# TFS 用户指南

## 什么是 Team Foundation Server

针对整个团队的协作软件开发工具

Team Foundation Server 提供一组协作软件开发工具，它与你现有的 IDE 或编辑器集成，从而使你的跨功能团队可以在所有大小的软件项目上高效工作。



TFS提供了一组集成的特性，可以通过web浏览器或支持的IDE访问这些特性，包括:

1、用于源代码的源代码控制的Git存储库

2、构建和发布管理以支持持续集成和交付应用程序

3、敏捷工具支持计划和跟踪您的工作，代码缺陷，以及使用看板和Scrum方法的问题

4、测试您的应用程序的各种工具，包括手动/探索性测试、负载测试和持续测试

5、用于共享进度和趋势的高度可定制的仪表板

此外，TFS生态系统还提供了对添加扩展的支持，集成了其他流行的服务，例如:篝火、Slack、Trello、UserVoice，以及开发你自己的自定义扩展。

在部署TFS时，还可以配置以下服务器或集成点:

构建服务器:支持本地和云托管的构建

SQL Server和SQL分析服务器:支持SQL Server报告和基于多维数据集创建Excel轴心图表的能力

在哪里开始呢?

首先，免费下载TFS Express

然后，上传代码来共享或源代码控制

或者，开始使用Scrum、看板或方法组合来跟踪你的工作

当您需要数据留在您的网络中，或者您想要访问与TFS数据和工具集成的SQL Server reporting services时，选择一个本地的TFS。

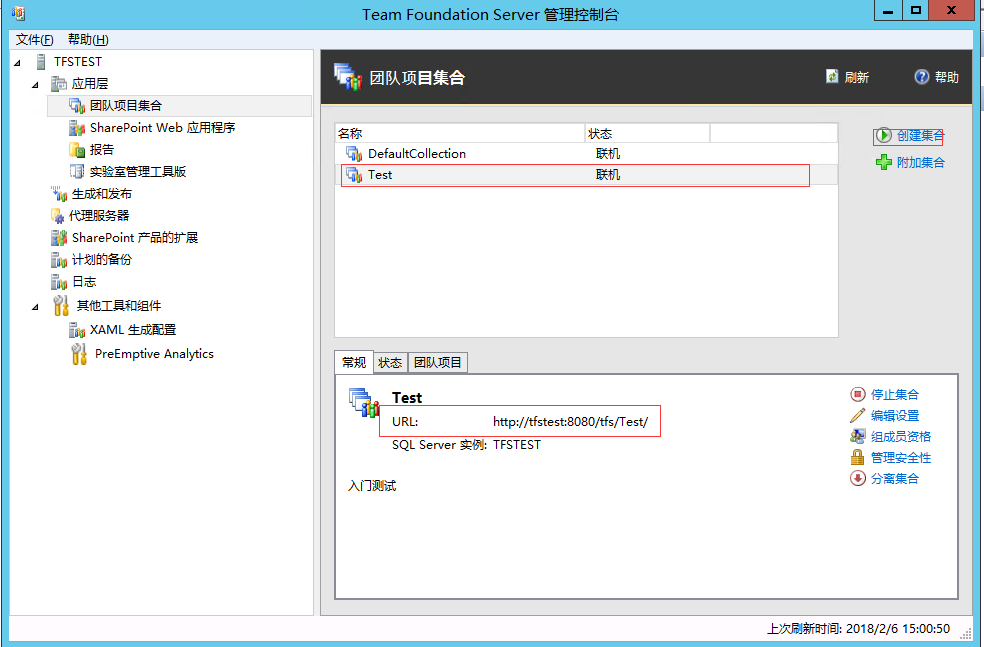
你可以通过下载TFS Express免费开始。

## Team Foundation Server Express

供个人或小团队免费使用，将 Team Foundation Server Express 提供给单个开发人员或五人（或五人以下）的小型团队使用。 在你的个人台式电脑或笔记本电脑上轻松安装，无需专用服务器。

