

783/636²
J7a2

NET

框架程序员参考手册·

网络编程篇

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 框架中有关网络编程的内容。全书共 11 章,主要内容包 括:维护系统安全,设置安全策略,管理系统账户,网络编程接口,使用 Windows 套接字,访问 Web 站点,管理 Web 缓存,ASP.NET 配置,发送 E-mail,实现 ASP.NET 安全以及构造和使用 Web 服务等。

本书主要供程序开发人员使用。

图书在版编目(CIP)数据

.NET 框架程序员参考手册·网络编程篇/吉尚戎等
编著. —北京:国防工业出版社,2002.1

ISBN 7-118-02725-1

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 081536 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 37¼ 885 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:58.00 元

(本书如有印装错误,我社负责调换)

关于 .NET 框架

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类足够强大的功能，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，而不作进一步详细说明。
- 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.Configuration`，`System.Globalization`，`System.ServiceProcess`，`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`

目 录

| | |
|---|----|
| 第 1 章 维护系统安全 | 1 |
| 1.1 System.Security 名称空间的类成员 | 2 |
| 1.1.1 CodeAccessPermission 类 | 2 |
| 1.1.2 NamedPermissionSet 类 | 9 |
| 1.1.3 PermissionSet 类 | 13 |
| 1.1.4 SecurityElement 类 | 22 |
| 1.1.5 SecurityException 类 | 30 |
| 1.1.6 SecurityManager 类 | 32 |
| 1.1.7 SuppressUnmanagedCodeSecurityAttribute 类 | 37 |
| 1.1.8 UnverifiableCodeAttribute 类 | 38 |
| 1.1.9 VerifierException 类 | 39 |
| 1.1.10 XMLSyntaxException 类 | 40 |
| 1.2 System.Security 名称空间的接口成员 | 42 |
| 1.2.1 IEvidenceFactory 接口 | 42 |
| 1.2.2 IPermission 接口 | 43 |
| 1.2.3 ISecurityEncodable 接口 | 45 |
| 1.2.4 ISecurityPolicyEncodable 接口 | 47 |
| 1.2.5 IStackWalk 接口 | 48 |
| 1.3 System.Security 名称空间的枚举成员 | 51 |
| 第 2 章 设置安全策略 | 52 |
| 2.1 System.Security.Policy 名称空间的类成员 | 53 |
| 2.1.1 AllMembershipCondition 类 | 53 |
| 2.1.2 ApplicationDirectory 类 | 56 |
| 2.1.3 ApplicationDirectoryMembershipCondition 类 | 58 |
| 2.1.4 CodeGroup 类 | 61 |
| 2.1.5 Evidence 类 | 67 |
| 2.1.6 FileCodeGroup 类 | 73 |
| 2.1.7 FirstMatchCodeGroup 类 | 77 |
| 2.1.8 Hash 类 | 80 |
| 2.1.9 HashMembershipCondition 类 | 82 |
| 2.1.10 NetCodeGroup 类 | 85 |
| 2.1.11 PermissionRequestEvidence 类 | 88 |

