NuGet

使用 NuGet 管理项目库

Phil Haack

无论多么努力，Microsoft 也没办法提供开发人员所需要的每一个库。 虽然 Microsoft 在全球的员工人数接近 90,000，但全球的开发人员数以百万计。 指望 Microsoft 满足每一个人的需求是不现实的，也不可想像。 因此，开发人员通常得自己动手解决问题，他们目前已经编写了成千上万的实用库，并将其发布到 Web 上。

如何共享如此多的库是一个令人头痛的问题。 共享和重用代码是一个很大的挑战。 不相信？ 请随便走进一间中型或大型工作室，问问他们有多少日志记录库。 访问多家公司后，您将发现他们拥有比例非常高的内部日志记录库，而这些库中有一些非常不错，例如，Log4Net、NLog 和 Error Logging Modules and Handlers（即 ELMAH）。

当一位开发人员开始新项目时，他将面对一张空白的画布。 他如何去发现这些有用的库？ 如何将库集成到当前项目中并管理库的依赖项和更新呢？

ELMAH 就是一个非常有用的库，是由开发人员自己编写的。 ELMAH 能够在出现异常时记录 Web 应用程序中所有未经处理的异常以及所有请求信息，例如，标头、服务器变量等。 假设您刚刚听说 ELMAH 并希望在下一个项目中使用它。

您可能会采取下列步骤：

1. **查找 ELMAH。**由于它名称独特，Bing 搜索的第一条搜索结果将是 ELMAH Google 代码页。
2. **下载正确的 zip 包。**该站点的下载页面有多个 zip 包。 您必须思考并选取正确的一个。 有时，您并不能一眼就看出正确的是哪个。
3. **“取消阻止”程序包。**从 Web 下载程序包后，您需要右键单击该文件，打开“属性”对话框，然后单击“取消阻止”按钮以从该文件删除“Web 的标记”。
4. **验证其哈希值是否与托管环境提供的哈希值相符。**Google 代码站点会显示代表该 zip 文件的 QR 代码。 在您认识的开发人员中，有多少会抽出时间来根据 QR 代码验证文件？
5. **将程序包的内容解压缩到解决方案中的特定位置。**大多数开发人员会避免将程序集解压缩到 bin 目录，这是因为该目录用于生成输出而非输入，并且不在版本控制的跟踪范围之内。 实际上，有必要将该依赖项添加到版本控制之下的文件夹，并从该位置引用该程序集。
6. **在项目中添加程序集引用。**必须在 Visual Studio 项目中添加对该程序集的引用，然后才能使用该程序集。
7. **使用正确的设置更新 web.config。**这可能意味着您要使用 Bing 或 Google 进行更多搜索才能找到配置文件所需的正确设置。

真是很麻烦！ 现在，假设您必须为 10 至 15 个依赖项执行这些操作。 当您的应用程序要发布新版本时，您需要花费大量时间为应用程序的依赖项搜索更新。

“非我发明”(NIH) 过去常常遭到非议，而在现在听起来却是不错的主意。

NuGet 应运而生

NuGet 是一种 Visual Studio 扩展，它能够简化在 Visual Studio 项目中添加、更新和删除库（部署为程序包）的操作。 NuGet 程序包是打包成一个文件的文件集，扩展名是 .nupkg，使用开放打包约定 (OPC) 格式。

OPC 仅仅是具有某些元数据的 zip 文件的首字母缩写词。 事实上，您可能早已熟悉 OPC，因为 Word 和 Excel 文档正是使用此格式。 如果您取一个 .docx 文件并将文件扩展名改为 .zip，您实际可以打开它并浏览里面的内容。 .nupkg 文件同样如此。

NuGet 产品同样随附能够轻松创建和发布程序包的实用工具。 现在，我先重点介绍如何使用 NuGet 发现和安装程序包。 之后，我将讲述如何创建和发布程序包。

安装 NuGet

要安装 NuGet，从“Tools”（工具）|“Extension Manager”（扩展管理器）菜单选项启动 Visual Studio Extension Manager。 单击“Online Gallery”（联机库）选项卡查看可用的 Visual Studio 扩展名，如**图 1** 中所示。 如您所见，NuGet 位于第一个屏幕，是排名最高的程序包。 如果情况不是如此，您可以使用右上角的搜索框找到它。 单击“Download”（下载）按钮安装 NuGet。