## Team Foundation Server Manangement

团队项目集合需要通过服务器管理控制台创建。

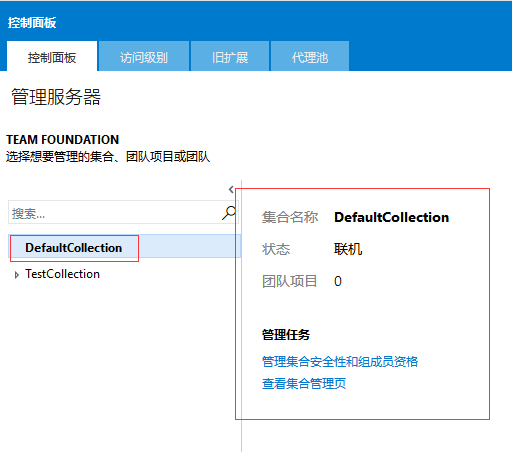


1、从服务TFS管理控制台中创建一个集合，如Test

2、在web部打开url ，域名若访问不了，改成IP访问。如：http://10.2.21.145:8080/tfs

### 集合管理（admin）

通过“管理”进入，在控制面板中可以看到多个集合，你可以在这里完成管理集合任务



#### 1、创建团队项目

在概述选项卡中，可以创建团队项目。

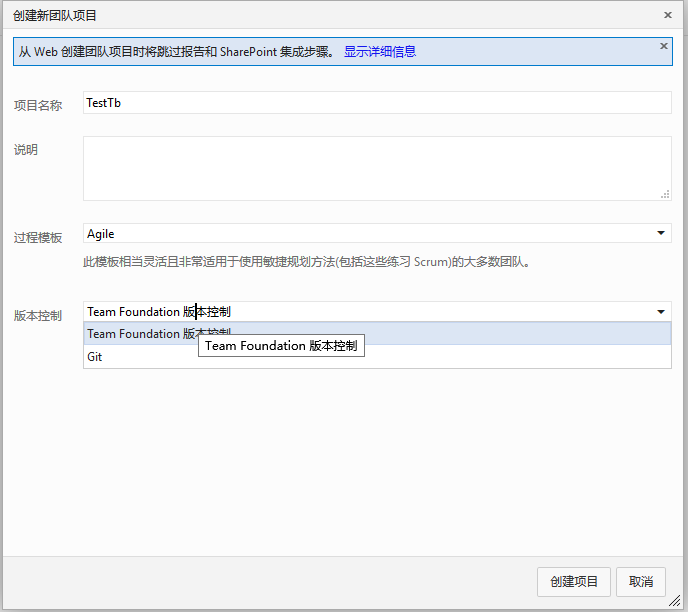
参考： VS 客户端创建

<https://msdn.microsoft.com/library/ms181477.aspx>

WEB 端创建，通过上图URL<http://10.2.21.145:8080/tfs> 进入，选择管理。



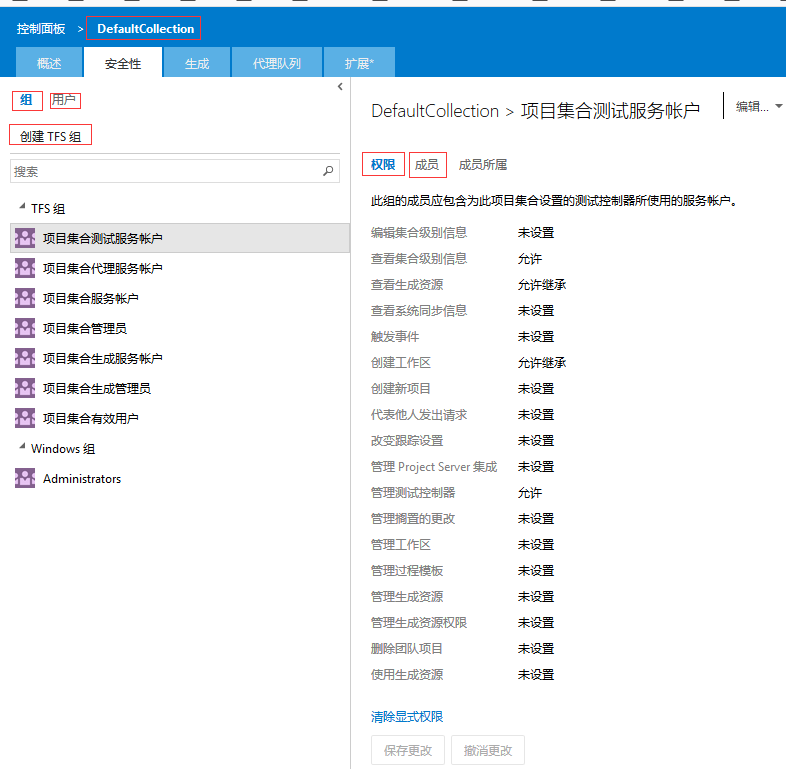
1)创建新团队项目可选择版本控制方式，这里有TFS和GIT两种版本控制方式



#### 2、安全性和组成员资格

权限是在权限和组中描述的集合、团队项目和对象级别设置的。因此，要查看您拥有的权限，您需要在对象、项目或集合级别上打开权限。

集合、团队项目或团队都有各自的安全性和组成员资格。在安全性选项卡中可以为它们创建TFS组或用户，并对它们进行权限设置。如图管理集合安全性



如团队项目或团队下，在安全性选项中添加用户和组

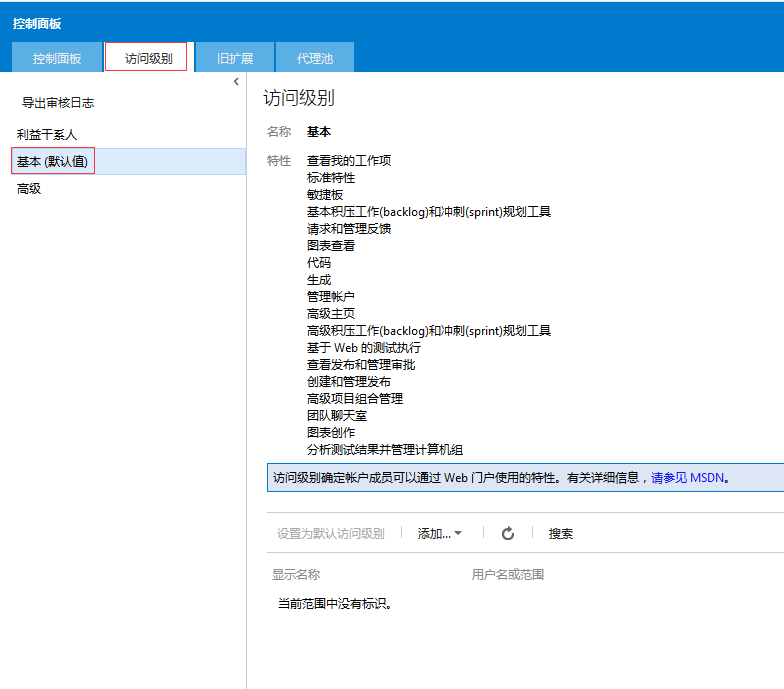


### 访问级别

访问级别使管理员能够为他们的用户提供他们需要的功能，并且只需要支付这些功能。为了连接和使用TFS提供的功能和特性，必须将用户添加到具有适当权限的组中。要使用select web portal特性，它们还必须属于允许访问该特性的访问级别。

当您将一个用户或组添加到一个团队或团队项目时，他们会自动获得访问默认访问级别所支持的那些功能，这是基本的。这为大多数用户提供了所需的所有特性.

