第一篇　概论

第一章 微软.NET 战略和 ASP.NET 简介

欢迎你阅读《ASP.NET 完全入门》，通过对本书的阅读，我们相信你能够对 ASP.NET 会有更深入的了解。

ASP.NET 又叫 ASP+，但并不仅仅是 ASP 的简单升级，而是 MicroSoft 推出的新一代 Active Server Pages 脚本语言。ASP.NET 是微软发展的新型体系结构.NET 的一部分，它的全新技术架构会让每一个人的网络生活都变得更简单。

首先需要特别指出的是，ASP.NET 不仅仅只是有了一个新界面并且修复了一些缺陷的 ASP3.0 的升级版本(即不同于 ASP2.0 升级到 ASP3.0 的转变)。更为重要的是，ASP.NET 吸收了 ASP 以前版本的最大优点并参照 Java、VB 语言的开发优势加入了许多新的特色，同时也修正了以前的 ASP 版本的运行错误。

要了解 ASP.NET 的真实面目，我们首先就得了解一下微软.NET 战略。

1.1.1 微软.NET 的历史

随着网络经济的到来，微软公司希望帮助用户，能够在任何时候、任何地方、利用任何工具都可以获得网络上的信息，并享受网络通信所带来的快乐。.NET 战略就是为着实现这样的目标而设立的。

微软公开宣布，今后将着重于网络服务和网络资源共享的开发工作，并称，将会为公众提供更加丰富、有用的网络资源与服务。

微软新一代平台的正式名称叫做“新一代 Windows 服务”(NGWS)，现在微软已经给这个平台注册了正式的商标——MicroSoft.Net。在.Net 环境中，微软不仅仅是平台和产品的开发者，并且还将作为架构服务提供商、应用程序提供商，开展全方位的 Internet 服务。在谈及这个平台中使用的新技术，微软透露，它将在.Net 环境中提供更多新产品和一揽子的全套服务。

MicroSoft .NET 平台的基本思想是：

侧重点从连接到互联网的单一网站或设备上，转移到计算机、设备和服务群组上，使其通力合作，提供更广泛更丰富的解决方案。用户将能够控制信息的传送方式、时间和内容。计算机、设备和服务将能够相辅相成，从而提供丰富的服务，而不是像孤岛那样，由用户提供唯一的集成。企业可以提供一种方式，允许用户将它们的产品和服务无缝地嵌入自己的电子构架中。这种思路将扩展二十世纪八十年代首先由 PC 赋予的个人权限。

MicroSoft .NET 将开创互联网的新局面，基于 HTML 的显示信息将通过可编程的基于 XML 的信息得到增强。XML 是经“万维网联盟”定义的受到广泛支持的行业标准，Web 浏览器标准也是由该组织创建的。微软公司为开发它投入了大量精力，但它并不是 MicroSoft 的专有技术。XML 提供了一种从数据的演示视图分离出实际数据的方式。这是新一代互联

网的关键，提供了开启信息的方式，以便对信息进行组织、编程和编辑；可以更有效地将数据分布到不同的数字设备；允许各站点进行合作，提供一组可以相互作用的“ Web 服务 ”。

1.1.2 微软.NET 的介绍

1.1.2.1 MicroSoft .NET 综述

MicroSoft .NET 平台包括用于创建和操作新一代服务的.NET 基础结构和工具；可以启用大量客户机的.NET User Experience；用于建立新一代高度分布式的数以百万计的.NET 积木式组件服务；以及用于启用新一代智能互联网设备的.NET 设备软件。

MicroSoft .NET 产品和服务—包括 Windows.NET，连同建立积木式服务的核心集成套件；MSNTM .NET；个人订购服务；Office.NET；Visual Studio .NET；以及用于.NET 的 bCentralTM。

.Net 环境中的突破性改进在于：

1． 使用统一的 Internet 标准（如 XML）将不同的系统对接；

2． 这是 Internet 上首个大规模的高度分布式应用服务架构；

3．使用了一个名为“ 联盟 ”的管理程序，这个程序能全面管理平台中运行的服务程序，并且为它们提供强大的安全保护后台；

.NET 平台包括如下组件：

1．用户数据访问技术。其中包括一个新的基于 XML 的、以浏览器为组件的混合信息架构，叫做“ 通用画板 ”；

2．基于 Windows DNA 2000 的构建和开发工具；

3．一系列模块化的服务，其中包括认证、信息传递、存储、搜索和软件送递功能；

4．一系列驱动客户设备的软件；

1.1.2.2 Microsoft.NET 平台带来的重要意义

我们来看一下 MicroSoft .NET 对开发人员、IT 专业人员、以及企业应用的巨大意义。

- **对于开发人员**

MicroSoft .NET 的策略是将互联网本身作为构建新一代操作系统的基础，对互联网和操作系统的设计思想进行合理延伸。这样，开发人员必将创建出摆脱设备硬件束缚的应用程序，以便轻松实现互联网连接。MicroSoft .NET 无疑是当今计算机技术通向计算时代的一个非常重要的里程碑。

.NET 的核心组件有：

· 一组用于创建互联网操作系统的构建块，其中包括 Passport.NET（用于用户认证）以及用于文件存储的服务、用户首选项管理、日历管理以及众多的其它任务

· 构建和管理新一代服务的基本结构和工具，包括 Visual Studio.NET、.NET 企业服务器、.NET 框架和 Windows.NET

· 能够启用新型智能互联网设备的.NET 设备软件

· .NET 用户体验

.NET 对最终用户来说非常重要，因为计算机的功能将会得到大幅度提升，同时计算机操作也会变得非常简单。特别地，用户将完全摆脱人为的硬件束缚：用户可以自由冲浪于互联网的多维时空，而不是束缚在便携式电脑的方寸空间——可通过任何桌面系统、任何便携式电脑、任何移动电话或 PDA 进行访问，并可对其进行跨应用程序的集成。

.NET 可使用户轻松进行互联网连接，并轻松完成那些在当今看来十分费时而且费力的事务，它们往往要求用户进行数据重输入并需运行几个小时才能完成。通过将多项安全数据流合并到单一的用户界面（或者甚至是可编程决策引擎），.NET 架构将用户从充斥于当今 Web 的数据竖井的束缚中解脱出来。用户可以自由访问、自由查看、自由使用他们的数据。

.NET 对开发人员来说也十分重要，因为它不但会改变开发人员的开发应用程序的方式，而且使得开发人员能创建出全新的各种应用程序。新型开发范例的核心是 Web 服务这个概念的引入。Web 服务是一种通过简单对象访问协议(SOAP)，在互联网上展露其功能性的、极为公开的服务。SOAP 是一种基于可扩展标记语言(XML)制定的协议。

在过去，开发人员通过集成本地系统服务来构建应用程序。在这种模型下，开发人员可以访问丰富的开发资源并能严格控制应用程序的行为。

如今，开发人员已在很大程度上挣脱了这种模型的束缚，致力于构建具有复杂结构的 n 层化系统，这种系统能将网络上众多的应用程序一并进行集成，大大提升了应用程序的价值。这样，开发人员便可把精力集中在充分挖掘软件独特的商业价值，而不是构建基本结构上。可喜的局面将应运而生：软件投放市场的时间大大缩短、开发人员的编程效率明显提高，最终把质量上乘的软件呈现给用户。

我们正在进入一个崭新的计算时代——一个由互联网（尤其是 Internet 核心技术 XML）实现的时代。利用 XML，能够创建出可供任何人从任何地方使用的、功能非常强大的应用程序。它极大地拓展了应用程序的功能，并实现了软件的动态提供。在这种情况下，软件已不完全指那些从光盘进行安装的程序，而是演变成了一种服务——类似于 ID 调用程序或按收看次数进行收费的电视——人们可通过通信媒体订购的服务。

n 层计算技术具有能够大幅度提高生产力、紧密耦合的特点，而 Web 概念具有面向消息、松散耦合的特点，我们将二者有机地糅合在一起，实现了上述构想。我们将这种计算风格称为 Web 服务，它的出现标志着人类已经迈入应用程序开发技术的新纪元。Web 服务是一种应用程序，它可以通过编程并使用标准的 Internet 协议，像超文本传输协议(HTTP)和 XML，将功能展示在互联网和企业内部网上。还可将 Web 服务视作 Web 上的组件编程。