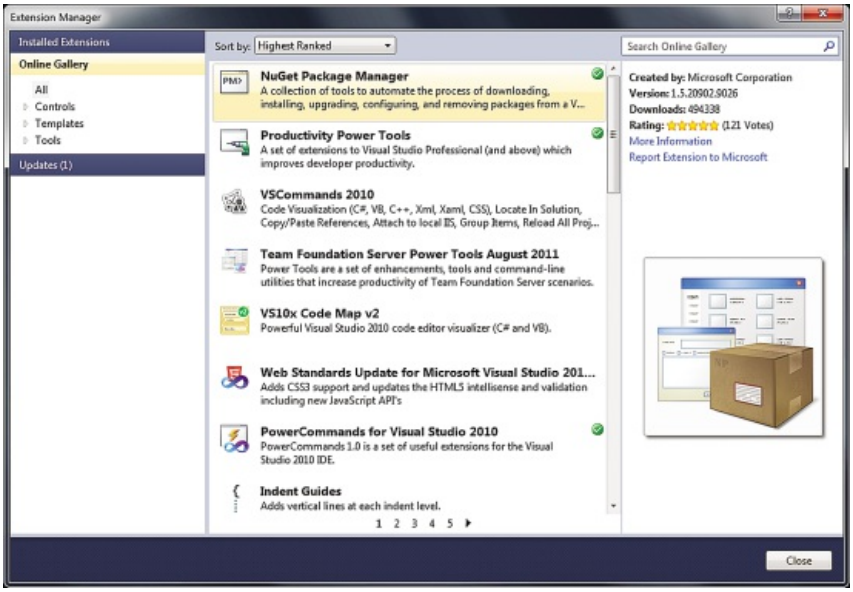

图 1：**Visual Studio Extension Manager**

如果您已经安装了 ASP.NET MVC 3，则您已经安装 NuGet。 ASP.NET MVC 3 包含 NuGet，并且 Microsoft 计划在下一版本的 Visual Studio 中包含 NuGet。

安装程序包

我们启动用户友好的 NuGet 对话框以安装程序包。 NuGet 同样内置基于 Windows PowerShell 的控制台，此控制台面向高级用户（将在下文提及）。

要启动 NuGet，请右键单击项目的引用节点，然后选择“Manage NuGet Packages”（管理 NuGet 程序包）选项（NuGet 1.4 之前的该选项具有不同的标签）。 这将启动“Manage NuGet Packages”（管理 NuGet 程序包）对话框，如**图 2** 中所示。