访问级别要设置为 基本 或 高级，否则新添加到团队中的用户（除属于administratrer）将无法访问源代码。



<https://docs.microsoft.com/zh-cn/vsts/security/access-levels>

### 版本控制

无限制、专用、安全

使用无限制专用存储库对代码进行存储并协作编写代码。 使用 Git 来进行分布式版本控制，或使用 Team Foundation 版本控制 (TFVC) 来实现集中式版本控制。 通过拉取请求和代码审阅轻松地与代码协作。 管理权限和策略以保护你的存储库。

#### 1、在 Team Foundation 版本控制中开发应用程序

参考： <https://msdn.microsoft.com/library/ms181382.aspx>

#### 2、Git命令版本控制开发应用程序

参考： <https://docs.microsoft.com/zh-cn/vsts/git/tutorial/creatingrepo>

创建好Git储存库后，我们可以通以Git命令将项目克隆到本地



使用Git命令推送更新到服务器

1. 选定一个文件夹，初始化git 本地文件夹

git init

2. 使用git clone命令将新建的git 项目信息下载到本地

git clone git %address%

3. 进入git项目的本地文件夹，查看状态

cd 本地文件夹

git status

4. 提交所有的新文件

git add -A (提交所有变化)

git add -u (提交被修改(modified)和被删除(deleted)文件，不包括新文件(new))

git add . (提交新文件(new)和被修改(modified)文件，不包括被删除(deleted)文件)

5. 通过在Git命令窗口中输入以下命令来提交您的更改

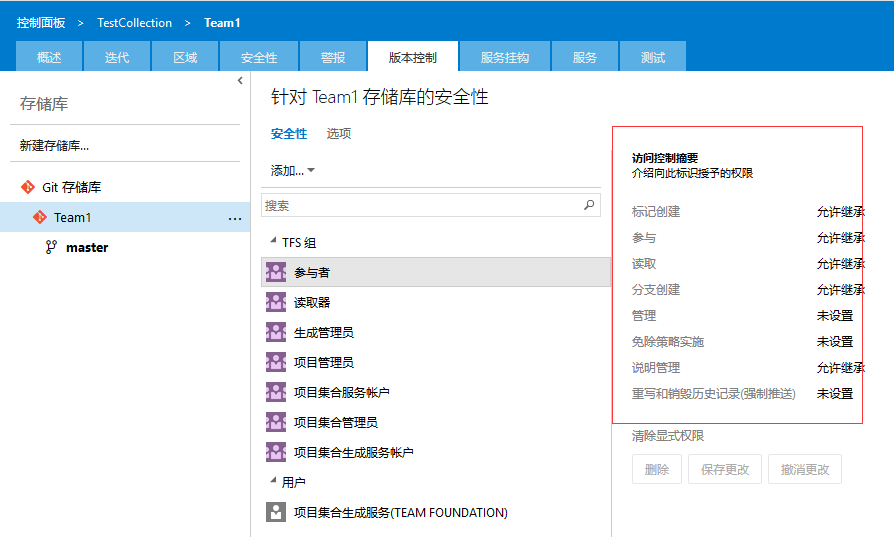
git commit -a -m "first commit"