从理论上讲，开发人员可通过调用 Web 应用编程接口(API)，将 Web 服务集成到应用程序中。其调用方法与调用本地服务类似，不同的是 Web API 调用可通过互联网发送给位于远程系统中的某一服务。例如，MicroSoft Passport(Passport)服务使得开发人员能够对应用程序进行认证。通过对 Passport 服务编程，开发人员可以充分利用 Passport 的基本结构，通过运行 Passport 来维护用户数据库，以确保其正常运行、定期备份等等。

.NET 正是根据这种 Web 服务原则而创建的，微软目前正着手提供这个基本结构，以便通过.NET 平台的每一部分来实现这种新型的 Web 服务。而 Visual Studio.NET、.NET 框架、Windows.NET 和.NET 企业服务器，正是为进行基于 Web 服务模型的应用程序开发而度身定做的新一代开发工具和基本结构。.NET 构建块服务、新增的.NET 设备支持以及即将到来的.NET 用户体验，将为人们彻底攻克这一难题划上一个圆满的句号，使人们能够充分利用 Web 服务模型，如愿以偿地开发出新一代应用程序。

- **.NET 对 IT 专业人员的重要意义**

目前，IT 专业人员能够利用与构建.NET 平台相同的技术。

.NET Enterprise Servers 和 Windows 2000 操作系统，为创建具有高度可管理性的、能迅速投入市场的应用程序提供了坚实基础。它们利用的是可扩展标记语言(XML)，因此随着 Web 体系结构的革新，在此平台上创建的程序依然很有价值。

.NET 平台的核心是，采用有效的、分门别类的方式来构建应用程序，达到其前所未有的规模。该平台上的 Web 服务模型指的是：企业应用程序的中心业务要素通常由本地管理，而支持它们的服务（如用户认证、文件存储、用户首选项管理、日历、邮件等等）却无须本地管理，可以被无缝订购。为了存储用户文件和邮件，IT 专业人员往往在服务器上安装新的独立磁盘冗余阵列（RAID 阵列），而有了.NET，他们在这一方面将会花费较少的精力，而更多地致力于怎样为公司增加效益。

该 Web 服务模型还将动态配置新软件的发布和更新。用户将以极其紧密的连接方式工作，因此更易于管理。而简化的管理又可使 IT 专业人员更能适应变幻莫测的业务需求。

开发应用程序的.NET Web 服务模型将为企业应用程序的创建开辟一条新路。通过企业内外多种服务的联合，很容易把企业内部数据和客户及合作伙伴的相关数据结合在一起，大大简化了应用程序的创建过程。这就为最终用户发掘了空前的功能涵盖性。例如，利用某公司的雇员福利程序，可以从其 HR 数据库订购信息，通过 Web 订购福利管理公司的服务、订购工资管理公司的服务。终端用户可以在简单、直观的界面下操作，而这个界面可以显示他们的累积休假时间、个人所得福利以及上次工资额。

- **.NET 对企业的重要意义**

MicroSoft .NET 平台将从根本上改善计算机和用户之间进行交互的方式，最大限度地发挥电子商务中计算技术的重要作用。首先，让我们来分析一下当前商务计算世界的现状：

人与计算机进行交互的手段极为有限——通常使用键盘和鼠标进行输入，使用监视器监控输出。

用户信息基本上是本地信息；如果从另一台机器进行登录，则无法获取用户的个人首选项设置、数据及应用程序。

用户必须亲自处理信息，而通过设置智能选项代表用户自动进行操作，则无异于是纸上谈兵。

同一用户存放于不同应用程序和站点的数据，很难（或根本不可能）进行自动合并和关联，用户无法统一进行查看。

想在家里或在路上工作的用户，不能方便地访问办公室电脑中的应用程序和数据。这无疑成为一道阻止人们获得更高工作效率的鸿沟。

不能使用其它设备访问专为特定设备设计的数据（这些设备包括 PC、寻呼机、移动电话以及 PDA 等）；最多可以定期进行同步。

.NET 将保证完全消除当今计算技术中的所有缺陷。.NET 定能实现确保用户从任何地点、任何设备都可访问其个人数据和应用程序的宏伟蓝图。除此之外，.NET 技术还可实现多个应用程序在逻辑上的松散耦合链接和紧密耦合链接。

用户可以通过手写、语音和图象技术与其个人数据进行交互。这些数据将安全地存放在互联网上，用户通过办公室（或家庭）PC，还可以通过移动电话或寻呼机、PDA、甚至是新发明的寻呼机——移动电话——PDA——PC 联合设备访问这些数据。应用程序可进行灵活的功能调整，以适应用户所用设备的功能状况。应用程序可根据用户预定义的选项集和指令集，完全代替用户自动执行相应的操作。

上述功能将协同作用，以便大幅度地提高用户使用计算技术的生产效率。根据设计，.NET 使得用户无需在如何与计算机进行交互上劳神，从而全身心地投入到使计算机自动执行任务、实现最终目标的工作中。通过使用 XML 行业标准，可将用户数据进行跨站点和应用程序的链接，从而轻松实现当前很难实现的操作。比如：对用户在数家不同银行、信用卡公司以及计费代理商那里的数据进行集中处理；这样，用户便可依据处理后的数据支付帐单，将费用明细报告归档。

.NET 把雇员、客户和商务应用程序整和成一个协调的、能进行智能交互的整体，而各公司无疑将是这场效率和生产力革命的最大受益者。简言之，.NET 承诺为人类创造一个消除任何沟鸿的商务世界。

1.1.2.3 MicroSoft .NET 的基本模块

◆ **网络服务一览**

通常说来，一个网络服务只是一个作为服务——通过 Internet 标准此服务能与其它网络服务集成在一起——发行的简单的应用程序。换句话说，它是可通过 URL 定位的自动将信息返回到需要它的客户端那里的一种资源。网络服务一个重要的特点是客户不需要知道一种服务是怎样实现的。在本节中，我将向你解释网络及网络服务如何把基于组件技术的最好的方面结合在一起的，并且介绍与网络服务通信所需的基本框架。

同组件一样，网络服务提供“ 黑匣子 ”函数，它可以被再次作用而不用关心此服务是怎样实现的。网络服务提供被称为契约的精确定义的接口，此接口描绘了所提供的服务。开发人员可以将远程服务、本地服务和定置代码组合在一起而集成应用程序。例如，某公司可以使用如下服务组建一在线商店：微软护照（原文：Passport）服务以验证用户身份，第三方个人化服务以使网页匹配每一个用户的参数，信用卡处理服务，销售税服务，对每个运输公司的包裹跟踪服务，链接公司内部库存管理程序的内部目录服务，以及少量定置代码以使他们的商店能脱颖而出。

然而，网络服务与现在的组件技术不同，它不使用需要在服务器和客户机有明确的、同类型基本构架的具体的对象模型协议，例如 DCOM、 RMI 或 IIOP 。尽管与具体组件技术紧密结合的实现在一个受控的环境中能很好地被接受，但它们在网络环境中变得不切实际。因为一个集成商业程序的参与者会发生变化，随着时间的推移，技术也在变化，所以在所有参与者间确保一个单一的、统一的体系架构就变得十分困难。网络服务采取了另外一种途径，它使用普便存在的网络协议和数据格式，如 HTTP 和 XML，进行通信。支持这些网络标准的任何系统都支持网络服务。

而且，网络服务契约描述的是以术语报文形式提供的服务，这些服务是由网络服务生成和接受的，而不是描述服务是如何实现的。通过把重点放在报文上，网络服务模板就完全对语言、平台和对象模板一无所知。 用任何一套编程语言、对象模型和平台的

完全特性集，都可实现网络服务。网络 服务可在任何平台被用任何语言所实现的应用程序使用。只要用于解释服务容量、报文序列和所期望协议的契约得到认同，那么所实现的网络服务及网络服务用户就可相互不同，而不会影响会话另一端的应用程序。

网络服务模板对最小体系架构的要求很低，以确保网络服务在使用任何技术和编程语言的平台上实现和访问。对网络服务互用性的解决可只依靠网络标准。然而，为了使应用程序更容易使用网络服务，简单地同意通过标准网络协议就可以访问网络服务是不够的。当网络服务和网络服使用者依靠标准的方式表示数据和命令、表示网络服务契约 、算出网络服务所提供的容量时，网络服务才容易使用。

