2. C#类库查询手册

1 Array 类

用括号声明数组是 C#中使用 Array 类的记号。在后台使用 C#语法，会创建一个派生于抽象基类 Array 的新类。这样，就可以使用 Array 类为每个 C#数组定义的方法和属性了。

Array 类实现了 IEumerable、ICollection 和 IList 接口，以访问和枚举数组中的元素。由于用定制数组创建的类派生于 Array 抽象类，所以能使用通过数组变量执行的接口中的方法和属性。

- IEumerable 接口是由 foreach 语句用于迭代数组的接口。
- ICollection 接口派生于 IEumerable 接口，这个接口主要用于确定集合中的元素个数，或用于同步。
- IList 接口派生于 ICollection 接口，Array 类实现 IList 接口的主要原因是 IList 接口定义了 Item 属性，以使用索引器访问元素。

Array 类包含的如下属性和方法可以用于每个数组实例。

属性及方法	说明
Length	Length 属性返回数组中的元素个数。如果是一个多维数组，该属性会返回所有阶的元素个数。如果需要确定一维中的元素个数，则可以使用 GetLength()方法。
LongLength	Length 属性返回 int 值，而 LongLength 属性返回 long 值。如果数组包含的元素个数超出了 32 位 int 值的取值范围，就需要使用 LongLength 属性，来获得元素个数。
Rank	使用 Rank 属性可以获得数组的维数。
CreateInstance()	如果事先不知道元素的类型，就可以使用该静态方法，因为类型可以作为 Type 对象传送给 CreateInstance()方法。
SetValue()	SetValue()方法设置数组的元素，其参数是每一维的索引。
Clone()	因为数组是引用类型，所以将一个数组变量赋予另一个数组变量，就会得到两个指向同一数组的变量。而复制数组，会使数组实现 ICloneable 接口。这个接口定义的 Clone()方法会创建数组的浅副本。
Sort()	Array 类实现了对数组中元素的冒泡排序。Sort()方法需要数组中的元素实现 IComparable 接口。简单类型，如 System.String 和 System.Int32 实现了 IComparable 接口，所以可以对包含这些类型的元素排序。
Count	Count 属性可确定集合中的元素个数，它返回的值与 Length 属性相同。
IsSynchronized SyncRoot	IsSynchronized 属性确定集合是否是线程安全的。对于数组，这个属性总是返回 false。对于同步访问，SyncRoot 属性可以用于线程安全的访问。
CopyTo()	利用 CopyTo()方法可以将数组的元素复制到现有的数组中。它类似于静态方法 Array.Copy()。
Add()	Add()方法用于在集合中添加元素。对于数组，该方法会抛出 NotSupportedException 异常。
Clear()	Clear()方法可清除数组中的所有元素。值类型设置为 0，引用类型设置为 null。
Contains()	Contains()方法可以确定某个元素是否在数组中。其返回值是 true 或 false。这个方法会对数组中的所有元素进行线性搜索，直到找到所需元素为止。

IndexOf()	IndexOf()方法与 Contains()方法类似，也是对数组中的所有元素进行线性搜索。不同的是，IndexOf()方法会返回所找到的第一个元素的索引。
Insert()	对于集合，Insert()方法用于插入元素，对于数组，这方法抛出 NotSupportedException 异常。
Remove() RemoveAt()	对于集合，Remove()和 RemoveAt()可删除元素。对于数组，这些方法抛出 NotSupportedException 异常。
IsFixedSize	数组的大小总是固定的，所以这个属性总是返回 true。
IsReadOnly	数组总是可以读/写的，所以这个属性返回 false。
Item	Item 属性可以用整型索引访问数组。
MoveNext()	MoveNext()方法移动到集合的下一个元素上，如果有这个元素，该方法就返回 true。如果集合不再有更多的元素，该方法就返回 false。
Current	属性 Current 返回光标所在的元素。
Reset()	Reset()方法将光标重新定位于集合的开头。许多枚举会抛出 NotSupportedException 异常。
GetEumerator()	数组或集合执行带 GetEumerator()方法的 IEumerable 接口。GetEumerator()方法返回一个执行 IEumerable 接口的枚举。接着，foreach 语句就可以使用 IEumerable 接口迭代集合了。

2 System.Text.String 类

System.String 是一个类，专门用于存储字符串，允许对字符串进行许多操作。由于这种数据类型非常重要，C#提供了它自己的关键字和相关的语法，以便于使用这个类来处理字符串。使用“+”运算符重载可以连接字符串。