当使用git提交时，-a 意味着提交所有修改过的文件，-m 指定一个提交消息。

6. 将您的更改推到Git服务器上

git push

选择储存库，可以对其设置访问控制。



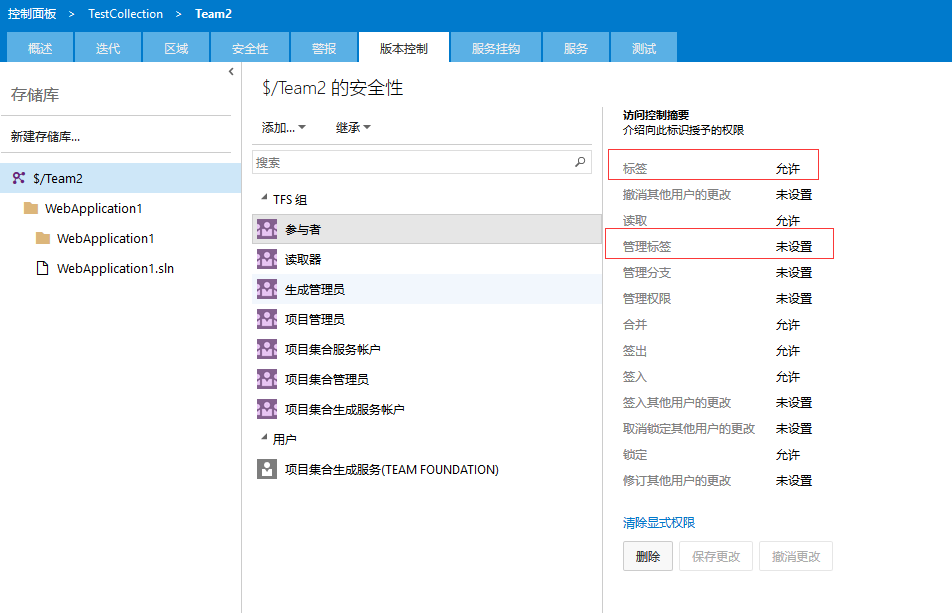
## 源码管理

### 标签

#### FTS 标签

使用标签，您可以获取文件快照，以便将来可以回头参考该快照。 通过使用标签，您可以查看、生成大量文件，甚至可以将这些文件回滚到应用标签时所处的状态。

若要应用标签，您必须将**“标签”**权限设置为**“允许”**。 若要查找、列出或查看标签，您必须将**“读取”**权限设置为**“允许”**。 若要修改或删除标签，您必须是标签的所有者或者将**“管理标签”**权限设置为**“允许”**。