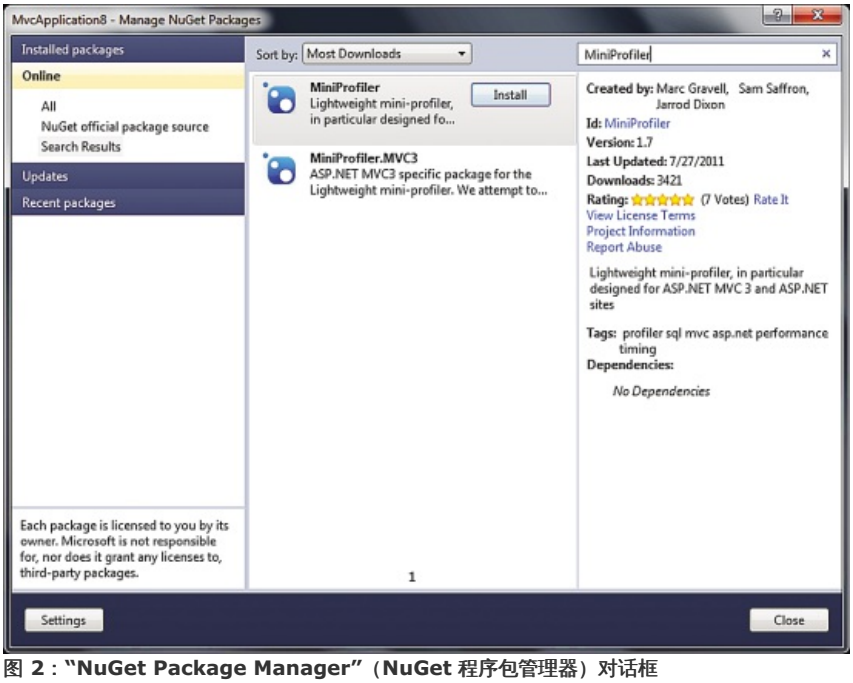

图 2：**“NuGet Package Manager”（NuGet 程序包管理器）对话框**

请确保选中“Online”（联机）选项卡，并在右上角输入搜索词（例如，搜索来自 StackOverflow.com 的实用库 MiniProfiler）。

找到程序包后，单击“Install”（安装）按钮安装该程序包。 NuGet 随后下载该程序包和它的依赖项，并将任何必要更改应用到程序包指定的项目中。

NuGet 执行下列步骤安装程序包：

1. **下载程序包文件及其所有依赖项。**有些程序包要求显式接受许可，并提示用户接受程序包的许可条款。大多数程序包支持隐式接受许可，并不发出提示。 如果解决方案或本地计算机缓存中已经存在该程序包，NuGet 将跳过下载程序包的步骤。
2. **提取程序包的内容。**NuGet 将内容提取到程序包文件夹中（在必要时创建文件夹）。 程序包文件夹在您的解决方案 (.sln) 文件的并列位置。 如果解决方案的多个项目中安装了同一个程序包，则仅提取该程序包一次并由各项目共享。
3. **引用程序包中的程序集。**依照惯例，NuGet 更新项目以引用程序包 *lib* 文件夹中的一个或多个特定程序集。 例如，将程序包安装到面向 Microsoft .NET Framework 4 的项目时，NuGet 将添加对 *lib/net40* 文件夹中的程序集的引用。
4. **将内容复制到项目中。**依照惯例，NuGet 将程序包的内容文件夹的内容复制到项目中。 这对包含 JavaScript 文件或图像的程序包十分有用。
5. **应用程序包转换。**如果任何程序包包含转换文件，例如用于配置的 app.config.transform 或

web.config.transform，则 NuGet 将在复制内容之前应用这些转换。 有些程序包所含的源代码经过转换可以在源文件中包含当前项目的命名空间。 NuGet 同样转换这些文件。

6. **运行程序包中关联的 Windows PowerShell 脚本。**有些程序包可能包含 Windows PowerShell 脚本，这些脚本使用设计时环境 (DTE) 自动化 Visual Studio，从而处理与 NuGet 无关的任务。

当 NuGet 执行所有这些步骤后，库将准备就绪。 很多程序包使用 WebActivator 程序包自行激活，从而最小化安装后所需的任何配置。

您可以卸载程序包，这将使项目回到安装程序包之前的状态。

更新程序包

在《软件工程的事实和谬误》（Addison-Wesley Professional，2002 年）一书中，Robert L. Glass 说："维护工作通常约占软件成本的 40-80%（平均是 60%）。 因此，维护可能是软件生命周期中最重要的阶段。"

安装程序包仅仅是故事的开始。 在今后维护这些库时，您必须及时更新应用程序，为库安装最新的 Bug 修补程序。 这需要您回答一个问题："此项目中的哪些依赖项有可用的最新更新？"

如前所述，回答此问题一直是一项耗时的工作。 然而，有了 NuGet，您只需启动对话框并单击"Updates"（更新）选项卡即可看到有可用更新的程序包列表，如**图 3** 所示。 单击"Update"（更新）按钮可将程序包更新到最新版本。

