562

TP3/348
J17a3

.NET 框架程序员参考手册·常规操作篇

吉尚戎　等编著

国防工业出版社

·北京·

内 容 简 介

本书详细介绍了.NET 框架中有关常规操作的内容。全书共 8 章，主要内容包括：使用配置文件，配置部件，定制组件安装程序，国际化支持，定制和安装服务程序，处理字符编码，设置规则表达式以及使用定时器等。

图书在版编目(CIP)数据

.NET 框架程序员参考手册·常规操作篇/吉尚戎等编著 .—北京：国防工业出版社，2002.1
ISBN 7-118-02746-4

Ⅰ.N... Ⅱ.吉... Ⅲ.计算机网络－程序设计
Ⅳ. TP393

中国版本图书馆 CIP 数据核字(2001)第 085403 号

国防工业出版社出版发行
(北京市海淀区紫竹院南路 23 号)
(邮政编码 100044)
北京奥隆印刷厂印刷
新华书店经售
*
开本 787×1092 1/16 印张 26½ 615 千字
2002 年 1 月第 1 版 2002 年 1 月北京第 1 次印刷
印数：1—4000 册 定价：39.00 元

(本书如有印装错误，我社负责调换)

Visual Studio.NET 7.0 是微软推出的新一代可视化集成开发环境，其中的.NET 就是指.NET 框架（.NET Framework）。.NET 框架是一种用于构建、配置、运行 Web 服务和应用程序的多语言环境，它主要由统一的编程类库、通用语言运行库（Common Language Runtime）和 ASP.NET（Active Server Pages.NET）三个部分组成。.NET 框架不但提供了诸如自动内存管理之类的很多强有力的功能，而且它的引入使得多语言间的无缝互用成为现实。

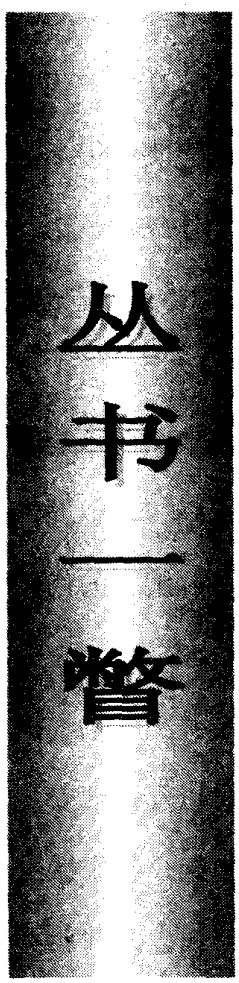
从某种程度上说，新版本的 Visual Studio 就是以.NET 框架为中心的：新引入的 C#语言本身并无类库，而是充分利用.NET 框架提供的功能； Visual Basic 7.0 语言上做了很大修改，而这些修改正是为了实现与.NET 框架的无缝兼容；Visual C++ 7.0 中提供了受控代码（Managed Extensions）编程，使得由 C++编写的代码也依然能够使用.NET 框架的服务。

.NET 框架中的类型有很多的功能，例如，封装数据结构、执行 I/O 操作、访问数据、控制服务器、获取类信息以及激活安全检查等。.NET 框架中既包括比较抽象的基类，也包括由基类派生的、具有实际功能的类。这些派生类已经提供足够强大的功能，但是如果需要，程序员依然可以通过继续派生提供更强大的功能。.NET 框架还包括接口及其默认实现。要使用接口的功能，程序员可以自己实现这些接口，也可以直接使用（或派生）运行库中的接口实现类。

本书约定

由于.NET 框架涉及面极宽，为了节省篇幅，使读者能以最少的费用得到最广泛的信息，本书有以下几点约定：

- 在介绍每个名称空间时，都会将其中定义的成员以表格形式给出。读者可以根据这些表格给出的信息，迅速确定将查阅的内容。而.NET 框架命名规则也使确定成员功能变得更加容易，例如 DirectorySeparatorChar 的意思为目录分隔符字符（Directory-Separator-Char）。
- 凡是“待提供”的名称空间成员，都只是在列表中简述，而不做进一步详细说明。
- 对于那些功能、形式类似的名称空间成员，只是选择其中最具代表性的进行介绍，其他的只列表简述。读者若需得知这些简述成员的详细形式，只要参考对应的详述成员即可。
- 另外，类库的每个层次都会自动继承其基层次中的所有成员，因此在介绍时也不再介绍这些继承所得的成员，而只是指出其派生层次。这样读者即可根据其派生层次确定其中包含有哪些继承成员。当然，对于那些被重载的继承成员，我们还是会加以详细介绍。