| | | |
|--------------|---|------------|
| 2.1.12 | PolicyException 类 | 91 |
| 2.1.13 | PolicyLevel 类 | 92 |
| 2.1.14 | PolicyStatement 类 | 96 |
| 2.1.15 | Publisher 类 | 100 |
| 2.1.16 | PublisherMembershipCondition 类 | 102 |
| 2.1.17 | Site 类 | 106 |
| 2.1.18 | SiteMembershipCondition 类 | 109 |
| 2.1.19 | SkipVerificationMembershipCondition 类 | 113 |
| 2.1.20 | StrongName 类 | 117 |
| 2.1.21 | StrongNameMembershipCondition 类 | 120 |
| 2.1.22 | UnionCodeGroup 类 | 125 |
| 2.1.23 | Url 类 | 127 |
| 2.1.24 | URLMembershipCondition 类 | 130 |
| 2.1.25 | WebPage 类 | 133 |
| 2.1.26 | WebPageMembershipCondition 类 | 134 |
| 2.1.27 | Zone 类 | 137 |
| 2.1.28 | ZoneMembershipCondition 类 | 140 |
| 2.2 | System.Security.Policy 名称空间的接口 | 144 |
| 2.2.1 | IIdentityPermissionFactory 接口 | 144 |
| 2.2.2 | IMembershipCondition 接口 | 145 |
| 2.3 | System.Security.Policy 名称空间的枚举成员 | 146 |
| 第 3 章 | 管理系统账户 | 148 |
| 3.1 | System.Security.Principal 名称空间的类成员 | 148 |
| 3.1.1 | GenericIdentity 类 | 148 |
| 3.1.2 | GenericPrincipal 类 | 149 |
| 3.1.3 | WindowsIdentity 类 | 150 |
| 3.1.4 | WindowsImpersonationContext 类 | 155 |
| 3.1.5 | WindowsPrincipal 类 | 155 |
| 3.2 | System.Security.Principal 名称空间的接口成员 | 157 |
| 3.2.1 | IIdentity 接口 | 157 |
| 3.2.2 | IPrincipal 接口 | 159 |
| 3.3 | System.Security.Principal 名称空间的枚举成员 | 160 |
| 3.3.1 | PrincipalPolicy 枚举 | 160 |
| 3.3.2 | WindowsAccountType 枚举 | 161 |
| 3.3.3 | WindowsBuiltInRole 枚举 | 161 |
| 第 4 章 | 网络编程接口 | 163 |
| 4.1 | System.Net 名称空间的类成员 | 164 |
| 4.1.1 | AuthenticationManager 类 | 164 |
| 4.1.2 | Authorization 类 | 167 |

| | | |
|--------|------------------------------------|-----|
| 4.1.3 | Cookie 类 | 170 |
| 4.1.4 | CredentialCache 类 | 171 |
| 4.1.5 | DNS 类 | 174 |
| 4.1.6 | DnsPermission 类 | 179 |
| 4.1.7 | DnsPermissionAttribute 类 | 182 |
| 4.1.8 | EndPoint 类 | 183 |
| 4.1.9 | EndpointPermission 类 | 185 |
| 4.1.10 | FileWebRequest 类 | 186 |
| 4.1.11 | FileWebResponse 类 | 194 |
| 4.1.12 | GlobalProxySelection 类 | 197 |
| 4.1.13 | HttpVersion 类 | 199 |
| 4.1.14 | HttpWebRequest 类 | 200 |
| 4.1.15 | HttpWebResponse 类 | 219 |
| 4.1.16 | IPAddress 类 | 225 |
| 4.1.17 | IPEndPoint 类 | 230 |
| 4.1.18 | IPHostEntry 类 | 234 |
| 4.1.19 | NetworkCredential 类 | 236 |
| 4.1.20 | ProtocolViolationException 类 | 238 |
| 4.1.21 | ServicePoint 类 | 239 |
| 4.1.22 | ServicePointManager 类 | 244 |
| 4.1.23 | SocketAddress 类 | 247 |
| 4.1.24 | SocketPermission 类 | 248 |
| 4.1.25 | SocketPermissionAttribute 类 | 253 |
| 4.1.26 | WebClient 类 | 256 |
| 4.1.27 | WebException 类 | 263 |
| 4.1.28 | WebHeaderCollection 类 | 264 |
| 4.1.29 | WebPermission 类 | 268 |
| 4.1.30 | WebPermissionAttribute 类 | 272 |
| 4.1.31 | WebProxy 类 | 274 |
| 4.1.32 | WebRequest 类 | 280 |
| 4.1.33 | WebResponse 类 | 290 |
| 4.2 | System.Net 名称空间的接口成员 | 294 |
| 4.2.1 | IAuthenticationModule 接口 | 294 |
| 4.2.2 | ICertificatePolicy 接口 | 296 |
| 4.2.3 | ICredentials 接口 | 297 |
| 4.2.4 | IWebProxy 接口 | 298 |
| 4.2.5 | IWebRequestCreate 接口 | 300 |
| 4.3 | System.Net 名称空间的枚举成员 | 300 |
| 4.3.1 | HttpStatusCode 枚举 | 301 |