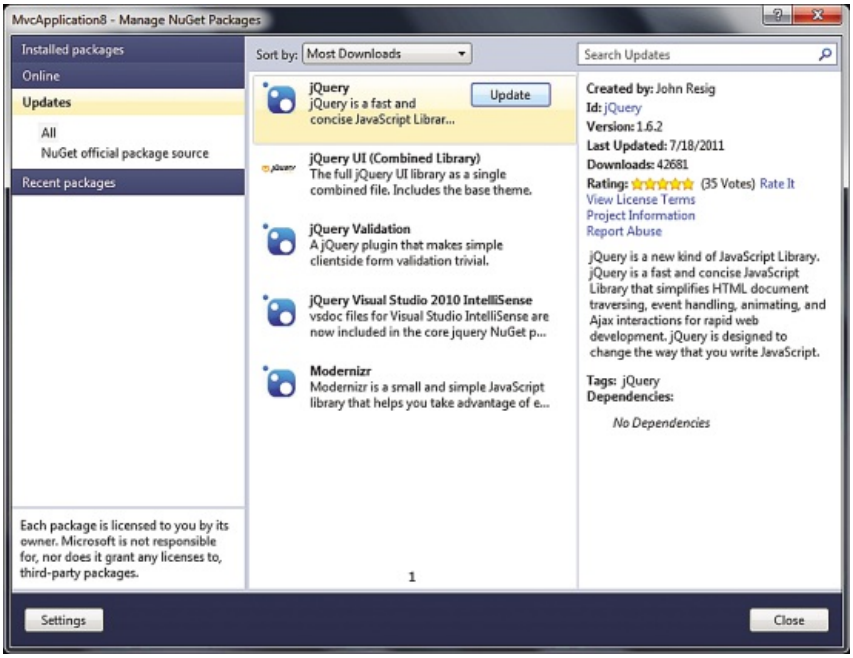

图 3 当前项目的可用更新

更新命令卸载旧程序包，然后安装新程序包，确保所有依赖项都根据需要得到更新。

NuGet 在程序包管理器控制台中提供便于控制更新的命令，例如，用以更新解决方案中的所有程序包或执行"安全"更新。

面向高级用户的 NuGet

虽然我对美观的 GUI 对话框非常痴迷，但我知道很多开发人员非常鄙视像我一样的鼠标操作者。 这些开发人员将命令行 shell 视作 UI，他们更愿意通过这些 shell 组成命令集。

NuGet 能够满足这种需求，它提供基于 Windows PowerShell 的控制台窗口（称作程序包管理器控制台）以及一组 Windows PowerShell 命令与 NuGet 进行交互。 Windows PowerShell 是基于 .NET 的脚本语言和命令行 shell，非常适合组成命令集，并能够处理对象。

要启动程序包管理器控制台，请导航至"Tools"（工具）|"Library Package Manager"（库程序包管理器）|"Package Manager Console"（程序包管理器控制台）菜单选项。

列出和安装程序包

要列出和搜索程序包，请使用 Get-Package 命令。 默认情况下，该命令列出当前项目中的已安装程序包。您可以通过指定 ListAvailable 标记和 Filter 标记联机搜索程序包。 下列命令行搜索所有包含"MVC"的程序包：

```
Get-Package -ListAvailable -Filter Mvc
```

找到要安装的程序包后，使用 Install-Package 命令。 例如，要将 ELMAH 安装到当前项目：

```
Install-Package Elmah
```

由于 Windows PowerShell 是动态语言，它能够提供 Tab 扩展功能，从而帮助您正确输入命令行参数。Tab 扩展等同于 C# 的 IntelliSense，但与基于编译时信息的 IntelliSense 不同，您可以在运行时填充 Tab 扩展。

例如，如果您输入 Install-Package Mvc{tab}，您将看见一个列表，包含可能来自程序包源的程序包名称，如**图 4** 所示。

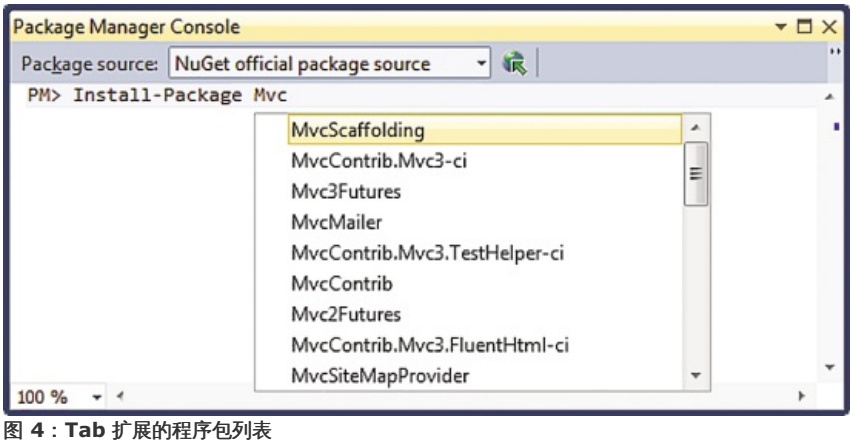

图 4：**Tab 扩展的程序包列表**

更新程序包

程序包管理器控制台还包含一个命令，与对话框相比，它提供更多的更新控制。 例如，无需参数即可调用此命令以更新解决方案的每个项目中的各程序包：

Update-Package

此命令尝试将每个程序包都更新到最新版本。 因此，如果您有 1.0 版本的程序包，而 1.1 和 2.0 版本在该程序包源中可用，则该命令将此程序包更新至最新的 2.0 版本。