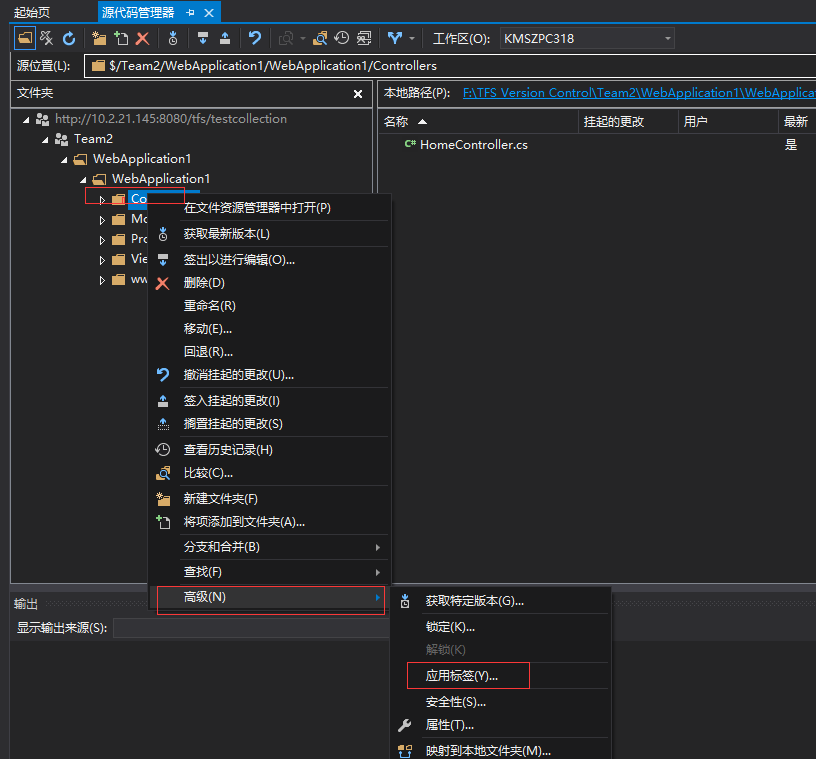


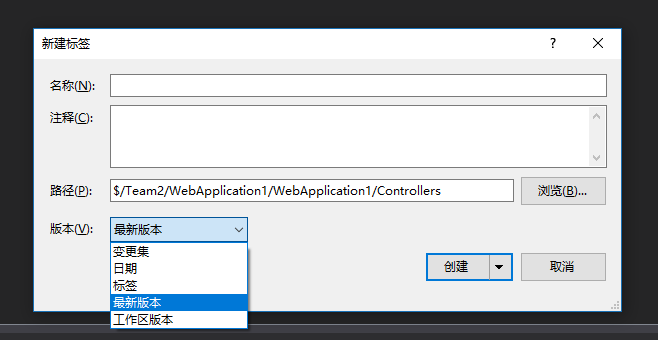
##### 应用标签

1. 在**“源代码管理资源管理器”**中，打开要应用标签的集合、团队项目、分支、文件夹或文件的快捷菜单，然后选择**“高级”**、**“应用标签”**。

此时将显示**“新建标签”**对话框。

1. 在**“名称”**框中键入标签的名称。
2. （可选）在**“注释”**框中键入注释。
3. 在**“版本”**列表中，默认情况下**“最新版本”**处于选中状态。 此选项是最常用的一个选项，但如果您需要标签来以特定版本作为目标，则选择：
   * **变更集**：在**“变更集”**框中指定变更集的编号。 或者，选择省略号 (**…**) 以打开**“查找变更集”**对话框。 有关详细信息，请参阅[查找和查看变更集](https://msdn.microsoft.com/zh-cn/library/ms181408.aspx)。
   * **日期**：在**“日期”**框中指定一个日期，或者从下拉菜单上的日历中选择一个日期。
   * **标签**：指定新标签将基于的现有标签。 在**“标签”**框中键入标签名称，或选择省略号 (**…**) 以打开**“查找标签”**对话框。 有关更多信息，请参见[查找、编辑和移除标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "find_label)。
   * **工作区版本**：为工作区中的版本创建标签。
4. 如果对设置感到满意，请执行以下步骤之一：
   * 若要应用标签，请选择**“创建”**。
   * 若要应用标签，然后在应用后进行修改，请选择**“创建并编辑”**。 这会应用标签，并且随后会显示**“标签”**窗口。 有关如何使用**“标签”**窗口编辑标签的更多信息，请参见[使用“标签”窗口处理标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "label_window)。
5. 当应用使用了已存在的名称的标签时，系统会警告您，并询问您是否继续。 如果选择**“确定”**继续，则未包含在标签应用路径中的所有项将不再具有标签。





##### 重命名标签

当不能修改现有标签的名称时，可以轻松地将新标签应用于另一标签所应用到的那些文件和版本。

基于现有标签应用标签

1. 在源代码管理资源管理器中，打开项目集合节点（根节点）的快捷菜单，然后选择**“高级”**、**“应用标签”**。

此时将显示**“新建标签”**对话框。

1. 在**“名称”**框中键入标签的名称。
2. （可选）在**“注释”**框中键入注释。
3. 在**“版本”**列表中选择**“标签”**。

此时将显示**“标签”**框。

1. 执行以下步骤之一：
   * 键入现有标签的名称。
   * 如果您不确定现有标签的名称，请选择**“标签”**框旁带有省略号 (…) 的浏览按钮。

将出现**“查找标签”**对话框。

使用**“查找标签”**对话框中的选项可对标签列表进行筛选，并查找要应用新标签的现有标签。（有关更多信息，请参见[查找、编辑和移除标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "find_label)）。

1. 选择**“创建”**。
2. （可选）创建新标签后，可以移除旧标签。有关更多信息，请参见[查找、编辑和移除标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "find_label)。

##### 列出、查找、查看、编辑和移除标签

可以使用两种方法列出和查找标签：**“查找标签”**对话框和**“历史记录”**窗口中的**“标签”**选项卡。 找到所需的标签后，即可查看、编辑、移除该标签或处理应用该标签的文件。

使用“查找标签”对话框查找和处理标签

1. 在源代码管理资源管理器中的菜单栏上，依次选择**“文件”**、**“源代码管理”**、**“查找”**、**“查找标签”**。

将出现**“查找标签”**对话框。

1. （可选）通过执行以下一个或多个步骤，缩小搜索范围：
   * 在**“名称”**文本框中，键入标签的名称。
   * 在**“项目”**列表中，选择包含要查找的标签的团队项目。
   * 在**“所有者”**文本框中，键入标签所有者的姓名或别名。
2. 选择**“查找”**。

此时**“结果”**列表会显示符合所输入的搜索条件的标签。

1. （可选）执行以下步骤之一：
   * 若要查看或编辑标签，请选择标签，然后选择**“编辑”**。

此时将显示**“标签”**窗口。 有关更多信息，请参见[使用“标签”窗口处理标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "label_window)。

* + 若要移除标签，请选择标签，然后选择**“移除”**。
  + 选择**“关闭”**。

有时您可能不太确定标签的名称，但知道标签已应用于处于版本控制下的特定项。 下面的过程解释如何在此情况中查找然后处理标签。

列出并处理应用于特定项的标签

1. 在源代码管理资源管理器中，打开要查找的标签所应用于的项的快捷菜单，然后选择**“查看历史记录”**。

此时将显示**“历史记录”**窗口。

1. 选择**“标签”**选项卡以显示应用于此项的标签的列表。
2. （可选）执行以下步骤之一：
   * 若要查看或编辑标签，请打开标签的快捷菜单，然后选择**“编辑标签”**。

这将显示“标签”窗口。 有关如何使用此窗口的信息，请参见[使用“标签”窗口处理标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "label_window)。

* + 若要移除标签，请打开标签的快捷菜单，然后选择**“移除标签”**。
  + 打开标签的快捷菜单，然后选择其他操作，例如**“获取此版本”**和**“比较”**。

有关如何使用“历史记录”窗口中的“标签”选项卡的更多信息，请参见[获取某项的历史记录](https://msdn.microsoft.com/zh-cn/library/ms245475.aspx)。

##### 将文件的带标签版本下载到工作区

对一组文件创建标签类似于获取这些文件在某一时间点的情况的快照。 在获取此快照的数周或数月后，您可能需要按照这些文件在该快照中存在的情况进行查看或处理。 下面的过程解释如何将一组文件的带标签版本下载到工作区。

将文件的带标签版本下载到工作区

1. 在源代码管理资源管理器中，打开包含下载项的项目集合、团队项目、分支或文件夹的快捷菜单，然后选择**“高级”**、**“获取特定版本”**。

这将显示**“获取”**对话框。

1. 从**“类型”**列表中，选择**“标签”**，然后在**“标签”**框中键入标签名称，或选择省略号 (**…**) 以打开**“查找标签”**对话框。 有关如何使用此对话框的更多信息，请参见[查找、编辑和移除标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "find_label)。
2. 查看**“获取”**对话框中的其他选项，并根据需要选择任意选项。
3. 选择**“获取”**。