C#还允许使用类似于索引器的语法来提取指定的字符。这个类可以完成许多常见的任务，例如替换字符、删除空白和把字母变成大写形式等。可用的方法如下：

方法	作用
Compare()	比较字符串的内容，考虑文化背景(区域)，确定某些字符是否相等。
CompareOrdinal()	与 Compare 一样，但不考虑文化背景。
Concat()	把多个字符串实例合并为一个实例。
CopyTo()	把特定数量的字符从选定的下标复制到数组的一个全新实例中。
Format()	格式化包含各种值的字符串和如何格式化每个值的说明符。
IndexOf()	定位字符串中第一次出现某个给定子字符串或字符的位置。
IndexOfAny()	定位字符串中第一次出现某个字符或一组字符的位置。
Insert()	把一个字符串实例插入到另一个字符串实例的指定索引处。
Join()	合并字符串数组，建立一个新字符串。
LastIndexOf()	与 IndexOf 一样，但定位最后一次出现的位置。
LastIndexOfAny()	与 IndexOfAny，但定位最后一次出现的位置。
PadLeft()	在字符串的开头，通过添加指定的重复字符填充字符串。
PadRight()	在字符串的结尾，通过添加指定的重复字符填充字符串。
Replace()	用另一个字符或子字符串替换字符串中给定的字符或子字符串。
Split()	在出现给定字符的地方，把字符串拆分为一个子字符串数组。
Substring()	在字符串中获取给定位置的子字符串。

ToLower()	把字符串转换为小写形式。
ToUpper()	把字符串转换为大写形式。
Trim()	删除首尾的空白。

3 System.Text.StringBuilder 类

String 对象是不可改变的。每次使用 System.String 类中的方法之一时，都要在内存中创建一个新的字符串对象，这就需要为该新对象分配新的空间。在需要对字符串执行重复修改的情况下，与创建新 String 对象相关的系统开销可能会非常昂贵。如果要修改字符串而不创建新的对象，则可以使用 System.Text.StringBuilder 类。

在使用 String 类构造一个字符串时，要给它分配足够的内存来保存字符串，但 StringBuilder 通常分配的内存会比需要的更多。开发人员可以选择显式指定 StringBuilder 要分配多少内存，但如果没有显式指定，存储单元量在默认情况下就根据 StringBuilder 初始化时的字符串长度来确定。它有两个主要的属性：

1) Length 指定字符串的实际长度。

2) Capacity 是字符串占据存储单元的最大长度。

主要的 StringBuilder 方法如下表所示。

名称	作用
Append()	给当前字符串添加一个字符串
AppendFormat()	添加特定格式的字符串
Insert()	在当前字符串中插入一个子字符串
Remove()	从当前字符串中删除字符
Replace()	在当前字符串中，用某个字符替换另一个字符，或者用当前字符串中的一个子字符串替换另一字符串。
ToString()	把当前字符串转换为 System.String 对象(在 System.Object 中被重写)

4 其他类

类	说明
ArrayList	使用大小可按需动态增加的数组实现 IList 接口。
BitArray	管理位值的压缩数组，该值表示为布尔值，其中 true 表示位是打开的(1)，false 表示位是关闭的(0)。
Hashtable	表示键/值对的集合，这些键/值对根据键的哈希代码进行组织。
Queue	表示对象的先进先出集合。
SortedList	表示键/值对的集合，这些键值对按键排序并可按照键和索引访问。
Stack	表示对象的简单的后进先出非泛型集合。
Capture	表示来自单个成功的子表达式捕获的结果。
CaptureCollection	表示一个捕获组做出的捕获的集合。
Group	表示来自单个捕获组的结果。
GroupCollection	返回一次匹配中捕获的组的集合。
Match	表示单个正则表达式匹配的结果。
MatchCollection	表示通过以迭代方式将正则表达式模式应用于输入字符串所找到的成功匹配的集合。