如果任何程序包包含重大改变，这会是一项非常重大的操作。 在多数情况下，您仅希望将各程序包更新至最新的修补程序版本。 这叫"安全"更新，前提是具有较大内部版本号或修订号（但具有相同的主版本号和次版本号）的程序包能够向后兼容。 仅添加 Safe 标记以执行安全更新，例如：

Update-Package -Safe

在这种情况下，如果您安装了 1.0.0 版本的程序包，而 1.0.1 和 1.1 版本在该程序包源中可用，则该程序包将安全地升级至 1.0.1 而非 1.1。

Update-Package 命令还提供更精细的控制，例如，将程序包更新至特定版本而非最新版本。

用新命令扩展 Visual Studio

虽然使用 Windows PowerShell 安装程序包的功能很不错，但这不是我们选择 Windows PowerShell 的最主要原因。 最主要原因之一是程序包能够将新命令添加至程序包管理器控制台。 这些命令能够与 Visual Studio DTE 交互以执行各种任务。

例如，安装 MvcScaffolding 程序包时，它会将新 Scaffold Controller 命令添加至控制台。 给定一个 Entity Framework (EF) Code First 对象，此命令生成一个控制器、控制器操作，以及 EF 对象的基本创建、读取、更新和删除 (CRUD) 操作对应的视图，如**图 5** 所示。

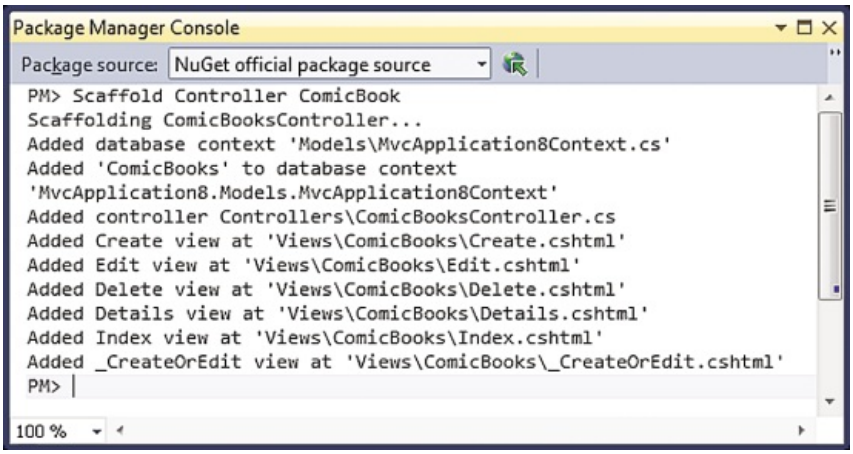

图 5：**运行中的 MvcScaffolding Custom Scaffold 命令**

您的组织中的 NuGet

由于用户仅关注 NuGet 如何简化与公共开发人员社区共享库，用户通常很容易忽视 NuGet 在企业中的作用。

毕竟，企业也没有特殊手段能够避免整个社区所面临的代码共享难题。 随着公司的成长，平均信息量也在增加。 在同一间公司，不同的组使用各自专有版本的公司"标准"库。 有些组可能会完全无视这些库，而是从头自己编写。

问题往往不在于库本身，而在于与其他团队共享这些库并通知他们更改时的麻烦。 听起来是不是很熟悉？

程序包来源

至此，我已讲完如何安装程序包，但尚未回答这样一个明显的问题：这些程序包在哪？ 它们位于 nuget.org 的官方 NuGet 程序包库中。 此库公开了一个 OData 源：packages.nuget.org/v1/FeedService.svc。

OData 格式使 NuGet 客户端能够在客户端上生成搜索程序包源的特定查询，但是会在服务器上执行这些查询。

要向 NuGet 添加更多程序包源，请导航至“Tools”（工具）|“Library Package Manager”（库程序包管理器）|“Package Manager Settings”（程序包管理器设置）菜单选项，单击“Package Sources”（程序包源）节点，如**图 6** 所示。

