共享的需求，慢慢发展可以满足作为一个应用的需求。

Web应用程序也就是基于网页的应用程序，用户无需安装任何专用程序，只用浏览器来访问服务器，通过浏览网页就可以实现业务流程，相对于原来的客户端/服务器(C/S)模式，这种新的程序开发模式称为浏览器/服务器(B/S)模式，它是随着 Internet技术的兴起，对传统结构的一种改进。

### 1.2.1 \*\*\*\*\*\*\*\*

System.Array类是C#中各种数组的基类，其属性和方法一览如图8.1所示。（注意，图编号和图都有专门的样式，要求图统一缩放为60%，而且必须在正文中引出“如图8.1所示。”，同时图的编号在整章都是连续的）

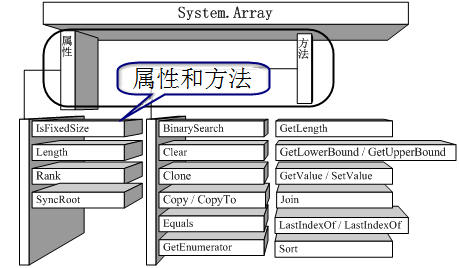


图8.1 System.Array类（图中的标注一定要清晰）

其常用属性和方法的简单说明如表8.1所示（表编号也必需在正文中引出）。先说明表的编号、表的意义，然后给出表，表编号和表内容的样式不同。

表8.1 Array类常用属性/方法说明（表头样式）

|  |  |
| --- | --- |
| 属性/方法 | 说明 |
| IsFixedSize | 指示 Array 是否具有固定大小（表格样式） |
| [Length](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasslengthtopic.htm) | 获得一个 32 位整数，表示 Array 的所有维数中元素的总数 |
| [Rank](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassranktopic.htm) | 获取 Array 的秩（维数） |
| [SyncRoot](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasssyncroottopic.htm) | 获取可用于同步对 Array 的访问的对象 |
| [BinarySearch](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassbinarysearchtopic.htm) | 使用二进制搜索算法在一维的排序 Array 中搜索值 |
| [Clone](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassclonetopic.htm) | 创建 Array 的浅表副本 |
| [Copy](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasscopytopic.htm)/[CopyTo](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasscopytotopic.htm) | 将一个 Array 的一部分复制到另一个 Array 中 |
| [GetLength](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassgetlengthtopic.htm) | 获取一个 32 位整数，表示 Array 的指定维中的元素。 |
| [GetLowerBound](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassgetlowerboundtopic.htm)/[GetUpperBound](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassgetupperboundtopic.htm) | 获取 Array 的指定维度的下/上限 |
| [GetValue](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassgetvaluetopic.htm)/[SetValue](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasssetvaluetopic.htm) | 获取/设置 Array 中的指定元素值 |
| [IndexOf](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclassindexoftopic.htm)/[LastIndexOf](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasslastindexoftopic.htm) | 返回一维 Array 或部分 Array 中某个值第一个/最后一个匹配项索引 |
| [Sort](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasssorttopic.htm) | 对一维 Array 对象中的元素进行排序 |

### 8.2.2 访问数组元素

访问数组的元素包括读取或设置某个元素的值，最基本的方法是通过下标定位元素，另外还可以使用GetValue/SetValue方法。

1．通过下标定位元素

C#中数组对其中的元素进行排序，并从0开始计数，这样每一个元素都会有一个唯一的下标，通过这个下标，就可以定位唯一的一个元素。通过示例来说明：

（1）一维数组

string[] myStrArr={"油" , "盐" , "酱", "醋", "毛毛熊"};

myStrArr[0]="油"

myStrArr[4]=" 毛毛熊"

如果试图访问超过下标范围的数据，则会出现如下异常：

System.IndexOutOfRangeException: 索引超出了数组界限。

（2）多维数组

string[,] myStrArr2={{"油","盐"},{"《围城》","《晨露》"},{"毛毛熊","Snoopy"}};

myStrArr2[0,0]="油"

myStrArr2[4,1]=" Snoopy "

（3）交错数组

01 int[][] myJaggedArray = new int [][]

02 {

03 new int[] {1,3,5,7,9},

04 new int[] {0,2,4,6},

05 new int[] {11,22}

06 };

则有：

01 myJaggedArray[0][0]=1

02 myJaggedArray[1][1]=2

03 myJaggedArray[2][1]=22

04 {

05 Console.WriteLine("item{0}",i);

06 for(int

07 j=myJaggedArray[i].GetLowerBound(0);j<=myJaggedArray[i].GetUpperBound(0);j++)

