﻿== CMD\_DESCRIPTION\_ACL ==

设置对象权限。

== CMD\_USAGE\_ACL ==

用法：

cm ^acl (--^user=<用户名> | --^group=<组名称>)

(-^allowed|-^denied|-^overrideallowed|-^overridedenied=+|-<权限>[,...])[,...]

<对象规格>

--^user 用户名。

--^group 组名称。

-^allowed 启用指定的一项或多项权限。使用

逗号对各个权限进行分隔。（使用 'cm ^showpermissions'

可显示所有可用权限。）

-^denied 拒绝指定的一项或多项权限。使用

逗号对各个权限进行分隔。（使用 'cm ^showpermissions'

可显示所有可用权限。）

-^overrideallowed 覆盖允许的一项或多项权限。使用

逗号对各个权限进行分隔。（使用 'cm ^showpermissions'

可显示所有可用权限。）

-^overridedenied 覆盖拒绝的一项或多项权限。使用

逗号对各个权限进行分隔。（使用 'cm ^showpermissions'

可显示所有可用权限。）

对象规格 要设置权限的对象。

此命令的有效对象为：

存储库服务器、存储库、分支、变更集、标签、项

和属性。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

安全路径的特殊用法：

cm ^acl [(--^user=<用户名> | --^group=<组名称>)

(-^allowed|-^denied|-^overrideallowed|-^overridedenied=+|-<权限>[,...])[,...]]

[--^delete] [--^branches=[+ | -]<分支>[,...]]

<规格>

--^delete 删除安全路径。

请参阅“备注”以了解更多信息。

--^branches 将安全路径权限设置为一个分支组。

使用逗号对各个分支进行分隔。

（可选）每个分支前面可添加 + 或 -

符号，从而指定在编辑时必须在列表中添加

还是删除某个分支。

请参阅“备注”以了解更多信息。

规格 要设置权限的安全路径。

== CMD\_HELP\_ACL ==

为了配置权限，需要了解 Plastic SCM 安全机制的工作原理。

请查看《安全指南》以了解权限的工作原理：

https://www.plasticscm.com/download/help/securityguide

备注：

此命令设置用户或组对于指定对象、存储库、分支、标签

和/或服务器路径的权限。

对象规格：

（使用 'cm ^help ^objectspec' 可了解如何指定对象。）

'^acl' 命令使用一种特殊类型的规格：安全路径。

- 安全路径规格：

^path:server\_path[#tag]

示例：^path:/src/foo.c

^path:/doc/pdf

^path:/doc/pdf#documents

权限操作：

使用 -^allowed 和 -^denied 指定要设置的权限。

使用 -^overrideallowed 和 -^overridedenied 参数指定

要覆盖的权限。

每个操作都需要一个权限列表（以逗号分隔）。

权限名称：

每个权限名称前面都带有 + 或 - 符号。

+ 符号表示设置权限，而 - 符号表示清除权限。

要查看对象的权限，请使用 'cm ^showacl' 命令。

覆盖的权限：

使用 -^overrideallowed 和 -^overridedenied

来覆盖权限可以绕过继承。

这对于绕过在存储库或服务器级别设置的权限

很有帮助。

示例：

cm ^acl --^user=vio -^allowed=+^ci -^overrideallowed=+^ci ^br:qa@test

（允许用户 'vio' 在存储库 'test' 上的分支 'qa' 上签入，

即使该用户的权限在存储库级别被拒绝也是如此。）

服务器路径权限（又称“安全路径”）：

- 允许为给定的服务器路径指定权限。

- 在签入操作期间会检查这些权限。

- 也可以在更新操作期间检查这些权限，

并可以将这些权限用作一种防止某些目录和文件

下载到工作区的方法。

- 对于要签入的每一项，服务器都会尝试将项路径

与安全路径匹配。如果匹配，则签入操作

将检查是否可以签入该项。

可为安全路径定义的权限

如下：

'^ci'、'^change'、'^add'、'^move'、'^rm'、'^read'

对于任何涉及的项，如果权限检查不成功，

则检入操作将回滚。

要将安全路径权限设置为一个分支组，请使用

--^branches 选项。

示例：

cm ^acl --^user=jo -^denied=+^ci ^path:/src#rule0 --^branches=main,main/rel0

要编辑与安全路径关联的 ACL，该标记很有用。

示例：

cm ^acl --^user=jo -^denied=+^rm ^path:/src#rule0

（如果没有标记，则需要再次指定

分支列表。）

安全路径的分支列表是可以编辑的。

示例：

cm ^acl ^path:/src#rule0 --^branches=-main,+main/rel1

（从列表中删除 'main' 并添加 'main/rel1'。）

要删除安全路径，请使用 --^delete 参数。

示例：

cm ^acl --^user=jo --^delete ^path:/src#rule0

继承：

继承是 Plastic SCM 3.0 时期的一种选项。

这是高级选项，但几乎已弃用。

该选项允许对象从任何其他对象继承权限，

从而覆盖默认的继承关系。

使用选项 -^cut 切断继承链。

使用选项 -^cutncpy 切断并复制当前继承的

权限。（这个选项受到 Windows 文件系统权限的启发，

在 Windows 文件系统中可以切断继承关系但保留实际权限。）

-^inherit 选项允许用户从对象规格继承。

示例：'-^inherit=对象规格'

示例：

cm ^acl --^user=danipen -^denied=+^ci ^rep:core

（拒绝用户 'danipen' 在存储库 'core' 上签入。）

cm ^acl --^group=developers -^allowed=+^view,-^read -^denied=+^chgperm ^br:main

（该命令向 'main' 分支中的 'developers' 组

授予查看权限、清除读取权限并拒绝 chgperm 权限。）

安全路径示例：

cm ^acl --^group=devs -^denied=+^ci ^path:/server#rel --^branches=main,main/2.0

（对于在分支 'main' 和 'main/2.0' 中与 '/server' 匹配的任何路径，

该命令拒绝向 'devs' 组授予签入权限。创建标记 '#rel'

是为了稍后能够进行引用。）

cm ^acl ^path:/server#rel --^branches=-/main,+/main/Rel2.1

（更新具有 'rel' 标记的安全路径 '/server'，

删除 'main' 分支，并将分支 'main/Rel2.1' 添加到

安全路径所应用到的分支组。就前面的示例而言，

现在分支列表将包含 'main/Rel2.1' 和 'main/2.0'。）

cm ^acl --^user=vsanchezm -^allowed=-^read -^overrideallowed=+^read ^path:/doc

（删除授权给 'vsanchezm' 的 '^read' 权限，在 '/doc' 路径中对其进行覆盖。）

== CMD\_DESCRIPTION\_ACTIVATEUSER ==

激活许可的用户。

== CMD\_USAGE\_ACTIVATEUSER ==

用法：

cm ^activateuser | ^au <用户名>[ ...][--^server=<存储库服务器规格>]