| | | |
|--------------|------------------------------------|------------|
| 4.3.2 | NetworkAccess 枚举..... | 303 |
| 4.3.3 | TransportType 枚举 | 304 |
| 4.3.4 | WebExceptionStatus 枚举 | 304 |
| 4.4 | System.Net 名称空间的 Delegate 成员 | 305 |
| 第 5 章 | 使用 Windows 套接字 | 307 |
| 5.1 | System.Net.Sockets 名称空间的类成员 | 308 |
| 5.1.1 | LingerOption 类 | 308 |
| 5.1.2 | MulticastOption 类 | 309 |
| 5.1.3 | NetworkStream 类 | 310 |
| 5.1.4 | Socket 类 | 320 |
| 5.1.5 | SocketException 类 | 340 |
| 5.1.6 | TCPClient 类 | 342 |
| 5.1.7 | TCPListener 类 | 347 |
| 5.1.8 | UDPClient 类 | 351 |
| 5.2 | System.Net.Sockets 名称空间的枚举成员 | 356 |
| 5.2.1 | AddressFamily 枚举 | 356 |
| 5.2.2 | ProtocolFamily 枚举 | 357 |
| 5.2.3 | ProtocolType 枚举 | 358 |
| 5.2.4 | SelectMode 枚举 | 359 |
| 5.2.5 | SocketFlags 类 | 360 |
| 5.2.6 | SocketOptionLevel 枚举 | 361 |
| 5.2.7 | SocketOptionName 枚举 | 361 |
| 5.2.8 | SocketShutdown 枚举 | 363 |
| 5.2.9 | SocketType 枚举 | 363 |
| 第 6 章 | 访问 Web 站点 | 365 |
| 6.1 | System.Web 名称空间的类成员 | 366 |
| 6.1.1 | HttpApplication 类 | 366 |
| 6.1.2 | HttpApplicationState 类 | 370 |
| 6.1.3 | HttpBrowserCapabilities 类 | 375 |
| 6.1.4 | HttpCachePolicy 类 | 384 |
| 6.1.5 | HttpCacheVaryByHeaders 类 | 391 |
| 6.1.6 | HttpCacheVaryByParams 类 | 393 |
| 6.1.7 | HttpClientCertificate 类 | 394 |
| 6.1.8 | HttpCompileException 类 | 397 |
| 6.1.9 | HttpContext 类 | 399 |
| 6.1.10 | HttpCookie 类 | 407 |
| 6.1.11 | HttpCookieCollection 类 | 411 |
| 6.1.12 | HttpException 类 | 415 |
| 6.1.13 | HttpFileCollection 类 | 417 |