##### 使用“标签”窗口处理标签

通过上述任一过程打开“标签”窗口后，即可开始处理标签。 在窗口顶部是一个显示注释的框，可以根据需要进行编辑。

可以使用“标签”窗口执行以下任务：

* 向标签中添加项
* 从标签中移除项
* 更改文件的版本
* 对项执行其他操作

向标签中添加项

1. 在**“标签”**窗口中，选择**“添加项”**。

将出现**“选择项版本”**对话框。

1. 在**“选择项版本”**对话框中，浏览至要添加的项，然后选择该项。

|  |
| --- |
| **System_CAPS_tip提示** |
| 可以按住 Ctrl 或 Shift 键，然后选择多个项。 |

1. （可选）如果需要标签来以特定版本作为目标，请从**“版本”**列表中进行选择：
   * **最新版本**：标签将应用于处于版本控制下的最新版本。
   * **变更集**：在**“变更集”**框中指定变更集的编号。 或者，选择省略号 (**…**) 以打开**“查找变更集”**对话框。 有关详细信息，请参阅[查找和查看变更集](https://msdn.microsoft.com/zh-cn/library/ms181408.aspx)。
   * **日期**：在**“日期”**框中指定一个日期，或者从下拉菜单上的日历中选择一个日期。
   * **标签**：指定新标签将基于的现有标签。 在**“标签”**框中键入标签名称，或选择省略号 (**…**) 以打开**“查找标签”**对话框。 有关更多信息，请参见[查找、编辑和移除标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "find_label)。
   * **工作区版本**：为工作区中的版本创建标签。
2. 选择**“确定”**。
3. 选择**“文件”**、**“保存”**。

从标签中移除项

1. 在**“标签”**窗口中，找到要从此标签中移除的集合、团队项目、分支、文件夹或文件。
2. 打开该项的快捷菜单，然后选择**“从标签中移除”**。

该标签即会从该项中移除。 如果该项是项目集合、团队项目、分支或文件夹，则也将从其包含的所有项中移除该标签。

1. 选择**“文件”**、**“保存”**。

更改文件的版本

1. 在**“标签”**窗口中找到要修改的文件。
2. 打开该文件的快捷菜单，然后选择**“更改已设置标签的版本”**。
3. 这将显示**“更改已设置标签的版本”**对话框。
4. 选择：
   * **最新版本**：将标签应用于处于版本控制下的最新版本。
   * **变更集**：在**“变更集”**框中指定变更集的编号。 或者，选择省略号 (**…**) 以打开**“查找变更集”**对话框。 有关详细信息，请参阅[查找和查看变更集](https://msdn.microsoft.com/zh-cn/library/ms181408.aspx)。
   * **日期**：在**“日期”**框中指定一个日期，或者从下拉菜单上的日历中选择一个日期。
   * **标签**：指定新标签将基于的现有标签。 在**“标签”**框中键入标签名称，或选择省略号 (**…**) 以打开**“查找标签”**对话框。 有关更多信息，请参见[查找、编辑和移除标签](https://msdn.microsoft.com/zh-cn/library/ms181439.aspx" \l "find_label)。
   * **工作区版本**：为工作区中的版本创建标签。
5. 选择**“文件”**、**“保存”**。

对带标签项执行其他操作

当在“标签”窗口中打开某个项的快捷菜单时，还可以选择下列命令之一：

* **查看**（仅限文件）
* [查看历史记录](https://msdn.microsoft.com/zh-cn/library/ms245475.aspx)
* [比较](https://msdn.microsoft.com/zh-cn/library/bb385990.aspx)（仅限文件）
* [批注](https://msdn.microsoft.com/zh-cn/library/bb385979.aspx)（仅限文件）
* [获取此版本](https://msdn.microsoft.com/zh-cn/library/ms181387.aspx)

#### Git 打标签

像其他版本控制系统（VCS）一样，Git 可以给历史中的某一个提交打上标签，以示重要。 比较有代表性的是人们会使用这个功能来标记发布结点（v1.0 等等）。 在本节中，你将会学习如何列出已有的标签、如何创建新标签、以及不同类型的标签分别是什么。

##### 列出标签

在 Git 中列出已有的标签是非常简单直观的。 只需要输入 git tag：

$ git tag

v0.1

v1.3

这个命令以字母顺序列出标签；但是它们出现的顺序并不重要。

你也可以使用特定的模式查找标签。 例如，Git 自身的源代码仓库包含标签的数量超过 500 个。 如果只对 1.8.5 系列感兴趣，可以运行：

$ git tag -l 'v1.8.5\*'

##### 创建标签

Git 使用两种主要类型的标签：轻量标签（lightweight）与附注标签（annotated）。

一个轻量标签很像一个不会改变的分支 - 它只是一个特定提交的引用。

然而，附注标签是存储在 Git 数据库中的一个完整对象。 它们是可以被校验的；其中包含打标签者的名字、电子邮件地址、日期时间；还有一个标签信息；并且可以使用 GNU Privacy Guard （GPG）签名与验证。 通常建议创建附注标签，这样你可以拥有以上所有信息；但是如果你只是想用一个临时的标签，或者因为某些原因不想要保存那些信息，轻量标签也是可用的。