08 {

09 Console.WriteLine(" item{0}{1}:{2}",i,j,myJaggedArray[i][j]);

10 }

11 }

2．使用GetValue/SetValue

GetValue方法定义如下：

* public object GetValue(params int[] indices);

其中，多个int型参数indices的含义为下标。方法返回一个object对象，这是C#中所有对象的基类，使用多态性，它可以指向所有的C#对象。

代码8-3使用GetValue方法，循环输出一个二维数组所有元素：

01 //定义二维数组

02 string[,] myStrArr2=new string[,]{{"油","盐"},{"《围城》","《晨露》"},{"毛毛熊","Snoopy"}};

03 //循环输出

04 for(int i=myStrArr2.GetLowerBound(0);i<=myStrArr2.GetUpperBound(0);i++)

05 {

06 Console.WriteLine("item{0}",i);

07 for(int j=myStrArr2.GetLowerBound(1);j<=myStrArr2.GetUpperBound(1);j++)

08 {

09 Console.WriteLine(" item{0}{1}:{2}",i,j,myStrArr2.GetValue(i,j));

10 }

11 }

SetValue的功能为数组的某个元素赋值，其定义及参数表同GetValue相似，不作赘述。

### 8.2.3 遍历数组

遍历数组是指全部访问数组中的元素一次且仅一次，可以在遍历的过程中完成许多操作，如查找等。有两种方式可以遍历整个数组：

1．使用GetLowerBound/GetUpperBound方法

GetLowerBound方法可以获取数组某一维上的最低下标，而GetUpperBound则可获取其最高下标，利用这两个参数和for语句，就可以实现数组的遍历，如对于二位数组：

01 //定义二维数组

02 string[,] myStrArr2=new string[,]{{"油","盐"},{"《围城》","《晨露》"},{"毛毛熊","Snoopy"}};

03 //遍历

04 for(int i=myStrArr2.GetLowerBound(0);i<=myStrArr2.GetUpperBound(0);i++)

05 {

06 for(int j=myStrArr2.GetLowerBound(1);j<=myStrArr2.GetUpperBound(1);j++)

07 {

08 //处理每一个元素

09 }

10 }

2．使用foreach

还可以使用更为简便的方法来实现数组的遍历，那就是foreach关键字，这种方法对于处理高维数组尤其方便。

foreach语句格式如下：

foreach (data\_typt item\_name in arr\_name)

{

//处理每一个元素

}

以下内容用来说明一些特殊技巧或声明。注意必须以“注意：说明：声明：技巧：”开头，并且使用模板提供的“注意说明技巧”样式。

注意：foreach语句获取的元素是最深层的原子元素，因此，无论处理几维的数组，使用一层的foreach循环就可以了。

例如，代码8-5实现同样的二维数组遍历：

01 //定义二维数组

02 string[,] myStrArr2=new string[,]{{"油","盐"},{"《围城》","《晨露》"},{"毛毛熊","Snoopy"}};

03 //遍历

04 foreach(string item in myStrArr2)

05 {

06 {

07 //处理每一个元素

08 }

### 8.2.5 数组排序

对数组进行排序，是指按照一定的排序规则，如递增或递减规则，重新排序数组中的所有元素。可以使用Array类的Sort方法完成这个功能。

Sort方法有多种重载方式，常用的形式如下：

* public static [void](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemvoidclasstopic.htm) Sort([Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) array);

其中，参数array为待排序的数组。下面的示例首先定义了一个数组，含有元素{5,4,3,2,1}，然后利用Sort方法对其排序。

01 /// 利用Sort方法进行数组排序

02 public void test1()

03 {

04 int[] myArr = { 5, 4, 3, 2, 1 }; //定义数组

05

06 //输出原始数组：原始数组:5->4->3->2->1->

07 Console.WriteLine( "原始数组:" );

08 for(int i=0;i<myArr.Length;i++)

09 Console.Write("{0}->",myArr[i]);

10

11 Array.Sort( myArr ); //对数组排序

12

13 //并输出排序后的数组：1->2->3->4->5->

14 Console.WriteLine( "排序以后数组:" );

15 for(int i=0;i<myArr.Length;i++)

16 Console.Write("{0}->",myArr[i]);

17 }

* public static [void](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemvoidclasstopic.htm) Sort([Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) keys, [Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) items);