用户名 要激活的一个或多个用户名。使用双引号 (" ")

指定包含空格的用户名。使用空格

对各个用户名进行分隔。

选项：

--^server=<存储库服务器规格> 激活指定服务器中的用户。

如果未指定服务器，则在 client.conf 文件

中的默认服务器中执行命令。

（使用 'cm ^help ^objectspec' 可进一步了解

存储库服务器规格。）

== CMD\_HELP\_ACTIVATEUSER ==

备注：

要激活用户，必须已事先禁用该用户。

默认情况下，用户第一次在 Plastic SCM 中执行写操作时

会被激活。仅当未超过最大用户数时，

才会自动激活用户。

有关停用 Plastic SCM 用户的更多信息，

请参阅 'cm ^help ^deactivateuser' 命令。

示例：

cm ^activateuser john

cm ^activateuser david "mary collins"

cm ^au peter --^server=localhost:8087

== CMD\_DESCRIPTION\_ADD ==

向版本控制中添加项。

== CMD\_USAGE\_ADD ==

用法：

cm ^add [-^R | -^r | --^recursive] [--^silent] [--^ignorefailed]

[--^skipcontentcheck] [--^coparent] [--^filetypes=<文件>] [--^noinfo]

[--^format=<格式字符串>] [--^errorformat=<格式字符串>]

<项路径>[ ...]

项路径 要移动的一个或多个项。使用双引号 (" ") 指定

包含空格的路径。使用空格对各个项进行分隔。

使用 \* 添加当前目录的所有内容。

选项：

-^R -^r --^recursive 以递归方式添加项。

--^silent 不显示任何输出。

--^ignorefailed 如果无法添加某一项，则添加操作将在没有

这一项的情况下继续进行。注意：如果无法添加

某个目录，则不会添加目录中的内容。

--^skipcontentcheck 如果不足以根据扩展名将文件设置为

文本或二进制，则会将该文件设置为二进制，

而不是检查内容来检测类型。这样做

是为了在进行大量签入时提高性能。

--^coparent 对要添加的项的父级进行签出。

--^filetypes 要使用的 filetypes 文件。请查看以下链接以了解

更多信息：

http://blog.plasticscm.com/2008/03/custom-file-types.html

--^noinfo 不打印进度信息。

--^format 检索特定格式的输出消息。请查看

示例以了解更多信息。

--^errorformat 检索特定格式的错误消息

（如果有）。请查看示例以了解更多信息。

== CMD\_HELP\_ADD ==

备注：

添加项的要求：

- 要添加的项的父目录必须已事先添加。

从 stdin 读取输入：

'^add' 命令可从 stdin 读取路径。为此，请传递一个破折号

"-"。

示例：cm ^add -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要添加的文件。

示例：

dir /S /B \*.c | cm ^add -

（在 Windows 中，添加工作区中的所有 .c 文件。）

示例：

cm ^add file1.txt file2.txt

（添加 'file1.txt' 和 'file2.txt' 项。）

cm ^add c:\workspace\file.txt

（添加路径 'c:\workspace' 中的 'file.txt' 项。）

cm ^add -^R c:\workspace\src

（以递归方式添加 'src'。）

cm ^add -^R \*

（以递归方式添加当前目录的所有内容。）

cm ^add -^R \* --^filetypes=filetypes.conf

（以递归方式添加当前目录的所有内容，使用

'filetypes.conf' 根据文件的扩展名为文件分配类型，

而不是检查文件内容。）

cm ^add --^coparent c:\workspace\dir\file.txt

（将 'file.txt' 添加到源代码管理中，并执行 'dir' 的签出。）

cm ^add -^R \* --^format="ADD {0}" --^errorformat="ERR {0}"

（以递归方式添加当前目录的所有内容，

对于成功添加的文件，打印 '^ADD <项>'，而对于无法添加的项，

则打印 '^ERR <项>'。）

== CMD\_USAGE\_ADDIGNOREPATTERN ==

用法：

cm ^addignorepattern <模式>[ ...]

[--^workspace=<工作区路径> | --^allworkspaces] [--^remove]

== CMD\_DESCRIPTION\_ADMIN ==

在服务器上执行管理命令。

== CMD\_USAGE\_ADMIN ==

用法：

cm ^admin <命令> [选项]

可用命令：

^readonly

要获取有关每条命令的更多信息，请运行：

cm ^admin <命令> --^usage

cm ^admin <命令> --^help

== CMD\_HELP\_ADMIN ==

备注：

只有服务器管理员才能执行管理命令。

示例：

cm ^admin ^readonly ^enter

cm ^admin ^readonly ^status

== CMD\_DESCRIPTION\_ADMIN\_READONLY ==

启用/禁用服务器只读模式。

== CMD\_USAGE\_ADMIN\_READONLY ==

用法：

cm ^admin ^readonly (^enter | ^leave | ^status) [<服务器>]

操作：

^enter 服务器进入只读模式。写操作将被拒绝。

^leave 服务器退出只读模式。

^status 显示服务器只读模式状态。

选项：

服务器 在指定服务器（服务器:端口）中执行命令。（使用

'cm ^help ^objectspec' 可进一步了解服务器规格。）

如果未指定服务器，则该命令适用于当前工作区的

服务器。

如果当前路径不在工作区中，则该命令适用于

client.conf 配置文件中定义的默认服务器。

== CMD\_HELP\_ADMIN\_READONLY ==

备注：

只有服务器管理员才能进入服务器只读模式。

示例：

cm ^admin ^readonly ^enter diana:8086

cm ^admin ^readonly ^leave

== CMD\_DESCRIPTION\_ANNOTATE ==

显示文件每一行上次修改所在的变更集及其作者。

== CMD\_USAGE\_ANNOTATE ==

用法：

cm ^annotate | ^blame <规格>[ ...]