| | | |
|--------------|--------------------------------------|------------|
| 6.1.14 | HttpModuleCollection 类 | 419 |
| 6.1.15 | HttpException 类 | 421 |
| 6.1.16 | HttpPostedFile 类 | 423 |
| 6.1.17 | HttpRequest 类 | 425 |
| 6.1.18 | HttpResponse 类 | 437 |
| 6.1.19 | HttpRuntime 类 | 452 |
| 6.1.20 | HttpServerUtility 类 | 453 |
| 6.1.21 | HttpStaticObjectsCollection 类 | 459 |
| 6.1.22 | HttpUnhandledException 类 | 463 |
| 6.1.23 | HttpUtility 类 | 464 |
| 6.1.24 | HttpWorkerRequest 类 | 468 |
| 6.1.25 | HttpWriter 类 | 477 |
| 6.1.26 | ProcessInfo 类 | 480 |
| 6.1.27 | TraceContext 类 | 484 |
| 6.2 | System.Web 名称空间的接口成员 | 487 |
| 6.2.1 | IHttpAsyncHandler 接口 | 487 |
| 6.2.2 | IHttpHandler 接口 | 488 |
| 6.2.3 | IHttpHandlerFactory 接口 | 489 |
| 6.2.4 | IHttpModule 接口 | 490 |
| 6.3 | System.Web 名称空间的枚举成员 | 491 |
| 6.3.1 | HttpCacheability 枚举 | 491 |
| 6.3.2 | HttpCacheRevalidation 枚举 | 492 |
| 6.3.3 | HttpValidationStatus 枚举 | 493 |
| 6.3.4 | ProcessShutdownReason 枚举 | 493 |
| 6.3.5 | ProcessStatus 枚举 | 494 |
| 6.3.6 | TraceModeEnum 枚举 | 495 |
| 6.4 | System.Web 名称空间的 Delegate 成员 | 495 |
| 第 7 章 | 管理 Web 缓存 | 496 |
| 7.1 | System.Web.Caching 名称空间的类成员 | 496 |
| 7.1.1 | Cache 类 | 496 |
| 7.1.2 | CacheDependency 类 | 501 |
| 7.2 | System.Web.Caching 名称空间的枚举成员 | 503 |
| 7.2.1 | CacheItemPriority 枚举 | 503 |
| 7.2.2 | CacheItemPriorityDecay 枚举 | 504 |
| 7.2.3 | CacheItemRemovedReason 枚举 | 504 |
| 7.3 | System.Web.Caching 名称空间的 Delegate 成员 | 505 |
| 第 8 章 | ASP.NET 配置 | 506 |
| 8.1 | AuthenticationMode 枚举 | 506 |
| 8.2 | FormsAuthPasswordFormat 枚举 | 507 |

| | | |
|---------------|--|------------|
| 8.3 | FormsProtectionEnum 枚举..... | 507 |
| 第 9 章 | 发送 E-mail..... | 509 |
| 9.1 | System.Web.Mail 名称空间的类成员..... | 509 |
| 9.1.1 | MailAttachment 类..... | 509 |
| 9.1.2 | MailMessage 类..... | 511 |
| 9.1.3 | SmtMail 类..... | 516 |
| 9.2 | System.Web.Mail 名称空间的枚举成员..... | 517 |
| 9.2.1 | MailEncoding 枚举..... | 517 |
| 9.2.2 | MailFormat 枚举..... | 518 |
| 9.2.3 | MailPriority 枚举..... | 519 |
| 第 10 章 | 实现 ASP.NET 安全..... | 520 |
| 10.1 | System.Web.Security 名称空间的类成员..... | 521 |
| 10.1.1 | DefaultAuthenticationEventArgs 类..... | 521 |
| 10.1.2 | DefaultAuthenticationModule 类..... | 522 |
| 10.1.3 | FileAuthorizationModule 类..... | 524 |
| 10.1.4 | FormsAuthentication 类..... | 525 |
| 10.1.5 | FormsAuthenticationEventArgs 类..... | 531 |
| 10.1.6 | FormsAuthenticationModule 类..... | 533 |
| 10.1.7 | FormsAuthenticationTicket 类..... | 535 |
| 10.1.8 | FormsIdentity 类..... | 539 |
| 10.1.9 | PassportAuthenticationEventArgs 类..... | 541 |
| 10.1.10 | PassportAuthenticationModule 类..... | 543 |
| 10.1.11 | PassportIdentity 类..... | 545 |
| 10.1.12 | UrlAuthorizationModule 类..... | 557 |
| 10.1.13 | WindowsAuthenticationEventArgs 类..... | 558 |
| 10.1.14 | WindowsAuthenticationModule 类..... | 560 |
| 10.2 | System.Web.Security 名称空间的 Delegate 成员..... | 562 |
| 10.2.1 | DefaultAuthenticationEventHandler Delegate..... | 562 |
| 10.2.2 | FormsAuthenticationEventHandler Delegate..... | 563 |
| 10.2.3 | PassportAuthenticationEventHandler Delegate..... | 563 |
| 10.2.4 | WindowsAuthenticationEventHandler Delegate..... | 564 |
| 第 11 章 | 构造和使用 Web 服务..... | 565 |
| 11.1 | WebMethodAttribute 类..... | 565 |
| 11.2 | WebService 类..... | 572 |
| 11.3 | WebServiceAttribute 类..... | 576 |
| 11.4 | WebServiceBindingAttribute 类..... | 579 |