XML 是定义一个标准的、可扩展的用于提供命令和典型数据的语言明显的一种选择。虽然为表示命令和典型数据可以定义使用其它技巧（比如编码为一种查询字符串）的规则，但 XML 被专门设计为描述数据的标准元语言。简单对象存取协议（SOAP）是以一种可扩展的方式使用 XML 表示数据和命令的工业标准。网络服务可选择用 SOAP 决定报文的格式。

XML 是网络服务契约的一种使能技术。服务契约语言 (SCL)是记录网络服务契约的 XML 语法。由于 SCL 是基于 XML 的，所以对开发者和开发工具来说，容易生成、解释契约。关于 SCL 细则的草案很快会出台（注意 :现在的 SOAP Toolkit for Visual Studio 6.0 支持称为 SDL 的 SCL 的早期版本）。

Disco 规范为服务提供者发布网络 服务契约和相应的机制描述了一个标准方式，这将使开发者或开发工具可找到契约文献。当你读到这里时， Disco 规范的草案应出台了。

象 SOAP， SCL 和 Disco 这样的标准有助于开发者，因为它们不需要明白和实现所使用的每一个网络服务的访问方式。支持这些标准的更好的、已充分测试的、高性能的体系架构将由开发平台提供，这会大大简化整个开发过程。

- **MicroSoft .NET Framework**

MicroSoft .NET 框架的目的是使你更容易建立网络应用程序和网络服务。图 2 显示了 MicroSoft .NET 框架的体系。建立在操作系统最上层的服务，是管理运行时代码需求的 common language runtime，这些代码可以用任何现代编程语言所写。Runtime 提供了许多服务，这些服务有助于简化代码开发和应用程序的开发同时也将提高应用程序的可靠性。.NET Framework 包括一套可被开发者用于任何编程语言的类库。在此之上是许多应用程序模板，这些模板特定地为开发网络站点和网络服务提供高级组件和服务。

- **Common Language Runtime**

运行语言(runtime)调入并运行用任何运行感知编程语言所写的代码。以运行为目标的代码被称为受控（managed ）代码，受控代码代码只是意味着在内部可执行代码与运行自身间存在已定义好的合作契约。对于象生成对象、调用方法等这样的任务，被委托给了运行语言，这使得在运行语言能为可执行代码增加额外的服务。

运行语言以交叉语言集成、自描述组件、简单配制和版本化及集成安全服务为特点。

运行语言使用一种新的能表达大部分现代编程语言语义的通用类型系统，通用类型系统定义了一套标准类型及生成新标准的规则。运行语言知道怎样生成、执行这些类型。编译器和解释器使用运行语言服务定义类型、管理对象、进行方法调用，而不是使用工

具或特定于语言的方法。

类型系统的主要设计目的是使多种语言能深度集成。用一种语言所写的代码能继承用另一种语言所写的类的实现，用一种语言所写的代码抛出的异常能被用另一种语言写的代码捕获，象调试和剖析之类的操作会在完全封闭下工作，而不用考虑代码所用的语言。这就意味着编写可重用类库的开发者，不再需要为每一种编程语言或编译器生成一个版本，并且使用类库的开发者不再受到为他们使用的编程语言开发的库的限制。

自描述组件——现在 MicroSoft .NET 框架上已成为可能——简化了开发和配制，并提高了系统的可靠性。许多由运行语言提供的服务是由元数据及用于补充可执行代码的信息所驱动。因为所有的信息都储存在一起，只有可执行的（代码）才被称为自描述组件。

自描述组件的一个主要优点是，使用它们并不需要其它文件。类的定义不需要单独的头文件；通过检查元数据对类的定义可以从组件自身获得。跨语言或过程边界访问组件并不需要各自的 IDL 文件、类型文件或 proxy/stubs；所必需的信息已存在于元数据之中。为识别开发者请示的服务属性，并不需要展开各自的配制信息。 最主要的是，由于元数据是在编译过程中由源代码生成，并与可执行代码储存在一起，它将永远和可执行部分同步。

除了改善对单个组件的配制，Microsft .NET 框架定义了一个应用程序配制模板，以解决定置应用程序安装和 DLL 版本化（通常被称为“ DLL Hell”）这一复杂过程的问题，运行语言提供了支持这个模板的服务。

Microsft .NET 框架 引入了组合体的概念。一个组合体是一组资源和类型，并包括有关这些资源和类型的元数据，也就是被作为一个单元配制的。元数据被称为组合体的名单，它包含象类型和资源表之类能被组合体外看得见的信息，这个名单也包括有关从属关系之类的信息，例如组合体建立时的版本号。开发人员可以指定版本策略，以指示运行语言是否装入系统上已安装的依赖于组合体的最新版本，装入一指定版本，或在编译时使用的版本。

某软件组件的多个拷贝总可以存在于同样的操作系统上，然而，通常说来，只有其中的一个拷贝能被操作系统注册、调入内存、执行。对系统来说，定位和调入内存的策略是全局性。.NET Framework Common Language Runtime 增加了所必须的体系架构以支持管理组件定位和调入的每个应用程序策略，这通常被称为并行配制。

组合体可以被一个应用程序私有，或被多个应用程序共享。一个组合体的多个版本可以同时配制在同一台机器上。应用程序配制信息定义了到何处去查找组合体，这样 runtime 就能为同时运行的两个不同的应用程序装入同一组合体的不同版本。这就消除了由组件版本的不兼容性引起的问题，提高了系统整体的稳定性。如果必要，如果必要，管理员可以为配制时刻的组合体增加配制信息，例如一个不同的版本策略，但是编译时提供的原始信息永远不会丢失。

因为组合体是自描述的，所以并不需要在系统上进行显式注册。应用程序的配制简单到只需将文件拷贝到目录中既可（如果为了使应用程序能够运行，必须安装未经组织过的组件的话，情况会稍微复杂一点）。配制信息保存在可被任何文本编辑器编辑的 XML 文件中。

最后，运行语言也提供完整的、普遍深入的安全服务，以确保未经授权的用户不能访问机器上的资源，并且代码不会执行未经允许的动作。这就提高了系统整体的安全性可靠性。 由于运行语言用于装入代码、生成对象、执行方法调用，所以当受控代码装入内存、执行时，运行语言能进行安全检查，强化安全策略。

Microsft .NET 框架不仅规定代码访问安全，还规定基于角色的安全。通过代码访

问安全机制，开发人员能为应用程序指定完成工作所必需的权限。例如，代码或许需要写文件或访问环境变量的权力。这类信息和有关代码标志的信息一起存储在配制级上的。当代码装入内存及执行方法调用时，运行语言验证是否能给予代码所要求的权限。如果不能，将记录一条安全冲突信息。给予权限的策略，这被称为信任策略，是由系统管理员建立的，并且是建立在关于代码的证据基础之上，比如：代码是谁发布的，是从什么地方获得的，以及在组合体中找到的代码标志和它要求的权限。开发人员可以指定他们显然不需要的权限，以防止其它人恶意使用他们的代码。如果所需要的权限依赖直到运行时刻才会知道的信息，那么就可写入纲邻性的安全检查。

除了代码访问安全，运行语言还支持基于角色的安全。基于角色的安全建立同代码访问安全一样的权限模板，只是这些权限是建立在用户的身份之上，而不是建立在代码的标志之上。角色表明了用户所属的类，并且可以在开发和配制阶段定义。给予权限的策略被分配到每个预定义的角色。在运行时刻，用户的身份被确定，代码将代表这个身份运行。运行语言决定用户是哪个角色的成员，然后给予基于这个角色的权限。

在查看 Microsft .NET 框架的可编程模板前，先看一下它所提供的服务。

- **服务框架**

在 Common Language Runtime 之上是服务框架，此框架提供能被任何现代编程语言调用的类。所有的类都遵循一套命名和设计方针，以大大减小开发人员的学习上的弯路。

框架包括一套开发人员希望在标准语言库中存在的基类库，例如：集合、输入/输出，字符串及数据类。另外，基类库提供访问操作系统服务如图画、网络、线程、全球化和加密的类。服务框架也包括数据访问类库，及开发工具，如调试和剖析服务，能够使用的类。本文章没有详细讨论所有的类，我将重点放在数据访问类上，因为大多数网络服务需要对数据的访问。当然，你可以在 MicroSoft .NET Framework SDK 中找到关于服务框架类库的附加信息。