[--^format=<格式字符串>]

[--^ignore=(^eol | ^whitespaces | ^"eol&whitespaces" | ^none)]

[--^dateformat=<日期格式字符串>]

[--^encoding=<名称>]

[--^stats]

[--^repository=<存储库规格>]

规格 要批注的文件的规格。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

使用双引号 (" ") 指定包含空格的路径。

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^ignore 设置指定的比较方法。

请参阅“备注”以了解更多信息。

--^dateformat 设置输出格式以打印日期。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^stats 显示统计信息。

--^repository 指定用于计算批注的存储库

规格。默认情况下，此命令使用的

存储库存储了工作区中已加载的

修订存储库。（使用 'cm ^help ^objectspec' 可进一步

了解存储库规格。）

== CMD\_HELP\_ANNOTATE ==

备注：

无法批注二进制文件。

--^ignore 选项：

^none 检测行尾和空格差异。

^eol 忽略行尾差异。

^whitespaces 忽略空格差异。

^"eol&whitespaces" 忽略行尾和空格差异。

--^format 选项：

此命令的输出参数如下：

{^owner} 上次更改行的用户。

{^rev} 行的源修订规格。

{^content} 行内容。

{^date} 签入行的日期。

{^comment} 行的源修订的注释。

{^changeset} 行的源修订的变更集。

{^line} 文件中的行号。

{^id} 项 ID。

{^parentid} 项的父级 ID。

{^rep} 项的存储库。

{^branch} 行的源修订的分支。

{^ismergerev} 是否在合并中创建了行的修订。

--^dateformat:

指定打印日期的输出格式。

请参阅以下网址中指明的受支持格式：

https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-date-and-time-format-strings

--^repository:

从远程存储库检索数据。适合用于分布式

场景。

示例：

cm ^blame c:\workspace\src --^ignore=^"eol&whitespaces" --^encoding=utf-8

cm ^annotate c:\workspace\file.txt --^ignore=^eol

cm ^annotate c:\workspace\file.txt --^format="{^owner} {^date, 10} {^content}"

（依次写入 owner 字段、一个空格、date 字段（右对齐）、

一个空格以及 content 字段。）

cm ^blame c:\workspace\file.txt --^format="{^owner, -7} {^comment} {^date}" \

--^dateformat=yyyyMMdd

（依次写入 owner 字段（占用 7 个空格，左对齐）、

一个空格、comment 字段、另一个空格以及结尾的

格式化日期（例如，20170329）。）

cm ^annotate c:\workspace\file.txt --^repository=centralRep@myserver:8084

cm ^blame ^serverpath:/src/client/checkin/Checkin.cs#^cs:73666

（使用服务器路径对变更集 73666 中开始的文件进行批注。）

== CMD\_DESCRIPTION\_APPLY\_LOCAL ==

检查是否有本地更改（本地移动、本地删除和本地更改）

并应用这些更改，以便 Plastic SCM 开始跟踪这些更改。

== CMD\_USAGE\_APPLY\_LOCAL ==

用法：

cm ^applylocal | ^al [--^dependencies] [<项路径>[ ...]]

[--^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

选项：

--^dependencies 将本地更改依赖项添加到要应用

的项。

项路径 要应用的项。使用空格对各个路径进行

分隔。使用双引号 (" ") 指定包含空格

的路径。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，指定

应如何分隔字段。

== CMD\_HELP\_APPLY\_LOCAL ==

备注：

如果未指定 --^dependencies 和 <项路径>，则该操作涉及

工作区中的所有本地更改。

始终以递归方式从给定路径应用。

示例：

cm ^applylocal foo.c bar.c

cm ^applylocal .

（应用当前目录中的所有本地更改。）

cm ^applylocal

（应用工作区中的所有本地更改。）

cm ^applylocal --^machinereadable

（应用工作区中的所有本地更改，并以易于解析的

简化格式打印结果。）

cm ^applylocal --^machinereadable --^startlineseparator=">" \

--^endlineseparator="<" --^fieldseparator=","

（应用工作区中的所有本地更改，并以易于解析的

简化格式打印结果，以指定的字符串作为行的开头和结尾以及

对各字段进行分隔。）

== CMD\_DESCRIPTION\_ARCHIVE ==

将数据存档在外部存储中。

== CMD\_USAGE\_ARCHIVE ==

用法：

cm ^archive | ^arch <修订规格>[ ...][-^c | --^comment=<注释字符串>]

[--^file=<基项文件>]

（从存储库中提取数据并将这些数据存储在外部存储中。）

cm ^archive | ^arch <修订规格>[ ...]--^restore

（将以前存档的修订还原到存储库中。）

修订规格 一个或多个修订规格。可通过 "-" 修饰符从 STDIN

进行读取。（使用 'cm ^help ^objectspec' 可

进一步了解修订规格。）

--^restore 从生成的存档文件中还原以前存档的

数据。

选项：

-^c | --^comment 在要创建的存档存储文件中设置注释。

--^file 新存档数据文件的名称前缀和

（可选）路径。

== CMD\_HELP\_ARCHIVE ==

备注：

此命令从存储库数据库中提取数据并将这些数据存储在

外部存储中，从而节省数据库空间。

此命令还可以将以前存档的修订还原到

存储库数据库中 (--^restore)。

使用 'cm ^help ^objectspec' 可了解如何指定修订规格。

运行此命令的用户必须是 Plastic SCM 服务器管理员

（存储库服务器所有者）才能完成操作。

来自指定修订中的每个数据段都将存储在

不同的文件中，且文件名以 --^file

参数定义的值开头。此参数可以包含完整路径值，其中包括

未来存档文件的前缀，也可以仅包含此前缀值。

存档后可以通过两种方式访问来自指定修订中的

数据：

- 从客户端：客户端将检测数据是否已存档，

并提示用户输入文件的位置。

用户可以通过创建一个名为 externaldata.conf 的

文件（在标准配置文件位置，使用适用于

client.conf 文件的相同规则）来配置外部数据位置，

其中包含已存档数据所在的路径。

- 从服务器：通过这种方式，用户将不必知道数据

是否已存档，因为请求将以透明的方式由服务器

解析。为此，管理员将在服务器目录中创建一个

名为 externaldata.conf 的文件，并在该文件中填充

已存档的卷所在的路径。

要取消存档（还原）一个修订（或一组修订），必须能够

从客户端访问已存档的文件。因此，无法取消存档

服务器正在解析的数据（方法 2），原因是客户端

无法将数据识别为已存档。如果使用了方法 2，

为了成功取消存档，管理员必须首先编辑

externaldata.conf 服务器文件，以便删除对必须取消存档的

已存档文件的访问权限。

设置 PLASTICEDITOR 环境变量可指定用于键入注释的

编辑器。

示例：

cm ^archive bigfile.zip#^br:/main

（在分支 'main' 中存档 'bigfile.zip' 的最后一个修订。）

cm ^archive ^rev:myfile.pdf#^cs:2 -^c="大型 PDF 文件" --^file=c:\arch\_files\arch

（将变更集 2 中 myfile.pdf 的修订存档在 'c:\archived\_files'