有时候需要进行所谓的关键字排序，例如，有两个数组arrSid和arrSname，分别代表一组学生的学号和姓名，如果想要根据学号顺序输出姓名，或反之，都需要使用数组的排序操作，那么，如何把这两个数组联系在一起排序呢？这时就可以使用Sort的这种形式进行关键字排序。

其中，参数keys代表关键字数组，而items代表另一个数组。利用Sort，下面的代码可实现上述需求：

01 /// 多个数组的关键字排序

02 public void test2()

03 {

04 //定义数组

05 int[] arrSid = { 5, 4, 3, 2, 1 };

06 string[] arrSname = { "张三", "李四", "王五", "麻子", "淘气" };

07

08 //输出原始数组：原始数组:张三(5)->李四(4)->王五(3)->麻子(2)->淘气(1)->

09 Console.WriteLine( "原始数组:" );

10 for(int i=0;i<arrSid.Length;i++)

11 Console.Write("{0}({1})->",arrSname[i],arrSid[i]);

12 Console.WriteLine();

13

14 //根据学号关键字排序

15 Array.Sort( arrSid,arrSname );

16

17 //并输出排序后的数组：淘气(1)->麻子(2)->王五(3)->李四(4)->张三(5)

18 Console.WriteLine( "排序以后数组:" );

19 for(int i=0;i<arrSid.Length;i++)

20 Console.Write("{0}({1})->",arrSname[i],arrSid[i]);

21 }

示例非常简单，输出已经在注释中给出，因此不作详细说明。

### 8.2.6 查找元素

在数组中查找元素，可以有两种解释，一是就是从整个数组中寻找到与给定值相同的元素来。可以使用Array类的BinarySearch方法完成这个功能。二是判断数组中是否含有一个特定的元素，可以用Contains方法实现。

1．BinarySearch方法

BinarySearch使用二进制搜索算法在一维的排序Array中搜索值，注意必须是已经排序的数组，如果找到给定的值，则返回其下标；否则，返回一个负整数。其常用形式如下：

* public static [int](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemint32classtopic.htm) BinarySearch([Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) array,[object](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemobjectclasstopic.htm) value);

其中，参数array为待搜索的数组，value为待寻找的元素值。下面的示例首先定义了一个数组，含有元素{5,4,3,2,1}，然后利用BinarySearch方法返回其中的元素3的下标（2）。

代码8-9 利用BinarySearch搜索数组元素示例：Class1.cs

01 /// 利用BinarySearch方法搜索元素

02 public void test1()

03 {

04 //定义数组

05 int[] myArr = { 5, 4, 3, 2, 1 };

06

07 //对数组排序

08 Array.Sort( myArr );

09

10 //搜索

11 int target=3;

12 int result=Array.BinarySearch(myArr,target); //2

13 Console.WriteLine("{0}的下标为{1}",target,result); //2

14 }

2．Contains方法

望文知义，Contains方法可以确定某个特定值是否包含在数组中，返回一个bool值。Array类的这个方法实际上是对IList接口中方法的实现，其常用形式为：

* bool IList.Contains([object](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemobjectclasstopic.htm) value);

其中，参数value代表所要验证的元素值，下面的示例判断学生数组arrSname中是否包含“王五”：

代码8-10 利用Contains判断数组是否包含某个元素示例：Class1.cs

01 /// 判断是否包含某个值

02 public void test2()

03 {

04 //定义数组

05 string[] arrSname = { "张三", "李四", "王五", "麻子", "淘气" };

06

07 //判断是否含有某值

08 string target="王五";

09 bool result=((System.Collections.IList)arrSname).Contains(target);

10 Console.WriteLine("包含{0}?{1}",target,result); //true

11 }

可以看到，在使用Contains方法时，需要首先将数组转换为IList（队列集合）对象。这是因为，本质上，数组是一种特殊的集合对象，因此可以把它转换为一个集合对象。对于集合，将在下一章中对其进行详细的讨论。

### 8.2.7 反转数组

反转数组是指，将一维数组中的全部或者部分元素的顺序，按照其逆序重新排列。可以使用Array类的Reverse静态方法完成这个功能。其常用的形式为：

* public static [int](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemint32classtopic.htm) Reverse([Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) array);

这个重载形式可以反转整个数组元素，参数array为待反转的数组。下面的示例首先定义了一个数组，含有元素{5,4,3,2,1}，然后利用Reverse方法进行反转。