由于.NET 框架类库的内容非常浩繁，为了向读者提供更具针对性的参考信息，《.NET 框架程序员参考手册》丛书分为以下 6 册：

- **框架基础篇**

本篇主要包括整个.NET 框架的根名称空间：System，以及微软的两个服务名称空间：Microsoft.ComServices 和 Microsoft.Win32。这些名称空间为用户提供了底层功能和服务支持，同时也是开发高级功能的基础。

- **数据访问篇**

本篇主要包括为数据 I/O 提供支持的名称空间：System.IO；为数据库访问提供支持的名称空间：System.Data 及其下属的三级名称空间。通过这些名称空间，用户能够方便地进行数据存取、数据库事务和 XML 编档。

- **网络编程篇**

本篇主要包括为进行网络和 Web 服务提供支持的名称空间：System.Security，System.Net，System.Web，及其下属的三级名称空间等。

- **用户界面篇**

本篇主要包括为用户界面提供支持的名称空间：System.Windows. Forms 等。

- **常规操作篇**

本篇主要包括为 Windows 下的常用操作提供支持的名称空间：System.Configration，System.Globalization，System.Service-Process，System.Text 和 System.Timers 等。

- **组件模型篇**

本篇主要包括为组件模型提供支持的名称空间：System. ComponentModel，System.Collections，System.Resources，System.Core 和 System.Threading 等。

试读结束：需要全本请在线购买： www.ertongbook.com

目　录

第 1 章　使用配置文件 1

1.1 System.Configuration 名称空间的类成员 1
1.1.1 ConfigurationException 类 1
1.1.2 ConfigurationSettings 类 5
1.1.3 DictionarySectionHandler 类 7
1.1.4 NameValueSectionHandler 类 9
1.1.5 SingleTagSectionHandler 类 11
1.2 System.Configuration 名称空间的接口成员 12

第 2 章　配置部件 14

2.1 System.Configuration.Assemblies 名称空间的结构成员 14
2.2 System.Configuration.Assemblies 名称空间的枚举成员 17
2.2.1 AssemblyHashAlgorithm 枚举 17
2.2.2 AssemblyVersionCompatibility 枚举 18

第 3 章　定制组件安装程序 20

3.1 System.Configuration.Install 名称空间的类成员 21
3.1.1 AssemblyInstaller 类 21
3.1.2 ComponentInstaller 类 27
3.1.3 InstallContext 类 29
3.1.4 Installer 类 32
3.1.5 InstallerCollection 类 45
3.1.6 InstallEventArgs 类 49
3.1.7 InstallException 类 51
3.1.8 TransactedInstaller 类 52
3.2 System.Configuration.Install 名称空间的枚举成员 54
3.3 System.Configuration.Install 名称空间的 Delegate 成员 55

第 4 章　国际化支持 56

4.1 System.Globalization 名称空间的类成员 57
4.1.1 Calendar 类 57

4.1.2 CompareInfo 类 74
4.1.3 CultureInfo 类 85
4.1.4 DateTimeFormatInfo 类 106
4.1.5 DaylightTime 类 127
4.1.6 GregorianCalendar 类 129
4.1.7 HebrewCalendar 类 140
4.1.8 HijriCalendar 类 152
4.1.9 JapaneseCalendar 类 163
4.1.10 JulianCalendar 类 174
4.1.11 KoreanCalendar 类 185
4.1.12 NumberFormatInfo 类 197
4.1.13 RegionInfo 类 218
4.1.14 SortKey 类 230
4.1.15 StringInfo 类 233
4.1.16 TaiwanCalendar 类 236
4.1.17 TextElementEnumerator 类 247
4.1.18 TextInfo 类 250
4.1.19 ThaiBuddhistCalendar 类 255
4.2 System.Globalization 名称空间的枚举成员 267
4.2.1 CalendarWeekRule 枚举 267
4.2.2 CompareOptions 枚举 268
4.2.3 CultureTypes 枚举 269
4.2.4 DateTimeStyles 枚举 270
4.2.5 GregorianCalendarTypes 枚举 271
4.2.6 NumberStyles 枚举 272
4.2.7 UnicodeCategory 枚举 273

第 5 章 定制和安装服务程序 275

5.1 System.ServiceProcess 名称空间的类成员 276
5.1.1 ServiceBase 类 276
5.1.2 ServiceController 类 288
5.1.3 ServiceInstaller 类 299
5.1.4 ServiceProcessDescriptionAttribute 类 306
5.1.5 ServiceProcessInstaller 类 307
5.2 System.ServiceProcess 名称空间的枚举成员 313
5.2.1 PowerBroadcastStatus 枚举 313
5.2.2 ServiceControllerStatus 枚举 314
5.2.3 ServiceStartMode 枚举 314
5.2.4 ServiceType 枚举 315