- **数据访问服务**

几乎所有的网络服务都需要查询和更新永久性数据，不论是以简单文件，还是以相关数据库，或是以其它的存储类型存在。为了提供对数据的访问，服务框架包括 ActiveX Data Objects+ (ADO.NET)类库。如同名子所暗示地那样，ADO.NET 由 ADO 发展而来。ADO+被设计为基于网络的可扩展的应用程序和服务提供数据访问服务。ADO.NET 为连接的指针风格的数据访问，同时也为更适合于把数据返回到客户端应用程序的无连接的数据模板提供高性能的 APIs 流，就象在以后介绍的那样。

就象其余几个部分一样，ADO.NET 定义了那些链接数据仓库、 对数据仓库发送命令及从中获取结果的类。这些类由受控数据提供者（managed data provider）实现。ADO+中链接和命令对象看上去和 ADO 中的是一样的，并且一个名为 DataReader 的新类提供了通过高性能 API 流获取结果的能力。DataReader 在功能上同前向、只读的 ADO 记录集（Recordset）是等同的，但是 DataReader 被设计用来最小化内存中生成的对象的数量，以提高性能，避免垃圾积累。在.NET Framework 中包含了针对 MicroSoft SQL Server™的受控数据提供者以及可通过 OLE DB 访问的任何数据仓库。

ADO.NET 的一个主要创新是引入了数据集（Dataset）。一个数据集是内存中提供数据关系图的高速缓冲区。数据集对数据源一无所知，它们可以由程序或通过从数据仓库中调入数据而被生成、填充。不论数据从何处获取，数据集都是通过使用同样的程序模板而被操作的，并且它使用相同的潜在的数据缓冲区。使用.NET 平台的开发人员能

够用数据集代替传统 ADO 中无连接的记录集。

受控数据提供者为数据仓库和数据集公开一名为 DataSetCommand 的接口对象。DataSetCommand 使用 ADO.NET 链接和命令以从数据仓库中填充数据集，并把在数据集中发生的变化解析到数据仓库中。

就象 DataReaders 显示了对于相关数据的有效的流访问一样，XmlReaders 显示了对 XML 数据的流访问。开发人员使用 DataNavigator 可以滚动和编辑内存中的 XML 文档。DataNavigator 在功能上和 W3C Document Object Model (DOM)是一样的，但它更有效，并提供了能很好映射关系数据表的对象模板。DataNavigator 支持 Xpath 语法以对数据流进行导航。ADO.NET 为那些希望继续使用 DOM 作为 XML 对象模板而不是使用更有效的 DataNavigator 模板的开发人员提供了一个 XMLDocument 类。

由于所有的数据都可被看作 XML，所以开发人员可以为任何数据使用转换和确认服务。ADO.NET 定义了一个消费 DataNavigator、生成一个新的 XmlReader 的通用转换体系。.NET Framework 提供了一个支持 W3C XSL Transformations (XSLT)细则的特殊转换组件。ADO.NET 同时提供了一使用 XML 简图确认 XmlReader 的确认引擎。ADO.NET 支持通过 DTDs， XSD 或 XDR 定义的简图。

- **表单应用模板**

从概念上讲，在服务框架的最上面是两个应用程序模板：Windows 应用程序模板和网络应用程序模板。尽管我把重点放在把微软.NET 框架用作开发网络服务和网络应用程序的一种途径上，但框架也可用于开发较传统的基于 Windows 的应用程序（当然，这些应用程序也能使用网络服务）。

编写 Windows 客户应用程序的开发人员可使用 Win 表单应用程序模板以利用 Windows 丰富的用户接口特点，包括现在的 ActiveX 控件和 Windows 2000 的新特点，如透明的、分层的、浮动窗口。可以选择传统的 Windows 或网络外观。得知它和现在的基于 Windows 表单包的相似性以后，开发人员会发现 Win 表单可编程模板和对设计阶段的支持非常直观。

Win 表单利用了 Microsft .NET 框架 runtime 以减少基于 Windows 的客户应用程序的开销。只要应用程序和组件是用 Win 所写或被 Win 表单应用程序使用，那么它们就能被框架安全模板在客户机上安全地执行。如果以这种方式使用或执行，那么某人从 Internet 下载下来的生猛游戏就不会对配制信息和数据产生破坏，否则会自动地给用户地址薄里的每一个人发送电子邮件。

Microsft .NET 框架 装配模板简化了应用程序的配制和版本化 。应用程序可被配制为使用它们在编译和测试所用的共享组件，而不是使用恰好在客户机器上安装的随便什么版本的组件，这就提高了应用程序的可靠性，减少了应用程序所支持调用的主要因素：用户接口控件和其它共享组件版本的不兼容性。

- **网络应用程序模板**

建立在 Microsft .NET 框架 上网络应用程序共享一个通用应用程序模板。在这个模型中，网络应用程序是一套起源于基 URL 的 URLs。因此它包含用于生成在浏览器中观看的网页的网络应用程序和网络服务。在本节中，我将详细介绍称为 Active Server Pages+ (ASP.NET)的网络应用程序可编程模板

如同你从名字猜到的那样，ASP.NET 是由活动服务器页面发展而来。ASP.NET 利

用 common language runtime 和服务框架网络应用程序提供了一个可靠的、自动化的、可扩展的主机环境。ASP.NET 也受益于 common language runtime 集成模板，简化了应用程序的配制。另外，它提供简化应用程序开发的服务（如状态管理服务）以及高水平的编程模板（如 ASP.NET Web Forms 和 ASP.NET Web Services）。

ASP.NET 的核心是 HTTP 运行语言，一个高性能的用于处理基于低级结构的 HTTP 请求的运行语言，而基于的结构与 MicroSoft Internet Information Services (IIS)所提供的 ISAPI 结构相似。如同你在图 5 所看到的，HTTP 运行语言是在象服务器上的 IIS 或客户机上的 IE 之类的 unmanaged 主机过程中运行的受控代码。HTTP runtime 负责处理引入的所有 HTTP 请求，并对每个请求应用程序的 URL 进行解析，然后把请求分配到应用程序以进行进一步的处理。HTTP 运行语言是多线程的，并异步处理请求，因此劣质的应用程序代码阻碍不了它对新请求的处理。而且 HTTP 运行语言假定失败必会发生，因此它被控制为尽最大力量自动地从访问冲突、内存泄漏、死锁等事故中恢复过来。除非是硬件故障，运行语言的目标是 100%的可靠性。

ASP.NET 使用基于构件的 Microsft .NET 框架配制模板，因此它获得了如 XCOPY 配制、构件并行配制、基于 XML 配制等优点。ASP.NET 另一个主要优点是，它支持应用程序的实时更新。管理员不必关掉网络服务器或者甚至不用停止应用程序的运行就可以更新应用文件。应用程序文件永远不会被加锁，因此甚至在程序运行时文件就可以被覆盖。当文件更新后，系统会温和地转换到新的版本。系统检测文件变化，并用新的应用程序代码建立一个新的应用程序实例，然后将引入的请求路由到应用程序。当所有被现存的应用程序实例处理的未完成的请求处理完后，该实例就被销毁了。

在应用程序中，HTTP 请求是通过 HTTP 模块的一个管道路由的，最终到达请求处理程序。HTTP 模块和请求处理程序是一些实现特殊接口的受控类，而这些接口是由 ASP.NET 定义的。这种管道结构使得为应用程序增加服务非常方便：只需补充一个 HTTP 模块。例如，安全，状态管理及跟踪都被实现为 HTTP 模块。高级可编程模块，如网络服务和网络表单，通常被实现为请求处理程序。一个应用程序能链接与多个请求处理程序——每个处理程序一个 URL，但是所有的 HTTP 请求都通过同样的管道路由。