文件夹中。存档的文件名将以 'arch' 开头（例如 arch\_11\_56）。）

cm ^find "^revs ^where ^size > 26214400" --^format="{^item}#{^branch}" \

--^nototal | cm ^archive --^comment="volume00" --^file="volume00" -

（将所有大于 25Mb 的文件存档在名称 'volume00' 开头的

文件中。）

cm ^find "^revs ^where ^size > 26214400 ^and ^archived='true'" \

--^format="{^item}#{^branch}" --^nototal | cm ^archive --^restore

（还原所有大于 25Mb 的存档文件。）

== CMD\_DESCRIPTION\_ATTRIBUTE ==

允许用户管理属性。

== CMD\_USAGE\_ATTRIBUTE ==

用法：

cm ^attribute | ^att <命令> [选项]

命令：

^create | ^mk

^delete | ^rm

^set

^unset

^rename

^edit

要获取有关每条命令的更多信息，请运行：

cm ^attribute <命令> --^usage

cm ^attribute <命令> --^help

== CMD\_HELP\_ATTRIBUTE ==

示例：

cm ^attribute ^create 状态

cm ^attribute ^set ^att:status ^br:/main/SCM105 未完成

cm ^attribute ^unset ^att:status ^br:/main/SCM105

cm ^attribute ^delete ^att:status

cm ^attribute ^rename ^att:status "构建状态"

cm ^attribute ^edit ^att:status "CI 管道中任务的状态"

== CMD\_DESCRIPTION\_CHANGELIST ==

对更改列表中的待定更改进行分组。

== CMD\_USAGE\_CHANGELIST ==

用法：

a) 管理更改列表对象：

cm ^changelist | ^clist [--^symlink]

（显示所有更改列表。）

cm ^changelist | ^clist ^add <更改列表名称>

[<更改列表描述>] [--^persistent | --^notpersistent] [--^symlink]

（创建更改列表。）

cm ^changelist | ^clist ^rm <更改列表名称> [--^symlink]

（删除所选的更改列表。如果此更改列表包含待定更改，

这些更改将移至 ^default 更改列表。）

cm ^changelist | ^clist ^edit <更改列表名称> [<操作名称> <操作值>]

[--^persistent | --^notpersistent] [--^symlink]

（编辑所选的更改列表。）

b) 管理给定更改列表的内容：

cm ^changelist | ^clist <更改列表名称> (^add | ^rm) <路径名称>[ ...]

[--^symlink]

（通过添加 ('^add') 或删除 ('^rm') 与给定路径名称匹配的更改

来添加所选的更改列表。使用空格

对各个路径名称进行分隔。使用双引号 (" ") 指定包含空格的

路径。路径的状态必须为 '^Added' 或 '^Checked-out'。）

选项：

更改列表名称 更改列表的名称。

更改列表描述 更改列表的描述。

操作名称 选择 '^rename' 或 '^description' 来编辑

更改列表。

操作值 在编辑更改列表时应用新名称或

新描述。

--^persistent 即使更改列表的内容已被签入或还原，

更改列表也会保留在工作区中。

--^notpersistent （默认值）即使更改列表的内容已被签入

或还原，更改列表也不会保留在

工作区中。

--^symlink 将操作应用于符号链接而不是

目标。

== CMD\_HELP\_CHANGELIST ==

备注：

'^changelist' 命令会处理工作区待定更改列表以及更改列表中

包含的更改。

示例：

cm ^changelist

（显示当前工作区更改列表。）

cm ^changelist ^add 配置更改 "dotConf 文件" --^persistent

（创建一个名为 '配置更改' 和描述为 'dotConf

文件' 的新更改列表，该更改列表在待定更改列表被签入

或还原后将在当前工作区中持久保留。）

cm ^changelist ^edit 配置更改 ^rename 配置文件 --^notpersistent

（编辑名为 '配置更改' 的更改列表，并将该更改列表重命名为

'配置文件'。此外还会将更改列表变为“非持久性”。）

cm ^changelist ^edit 配置更改 --^notpersistent

（编辑名为 '配置更改' 的更改列表，并将该更改列表变为“非持久性”。）

cm ^changelist ^rm 配置文件

（从当前工作区中删除待定更改列表 '配置文件'。）

cm ^changelist 配置文件 ^add foo.conf

（将文件 'foo.conf' 添加到 '配置文件' 更改列表中。）

cm ^changelist 配置文件 ^rm foo.conf readme.txt

（从 '配置文件' 更改列表中删除文件 'foo.conf' 和 'readme.txt'，

并将这些文件移至系统默认更改列表。）

== CMD\_DESCRIPTION\_CHANGESET ==

对变更集执行高级操作。

== CMD\_USAGE\_CHANGESET ==

用法：

cm ^changeset <命令> [选项]

命令：

^move | ^mv

^delete | ^rm

^editcomment | ^edit

要获取有关每条命令的更多信息，请运行：

cm ^changeset <命令> --^usage

cm ^changeset <命令> --^help

== CMD\_HELP\_CHANGESET ==

示例：

cm ^changeset ^move ^cs:15@myrepo ^br:/main/scm005@myrepo

cm ^changeset ^delete ^cs:2b55f8aa-0b29-410f-b99c-60e573a309ca@devData

== CMD\_DESCRIPTION\_CHANGESET\_EDIT\_COMMENT ==

修改变更集的注释。

== CMD\_USAGE\_CHANGESET\_EDIT\_COMMENT ==

用法：

cm ^changeset ^editcomment | ^edit <变更集规格> <新注释>

选项：

变更集规格 要修改注释的目标变更集。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

新注释 要添加到目标变更集的

新注释。

== CMD\_HELP\_CHANGESET\_EDIT\_COMMENT ==

备注：

- 目标变更集规格必须有效。

示例：

cm ^changeset ^editcomment ^cs:15@myrepo "我忘了添加签入详细信息"

cm ^changeset ^edit ^cs:cb11ecdb-1aa9-4f11-8698-dcab14e5885a \

"此注释文本将替代先前的注释文本。"