第 1 章 维护系统安全

System.Security 名称空间为 .NET 框架的安全系统提供了基础结构，这包括接口、标志、异常和许可基类。System.Security 名称空间的组成如表 1-1 所示。

表 1-1 System.Security 名称空间成员

| 成员类型/成员名称 | 描述 |
|--|--|
| 类 | |
| CodeAccessPermission | 定义了所有代码访问许可的基础结构 |
| NamedPermissionSet | 定义了具有相关名称和描述的许可集 |
| PermissionSet | 代表能包含多种不同许可的集合，并提供使用和修改这些许可的方法 |
| SecurityElement | 代表编码安全对象的 XML 对象模型 |
| SecurityException | 当检测到安全错误时，将抛出由本类代表的异常 |
| SecurityManager | 为类与安全系统的交互提供了主访问点 |
| SuppressUnmanagedCodeSecurityAttribute | 允许受控代码无需安全检查即可调用非受控代码 |
| UnverifiableCodeAttribute | 用以标记包含无法检验的代码的模块 |
| VerifierException | 当安全策略需要代码为类型安全，而验证进程却无法验证代码为类型安全时，将抛出由本类所代表的异常 |
| XMLSyntaxException | 当解析 XML 时出现语法错误，将抛出由本类所代表的异常 |
| 接口 | |
| IevidenceFactory | 用以获取对象的 Evidence |
| Ipermission | 定义了所有许可对象的基础功能 |
| IsecurityEncodable | 定义了许可对象状态与 XML 元素之间进行转换的方法 |
| IsecurityPolicyEncodable | 为许可对象状态与 XML 元素间的转换方法提供支持 |
| IstackWalk | 定义了处理检查堆栈的方法，从而防止受控代码受到恶意攻击，和防止其他方法重载这些性能 |
| 枚举 | |
| PolicyLevelType | 待提供 |
| SecurityZone | 定义了由安全策略使用的、与安全区域相关的整数值 |

本章主要介绍 System.Security 名称空间中包括的类、接口和枚举。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.Security 名称空间对应的组件为：mscorlib.dll。

1.1 System.Security 名称空间的类成员

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

1.1.1 CodeAccessPermission 类

原型:

[Visual Basic]

MustInherit Public Class CodeAccessPermission Implements IPPermission, ISecurityEncodable, IStackWalk

[C#]

public abstract class CodeAccessPermission : IPPermission, ISecurityEncodable, IStackWalk

[C++]

public __gc __abstract class CodeAccessPermission : public IPPermission, ISecurityEncodable, IStackWalk

[JScript]

public abstract class CodeAccessPermission implements IPPermission, ISecurityEncodable, IStackWalk

用途:

CodeAccessPermission 类定义了所有代码访问许可的基础结构。

说明:

CodeAccessPermission 类是 Object 的子层次。所谓代码访问许可，就是通过检查堆栈实现 Demand 方法，从而确保了所有代码调用者都被授予许可。

在创建 CodeAccessPermission 的派生类时，必须重载如下成员：Copy、Intersect、IsSubsetOf、ToXml、FormXml 和 Union。

成员:

构造函数

原型:

[Visual Basic]

Protected Sub New()

[C#]

protected CodeAccessPermission();

[C++]

protected: CodeAccessPermission();

[JScript]

protected function CodeAccessPermission();

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

说明: 该构造函数应由 CodeAccessPermission 的派生类调用，以初始化该类型的状态。

Assert 方法**原型:**

[Visual Basic]

NotOverridable Public Sub Assert()

[C#]

public void Assert();