代码8-11 利用Reverse反转数组示例：Class1.cs

1 /// 利用Reverse方法反转数组

2 public void test1()

3 {

4 //定义数组

5 int[] myArr = { 5, 4, 3, 2, 1 };

6

7 //输出原始数组：原始数组:5->4->3->2->1->

8 Console.WriteLine( "原始数组:" );

9 for(int i=0;i<myArr.Length;i++)

10 Console.Write("{0}->",myArr[i]);

11 Console.WriteLine();

12

13 //对数组反转

14 Array.Reverse( myArr );

15

16 //并输出反转后的数组：1->2->3->4->5->

17 Console.WriteLine( "反转以后数组:" );

18 for(int i=0;i<myArr.Length;i++)

19 Console.Write("{0}->",myArr[i]);

20 }

另外，还可以反转数组的一部分元素，这时的Reverse重载形式为：

* public static [int](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemint32classtopic.htm) Reverse([Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) array,int index, int length);

其中，参数index指定所要反转元素的其实下标，而length指定所要反转的元素个数。

### 8.2.8 复制数组

复制数组可以得到一个和原数组完全一样的新的数组，可以用Array的Copy或CopyTo方法来实现。

1．Copy方法

在使用Copy方法进行数组复制操作之前，必须首先为新的数组分配空间，然后再通过复制操作向新的数组空间中填入元素值。

静态方法Copy的常用重载形式为：

* public static [void](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemvoidclasstopic.htm) Copy([Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) sourceArray,[Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) destinationArray,[int](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemint32classtopic.htm) length);

其中，参数sourceArray为源数组，destinationArray为目标数组，而length为要复制的元素数目，默认的复制操作从第一个元素开始。

下面的示例首先定义了一个数组，含有元素{5,4,3,2,1}，然后利用Copy方法获取一个新的数组，包含了源数组的前三项。

代码8-12 利用Copy复制数组示例：Class1.cs

1 /// 利用Copy静态方法复制数组

2 public void test1()

3 {

4 //定义数组

5 int[] myArr = { 5, 4, 3, 2, 1 };

6

7 //输出原始数组：原始数组:5->4->3->2->1->

8 Console.WriteLine( "原始数组:" );

9 for(int i=0;i<myArr.Length;i++)

10 Console.Write("{0}->",myArr[i]);

11 Console.WriteLine();

12

13 //复制数组

14 int[] newArr=new int[3];

15 Array.Copy(myArr,newArr,3);

16

17 //并输出反复制的数组：5->4->3->

18 Console.WriteLine( "复制数组:" );

19 for(int i=0;i<newArr.Length;i++)

20 Console.Write("{0}->",newArr[i]);

21 }

注意：在使用Copy进行数组之前，必须要首先定义一个新的数组，并对其分配空间。另外，还需保证所分配的空间足够容纳所要复制的元素，否则，在复制时将出现异常：“System.ArgumentException: 目标数组的长度不够。”。

2．CopyTo方法

CopyTo和Copy方法的功能类似，但它是一个实例方法，即需要在数组对象上引用。另外的区别是，它只能复制源数组所有的元素，并可以空值元素在目标数组中存放的起始位置。其常用重载形式为：

* public virtual [void](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemvoidclasstopic.htm) CopyTo( [Array](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemarrayclasstopic.htm) array, [int](ms-help://MS.VSCC.2003/MS.MSDNQTR.2003FEB.2052/cpref/html/frlrfsystemint32classtopic.htm) index);

其中，参数array代表所要复制的目标数组，index则表示开始复制的元素下标。下面的代码实现对对源数组的复制，并从从第三个位置开始存放。

代码8-13 利用CopyTo复制数组示例：Class1.cs

1 /// 利用CopyTo实例方法复制数组

2 public void test2()

3 {

4 //定义数组

5 int[] myArr = { 5, 4, 3, 2, 1 };

6

7 //输出原始数组：原始数组:5->4->3->2->1->

8 Console.WriteLine( "原始数组:" );

9 for(int i=0;i<myArr.Length;i++)

10 Console.Write("{0}->",myArr[i]);

11 Console.WriteLine();

12

13 //复制数组

14 int[] newArr=new int[7];

15 myArr.CopyTo(newArr,2);

16

17 //并输出反复制的数组：0->0->5->4->3->2->1->

18 Console.WriteLine( "复制数组:" );

19 for(int i=0;i<newArr.Length;i++)