== CMD\_DESCRIPTION\_CHANGESET\_MOVE ==

将变更集及其所有后代移动到另一个分支。

== CMD\_USAGE\_CHANGESET\_MOVE ==

用法：

cm ^changeset ^move | ^mv <变更集规格> <分支规格>

选项：

变更集规格 要移动到另一个分支的第一个变更集。All

同一分支中的所有后代变更集也将作为

该命令的目标。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

分支规格 存储目标变更集的目标

分支。目标分支需要为空或不存在；

如果目标分支不存在，该命令将创建

目标分支。

（使用 'cm ^help ^objectspec' 可进一步了解分支

规格。）

== CMD\_HELP\_CHANGESET\_MOVE ==

备注：

- 目标变更集规格必须有效。

- 目标分支必须为空或不存在。

- 如果目标分支不存在，则会创建目标分支。

- 合并链接将保持不变，因为分支不会影响它们。

示例：

cm ^changeset ^move ^cs:15@myrepo ^br:/main/scm005@myrepo

cm ^changeset ^move ^cs:cb11ecdb-1aa9-4f11-8698-dcab14e5885a ^br:/hotfix/TL-352

== CMD\_DESCRIPTION\_CHANGESET\_DELETE ==

从存储库中删除变更集。

== CMD\_USAGE\_CHANGESET\_DELETE ==

用法：

cm ^changeset ^delete | ^rm <变更集规格>

选项：

变更集规格 要删除的目标变更集。必须满足

某些特定条件。请参阅“备注”以了解更多信息。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

== CMD\_HELP\_CHANGESET\_DELETE ==

备注：

- 目标变更集必须是其分支中的最后一个变更集。

- 目标变更集不能是任何其他变更集的父级。

- 目标变更集既不能作为合并链接的源，也不能

在间隔合并中作为源。

- 不得将任何标签应用于目标变更集。

- 目标变更集不能是根变更集 ('^cs:0')

示例：

cm ^changeset ^rm ^cs:4525@myrepo@myserver

cm ^changeset ^delete ^cs:cb11ecdb-1aa9-4f11-8698-dcab14e5885a

== CMD\_DESCRIPTION\_CHANGEUSERPASSWORD ==

更改用户密码 (UP)。

== CMD\_USAGE\_CHANGEUSERPASSWORD ==

用法：

cm ^changepassword | ^passwd

== CMD\_HELP\_CHANGEUSERPASSWORD ==

备注：

仅当安全性配置为 UP（用户/密码）时，

此命令才可用。请参阅《管理指南》以了解更多信息。

需要新旧密码。

示例：

cm ^passwd

== CMD\_DESCRIPTION\_CHECKCONNECTION ==

检查与服务器的连接。

== CMD\_USAGE\_CHECKCONNECTION ==

用法：

cm ^checkconnection | ^cc

== CMD\_HELP\_CHECKCONNECTION ==

备注：

- 此命令返回一条消息，指示是否存在与配置的

Plastic SCM 服务器之间的有效连接。

- 此命令检查配置的用户是否有效。此外

还会检查与服务器的版本兼容性。

== CMD\_DESCRIPTION\_CHECKDB ==

检查存储库的完整性。

== CMD\_USAGE\_CHECKDB ==

用法：

cm ^checkdatabase | ^chkdb [<存储库服务器规格> | <存储库规格>]

使用 'cm ^help ^objectspec' 可进一步了解存储库服务器和存储库规格。

== CMD\_HELP\_CHECKDB ==

备注：

- 如果未指定存储库服务器规格和存储库规格，

则将在 client.conf 文件内指定的服务器中执行检查。

示例：

cm ^checkdatabase ^repserver:localhost:8084

cm ^chkdb ^rep:default@localhost:8084

== CMD\_DESCRIPTION\_CHECKIN ==

将更改存储在存储库中。

== CMD\_USAGE\_CHECKIN ==

用法：

cm ^checkin | ^ci [<项路径>[ ...]]

[-^c=<注释字符串> | -^commentsfile=<注释文件>]

[--^all|-^a] [--^applychanged] [--^private] [--^update] [--^symlink]

[--^noshowchangeset]

[--^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

选项：

项路径 要签入的项。使用双引号 (" ")

指定包含空格的路径。使用空格

对各个项路径进行分隔。

使用 . 将签入应用到当前目录。

-^c 将指定的注释应用于在签入操作中创建的

变更集。

-^commentsfile 将指定文件中的注释应用于在签入操作中

创建的变更集。

--^all | -^a 还包括在给定路径上进行了本地更改、移动

和删除的项。

--^applychanged 将签入操作应用于在工作区中

检测到的已更改项以及

签出的项。

--^private 还包括在工作区中检测到的

私有项。

--^update 自动处理更新合并

（如果最终发生此行为）。

--^symlink 将签入操作应用于符号链接而不是

目标。

--^noshowchangeset 不打印结果变更集。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，指定

应如何分隔字段。

== CMD\_HELP\_CHECKIN ==

备注：

- 如果未指定 <项路径>，则签入操作涉及

工作区中的所有待定更改。

- 签入操作始终以递归方式从给定路径应用。

- 要签入项，必须满足以下条件：

- 项必须受源代码管理。

- 如果是私有项（不受源代码管理），则必须有 --^private

标志才能签入该项。

- 必须签出项。

- 如果已更改但未签出项，则除非 <项路径> 是目录

或包含通配符 ('\*')，否则不需要

--^applychanged 标志。

修订内容必须与以前的修订不同才能

签入。

设置 PLASTICEDITOR 环境变量可指定用于键入注释的

编辑器。

从 stdin 读取输入：

'^checkin' 命令可从 stdin 读取路径。为此，请传递一个

破折号 "-"。

示例：cm ^checkin -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要签入的文件。

示例：

dir /S /B \*.c | cm ^checkin --^all -

（在 Windows 中，签入工作区中的所有 .c 文件。）

示例：

cm ^checkin file1.txt file2.txt

（签入已签出的文件 'file1.txt' 和 'file2.txt'。）

cm ^checkin .-^commentsfile=mycomment.txt

（签入当前目录并在 'mycomment.txt' 文件中

设置注释。）

cm ^checkin 链接 --^symlink

（签入 '链接' 文件而不是目标；适用于 UNIX

环境。）

cm ^ci file1.txt -^c="我的注释"

（签入 'file1.txt' 并包含注释。）

cm ^status --^short --^compact --^changelist=pending\_to\_review | cm ^checkin -

（列出名为 'pending\_to\_review' 的更改列表中的路径，并将此列表

重定向到 checkin 命令的输入。）

cm ^ci .--^machinereadable

（签入当前目录，并以易于解析的简化格式

打印结果。）

cm ^ci .--^machinereadable --^startlineseparator=">" --^endlineseparator="<" --^fieldseparator=","

（签入当前目录，并以易于解析的简化格式

打印结果，以指定的字符串作为行的开头和结尾以及

对各字段进行分隔。）

== CMD\_DESCRIPTION\_CHECKOUT ==

将文件标记为修改就绪。

== CMD\_USAGE\_CHECKOUT ==

用法：

cm ^checkout | ^co [<项路径>[ ...]] [-^R | -^r | --^recursive]