第 6 章　处理字符编码.....317

6.1　ASCIIEncoding 类.....317
6.2　Decoder 类.....322
6.3　Encoder 类.....325
6.4　Encoding 类.....328
6.5　StringBuilder 类.....341
6.6　UnicodeEncoding 类.....354
6.7　UTF7Encoding 类.....360
6.8　UTF8Encoding 类.....365

第 7 章　设置规则表达式.....371

7.1　System.Text.RegularExpressions 名称空间的类成员.....371
7.1.1　Capture 类.....371
7.1.2　CaptureCollection 类.....373
7.1.3　Group 类.....376
7.1.4　GroupCollection 类.....378
7.1.5　Match 类.....381
7.1.6　MatchCollection 类.....384
7.1.7　Regex 类.....387
7.1.8　RegexCompilationInfo 类.....400
7.2　System.Text.RegularExpressions 名称空间的枚举成员.....403
7.3　System.Text.RegularExpressions 名称空间的 Delegate 成员.....404

第 8 章　使用定时器.....405

8.1　System.Timers 名称空间的类成员.....405
8.1.1　ElapsedEventArgs 类.....405
8.1.2　Timer 类.....406
8.1.3　TimersDescriptionAttribute 类.....413
8.2　System.Timers 名称空间的 Delegate 成员.....414

第 1 章　使用配置文件

System.Configuration 名称空间为访问.NET 框架配置和处理配置文件（.config）中的错误提供了类和接口。该名称空间的组成如表 1-1 所示。

表 1-1　System.Configuration 名称空间成员

成员类型/成员名称	描　　述
类	
ConfigurationException	由配置段的处理函数使用，以报告配置设置中的错误
ConfigurationSettings	为读取指定配置段中的设置提供了方法
DictionarySectionHandler	用以读取键-值对
IgnoreSectionHandler	待提供
NameValueSectionHandler	用以为指定配置段提供名称-值对配置信息
SingleTagSectionHandler	提供了从配置段中读取所有键-值对的手段
接口	
IConfigurationSectionHandler	定义了配置段处理函数必须实现的“协议”，以参与配置设置的解析

本章主要介绍 System.Configuration 名称空间中的类和接口。本名称空间中的成员都能用于 Windows 98、Windows NT 4.0、Windows Millennium Edition、Windows 2000 和 Windows XP 等操作系统。与 System.Configuration 名称空间对应的组件为：System.dll。

1.1　System.Configuration 名称空间的类成员

本节将向读者详细介绍 System.Configuration 名称空间中类成员的定义和使用。

1.1.1　ConfigurationException 类

原型：

[Visual Basic]

```
Public Class ConfigurationException Inherits SystemException
```

[C#]

```
public class ConfigurationException : SystemException
```

[C++]

```
public __gc class ConfigurationException : public SystemException
[JScript]
public class ConfigurationException extends SystemException
```

用途：

ConfigurationException 类由配置段的处理函数使用，以报告配置设置中的错误。

说明：

ConfigurationException 类是 Object/Exception/SystemException 的子层次。

成员：

构造函数

原型：

```
[Visual Basic]
Public Sub New()
Public Sub New(ByVal message As String)
Public Sub New(ByVal message As String,ByVal inner As Exception)
Public Sub New(ByVal message As String,ByVal node As XmlNode)
Public Sub New(ByVal message As String,ByVal inner As Exception,ByVal node As XmlNode)
Public Sub New(ByVal message As String,ByVal filename As String,ByVal line As Integer)
Public Sub New(ByVal message As String,ByVal inner As Exception,ByVal filename As String, ByVal line As Integer)
[C#]
public ConfigurationException();
public ConfigurationException(string message);
public ConfigurationException(string message,Exception inner);
public ConfigurationException(string message,XmlNode node);
public ConfigurationException(string message,Exception inner,XmlNode node);
public ConfigurationException(string message,string filename,int line);
public ConfigurationException(string message,Exception inner,string filename,int line);
[C++]
public: ConfigurationException();
public: ConfigurationException(String* message);
public: ConfigurationException(String* message,Exception* inner);
public: ConfigurationException(String* message,XmlNode* node);
public: ConfigurationException(String* message,Exception* inner,XmlNode* node);
public: ConfigurationException(String* message,String* filename,int line);
public: ConfigurationException(String* message,Exception* inner,String* filename, int line);
[JScript]
public function ConfigurationException();
public function ConfigurationException(message : String);
public function ConfigurationException(message : String,inner : Exception);
public function ConfigurationException(message : String, node : XmlNode);
```

```
public function ConfigurationException(message : String,inner : Exception,node : XmlNode);
public function ConfigurationException(message : String,filename : String,line : int);
public function ConfigurationException(message : String,inner : Exception,filename : String,    line : int);
```