20 Console.Write("{0}->",newArr[i]);

21 }

另外，从输出结果还可以看到，在初始化int型数组时，其默认值为0。那么对于字符串数组，默认值是什么呢？请读者自己试验。

## 8.3 综合示例

【本节示例参考：\源代码\C8\Example\_BookList】

本节将利用上一章和本章所介绍的字符串和数组操作技术，实现一个完整的示例：我的书房之图书排序。其功能为按照一定的排列顺序显示一系列图书信息，最终实现结果如图8.2所示。

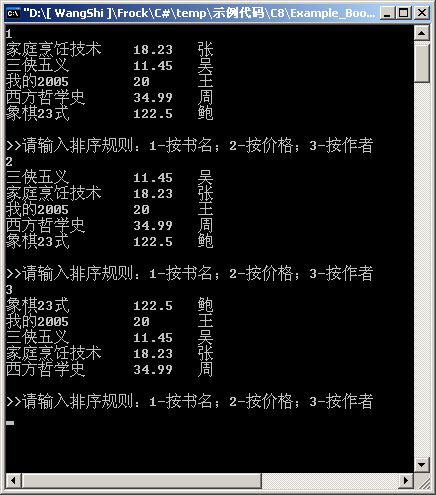


图8.2 图书列表示例最后结果

整体的思路是：首先，每本图书的信息用一个图书对象来表示，包括书名、价格、作者三项，然后所有的图书信息放在一个数组arrBooks中，最后通过一个BookList类来完成图书显示。如果实例比较大，必须先给出一个实例的实现流程，最好用Visio画，编辑可以直接修改。



图8.3 本例的实现流程

1．Book类

Book类如图8.3所示。



图8.3 Book类图

属性介绍：

* strName：图书名
* intPrice：图书价格
* strAuthor：图书作者

图书类Book的实现代码参考如下。

代码8-15 图书类Book的实现代码：Class1.cs

1. /// <summary>

2. /// 图书类

3. /// </summary>

4. class Book

5. {

6. public string strName; //图书名

7. public double dblPrice; //图书价格

8. public string strAuthor; //图书作者

9.

10. /// <summary>

11. /// 构造函数

12. /// </summary>

13. /// <param name="\_strName">图书名</param>

14. /// <param name="\_dblPrice">图书价格</param>

15. /// <param name="\_strAuthor">图书作者</param>

16. public Book(string \_strName,double \_dblPrice,string \_strAuthor)

17. {

18. this.strName=\_strName;

19. this.dblPrice=\_dblPrice;

20. this.strAuthor=\_strAuthor;

21. }

22. }

图书类的实现非常简单，它有三个公共属性（第6-9行），还有一个构造函数（第10-21）行。

2．BookList类

BookList类，用于按照不同的排序规则显示图书列表，它有三个静态方法，输入参数都是一个图书数组，功能为把所有书目进行排序并显示出来。BookList类如图8.4所示。



图8.4 BookList类

这三个排序显示的静态方法为：

* DisplayByName：根据图书名顺序显示多本图书
* DisplayBy：根据图书价格顺序显示图书
* DisplayBy：根据图书作者顺序显示图书

下面单独来看DisplayByName的实现，代码如下。

代码8-16 按图书名显示图书实现代码：Class1.cs

1. /// <summary>

2. /// 按照图书名显示一个图书数组中的多本图书

3. /// </summary>

4. /// <param name="arrBooks">图书数组</param>

5. public static void DisplayByName(Book[] arrBooks)

6. {

7. //获取图书数目，用户动态建立“书名”数组

8. int bookNumber=arrBooks.GetUpperBound(0)-arrBooks.GetLowerBound(0)+1;

9.

10. //使用CreateInstance方法，动态建立“书名”数组

11. int[] lengths=new int[]{bookNumber};

12. int[] lowerBounds=new int[]{0};

13. Array arrNames=

Array.CreateInstance(Type.GetType("System.String"),lengths,lowerBounds);

14.

15. //为“书名”数组赋值

16. for(int i=arrBooks.GetLowerBound(0);i<=arrBooks.GetUpperBound(0);i++)

17. arrNames.SetValue(arrBooks[i].strName,i);

18.

19. //利用Sort方法，以“书名”为键，将图书排序

20. Array.Sort(arrNames,arrBooks);

21.