[--^format=<格式字符串>]

[--^errorformat=<格式字符串>] [--^resultformat=<格式字符串>]

[--^silent] [--^symlink] [--^ignorefailed]

[--^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

选项：

项路径 要签出的项。使用双引号 (" ")

指定包含空格的路径。使用空格

对各个项路径进行分隔。

使用 . 将签出应用到当前目录。

-^R 以递归方式签出文件。

--^format 检索特定格式的输出进度

消息。请查看示例以了解更多信息。

--^errorformat 检索特定格式的错误消息

（如果有）。请查看示例以了解更多信息。

--^resultformat 检索特定格式的输出结果

消息。请查看示例以了解更多信息。

--^silent 不显示任何输出。

--^symlink 将签出操作应用于符号链接而不是

目标。

--^ignorefailed 如果无法锁定某项（无法执行独占签出），

则签出操作将在没有这一项的情况下

继续进行。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，指定

应如何分隔字段。

== CMD\_HELP\_CHECKOUT ==

备注：

要签出项，必须满足以下条件：

- 项必须受源代码管理。

- 必须签入项。

如果在服务器上配置了锁（存在 lock.conf），则每次

在某条路径上进行签出时，Plastic 都会检查是否符合规则，

如果符合，该路径将处于独占签出（锁定）状态，以便任何其他人

都无法同时进行签出。

可使用 'cm ^lock ^list' 获取服务器中的所有锁。

请参阅《管理员指南》以了解更多信息：

https://www.plasticscm.com/download/help/adminguide

格式字符串将占位符 '{0}' 替换为要签出的

项的路径。请查看示例以了解具体用法。

从 stdin 读取输入：

'^checkout' 命令可从 stdin 读取路径。为此，请传递一个

破折号 "-"。

示例：cm ^checkout -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要签出的文件。

示例：

dir /S /B \*.c | cm ^checkout -

（在 Windows 中，签出工作区中的所有 .c 文件。）

示例：

cm ^checkout file1.txt file2.txt

（签出 'file1.txt' 和 'file2.txt' 文件。）

cm ^co \*.txt

（签出所有文本文件。）

cm ^checkout .

（签出当前目录。）

cm ^checkout -^R c:\workspace\src

（以递归方式签出 'src' 文件夹。）

cm ^co file.txt --^format="正在签出项 {0}"

--^errorformat="签出项 {0} 时出错" /

--^resultformat="项 {0} 已签出"

（签出 'file.txt'，并使用指定的格式字符串

显示操作的进度、结果和错误。）

cm ^checkout 链接 --^symlink

（签出 '链接' 文件而不是签出到目标；适用于 UNIX

环境。）

cm ^checkout .-^R --^ignorefailed

（以递归方式签出当前文件夹，忽略无法签出的

文件。）

cm ^co .--^machinereadable --^startlineseparator=">"

（签出当前目录，并以易于解析的简化格式打印结果，

以指定的字符串作为行的开头。）

== CMD\_DESCRIPTION\_CHECKSELECTORSYNTAX ==

检查选择器的语法。

== CMD\_USAGE\_CHECKSELECTORSYNTAX ==

用法：

cm ^checkselectorsyntax | ^css --^file=<选择器文件>

（检查选择器文件语法。）

^cat <选择器文件> | cm ^checkselectorsyntax | ^css -

（适用于 Unix。在标准输入中检查选择器文件。）

^type <选择器文件> | cm ^checkselectorsyntax | ^css -

（适用于 Windows。在标准输入中检查选择器文件。）

--^file 要从中读取选择器的文件。

== CMD\_HELP\_CHECKSELECTORSYNTAX ==

备注：

此命令在文件或标准输入中读取选择器，并检查

选择器是否为有效语法。如果语法检查结果为失败，则会将原因

打印在标准输出中。

示例：

cm ^checkselectorsyntax --^file=myselector.txt

（检查 'myselector.txt' 文件的语法。）

^cat myselector.txt | cm ^checkselectorsyntax

（在标准输入中检查 'myselector.txt' 的语法。）

== CMD\_DESCRIPTION\_CHGREVTYPE ==

更改某个项的修订类型（二进制或文本）。

== CMD\_USAGE\_CHGREVTYPE ==

用法：

cm ^changerevisiontype | ^chgrevtype | ^crt <项路径>[ ...]--^type=(^bin | ^txt)

项路径 要更改修订类型的项。使用双引号 (" ")

指定包含空格的路径。使用空格

对各个项路径进行分隔。

--^type 目标修订类型。选择 '^bin' 或 '^txt'。

== CMD\_HELP\_CHGREVTYPE ==

备注：

此命令只能应用于文件，而不能应用于目录。

指定的类型必须是系统支持的类型：'^bin' 或 '^txt'（二进制

或文本）。

示例：

cm ^changerevisiontype c:\workspace\file.txt --^type=^txt

（将 'file.txt' 修订类型更改为文本。）

cm ^chgrevtype comp.zip "image file.jpg" --^type=^bin

（将 'comp.zip' 和 "image file.jpg" 修订类型更改为二进制。）

cm ^crt \*.\* --^type=^txt

（将所有文件的修订类型更改为文本。）

== CMD\_DESCRIPTION\_TRIGGER\_EDIT ==

编辑触发器。

== CMD\_USAGE\_TRIGGER\_EDIT ==

用法：

cm ^trigger | ^tr ^edit <子类型\_类型> <位置编号>

[--^position=<新位置>]

[--^name=<新名称>] [--^script=<脚本路径>]

[--^filter=<筛选器字符串>] [--^server=<存储库服务器规格>]

子类型\_类型 触发器执行和触发器操作。

键入 'cm ^showtriggertypes' 可查看触发器类型

列表。

位置编号 要修改的触发器占用位置。

选项：

--^position 指定触发器的新位置。

此位置不能正在由相同类型的另一个

触发器使用。

--^name 指定触发器的新名称。

--^script 指定触发器脚本的新执行路径。

如果脚本以 "^webtrigger " 开头，则认为

这是 Web 触发器。请参阅“备注”以了解

更多详细信息。

--^filter 仅检查与指定筛选条件匹配的项。

--^server 修改指定服务器上的触发器。

如果未指定服务器，则在客户端上配置的

服务器上执行命令。

（使用 'cm ^help ^objectspec' 可进一步了解服务器

规格。）

== CMD\_HELP\_TRIGGER\_EDIT ==

备注：

Web 触发器：创建 Web 触发器的方法是键入 "^webtrigger <目标-URI>"