[C++]

public: __sealed void Assert();

[JScript]

public function Assert();

用途: 调用 `Assert` 方法, 以断言调用代码能访问由当前许可对象标识的资源, 即使在堆栈中处于较高位置的调用者不具备访问资源的授权。

异常: 如果调用代码不具备 `SecurityPermissionFlag.Assertion`, 则方法将抛出 `SecurityException` 异常。

说明: 调用堆栈通常按照由上向下的顺序, 因此在调用堆栈中位置较高的方法, 将调用堆栈中位置较低的方法。`Assert` 方法能终止对调用链中前一个调用者的检查。因此, 即使前一个调用者不具备必需的许可, 它们依然可以访问资源。为了断言许可, 代码必须被授予 `SecurityPermission` 许可, 以具备调用 `Assert` 方法的能力。

对 `Assert` 的调用将保持有效, 直到调用代码返回给调用者。同样, `RevertAssert` 或 `RevertAll` 将删除被挂起的 `Assert`。

对于未被授权的许可 `Assert` 将被忽略, 这是因为对该许可的请求无法成功。如果在调用堆栈中处于较低位置的代码为该许可调用 `Demand`, 那么当堆栈行至试图调用 `Assert` 的代码时, 则将抛出 `SecurityException` 异常。这也是因为调用 `Assert` 的代码不具备许可, 即使它已试图调用 `Assert` 方法断言该许可。

由于会使调用链中所有代码无需具备必要的授权, 即可访问指定资源, 那么如果使用 `Assert` 方法不当, 则将打开安全漏洞。

`Assert` 方法不能被继承。

Copy 方法**原型:**

[Visual Basic]

MustOverride Public Function Copy() As IPermission

[C#]

public abstract IPermission Copy();

[C++]

public: virtual IPermission* Copy() = 0;

[JScript]

public abstract function Copy() : IPermission;

用途: 当由派生类实现时, `Copy` 方法用以创建并返回当前许可对象的一个拷贝。

返回值: 当前许可对象的拷贝。

说明: Copy 方法返回的对象拷贝具有对资源相同的访问权限。该方法必须由派生类实现。

Demand 方法

原型:

[Visual Basic]

```
Overridable Public Sub Demand()
```

[C#]

```
public virtual void Demand();
```

[C++]

```
public: virtual void Demand();
```

[JScript]

```
public function Demand();
```

用途: 调用 Demand 方法, 以在运行时确定调用链中的所有调用者, 是否都已被授予由当前许可对象指定的许可。

异常: 如果调用链中的调用者不具备请求的许可; 或调用链中的调用者已经对被请求的许可调用了 Deny 或 PermitOnly, 则方法将抛出 SecurityException 异常。

说明: 该方法通常由安全库使用, 以确保调用者具备访问资源的权限。例如, 安全类库中的某个文件类将在执行由调用者请求的文件操作前, 为 FileIOPermission 调用 Demand。

调用 Demand 方法的代码许可并不会被检查。检查将从与该代码最近的调用者开始, 在堆栈中自下而上地进行。堆栈中位置较高的方法将调用堆栈中位置较低的方法。只有无 SecurityException 异常产生时, Demand 方法才会成功。

Demand 方法不能被重载。

Deny 方法

原型:

[Visual Basic]

```
NotOverridable Public Sub Deny()
```

[C#]

```
public void Deny();
```

[C++]

```
public: __sealed void Deny();
```

[JScript]

```
public function Deny();
```

用途: 调用 Deny 方法, 以拒绝堆栈中的高位调用者, 通过调用本方法的代码访问由当前许可对象指定的资源。

说明: Deny 方法阻止调用堆栈中的高位调用者, 通过调用 Deny 方法的代码访问被保护的资源, 即使这些调用者具备访问这些资源的权限。

Deny 方法能减少程序员的责任, 或防止意外的安全攻击。这是因为, 它能阻止调用

Deny 的方法，被用于访问由禁止许可保护的资源。如果某个方法对特定许可调用了 Deny 方法，而调用堆栈中的低位调用者为该许可调用了 Demand 方法，那么当安全检查到达 Deny 时将失败。