##### 附注标签

在 Git 中创建一个附注标签是很简单的。 最简单的方式是当你在运行 tag 命令时指定 -a 选项：

$ git tag -a v1.4 -m 'my version 1.4'

$ git tag

-m 选项指定了一条将会存储在标签中的信息。 如果没有为附注标签指定一条信息，Git 会运行编辑器要求你输入信息。

通过使用 git show 命令可以看到标签信息与对应的提交信息：

$ git show v1.4

输出显示了打标签者的信息、打标签的日期时间、附注信息，然后显示具体的提交信息。

##### 轻量标签

另一种给提交打标签的方式是使用轻量标签。 轻量标签本质上是将提交校验和存储到一个文件中 - 没有保存任何其他信息。 创建轻量标签，不需要使用 -a、-s 或 -m 选项，只需要提供标签名字：

$ git tag v1.4-lw

$ git tag

这时，如果在标签上运行 git show，你不会看到额外的标签信息。 命令只会显示出提交信息：

$ git show v1.4-lw

##### 后期打标签

你也可以对过去的提交打标签。 假设提交历史是这样的：

$ git log --pretty=oneline

现在，假设在 v1.2 时你忘记给项目打标签，也就是在 “updated rakefile” 提交。 你可以在之后补上标签。 要在那个提交上打标签，你需要在命令的末尾指定提交的校验和（或部分校验和）:

$ git tag -a v1.2 9fceb02

可以看到你已经在那次提交上打上标签了：

$ git tag

$ git show v1.2

##### 共享标签

默认情况下，git push 命令并不会传送标签到远程仓库服务器上。 在创建完标签后你必须显式地推送标签到共享服务器上。 这个过程就像共享远程分支一样 - 你可以运行 git push origin [tagname]。

$ git push origin v1.5

如果想要一次性推送很多标签，也可以使用带有 --tags 选项的 git push 命令。 这将会把所有不在远程仓库服务器上的标签全部传送到那里。

$ git push origin --tags

现在，当其他人从仓库中克隆或拉取，他们也能得到你的那些标签。

##### 检出标签

在 Git 中你并不能真的检出一个标签，因为它们并不能像分支一样来回移动。 如果你想要工作目录与仓库中特定的标签版本完全一样，可以使用 git checkout -b [branchname] [tagname] 在特定的标签上创建一个新分支：

$ git checkout -b test v2.0.0

当然，如果在这之后又进行了一次提交，version2 分支会因为改动向前移动了，那么 version2 分支就会和 v2.0.0 标签稍微有些不同，这时就应该当心了。

之后可以分支方式签出标签

$ git checkout test

签出原master分支

$ git checkout master

### 分支

#### FTS分支

分支是一种允许一个文件集合分化到两个或更多分叉路径中的功能。当团队必须维护两个或更多相似基本代码（发布了一个产品且必须开始下一个版本的工作时会出现这种情况）时，经常使用分支。

Team Foundation 源代码管理 中的分支操作与文件系统复制操作相似。分支操作将保留您正进行分支的文件夹和文件的历史记录，并允许将旧文件中的更改合并到新文件中。分支操作可以通过源代码管理资源管理器执行。

分支操作有一个源和一个目标。源提供要进行分支的文件或文件夹的名称和版本，而目标提供目标文件或文件夹的位置。如果指定的目标是服务器中已存在的文件夹的名称，将在该文件夹中创建实际的目标；否则，实际目标就是指定的目标。

分支操作是在 Team Foundation 源代码管理 中创建了一个挂起的更改操作。在将变更集内挂起的更改提交到服务器之前，并不会真正创建分支。

将分支签入服务器之前，可以修改、重命名、删除和撤消文件及文件夹。

文件夹或文件必须处于未删除状态才能进行分支。对文件树进行分支不会导致同时分支该树中已删除的项。以后，如果在源树中撤消删除已删除的项，则合并操作可以将该文件分支到目标分支。

##### 分支文件和文件夹

分支是一项 Team Foundation 源代码管理功能，用于基于现有的文件夹或文件创建新的文件夹或文件。使用分支的原因有多种，例如，可能中断生成的维护版本和更改。

1. 在“视图”菜单上单击“其他窗口”，然后单击“源代码管理资源管理器”。
2. 在源代码管理资源管理器中，定位到要分支的文件夹或文件，单击鼠标右键，然后单击“分支”。
3. 在“分支”对话框的“目标”文本框中，修改新分支的位置和名称。您也可以单击“浏览...”来找到一个位置。
4. 在“分支起源版本”部分的“依据”列表框中，单击创建新分支所依据的源版本。
5. （可选）选择“创建新分支的本地工作副本”选项，该选项将在本地工作区上创建源代码的副本。如果不需要本地副本，请不要选中该复选框，这样可以避免将大量选定项下载到您的计算机上，从而提高性能。
6. 单击“确定”；会创建新分支，并将它显示在源代码管理资源管理器中。
7. （可选）如果选择了“创建新分支的本地工作副本”选项，将出现“浏览文件夹”窗口。请浏览文件夹，或者单击“新建文件夹”并指定要同步到源文件夹的新文件夹。单击“确定”。