网络基本上是一个无状态模型，并且在 HTTP 请求间没有联系，这使得编写网络应用程序很困难，因为应用程序通常需要维护跨多个请求的状态。ASP.NET 增强了由 ASP 引入的状态管理服务，以便为网络应用程序提供三种类型的状态：应用程序、会话、用户。就象在 ASP 中一样，应用程序状态特定于一个应用程序实例，并且不会持久。会话状态是特定于一个用户与应用程序间的会话的。与 ASP 会话状态不同，ASP.NET 会话状态储存在一个独立的过程中，并且可把它配制成可以储存到一个独立的机器上。这使得会话状态当应用程序在网络群（Web farm）扩展时非常有用。用户状态类似于会话状态，但通常它不会超时，并且是永久性的。因此，用户状态对储存用户参数和其它个性化的信息是有用的。所有状态管理服务都被实现为 HTTP 模块，因此它们容易增加到应用程序管道中，或从中删除。如果除了由 ASP.NET 提供的服务外，还需要额外的状态管理服务，那么可由第三方的模块提供。

ASP.NET 同样提供高速缓冲服务，以改善性能。输出缓冲可完全节省网页翻译，段缓冲储存部分的网页。由于提供了相应的类，所以只要需要，应用程序、HTTP 模块以及请求处理程序可以在高速缓存中储存任意数量的对象。

下面快速浏览一下建立在 ASP.NET 可编程模块之上的两个高级可编程模块：ASP.NET 网络 表单和 ASP.NET 网络 服务。

- **ASP.NET 网络表单**

网络表单把基于 Visual Basic®的表单的高生产性的优点带到了网络应用程序的开发中来。网络表单支持传统的将 HTML 内容与角本代码混合的 ASP 语法，但是它提出了一种将应用程序代码和用户接口内容分离的更加结构化的方法。引入的网络表单控件用于为封装通用用户接口元素提供了一种机制。这些新的特点使得开发工具在支持 VB 小应用程序的同时，也支持设计时模块，使得 WUSIWYG 工具支持网页布局。

网络表单控件负责生成用户接口，典型情况是在 HTML 表单中。ASP.NET 是提供了一套映射传统的 HTML 用户接口小部件（包括列表框，文本框和按钮）的网络表单控件和一套附加的更加复杂的网络控件（如日历和广告转板）。这些控件的一个重要特点是，它们可以被编写以适应客户端的能力；同一网页把大范围的客户端平台和表单因素作为目标。换句话说，网络表单控件能“嗅”到正在查找表单的客户，然后返回合适的用户经验——可能是适合低级浏览器的 HTML3.2 或是适于 IE5.0 的动态 HTML。

考虑到网络是一种无状态的联接模型，网络应用程序开发人员所面临的一个很复杂的问题是，他们要对用户与基于网络的接口的交互作用作出反应。网络利用 ASP.NET 的体系架构提供了一套丰富的服务，以帮助开发人员建立交互式网页。这些服务的净作用是使基于组件的、事件驱动的可编程模块，对开发人员来说，非常象客户端的表单程序设计。用户与网页交互作用的状态管理的复杂性被 ASP.NET 网络 表单和网络表单控件隐藏起来了。对开发人员来说，提供的丰富数据绑定服务使得显示通过数据访问服务得到的数据变得非常容易。

代码与内容的分离使 ASP.NET 网页能动态地编译到受控类中，用以提高性能。每个引入的 HTTP 请求都被传递到一个新的网页实例，因此开发人员不需要关心代码中的线程安全性。

- **ASP.NET 网络 服务**

ASP.NET 网络 服务体系架构为用 ASP.NET 建立网络 服务提供了一高级可编程模板。虽然建立网络服务并不需要使用网络 服务平台，但是它提供许多的优点将简化开发过程，并且它使用的编程模型对用 ASP 或 VB 工作的开发人员来说是很熟悉的。使用这个可编程模型，开发人员不需要理解 HTTP、SOAP 或其它任何网络服务规范。

开发人员用 ASP.NET 生成一个扩展名为.asmx 的文件，并把此文件配制为网络应用程序的一部分，就建立起了一个网络 服务。ASMX 文件或者包含对在其它地方定义的受控类的引用，或者包含这个类的定义。这个类是由 ASP.NET 提供的 WebService 类所派生。公有的类方法在标记上 WebMethod 属性后，就会成为网络服务方法，把 HTTP 请求发送到 ASMX 文件中的 URL 后，这些方法就会被调用。你不必手工为你的网络服务建立一个契约。当被调用者请求时，ASP.NET 检查类的元数据，以自动生成 SCL 文件。

客户可通过 SOAP，HTTP GET 和 HTTP POST 提交请求。对方法和参数进行编码的约定是：对 HTTP GET，将被编码为查询字符串；对 HTTP POST，将被编码为表单数据。HTTP GET 和 HTTP POST 的机制不如 SOAP 有力，但是它们使得客户在访问网络服务时不必支持 SOAP。

ASP.NET 网络服务模型假定了一个无状态服务结构。无状态结构通常比有状态结构更具可扩展性。每次收到一个服务请求后，就生成一个新对象，请求被转化为一个方法调用，当方法调用返回时对象被销毁。如果这些服务需要跨请求维护状态，那么它们将使用 ASP.NET 状态管理服务。基于 ASP.NET 的网络服务在网络应用程序模型中运行，

因此它们得到了该模型的所有安全、配制和其它优点。

ASP.NET 网络服务还提供了一个为在 SCL 文件中描述的网络服务生成分类的受控代理工具。代理生成器把 SCL 文件中描述的消息映射成受控类中的方法。代理对应用程序代码隐藏了所有的网络和引导设备，因此使用网络服务看起来就象使用其它受控代码一样。代理将优先使用 SOAP 链接网络服务，但是它同样支持 HTTP GET 和 HTTP POST 机制。因此 HTTP GET 和 HTTP POST 同样也能被使用。

网络服务为在 Internet 上绑定应用程序提供了一个利用现存体系架构和应用程序的简单的、灵活的、基于许多标准的模型。网络应用程序很容易与当地开发的服务或已存在的服务集成在一起，而不用考虑开发平台、开发语言或使用的对象模型，以用于实现任何组成的服务或应用程序。

Microsft .NET 框架在现有开发人员技巧之上，提供了一个应用程序模板和关键技术，用于简化安全、可靠、可扩展、高可用性的网络服务的建立、部署和不断的发展。

通过上面的介绍，我们能够感觉到 MicroSoft .NET 对于我们今后的程序设计将产生巨大的影响。

1.1.3 ASP.NET 历史

我们在讲述 ASP.NET 历史之前，让我们来回顾一下 ASP。

ASP 的第一个版本是 0.9 测试版。它给 WEB 开发带来一阵暴风，它能够将代码直接嵌入 HTML，使得设计 WEB 页面变得更简单，更强大，并且通过内置的组件能够实现强大功能，最明显的就是 ActiveX Data Objects （ADO），它使得建立一个动态页面如小孩子玩游戏一样简单。

最终出场的是 Active Server Page 1.0，它做为 IIS 的附属产品免费发送。并且不久就在 Windows 平台上广泛使用。ASP 与 ADO 的结合使用开发者很容易地在一个数据库中建立和打开一个记录集。这不无疑是它如此快就被大众接受的因素，因为你现在能使用这些脚本建立和打开一个记录集，处理和输出任何数据，以任何顺序，几乎只要你能想到的，它就能完成。

1998 年，微软公司又发布了 ASP 2.0 。ASP 1.0 和 ASP 2.0 主要区别是外部的组件需要实例化。有了 ASP 2.0 和 IIS 4.0，我们就有可能建立 ASP 应用了，而且每个组件就有了自已单独的内存空间。内置的 MicroSoft Transaction Server(MTS)也使用制做组件便得简单。

微软公司接着开发了 Windows 2000 操作系统。这个 Windows 版本给我们带上了 IIS 5.0 以及 ASP 3.0。此次并不是简单对 ASP 进行补充，核心的不同实际上是把很多的事情交给了 COM 来做。在 windows 2000 中，微软结合了 MTS 与 COM 核心环境做出了 COM+，这就让主机有了一种新的方法来使用组件，同样给主机带来了更多的稳定性，成了一个可以升级的效率高的工作平台。IIS 5.0 在表面上似乎没有改什么，但是在接口上动的手术比较大。在内部，它使用 COM+ 组件服务来对组件提供一个更好的执行的环境。

有了这些，微软公司推出了 ASP.NET，ASP.NET 又叫 ASP.NET，他不是 ASP 的简单升级，而是 MicroSoft 推出的新一代 Active Server Pages。ASP.NET 是微软发展的新的体系结.NET 的一部分，其中全新的技术架构会让每个人的编程生活变得更的简单

1.1.4 小结

在本章中，我们介绍了微软.NET 的历史，以及对.NET 的构成、性能进行了一个详细的介绍，同时，我们还详细介绍了 ASP.NET 的历史。在下面的章节中，我们将按实例一步一步的讲解 ASP.NET。

第二章 .NET 的安装与运行环境

1.2.1 运行环境配置

- **ASP.NET 的调试环境**

操作系统：

Windows 2000 Professional，Windows 2000 Server ，Windows 2000 Advanced Server

浏览器：

IE 5.5

NGWS

- **支持哪几种语言**

ASP.NET 目前能支持 3 种与语言， C# (读作 "C Sharp")， Visual Basic，and Jscrip。.