Regex	表示不可变的正则表达式。
SqlBulkCopy	使您可以用其他源的数据有效批量加载 SQL Server 表。
SqlCommand	表示要对 SQL Server 数据库执行的一个 Transact-SQL 语句或存储过程。
SqlCommandBuilder	用于将对 DataSet 所做的更改与关联的 SQL Server 数据库的更改相协调。
SqlConnection	表示 SQL Server 数据库的一个打开的连接。无法继承此类。
SqlConnectionStringBuilder	为创建和管理由 Sql Connection 类使用的连接字符串的内容提供了一种简单方法。
SqlDataAdapter	表示用于填充 DataSet 和更新 SQL Server 数据库的一组数据命令和一个数据库连接。
SqlDataReader	提供一种从 SQL Server 数据库读取行的只进流的方式。
SqlError	收集与 SQL Server 返回的警告或错误有关的信息。
SqlErrorCollection	收集 SQL Server .NET Framework 数据提供程序生成的所有错误。
SqlException	当 SQL Server 返回警告或错误时引发的异常。无法继承此类。
SqlParameter	表示 SqlCommand 的参数，也可以是它到 DataSet 列的映射。
SqlTransaction	表示要在 SQL Server 数据库中处理的 Transact-SQL 事务。
Constraint	表示可在一个或多个 DataColumn 对象上强制的约束。
ConstraintCollection	表示 DataTable 的约束的集合。
DataColumn	表示 DataTable 中列的架构。
DataColumnCollection	表示 DataTable 的 DataColumn 对象的集合。
DataException	表示使用 ADO.NET 组件发生错误时引发的异常。
DataRelation	表示两个 DataTable 对象之间的父/子关系。
DataRelationCollection	表示此 DataSet 的 DataRelation 对象的集合。
DataRow	表示 DataTable 中的一行数据
DataRowCollection	表示 DataTable 的行的集合
DataRowView	表示 DataRow 的自定义视图
DataSet	表示数据在内存中的缓存
DataTable	表示内存中数据的一个表
DataTableCollection	表示 DataSet 的表的集合
DataTableReader	DataTableReader 以一个或多个只读、只进结果集的形式获取一个或多个 DataTable 对象的内容
DataView	表示用于排序、筛选、搜索、编辑和导航的 DataTable 的可绑定数据的自定义视图
XmlAttribute	表示一个属性。此属性的有效值和默认值在文档类型定义(DTD)或架构中进行定义。
XmlAttributeCollection	表示可以按名称或索引访问的属性的集合。
XmlDataDocument	允许通过相关的 DataSet 存储、检索和操作结构化数据。
XmlDeclaration	表示 XML 声明节点：<?xml version='1.0'...?>。
XmlDocument	表示 XML 文档。
XmlDocumentFragment	表示对树插入操作有用的轻量对象。

XmlDocumentType	表示文档类型声明。
XmlElement	表示一个元素。
XmlException	返回有关最后一个异常的详细信息。
XmlNode	表示 XML 文档中的单个节点。
XmlNodeReader	表示提供对 XmlNode 中的 XML 数据进行快速、非缓存的只进访问的读取器。
XmlReader	表示提供对 XML 数据进行快速、非缓存、只进访问的读取器。
XmlReaderSettings	指定在 Create 方法创建的 XmlReader 对象上支持的一组功能。
XmlText	表示元素或属性的文本内容。
XmlTextReader	表示提供对 XML 数据进行快速、非缓存、只进访问的读取器。
XmlTextWriter	表示提供快速、非缓存、只进方法的编写器，该方法生成包含 XML 数据（这些数据符合 W3C 可扩展标记语言(XML)1.0 和“XML 中的命名空间”建议）的流或文件。
XmlWriter	表示一个编写器，该编写器提供一种快速、非缓存和只进的方式来生成包含 XML 数据的流或文件。
XmlWriterSettings	指定在由 System.Xml.XmlWriter.Create 方法创建的 XmlWriter 对象上支持的一组功能。
BinaryReader	用特定的编码将基元数据类型读作二进制值。
BinaryWriter	以二进制形式将基元类型写入流，并支持用特定的编码写入字符串。
BufferedStream	给另一流上的读写操作添加一个缓冲层。
Directory	公开用于创建、移动和枚举通过目录和子目录的静态方法。
DirectoryInfo	公开用于创建、移动和枚举目录和子目录的实例方法。
DriveInfo	提供对有关驱动器的信息的访问。
File	提供用于创建、复制、删除、移动和打开文件的静态方法，并协助创建 FileStream 对象。
FileInfo	提供创建、复制、删除、移动和打开文件的实例方法，并且帮助创建 FileStream 对象。无法继承此类。
FileStream	公开以文件为主的 Stream，既支持同步读写操作，也支持异步读写操作。
IOException	发生 I/O 错误时引发的异常。
MemoryStream	创建其支持存储区为内存的流。
Path	对包含文件或目录路径信息的 String 实例执行操作。这些操作是以跨平台的方式执行的。
StreamReader	实现一个 TextReader，使其以一种特定的编码从字节流中读取字符。
StreamWriter	实现一个 TextWriter，使其以一种特定的编码向流中写入字符。
StringReader	实现从字符串进行读取的 TextReader。
StringWriter	实现一个用于将信息写入字符串的 TextWriter。该信息存储在基础 StringBuilder 中。
TextReader	表示可读取连续字符系列的读取器。
TextWriter	表示可以编写一个有序字符系列的编写器。该类为抽象类。