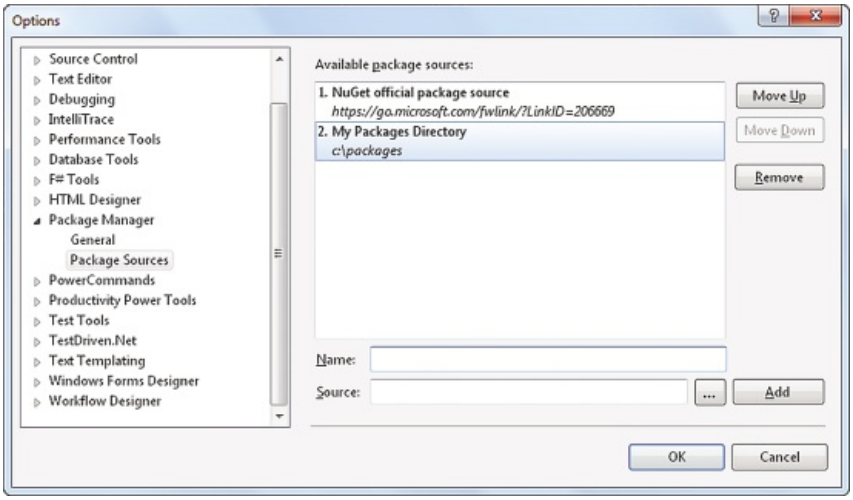

图 6：程序包管理器设置

默认的程序包源位于 Web 上的 OData 端点，但示例屏幕快照同样将本地文件夹显示为程序包源。 NuGet 将文件夹（无论位于本地还是网络共享位置）视为程序包源，并在“Online”（联机）窗格中列出文件夹中的每个程序包。 这样一来，只需将代码放入一个文件夹即可与他人共享，并且在测试您自己创建的程序包时同样有用。

托管您自己的 NuGet 服务器

除了在网络共享上托管程序包之外，您还可以将网站设置为程序包源，使用网站与组织中的其他人共享程序包。

如果有很多任务，还有一个程序包可以在此处帮到您。 首先，在 Visual Studio 中创建一个空的 ASP.NET Web 应用程序（面向 ASP.NET 4）。 使用 NuGet 安装程序包 NuGet.Server。 此程序包将简单的 OData 端点添加到空 Web 应用程序中。

接着，将程序包文件添加到 Web 应用程序的 Packages 文件夹，以便发布它们并部署网站。 有关如何设置的详细信息，请参阅 NuGet 的文档站点 bit.ly/jirmLO。

如果您希望部署类似 nuget.org 的完整库体验，NuGet 库代码还可通过 nugetgallery.codeplex.com 项目作为开放源项目提供。

通过托管专用 NuGet 服务器或库实施，您可以方便地在公司内部共享专有代码，无需公开发布。

创建程序包

NuGet 发挥作用的前提是有程序包可供安装。 NuGet 中的有用程序包越多，NuGet 对每一位开发人员的价值越大。 这就是 NuGet 团队不辞劳苦尽可能创建使用方便的程序包的原因。 虽然程序包的创建不难，但 NuGet 团队一直在对其功能进行投资，以使之更加简单。 他们已开发若干用于创建 NuGet 程序包的工具。 例如，Package Explorer 是 NuGet 团队的开发人员编写的一个 ClickOnce 应用程序，使用者可以凭借它在创建或测试程序包时进行可视化操作。 您可以在 npe.codeplex.com 下载该应用程序。

NuGet.exe

由于大多数程序包作者都希望将程序包的创建集成到生成流程，让我们看看使用 NuGet 命令行实用工具的其他方法。 您仅需从 bit.ly/gmw54b 下载一次该实用工具。 下载 NuGet.exe 后，请确保将其放入已添加至 PATH 环境变量的文件夹。 我针对此类实用工具创建了一个名为“utils”的文件夹。