- **使你的机器持 ASP.NET，必须满足以下配置：**

硬件要求：

1、 CPU: Intel Pentium II-class 300 MHz (最好 Intel Pentium III-class 600 MHz)
2、 内存: 96 MB (最好 128 MB)
3、 磁盘空间: 250 MB(完全安装) 155 MB(快速安装)
4、 显示: 800x600， 256 colors
5、 CD-ROM: required

软件要求：

1、 MicroSoft Windows 2000 + SP1
2、 MicroSoft Internet Explorer 5.5
3、 IIS5.0
4 、 其它: MDAC 2.6 Beta 2

- **.NET 是运行库，还是开发平台？**

微软的宏伟目标是让 MicroSoft.NET 彻底改变软件的开发方式、发行方式、使用方式等等，并且不止是针对微软一家，而是面向所有公司！2000 年 7 月份在 PDC 展会上分发的是

“ .NET 架构 ”包，“ .NET 架构 ”是 MicroSoft.NET 计划中首先问世的一部分，它包括了两方面的组件：“ .NET 通用运行库 ”和“ .NET 类库 ”。最近传来好消息说这两个组件已经被打包到“ .NET 架构 SDK”中，放在微软的站上免费供大家下载，有兴趣的朋友一定要去试试看哦！另外，这个 SDK 中还包括 C#、C++、JavaScript 和 VB 的命令行编译器，使用这些编译器就可以开发应用程序和组件了，从这个角度来看，.NET 架构首先是一个开发平台，因为它提供了运行库和类库，并且，下一个即将面市的就是 Visual Studio.NET，其中包括了更加全面的 SDK 和图形化的开发界面、向导、工具等等，更象一个开发平台了。但是.NET 的运行库其实已经融合到操作系统中，所以说它为运行库也是可以的。

- **什么是 NGWS？**

ASP.NET 实际上是一个崭新的运行结构的一部分，这个结构提供对所有 windows 应用程序的支持。这个结构是 MicroSoft's Next Generation Web Services （ NGWS）关建部分。当你安装了这个结构，你就获得了 ASP.NET。这个结构同样支持所有其它服务器程序技术。

NGWS 结构通过对可升级分布式应用添加 [新的和增强的服务] 来扩展 COM 的结构，此种结构常用做编写可重复调用的可共同使用的软件组件，这些新的和增强的服务有：

一套统一的丰富的程序库

一个支持多语言的运行引擎

简单地应用建立，调试，以及维护

对分布式应用加强了可升级性

保护现已存在的软件和投资

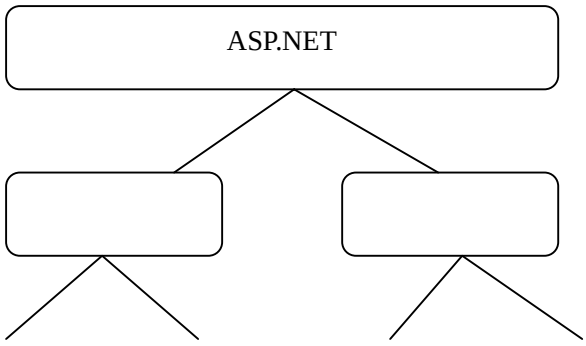
- **在 ASP.NET 引入了 namespace 的概念，那么 namespace 是什么？**

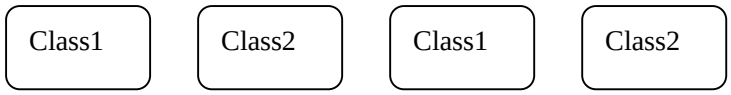
对象一直是 Windows 开发环境中，程序开发的中心。不论在 VB、VBA、VC++、VBScript 等，都是如此，不同的开发环境有不同的对象，这些对象均是各个语法所提供的的“ 资源 ”，程序开发人员可以利用这些资源，来编写所需的系统，就象我们在盖房屋的一样，建筑师使用使用同样的素材，然而盖好的房子可能不尽相同。

在过去的 SP 中，仅有 Server、Request、Response…等七个对象。而在 ASP.NET 的对象库中却分得很细。

例如在 ASP.NET 网页中要通过 SQL 语句获得数据库中的数据，必须使用“ System.Data.SQL ”，这是 NameSpace 名称。 在 System.Data.SQL 下，又有很多类（Class）。每个 Class 可视为一个对象，因为 Class 下有属性、方法和事件等

所以，最上层的 NameSpace 是看作是同类型对象的集合，一个 NameSpace 之下可拥有多个 Class。他们之间的关系如图：





通过此图，我们了解了 NameSpace 及 Class 的概念，二者分别是表示对象集合和对象。

- **如何应用名字空间（NameSpace）？**

```
<%@ Import Namespace="System.Globalization"%>

<%@ Import Namespace="DataEmployee" %>

<%@ Import Namespace="System.Data" %>

<%@ Import Namespace="System.Data.ADO" %>
```

以上表示在 ASP.NET 网页中使用了四个 NameSpace，接下来我们要申明变量，但此变量必须是已引用的四个 NameSpace 所属的 Class，如：

```
Dim MyConnection As ADOConnection

Dim MyCommand As ADODataSetCommand
```

说明：ADOConnection 及 ADODataSetCommand 都是 System.Data.ADO 之下的 Class。

- **ASP.NET 中的文件类型？**

ASP 的文件类型只有一种，其扩展名是.asp 文件。那么在 ASP.NET，就有很多的文件名：

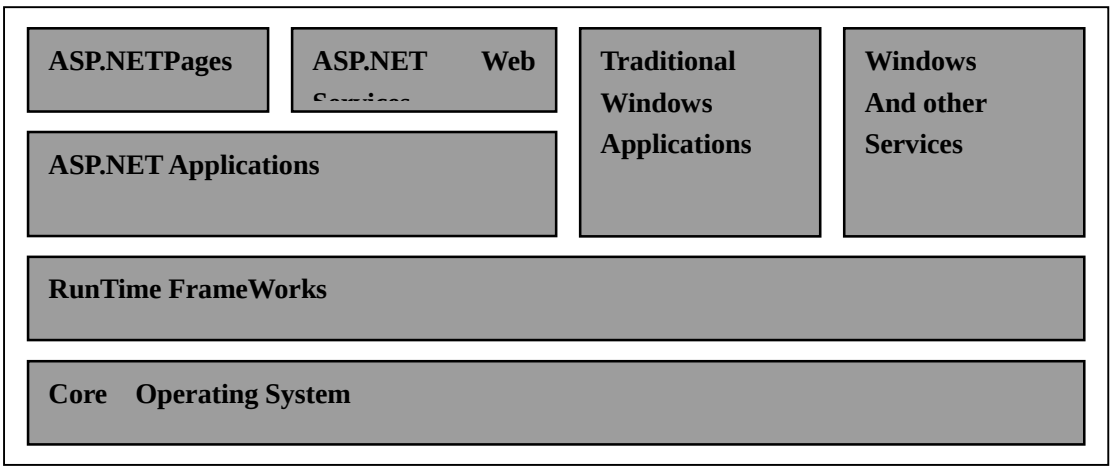
文件扩展名	用途及说明
Global.asax	ASP.NET 系统环境设置文件，相当与 ASP 中的 Global.asa。
.aspx	內含 ASP 程序代码的文件，如同過去的.asp，浏览器可执行此类文件，向服务器 提出浏览請求
.asmx	制作 Web Service 的原始文件
.sdl	制作 Web Service 的 XML 格式的文件
Vb 或.cs	在非 ASP.NET 环境下，执行 Web Service 的文件
.aspc	可重覆使用在多个.aspx 的文件，此文件內可含有控件
.ascx	內含 User Control 的文件，可內含在多個.aspx 文件中

- **什么是 NGWS Framework?**

ASP 的综合性能明显的要好于以前的版本。到目前为至，ASP 是通过一个名叫 asp.dll 的 ISAPI DLL 来执行的，另外还加上一些系统文件和 ASP 用户组件。

这个新的 NGWS 结构反映了行业信息技术观点对于建立，调试以及维护各种 WEB 服务的需要的转变，这些服务包括简单的客户应用到复杂的分布式结构。上面所有的概念和策略只是 Windows Distributed Internet Applications (DIA)部分结构.