一个分支就是一个挂起的更改。为了使其他团队成员能够查看分支，您必须执行签入。

##### 合并文件和文件夹

合并是将两个不同分支中的更改结合在一起的过程。合并操作获取源分支中已发生的更改，并将这些更改集成到目标分支中。合并操作集成源分支中所有类型的更改，其中包括更改名称、编辑文件、添加文件、删除文件以及撤消删除更改。如果同时在源分支和目标分支中修改了项，则系统将提示解决冲突。在合并期间执行的操作

在合并期间，会执行下列操作：

* 合并将标识源分支中所有已添加的文件或文件夹，然后尝试添加目标分支中的相应项。如果从源分支添加的项与已添加到目标分支的项共享同一名称，则可能会发生命名空间冲突。Team Foundation 不处理解决此类型冲突的事宜，但会记录一条错误消息。
* 合并操作将查看源分支中也存在于目标分支内的每个项的历史记录。对于每个项，如果在源分支中所做的更改并不存在于目标分支中，则将这些更改合并到目标分支中。如果已在目标分支上修改了该项，则将会检测到冲突。在合并操作过程中，可以选择合并项的特定版本或所有更改

合并操作：

1. 在源代码管理资源管理器中，右击要与另一个团队项目、文件夹或文件版本合并的团队项目、文件夹或文件版本，并选择“合并”。
2. 在“源代码管理合并向导”的“源分支”框中，键入团队项目源分支的名称，或者单击“浏览”以定位到该分支。
3. 在“目标分支”下拉列表中，选择源分支要合并到的所需目标团队项目分支。
4. 通过选择“对特定版本的所有更改”或“所选变更集”选项，选择要合并到目标分支的更改类型，然后单击“下一步”。
   * 如果在“选择合并操作的源分支和目标分支”步骤中选择了“对特定版本的所有更改”，则选择要使用的源项的版本。

|  |  |
| --- | --- |
| 变更集 | 通过指定变更集版本执行合并。 |
| 日期 | 通过指定日期版本执行合并。 |
| 标签 | 通过指定标签版本执行合并。 |
| 最新版本 | 通过指定最新版本执行合并。 |
| 工作区 | 通过指定工作区版本执行合并。 |

* + 如果在“选择合并操作的源分支和目标分支”步骤中选择了“所选变更集”，则选择要合并的变更集。
  + 如果可能，请选择“对特定版本的所有更改”，因为这样做将减少以后在合并调用中发生冲突的可能性。

1. 单击“下一步”，然后在“执行合并操作”步骤中单击“完成”。
2. 如果发生合并冲突，必须解决这些冲突。

若要执行这些过程，您必须将源树中项的“读取”权限和目标树中项的“签出”权限都设置为“允许”。如果要重命名目标树中的项，您必须将源树和目标树的“签出”权限都设置为“允许”。如果涉及锁定，则您必须将“锁定”权限设置为“允许”

#### Git分支

##### 新建分支

要新建并切换到该分支，运行 git checkout 并加上 -b 参数：

$ git checkout -b test-branch

这相当于执行下面这两条命令：

$ git branch test-branch

$ git checkout test-branch

##### 切换分支

要切换分支，运行 git checkout ：

$ git checkout test-branch

##### 删除分支

要删除分支，运行 git checkout 并加上 -d 参数 ：

$ git branch -d test-branch

##### 合并分支

运行 git merge 命令指定要合并进来的分支，将test-branch合并到master：

$ git checkout master

$ git merge test-branch

##### 解决冲突

有时候合并操作并不会如此顺利。如果在不同的分支中都修改了同一个文件的同一部分，Git 就无法干净地把两者合到一起（这种问题只能由人来裁决）。

Git 会在有冲突的文件里加入标准的冲突解决标记，可以通过它们来手工定位并解决这些冲突。可以看到此文件包含类似下面这样的部分：

<<<<<<< HEAD

Master content

=======

Branch content

>>>>>>> test-branch

可以看到 ======= 隔开的上半部分，是 HEAD（即 master 分支，在运行 merge 命令时所切换到的分支）中的内容，下半部分是在 test-branch分支中的内容。解决冲突的办法无非是二者选其一或者由你亲自整合到一起。而且还要删除 <<<<<<<，======= 和 >>>>>>>这些行。在解决了所有文件里的所有冲突后，运行 git add 将把它们标记为已解决状态。实际上就是来一次快照保存到暂存区域，一旦暂存就表示冲突已经解决。

## 持续集成

版本、程序包、测试、发布、重复

通过持续集成 (CI) 版本在早期发现质量问题。 使用发布管理自动化跟踪你的所有部署。 使用我们广泛的测试工具组来维护较高的质量。 通过重复使用代码和模块更快地交付程序包管理。

## Java 支持

对任何平台进行开发

使用你选择的 IDE：Eclipse、IntelliJ、Android Studio、Visual Studio Code 等。 从你的 Git、Subversion 和 TFVC 存储库使用 Ant、Maven 和 Gradle 构建你的代码。 借助本机功能或通过与系统（如 Jenkins）的集成来实现 CI/CD。 支持跨平台和移动语言，包括：C++、PHP、Python、Go、Swift 等更多语言。