作为触发器命令。在这种情况下，触发器将针对指定的 URI

执行 POST 查询，其中，请求主体包含带有触发器

环境变量的 JSON 字典，还有一个指向字符串数组的

固定 INPUT 键。

示例：

cm ^trigger ^edit ^after-setselector 6 --^name="Backup2 管理器" --^script="/new/path/al/script"

cm ^tr ^edit ^before-mklabel 7 --^position=4 --^server=myserver:8084

cm ^trigger ^edit ^after-add 2 --^script="^webtrigger http://myserver.org/api"

== CMD\_DESCRIPTION\_CODEREVIEW ==

创建、编辑或删除代码审查。

== CMD\_USAGE\_CODEREVIEW ==

用法：

cm ^codereview <规格> <标题> [--^status=<状态名称>]

[--^assignee=<用户名>] [--^format=<格式字符串>]

[--^repository=<存储库规格>]

（创建代码审查。）

cm ^codereview -^e <ID> [--^status=<状态名称>] [--^assignee=<用户名>]

[--^repository=<存储库规格>]

（编辑代码审查。）

cm ^codereview -^d <ID> [ ...][--^repository=<存储库规格>]

（删除一个或多个代码审查。）

规格 可以是变更集规格或分支规格。此规格

将作为新代码审查的目标。（使用

'cm ^help ^objectspec' 可进一步了解变更集规格

或分支规格。）

标题 用作新代码审查的标题的

文本字符串。

ID 代码审查标识号。也可以使用

GUID。

选项：

-^e 编辑现有代码审查的参数。

-^d 删除一个或多个现有代码审查。使用

空格对各个代码审查 ID 进行分隔。

--^status 设置代码审查的新状态。请参阅“备注”

以了解更多信息。

--^assignee 设置代码审查的新被分派人。

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^repository 设置要用作默认存储库的存储库。（使用

'cm ^help ^objectspec' 可进一步了解存储库

规格。）

== CMD\_HELP\_CODEREVIEW ==

备注：

此命令允许用户管理代码审查：为变更集或分支

创建、编辑和删除代码审查。

要创建新的代码审查，必须提供变更集/分支规格和

标题。也可以设置初始状态和被分派人。结果将返回

一个 ID（如果请求返回 GUID，则返回 GUID）。

要编辑或删除现有代码审查，必须提供目标代码

审查 ID（或 GUID）。如果没有错误，则不会显示任何消息。

状态参数只能是以下值之一：^"Under review"

（默认值）、^"Reviewed" 或 ^"Rework required"。

repository 参数可用于设置默认的工作

存储库。当用户想要在其他服务器上而不是在与当前工作区

关联的服务器上管理审查时，或者根本没有

当前工作区时，此参数很有用。

输出格式自定义：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} ID

{1} GUID

请注意，'--^format' 参数仅在创建新的代码

审查时生效。

示例：

cm ^codereview ^cs:1856@myrepo@myserver:8084 "我的代码审查" --^assignee=dummy

cm ^codereview ^br:/main/task001@myrepo@myserver:8084 "我的代码审查" \

--^status=^"Rework required" --^assignee=newbie --^format="{^id} -> {^guid}"

cm ^codereview 1367 -^e --^assignee=新的被分派人

cm ^codereview -^e 27658884-5dcc-49b7-b0ef-a5760ae740a3 --^status=已审查

cm ^codereview -^d 1367 --^repository=myremoterepo@myremoteserver:18084

cm ^codereview 27658884-5dcc-49b7-b0ef-a5760ae740a3 -^d

== CMD\_DESCRIPTION\_CRYPT ==

对密码进行加密。

== CMD\_USAGE\_CRYPT ==

用法：

cm ^crypt <我的密码>

我的密码 要加密的密码。

== CMD\_HELP\_CRYPT ==

备注：

此命令用于加密作为参数传递的给定密码。

旨在对配置文件中的密码进行加密并提高

安全性。

示例：

cm ^crypt dbconfpassword -> ENCRYPTED: encrypteddbconfpassword

（对数据库配置文件 'db.conf' 中的密码进行加密。）

== CMD\_DESCRIPTION\_DEACTIVATEUSER ==

停用许可的用户。

== CMD\_USAGE\_DEACTIVATEUSER ==

用法：

cm ^deactivateuser | ^du <用户名>[ ...][--^server=<名称:端口>]

[--^nosolveuser]

用户名 要停用的一个或多个用户名。使用空格

对各个用户名进行分隔。

如果是 SID，则需要 '--^nosolveuser'。

选项：

--^server 停用指定服务器上的用户。

如果未指定服务器，则在客户端上配置的

服务器上执行命令。

--^nosolveuser 使用此选项时，该命令将不会检查身份验证系统上

是否存在相应的用户名。而且

<用户名> 必须是用户 SID。

== CMD\_HELP\_DEACTIVATEUSER ==

备注：

此命令将用户设置为非活动状态，从而禁止该用户使用

Plastic SCM。

有关激活 Plastic SCM 用户的更多信息，请参阅 'cm ^activateuser'

命令。

此命令检查用户是否在基础身份验证系统（例如 ActiveDirectory、

LDAP、用户/密码...）上存在。

要强制停用在身份验证系统上不再存在的

用户，可以使用 '--^nosolveuser' 选项。

示例：

cm ^deactivateuser john

cm ^du peter "mary collins"

cm ^deactivateuser john --^server=myserver:8084

cm ^deactivateuser S-1-5-21-3631250224-3045023395-1892523819-1107 --^nosolveuser

== CMD\_DESCRIPTION\_DIFF ==

显示文件、变更集和标签之间的差异。

== CMD\_USAGE\_DIFF ==

用法：

cm ^diff <变更集规格> | <标签规格> | <搁置规格> [<变更集规格> | <标签规格> | <搁置规格>]

[<路径>]

[--^added] [--^changed] [--^moved] [--^deleted]

[--^repositorypaths] [--^download=<下载路径>]

[--^encoding=<名称>]

[--^ignore=(^eol | ^whitespaces | ^"eol&whitespaces" | ^none)]

[--^clean]

[--^format=<格式字符串>] [--^dateformat=<格式字符串>]

显示“源”变更集或搁置集与“目标”变更集或搁置集

之间的差异。可使用变更集或标签规格

来指定变更集。

如果给出了两个规格，则第一个是差异比较的“源”，

第二个是“目标”。

如果仅给出一个规格，则“源”将是指定“目标”的

父变更集。

如果指定了可选路径，则将启动“差异比较窗口”以显示

该文件的两个修订之间的差异。

cm ^diff <修订规格1> <修订规格2>

显示一对修订之间的差异。这些差异

显示在“差异比较窗口”中。指定的第一个修订将

显示在左侧。

cm ^diff <分支规格> [--^added] [--^changed] [--^moved] [--^deleted]