在这里我们最重要需要认识的问题是这里所说的结构（framework)不是我们所说 ASP.NET。它只是做为 windows 系统中所有应用的基础。下面的图表给我们演示了 framework 是如何支持 ASP.NET 应用的。



- **ASP.NET 对于 asp 来说有什么突破呢？**

  - **运行机制不同**

asp 属于一种解释型的编程框架，它的核心是 vbs 和 js，受这两种脚本语言的限制，决定了 asp 先天不足，它无法进行象传统编程语言那样的底层操作，所以如果你需要进行一些诸如 socket、文件等的操作时不得不借助于用其他传统编程语言如 C++、VB 、JAVA 等编写的组件，并且由于它是解释执行的，所以在运行效率上大打折扣。而 ASP.NET 呢，它是一种编译型的编程框架，它的核心是 NGWS runtime ，除了和 asp 一样可以采用 vbs 和 js 作为编程语言外，还可以用 VB 和 C# 来编写，这就决定了它功能的强大，可以进行很多低层操作而不必借助于其他编程语言。

  - **执行效率**

由于它是编译后运行的，所以执行效率要比 asp 高得多。

- **C#编译器选项全解**

可以使用 CSC.exe/?来察看可选项.

◆ 输出文件相关选项 ：

/out:<file> 输出文件名(如果不指定则从第一个源文件名中取得)

/target:exe 建立一个控制台可执行程序(这是默认选项)(可以缩略写作 /t:exe)

/target:winexe 建立一个 windows 可执行程序(可以缩略写作 /t:winexe)

/target:library 建立一个库(可以缩略写作 /t:library)

/target:module 建立一个可以加到其他汇编文件的模块(可以缩略写 作 /t:module)

/win32icon:<file> 指定一个图标作为输出文件的图标

/nooutput[+|-]　只检查代码中的错误，并不生成可执行程序
/define:<symbol file>　定义条件编译符号(可以缩略写作 /d)
/doc:<file>　生成 XML 文档

◆ 输入文件相关选项：
/recures:<wildcard>　包括当前目录及其子目录下所有符合指定的通配符规则的文件
/main:<type>　指定包含入口点的类型(忽略其他所有可能的入口点)(可以缩略写作 /m)
/reference:<file list>　参考由给出的汇编文件所指定的元数据(可以缩略写作 /r)
/addmodule:<file list>　链接指定的模块到汇编文件中

◆ 资源相关选项 ：
/resource:<resinfo>　嵌入特定的资源(可以缩略写作 /res)
/linkresource:<resinfo>　链接指定的资源到汇编文件中(可以缩略写作 /linkers)

◆ 代码生成相关选项
/debug[+|-]　产生调试信息
/optimize[+|-]　提供优化(可以缩略写作 /o)
/incremental[+|-]　进行增量编译，也就是只编译改变的部分(可以缩略写作 /incr)

◆ 错误和警告相关选项
/warnaserror[+|-]　对警告与错误作相同处理
/warn:<n>　设定警告级别(0-4)(可以缩略写作 /w)
/nowarn:<warning list>　禁止特定的警告消息

◆ 语言相关选项
/checked[+|-]　对上溢和下溢进行检查
/unsafe[+|-]　允许"不安全"的代码

◆ 其他方面的选项
@<file>　读取相应文件以获取更多选项
/help　显示帮助文件(可以缩略写作 /?)
/nologo　禁止编译版权信息

◆ 增强的选项
/baseaddress:<address>　指定被编译库的基地址
/win32res:<file>　通常用来指定存放版本和图标信息的 WIN32 资源文件
/bugreport:<file>　建立"错误报告"文件
/codepage:<n>　指定打开源文件时使用的代码页
/fullpath　指定程序生成的完整路径
/nostdlib[+|-]　不参考标准库(mscorlib.dll)

1.2.2 Visual Studio.NET 7.0 安装

Visual Studio.NET 7.0 的安装，机器必须满足下面的要求，

1、硬件要求：

① CPU: Intel Pentium II-class 300 MHz (最好 Intel Pentium III-class 600 MHz)
② 内存: 96 MB (最好 128 MB)
③ 磁盘空间: 250 MB(完全安装) 155 MB(快速安装)
④ 显示: 800x600, 256 colors
⑤ CD-ROM: required

2、软件要求：

① MicroSoft Windows 2000 + SP1
② MicroSoft Internet Explorer 5.5
③ IIS5.0
④ 其它: MDAC 2.6 Beta 2

ASP.NET 的安装过程很简单，只需按照简单提示安装即可。但是，如果你的机器安装了 OFFICE2000，在此建议安装 ASP.NET 之前先备份\MicroSoft Office\Office\mso9.dll 这个文件，因为安装完 ASP.NET 后，OFFICE 会提示你注册，否则的话 OFFICE2000 就会出现限制使用 50 次。此时将备份的 mso9.dll 文件覆盖掉原来的文件即可。

ASP.NET(NGWS SDK)的下载地址：

http://download.microsoft.com/download/platformsdk/Trial/1812.10full/NT5/EN-US/Setup.exe

安装微软的 VisualStudio.NET Beta1 和安装 ASP.NET 很多地方有惊人的相似，所以在此简单地提一下。

安装 beta1 版本的记得必须先安装以下内容：

1、windows2000 sp1
2、安装 IE5.5
3、必须要装有 iis,而且 iiS 要带 front page 扩展
4、front page 服务扩展的补丁 QFE

相同地，安装完 VisualStudio.NET 后同样会出现 OFFICE2000 的 50 次限制，所以可以用同样的方法，先备份 mso9.dll 文件，然后安装完后覆盖掉原来的文件。