对 Deny 的调用将保持有效，直到调用代码返回给调用者。同样，RevertDeny 或 RevertAll 将删除未完成的 Deny。

对于未被授权的许可 Deny 将被忽略，这是因为对该许可的请求无法成功。

FromXml 方法

原型：

[Visual Basic]

```
MustOverride Public Sub FromXml(ByVal elem As SecurityElement)
```

[C#]

```
public abstract void FromXml(SecurityElement elem);
```

[C++]

```
public: virtual void FromXml(SecurityElement* elem) = 0;
```

[JScript]

```
public abstract function FromXml(elem : SecurityElement);
```

用途：当由派生类实现时，FromXml 方法用以重构具有由 XML 编码指定的状态的安全对象。

参数：elem——用以重构安全对象的 XML 编码。

说明：用以扩展安全对象的定制代码需要实现 ToXml 和 FromXml 方法，以使得对象可被安全编码。

在 CodeAccessPermission 的派生类中，必须重载本方法。

Intersect 方法

原型：

[Visual Basic]

```
MustOverride Public Function Intersect(ByVal target As IPPermission) As IPPermission
```

[C#]

```
public abstract IPPermission Intersect(IPPermission target);
```

[C++]

```
public: virtual IPPermission* Intersect(IPPermission* target) = 0;
```

[JScript]

```
public abstract function Intersect(target : IPPermission) : IPPermission;
```

用途：当由派生类实现时，Intersect 方法用以创建并返回一个许可，该许可为当前许可对象和目标许可对象的交集。

参数：target——将与当前许可对象取交集的许可对象。该参数必须与当前许可对象的类型相同。

返回值：代表当前许可对象和目标许可对象交集的新许可对象。如果交集为空，则新许可对象为空引用。

异常：如果 target 为空引用，并且与当前许可对象类型不同，则方法将抛出

ArgumentException 异常。

说明：

两个许可的交集就是它们所共有的操作集合。

在 CodeAccessPermission 的派生类中，必须实现 Intersect 方法。

IsSubsetOf 方法

原型：

[Visual Basic]

MustOverride Public Function IsSubsetOf(ByVal target As IPermission) As Boolean

[C#]

public abstract bool IsSubsetOf(IPermission target);

[C++]

public: virtual bool IsSubsetOf(IPermission* target) = 0;

[JScript]

public abstract function IsSubsetOf(target : IPermission) : Boolean;

用途：当在派生类中实现时，IsSubsetOf 方法用以确定当前许可对象是否为指定许可的子集。

参数：target——指定了将测试的许可对象。该参数必须与当前许可对象的类型相同。

返回值：如果当前许可为 target 的子集，则返回 true，否则返回 false。

异常：如果 target 与当前许可对象类型不同，则方法将抛出 ArgumentException 异常。

说明：如果由当前许可对象所指定的操作，都包含于指定许可对象中，那么就称当前许可对象为指定许可对象的子集。例如，代表访问 C:\example.txt 权限的许可对象，就是代表访问 C:\ 权限的许可对象的子集。如果 IsSubsetOf 方法返回 true，则当前许可对象不会比 target 具有更多的访问权限。

在 CodeAccessPermission 的派生类中，必须实现此方法。

PermitOnly 方法

原型：

[Visual Basic]

NotOverridable Public Sub PermitOnly()

[C#]

public void PermitOnly();

[C++]

public: __sealed void PermitOnly();

[JScript]

public function PermitOnly();

用途：调用 PermitOnly 方法，以确保只有由本许可对象指定的资源才能被访问，即使代码具有访问其他资源的权限。

说明：PermitOnly 实际是 Deny 方法的另一种形式，它们都将导致本来可以成功的堆栈检查失败。两者的不同在于：Deny 指定了会导致检查失败的许可；而 PermitOnly 则只指定了能成功的许可——其他许可将失败。