用途：

调用 ConfigurationException 类的构造函数，以初始化该类的一个新实例。

参数：

message——当抛出异常时为客户显示的消息。

inner——抛出当前异常的内部异常。

node——包含错误的配置段节点名。

filename——包含错误的配置文件的名称。

line——配置文件中包含错误的行号。

说明：

ConfigurationException 类的第一个（默认）构造函数，将其所有字段都初始化为其默认值。

BareMessage 属性

原型：

```
[Visual Basic]
Public ReadOnly Property BareMessage As String
[C#]
public string BareMessage {get;}
[C++]
public: __property String* get_BareMessage();
[JScript]
public function get BareMessage() : String;
```

用途：

使用 BareMessage 属性，以获取基础错误消息。

Filename 属性

原型：

```
[Visual Basic]
Public ReadOnly Property Filename As String
[C#]
public string Filename {get;}
[C++]
public: __property String* get_Filename();
[JScript]
public function get Filename() : String;
```

用途：

使用 Filename 属性，以获取其中发生错误的配置文件的名称。

试读结束：需要全本请在线购买：　www.ertongbook.com

Line 属性

原型：

[Visual Basic]

```
Public ReadOnly Property Line As Integer
```

[C#]

```
public int Line {get;}
```

[C++]

```
public: __property int get_Line();
```

[JScript]

```
public function get Line() : int;
```

用途：

使用 Line 属性，以返回发生错误的行号。

Message 属性

原型：

[Visual Basic]

```
Overrides Public ReadOnly Property Message As String
```

[C#]

```
public override string Message {get;}
```

[C++]

```
public: __property virtual String* get_Message();
```

[JScript]

```
public function get Message() : String;
```

用途：

使用 Message 属性，以获取或设置包含发生错误的文件名和行号的字符串。

GetXmlNodeFilename 方法

原型：

[Visual Basic]

```
Public Shared Function GetXmlNodeFilename(ByVal node As XmlNode) As String
```

[C#]

```
public static string GetXmlNodeFilename(XmlNode node);
```

[C++]

```
public: static String* GetXmlNodeFilename(XmlNode* node);
```

[JScript]

```
public static function GetXmlNodeFilename(node : XmlNode) : String;
```

用途：

调用 GetXmlNodeFilename 方法，以返回包含配置段节点的文件的名称。

参数：

node——包含错误的配置段节点的名称。

GetXmlNodeLineNumber 方法

原型：

[Visual Basic]

```
Public Shared Function GetXmlNodeLineNumber(ByVal node As XmlNode) As Integer
```

[C#]

```
public static int GetXmlNodeLineNumber(XmlNode node);
```

[C++]

```
public: static int GetXmlNodeLineNumber(XmlNode* node);
```

[JScript]

```
public static function GetXmlNodeLineNumber(node : XmlNode) : Int;
```

用途：

调用 GetXmlNodeLineNumber 方法，以返回包含配置段节点的行号。

参数：

node——包含错误的配置段节点的名称。

1.1.2　ConfigurationSettings 类

原型：

[Visual Basic]

```
NotInheritable Public Class ConfigurationSettings
```

[C#]

```
public sealed class ConfigurationSettings
```

[C++]

```
public __gc __sealed class ConfigurationSettings
```

[JScript]

```
public class ConfigurationSettings
```

用途：

ConfigurationSettings 类为读取指定配置段中的设置提供了方法。

说明：

ConfigurationSettings 类是 Object 的子层次。

成员：

构造函数

原型：

[Visual Basic]

```
Public Sub New()
```

[C#]

```
public ConfigurationSettings();
```

[C++]

```
public: ConfigurationSettings();
[JScript]
public function ConfigurationSettings();
```

用途：

调用 ConfigurationSettings 类的构造函数，以初始化该类的一个新实例。

说明：

本默认构造函数，将所有字段都初始化为其默认值。

AppSettings 属性

原型：

```
[Visual Basic]
Public Shared ReadOnly Property AppSettings As NameValueCollection
[C#]
public static NameValueCollection AppSettings {get;}
[C++]
public: __property static NameValueCollection* get_AppSettings();
[JScript]
public static function get AppSettings() : NameValueCollection;
```