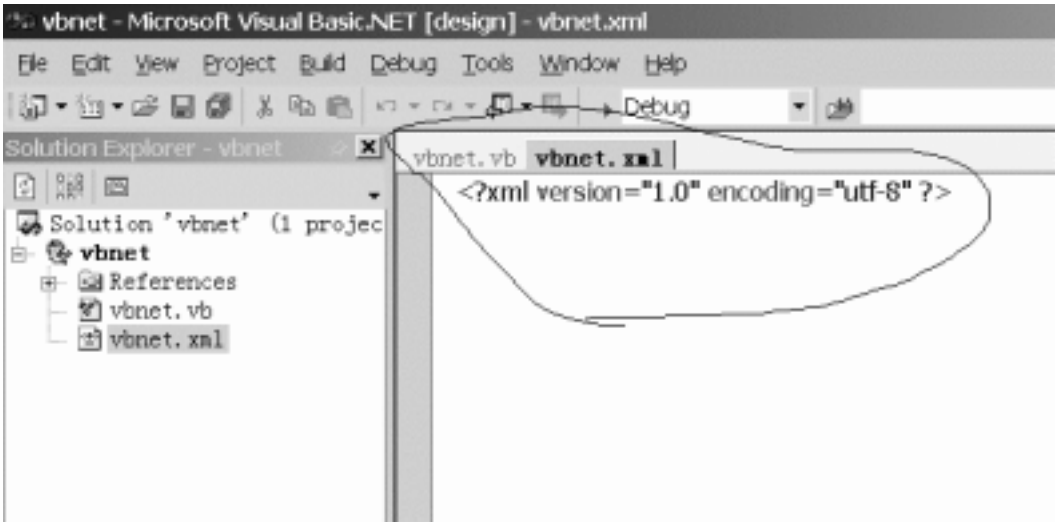
1.2.3 运行环境 IDE

微软的 vs.net7.0 IDE 是一个非常丰富的变成环境，可以进行 C#/VC++、VB.NET、ASPX 等的编程，你甚至也可以编写 ASP 文件。

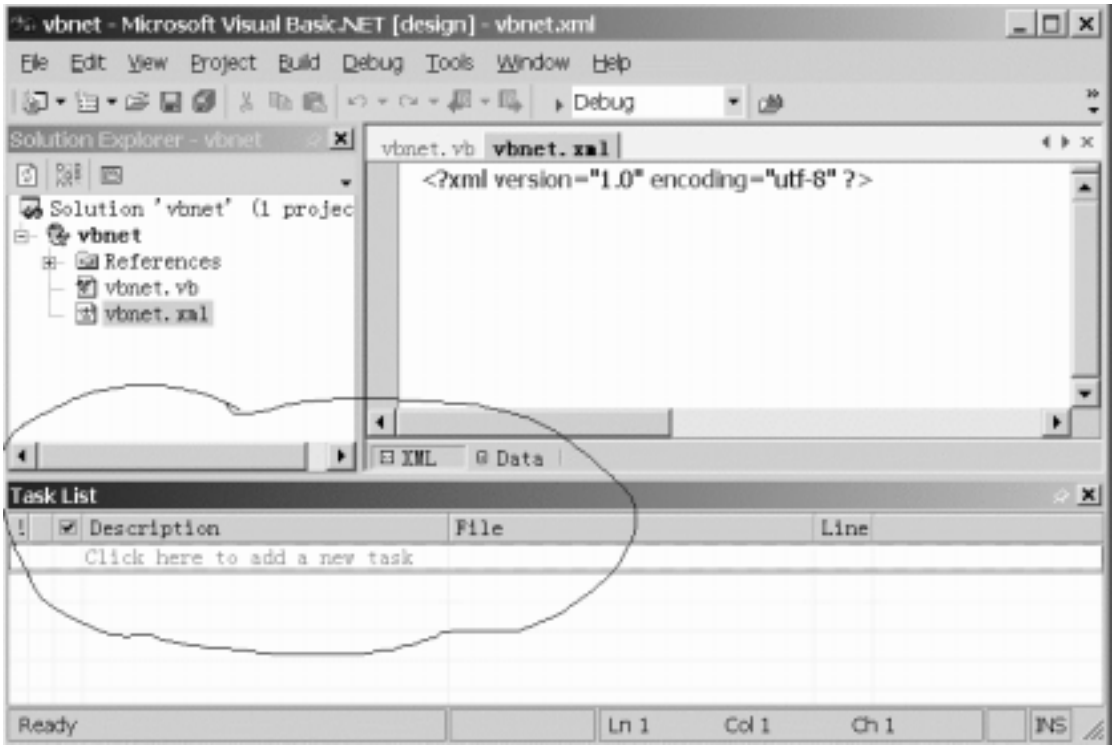
你首先看到的是 IDE，IDE 看起来很熟悉，开发 VS.NET IDE 的开发人员以前曾开发过 VB 的 IDE，它在 VB IDE 的基础上又有了新的提高。

然而，IDE 的变化绝非是表面性的。所有的.NET 语言都使用同一个 IDE，其中的新工具的功能是强大而全面的，你可以把任何一个设计窗口设定为自动隐藏（就象 Windows 中的任务条一样），这样就可以使桌面显得不太凌乱，如下面所示：

主工作区是一系列的标签，也就是说 IDE 不会同时显示许多的窗体或代码模块，在打开对象的源代码时，IDE 就会在相应对象的主区内增加新按钮，如下面所示：



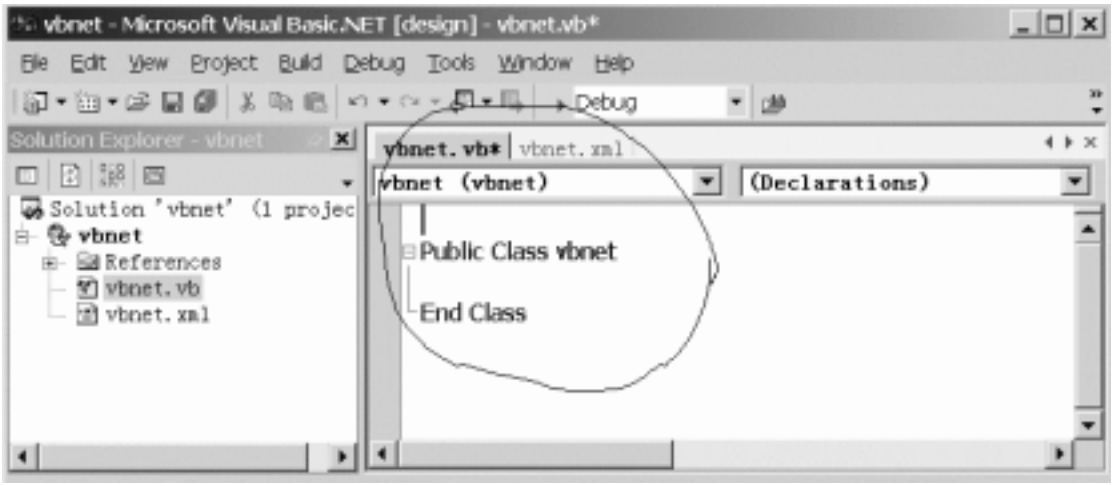
IDE 中还有一个新的被称作任务清单的窗口，其中的内容包括你和 IDE 创建的项目。例如，如果在编译一个 VB 项目时收到一个错误信息，VB 就会在任务清单中创建一个条目来解释这个问题，如下所示：



你还可以直接在任务清单中添加一个条目，或者通过在代码中建立以"TODO:"开头的注释把代码中的一个位置与任务联系起来。我非常喜欢微软添加的任务清单，它能使我节约不少的时间，并有助于我能够更好地调试自己的软件。

另一个会立刻感受到的变化是.NET IDE 中的窗体。微软抛弃了原来的窗体引擎，而采用了 Windows 风格的窗体，所有的基于 CLR 的语言都使用 Windows 的窗体引擎，与 VB6 等中的使用的窗体引擎相比，它有几个明显的优点。例如，Windows 的窗体可以自动地改变其中的组件的大小，而且可以把控制锁定在特定的位置，也就是说，我们无需借助第三方的工具来完成相应的工作了。另外，Windows 的窗体还可以使我们完成另外一些很"酷"的工作，例如创建透明的窗体。

过去，VB 隐藏了创建一个窗体所必需的全部工作。我们使用 IDE 创建一个窗体，并在 Initialize 事件处理程序中添加代码，但对于发生在这两者之间的过程则无能为力。在 VB.NET 中，窗体成了一个类，包含创建窗体的全部代码，我认为这些代码是"鸡肋"，原因是大多数的开发人员都不想去理它。如果说有一种东西一定能让你的软件出问题，那就是这些代码了。一些高级开发人员可以通过这些代码完成一些很"酷"的工作，因为它可以让你"看到"VB 创建窗体的全部情况。如果不想看，你并非必须看这些代码，新的代码编辑器可以扩展或消除一些代码区，在缺省状态下这些代码是不会显示的。代码编辑器还包括一些新的特性，例如它可以自动地对编辑的源代码进行"缩进"处理，而且可以显示源代码的行号，如下面所示：



还有有了这个 IDE 之后，我们就可以不用手工编写编译语句了，直接就可以把我们的.vb 或者.cs 文件编译成.dll 或者.exe 文件，等等。

总之，微软的.NET IDE 是一个很酷的编程环境，如果一个一个的介绍，那可得写几本书了，大家只有多用才会熟练啊。

总之，微软的.net 是一个很酷的变成环境，如果一个一个的介绍，那可的小写基本书了，大家只有多用才会熟练。

1.2.4 小结

MicroSoft .NET 的策略是将互联网本身作为构建新一代操作系统的基础，对互联网和操作系统的设计思想进行合理延伸。这样，开发人员必将创建出摆脱设备硬件束缚的应用程序，以便轻松实现互联网连接。MicroSoft .NET 无疑是当今计算机技术通向计算时代的一个非常重要的里程碑。通过上面的介绍，相信大家对.net 以及 asp.net 有了一定的了解。在下面的内容，我们将带大家进入一个崭新的 ASP.NET 世界。