22. //显示排序后的图书列表

23. foreach(Book item in arrBooks)

24. {

25. Console.WriteLine("{0} {1} {2}",item.strName,item.dblPrice,item.strAuthor);

26. }

27. }

第8行，方法首先获取了输入参数arrBooks的图书数目，这是为了后面动态建立书名数组而准备。

接下来，第10－13行利用Array的CreateInstance静态方法，动态建立了一个一维数组arrNames，下标从0开始，用于存储图书的书名信息，这是为了后面利用Sort方法进行排序作准备。

第15－17行对书名数组进行赋值。

第20行进行了排序，这是一个键值排序，以书名为键，排序所有的图书。

第22－26行顺序显示排序好后的图书信息。

另外的两个方法DisplayByPrice和DisplayByAuthor与上面所介绍的DisplayByName非常类似，此处不再进行详细的说明。

而整个BookList类的实现如下。

代码8-17 BookList类完整实现代码：Class1.cs

1. class BookList

2. {

3. /// <summary>

4. /// 按照图书名显示一个图书数组中的多本图书

5. /// </summary>

6. /// <param name="arrBooks">图书数组</param>

7. public static void DisplayByName(Book[] arrBooks)

8. {

9. //…

10. }

11.

12. /// <summary>

13. /// 按照图书价格显示一个图书数组中的多本图书

14. /// </summary>

15. /// <param name="arrBooks">图书数组</param>

16. public static void DisplayByPrice(Book[] arrBooks)

17. {

18. //…

19. }

20.

21. /// <summary>

22. /// 按照图书名显示一个图书数组中的多本图书

23. /// </summary>

24. /// <param name="arrBooks">图书数组</param>

25. public static void DisplayByAuthor(Book[] arrBooks)

26. {

27. //…

28. }

29. }

类分别实现了上面介绍的三个函数，除此之外，没有其他更多的内容。

3．主函数

主测试函数首先使用Book类，定义了5本数，并把这5本书放在一个数组内，然后通过BookList类的静态方法进行排序输出。实现代码如下所示。

代码8-18 Main函数：Class1.cs

1. /// <summary>

2. /// 应用程序的主入口点。

3. /// </summary>

4. static void Main(string[] args)

5. {

6. Book b1=new Book("我的2005",20,"王");

7. Book b2=new Book("家庭烹饪技术",18.23,"张");

8. Book b3=new Book("西方哲学史",34.99,"周");

9. Book b4=new Book("三侠五义",11.45,"吴");

10. Book b5=new Book("象棋23式",122.50,"鲍");

11.

12. Book[] myBooksArr=new Book[]{b1,b2,b3,b4,b5};

13.

14. Console.WriteLine("\n>>请输入排序规则：1-按书名；2-按价格；3-按作者");

15. string type=Console.ReadLine();

16. switch(Convert.ToInt32(type))

17. {

18. case 1:

19. BookList.DisplayByName(myBooksArr);

20. break;

21. case 2:

22. BookList.DisplayByPrice(myBooksArr);

23. break;

24. case 3:

25. BookList.DisplayByAuthor(myBooksArr);

26. break;

27. default:

28. break;

29. }

30. }

示例使用了switch语句，来判断用户的输入，然后根据用户的选择，调用BookList不同的方法。

4．扩展

读者可以对从以下方面这个应用进行进一步的扩展：

（1）添加一本新图书到图书列表中；

（2）显示图书价格大于参数min的所有图书（参考8.2.2、8.2.3内容）

（3）根据书名、作者，或者价格，查找某本图书（参考8.2.6）

## 8.4 温故知新

* 学完本章中，读者需要回答：

1．什么是数组？

2．.NET基础类库中的System.Array和数组是什么关系？

3．能够使用数组完成（1）遍历（2）同步（3）排序（4）查找（5）反转（6）复制操作吗？

4．如何根据不同的情况动态生成数组？

* 在下一章中，读者会了解到：

1．C#中的集合命名空间System.Collections；

2．使用ArrayList类完成（1）添加元素（2）插入元素（3）删除元素（4）排序元素（5）查找元素（6）遍历列表操作；

3．Queue类和Stack类的作为功能受限的列表，其典型操作的实现。

4．哈希表的概念，以及使用Hashtable和SortedList类实现普通的哈希表和排序的哈希表。

## 8.5 习题

通过下面的习题来检验本章的学习，习题答案参考光盘。

【本章习题答案在光盘中 \源代码\C8\习题】

1．创建一个学生数组，并从控制台程序中输出这些学生。

2．有以下一个数值数组，从控制台程序中输出大于25的数。

3．XXXX

4．