我之所以说只需下载 NuGet.exe once 一次（每台计算机一次），是因为它是自行更新的可执行程序。 如果出现更新的版本，仅需运行以下命令，NuGet 即会联机检查并自行更新至最新版本：

nuget update –self

该命令行工具能够查询类似程序包管理器控制台的联机源。 例如，要搜索带“MVC”的所有程序包，可使用以下命令：

nuget list Mvc

NuGet.exe 甚至能够下载程序包和依赖项并解压缩它们，但它不能将项目修改为引用已下载的程序包程序集或运行程序包中包含的任何 Windows PowerShell 脚本。

从项目创建程序包

程序包在 90% 的情况下仅包含一个程序集（据本人统计）。 此部分讲述使用 NuGet.exe 针对项目文件（例如，.csproj 或 .vbproj 文件）创建上述程序包的简单流程。

有关创建较复杂程序包（例如，针对不同 .NET Framework 版本的单个程序包）的详细信息，请参阅 docs. nuget.org 的 NuGet 文档网站。

创建程序包的基本步骤：

1. 创建一个类库项目。
2. 从项目生成 NuSpec 清单。
3. 更新项目的程序集元数据。
4. 使用 NuGet.exe 创建程序包。

创建类库项目。要共享程序集，首先要创建类库项目。 NuGet 可以压缩多个项目类型，但共享代码时最常见的情况是使用类库。 创建程序包后，务必打开 AssemblyInfo.cs 文件以更新程序集的元数据。

创建 NuSpec 清单。NuSpec 文件是程序包清单，包含与程序包有关的重要元数据（例如，作者、描述和程序包依赖项）。 自己动手创建 NuSpec 格式很简单，但使用 spec 命令创建此文件更加方便。 确保在项目文件所在目录中运行命令：

```
nuget spec
```

在此特定情况下，由于 spec 命令从项目文件生成 NuSpec，它会包含某些元数据的占位符，如**图 7** 中所示。

图 7：**生成的 NuSpec 文件**

```
<?xml version="1.0"?>
<package xmlns=
  "http://schemas.microsoft.com/packaging/2010/07/nuspec.xsd">
  <metadata>
    <id>$id$</id>
    <version>$version$</version>
    <title>$title$</title>
    <authors>$author$</authors>
    <owners>$author$</owners>
    <licenseUrl>http://LICENSE_URL_HERE_OR_DELETE_THIS_LINE</licenseUrl>
    <projectUrl>http://PROJECT_URL_HERE_OR_DELETE_THIS_LINE</projectUrl>
    <iconUrl>http://ICON_URL_HERE_OR_DELETE_THIS_LINE</iconUrl>
    <requireLicenseAcceptance>false</requireLicenseAcceptance>
    <description>$description$</description>
    <copyright>Copyright 2011</copyright>
    <tags>Tag1 Tag2</tags>
  </metadata>
</package>
```

请勿编辑包含占位符的字段，但应在其他字段（例如，licenseUrl、projectUrl、iconUrl 和 tags）中填入正确的值。

更新项目的程序集元数据。每个程序集都有与其关联的元数据。 NuGet 能够读取这些程序集元数据并在创建程序包时将其合并到 NuSpec 清单，从而确保了此信息始终在您的程序包和程序集中保持同步。

如此前所述，此信息通常位于名为 AssemblyInfo.cs 的文件中。 **图 8** 中的表显示了程序集元数据和 NuSpec 占位符值之间的对应关系。

图 8：**映射到 NuSpec 的程序集元数据**

标记	源
id	程序集名称。
$title$	AssemblyTitleAttribute 中指定的程序集标题。
$version$	程序集的 AssemblyVersionAttribute 中指定的程序集版本。
$author$	AssemblyCompanyAttribute 中指定的公司。
$description$	AssemblyDescriptionAttribute 中指定的描述。

与其他字段不同，id 字段并非从程序集属性提取，而是设置为程序集名称。

创建程序包。在项目文件和 NuSpec 文件所在目录中，运行以下命令以创建程序包：

```
nuget pack ProjectName.csproj
```

如果同一个目录中只有一个项目文件，则在运行命令时可以省略项目文件名称。

如果尚未编译项目，可先用 Build 标记编译项目，然后压缩它。 这将在运行 pack 命令之前编译项目：

```
nuget pack ProjectName.csproj -Build
```

此命令将生成名为 ProjectName.{version}. nupkg 的文件，其中，{version} 的值与 AssemblyVersionAttribute 中指定的值相同。 例如，如果版本是 1.0.0，您的程序包将命名为 ProjectName.1.0.0. nupkg。 您可以在操作后使用 Package Explorer 检查程序包，以确保创建正确。