用途：

使用 AppSettings 属性，以获取 TBD 配置段中的配置设置。

说明：

AppSettings 属性的值是包含配置设置的名称-值对的 NameValueCollection。

GetConfig 方法

原型：

```
[Visual Basic]
Public Shared Function GetConfig(ByVal sectionName As String) As Object
[C#]
public static object GetConfig(string sectionName);
[C++]
public: static Object* GetConfig(String* sectionName);
[JScript]
public static function GetConfig(sectionName : String) : Object;
```

用途：

调用 GetConfig 方法，以返回用户自定义配置段中的配置设置。

参数：

sectionName——将读取其配置设置的配置段名。

返回值：

包含配置设置的对象。

1.1.3　DictionarySectionHandler 类

原型：

[Visual Basic]

```
Public Class DictionarySectionHandler Implements IConfigurationSectionHandler
```

[C#]

```
public class DictionarySectionHandler : IConfigurationSectionHandler
```

[C++]

```
public __gc class DictionarySectionHandler : public IConfigurationSectionHandler
```

[JScript]

```
public class DictionarySectionHandler implements IConfigurationSectionHandler
```

用途：

DictionarySectionHandler 类用以读取键-值对。

说明：

DictionarySectionHandler 类是 Object 的子层次。

成员：

构造函数

原型：

[Visual Basic]

```
Public Sub New()
```

[C#]

```
public DictionarySectionHandler();
```

[C++]

```
public: DictionarySectionHandler();
```

[JScript]

```
public function DictionarySectionHandler();
```

用途：

调用 DictionarySectionHandler 类的构造函数，以初始化该类的一个新实例。

说明：

本默认构造函数，将所有字段都初始化为其默认值。

KeyAttributeName 属性

原型：

[Visual Basic]

```
Overridable Protected ReadOnly Property KeyAttributeName As String
```

[C#]

```
protected virtual string KeyAttributeName {get;}
```

[C++]

```
protected: __property virtual String* get_KeyAttributeName();
[JScript]
protected function get KeyAttributeName() : String;
```

用途：

使用 KeyAttributeName 属性，以获取键标志的标签名。

说明：

KeyAttributeName 属性可以由派生类重载，以修改键标志标签的名称。该属性的默认值为"key"。

ValueAttributeName 属性

原型：

```
[Visual Basic]
Overridable Protected ReadOnly Property ValueAttributeName As String
[C#]
protected virtual string ValueAttributeName {get;}
[C++]
protected: __property virtual String* get_ValueAttributeName();
[JScript]
protected function get ValueAttributeName() : String;
```

用途：

使用 ValueAttributeName 属性，以获取值标签名。

说明：

ValueAttributeName 属性可以由派生类重载，以修改值标签的名称。该属性的默认值为"value"。

Create 方法

原型：

```
[Visual Basic]
Overridable Public Function Create(ByVal parent As Object,ByVal context As Object,ByVal section As XmlNode) As Object
[C#]
public virtual object Create(object parent,object context,XmlNode section);
[C++]
public: virtual Object* Create(Object* parent,Object* context,XmlNode* section);
[JScript]
public function Create(parent : Object, context : Object,section : XmlNode) : Object;
```

用途：

调用 Create 方法，以创建一个新 DictionarySectionHandler 对象并将其添加到集合中。

参数：

parent——父配置段。

context——提供对虚路径的访问，配置段处理函数使用它以计算配置值。一般来说，此参数被保留（空值）。

section——包含将被处理的配置信息的 XML 节点。提供对配置段的 XML 内容的直接访问。

1.1.4　NameValueSectionHandler 类

原型：

[Visual Basic]

```
Public Class NameValueSectionHandler Implements IConfigurationSectionHandler
```

[C#]

```
public class NameValueSectionHandler : IConfigurationSectionHandler
```

[C++]

```
public __gc class NameValueSectionHandler : public IConfigurationSectionHandler
```

[JScript]

```
public class NameValueSectionHandler implements IConfigurationSectionHandler
```

用途：

NameValueSectionHandler 类用以为指定配置段提供名称-值对配置信息。

说明：

NameValueSectionHandler 类是 Object/NameValueSectionHandler 的子层次。

成员：

构造函数

原型：

[Visual Basic]

```
Public Sub New()
```

[C#]

```
public NameValueSectionHandler();
```

[C++]

```
public: NameValueSectionHandler();
```

[JScript]

```
public function NameValueSectionHandler();
```

用途：

调用 NameValueSectionHandler 类的构造函数，以初始化该类的一个新实例。

说明：

本默认构造函数，将所有字段都初始化为其默认值。

KeyAttributeName 属性

原型：

[Visual Basic]

```
Overridable Protected ReadOnly Property KeyAttributeName As String
```


试读结束：需要全本请在线购买：　www.ertongbook.com