为了方便开发人员安装您的程序包，请考虑使用 Symbols 标记创建带调试器符号的程序包：

```
nuget pack ProjectName.csproj -Build -Symbols
```

除了主程序包之外，此命令还创建符号程序包。 这使安装您的程序包的其他人在调试其应用程序时能够单步执行程序包代码。

发布程序包

创建程序包后，您可能希望与全世界共享。 NuGet.exe 专门针对此目的提供一条发布命令。 发布之前，您需要在 nuget.org 上创建一个帐户。

注册帐户后，单击指向您的帐户的链接以查看您的访问密钥。 此密钥非常重要，因为向 nuget.exe 命令标识库，并且是可撤消的密码。

一旦拥有自己的密钥后，请使用以下命令将其存储在安全的位置：

nuget setApiKey b688a925-0956-40a0-8327-ff2251cf5f9a

存储密钥后，使用 push 命令将您的程序包发布到库：

nuget push ProjectName.1.0.0. nupkg

在库上载程序包之前，该命令将验证库的 API 密钥。 如果您创建了前述符号程序包，则应在对程序包执行 push 命令时指定 Symbols 标记：

nuget push ProjectName.1.0.0. nupkg -Symbols

确保指定主程序包名称而非符号程序包名称。 依照惯例，此命令查找特定的符号程序包。 此命令将主程序包推送到 NuGet 库，并将符号程序包推送到合作伙伴的 symbolsource.org 存储库。

后序

在本文中，我演示了 NuGet 如何从 NuGet 库提取有用的库以助推新项目开发的启动。 在企业内部，NuGet 可用于在组织中的不同开发人员之间共享代码。

但是我需要指出，大家对 NuGet 存在一种固有的误解：NuGet 仅适合 Web 开发人员。 该误解可能源于它随附在 ASP.NET MVC 3 版本中，但这种观点是错误的。 NuGet 并非仅针对 Web 开发人员，而是所有开发人员。 NuGet 支持 Windows Phone、Silverlight、Windows Presentation Foundation 以及其他项目类型，并将在今后支持新的项目类型。

NuGet 是社区驱动的开放源码项目，通过 Apache 2 许可注册。 该项目属于 Outercurve Foundation，但已集成到 Microsoft 的产品，并且若干 Microsoft 开发人员已成为它的核心参与者。

若希望帮助 NuGet 的开发，请访问 nuget.codeplex.com 以了解如何参与其中以及如何对 NuGet 做出贡献。

本文章仅对 NuGet 的功能进行了初步探讨。 要了解更多信息，请访问 docs. nuget.org 的 NuGet 文档网站（开发团队在对其进行精心维护，并接受对其文档的贡献）。 开发团队积极参与 CodePlex 讨论论坛中有关 NuGet 项目的讨论，并欢迎您提出有关问题。

Phil Haack *是 Microsoft ASP.NET 团队的高级项目经理，专注于为开发人员创建优秀的产品。他参与 ASP.NET 很多领域的工作，他的主要项目是 ASP.NET MVC 和 NuGet 程序包管理器，这两个产品均通过 OSS 许可证发布。在工作之余，他会在博客 (haacked.com) 上写有关软件的文章，并研究 Subtext 开放源码博客引擎。*

衷心感谢以下技术专家对本文的审阅： ***David Fowler***

MSDN Magazine Blog

14 Top Features of Visual Basic 14: The Q&A
Wednesday, 1月 7
Big Start to the New Year at MSDN Magazine
Friday, 1月 2

More MSDN Magazine Blog entries >

Current Issue

Browse All MSDN Magazines

Subscribe to MSDN Flash newsletter

Receive the MSDN Flash e-mail newsletter every other week, with news and information personalized to your interests and areas of focus.

Windows

Office

Visual Studio

Microsoft Azure

更多...

学习资源

Microsoft 虚拟学院

第 9 频道

MSDN 杂志

社区

论坛

博客

Codeplex

支持

自助支持

计划

BizSpark（针对新创企业）

DreamSpark

创新杯

中国 (简体中文)

新闻稿 隐私& Cookie 使用条款 商标

Microsoft