[--^repositorypaths] [--^download=<下载路径>]

[--^encoding=<名称>]

[--^ignore=(^eol | ^whitespaces | ^"eol&whitespaces" | ^none)]

[--^clean]

[--^format=<格式字符串>] [--^dateformat=<格式字符串>]

[--^fullpaths | --^fp]

显示分支差异。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

选项：

--^added 仅打印包含已添加到存储库的项

的差异。

--^changed 仅打印包含已更改的项

的差异。

--^moved 仅打印包含已移动或已重命名的项

的差异。

--^deleted 仅打印包含已删除的项

的差异。

如果未指定 '--^added'、'--^changed'、'--^moved' 或 '--^deleted'，

则该命令将打印所有差异。

'^A' 表示已添加的项。

'^C' 表示已更改的项。

'^D' 表示已删除的项。

'^M' 表示已移动的项。左侧的项是原始项，

右侧是目标项。

--^repositorypaths 打印存储库路径而不是工作区路径。

（此选项将覆盖 '--^fullpaths' 选项。）

--^download 将差异内容存储在指定的输出

路径中。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^ignore 设置指定的比较方法。

请参阅“备注”以了解更多信息。

--^clean 不考虑由于合并而产生的

差异，仅考虑简单签入所创建的

差异。

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^dateformat 用于输出日期的格式。

--^fullpaths, --^fp 如果可能，强制打印文件和目录的完整

工作区路径。

== CMD\_HELP\_DIFF ==

备注：

比较方法：

^eol 忽略行尾差异。

^whitespaces 忽略空格差异。

^"eol&whitespaces" 忽略行尾和空格差异。

^none 检测行尾和空格差异。

此命令接受格式字符串以显示输出。

此命令的参数如下：

{^path} 项路径。

{^date} 更改日期/时间。

{^owner} 更改作者。

{^revid} 在差异比较中被视为目标的修订的

修订 ID。

{^parentrevid} 在差异比较中被视为目标的修订的

父级修订 ID。

{^baserevid} 在差异比较中被视为源的修订的

修订 ID。

{^srccmpath} 移动项（移动操作）之前的服务器路径。

{^dstcmpath} 移动项（移动操作）之后的服务器路径。

{^type} 项类型：

^D 目录，

^B 二进制文件，

^F 文本文件，

^S 符号链接，

^X Xlink。

{^repository} 项的存储库。

{^status} 项状态：

^A 已添加，

^D 已删除，

^M 已移动，

^C 已更改。

{^fsprotection} 显示项权限 (Linux/Mac chmod)。

{^srcfsprotection} 显示父修订项权限。

{^newline} 插入一个新行。

有关 '^revid' 的注意事项：

对于已添加的项，'^baserevid' 和 '^parentrevid' 将为 -1，因为

在这种情况下不存在以前的修订。

对于已删除的项，'^revid' 是源修订的 ID，而

'^baserevid' 将为 -1，因为没有目标修订。

对于 Xlink，'^baserevid' 和 '^parentrevid' 都始终为 -1。

示例：

比较分支：

cm ^diff ^br:/main/task001

cm ^diff ^br:/main/task001 \doc\readme.txt

比较变更集树：

cm ^diff 19

cm ^diff 19 25

cm ^diff ^cs:19 ^cs:25 --^format="{^path} {^parentrevid}"

cm ^diff ^cs:19 ^cs:23 --^format="{^date} {^path}" --^dateformat="yy/dd/MM HH:mm:ss"

cm ^diff ^cs:19 ^cs:23 --^changed

cm ^diff ^cs:19 ^cs:23 --^repositorypaths

cm ^diff ^cs:19 ^cs:23 --^download="D:\temp"

cm ^diff ^cs:19 ^cs:23 --^clean

cm ^diff ^cs:19 ^cs:23 \doc\readme.txt

比较标签树：

cm ^diff ^lb:FirstReleaseLabel ^lb:SecondReleaseLabel

cm ^diff ^lb:tag\_193.2 ^cs:34214

cm ^diff ^cs:31492 ^lb:tag\_193.2

比较搁置树：

cm ^diff ^sh:2

cm ^diff ^sh:2 ^sh:4

比较修订规格：

cm ^diff ^rev:readme.txt#^cs:19 ^rev:readme.txt#^cs:20

cm ^diff ^serverpath:/doc/readme.txt#^cs:19@myrepo \

^serverpath:/doc/readme.txt#^br:/main@myrepo@localhost:8084

== CMD\_DESCRIPTION\_DIFFMETRICS ==

显示两个修订之间的差异比较指标。

== CMD\_USAGE\_DIFFMETRICS ==

用法：

cm ^diffmetrics | ^dm <修订规格 1> <修订规格 2> [--^format=<格式字符串>]

[--^encoding=<名称>]

[--^ignore=(^eol | ^whitespaces | ^"eol&whitespaces" | ^none)]

修订规格 用于比较的修订。

（使用 'cm ^help ^objectspec' 可进一步了解修订规格。）

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^ignore 设置指定的比较方法。

请参阅“备注”以了解更多信息。

== CMD\_HELP\_DIFFMETRICS ==

备注：

指标为：已更改、添加和删除的行数。

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} 已更改的行数。

{1} 已添加的行数。

{2} 已删除的行数。

示例：

cm ^diffmetrics file.txt#^cs:2 file.txt#^br:/main/scm0211 \

--^format="已更改 {0} 行，已添加 {1} 行，已删除 {2} 行。"

（检索已格式化的差异比较指标结果。）

cm ^dm file.txt#^cs:2 file.txt#^cs:3 --^encoding=utf-8 --^ignore=^whitespaces

== CMD\_DESCRIPTION\_FASTEXPORT ==

以快速导出格式导出存储库。

== CMD\_USAGE\_FASTEXPORT ==

用法：

cm ^fast-export | ^fe <存储库规格> <快速导出文件>

[--^import-marks=<标记文件>]

[--^export-marks=<标记文件>]

[--^branchseparator=<分隔符>]

[--^nodata] [--^from=<变更集ID>] [--^to=<变更集ID>]

选项：

存储库规格 要从中导出数据的存储库。

（使用 'cm ^help ^objectspec' 可进一步了解存储库规格。）

快速导出文件 具有存储库数据并采用 Git 快速导出格式的

文件。

--^import-marks 用于增量导入的标记文件。此文件

已在先前由 '--^export-marks' 导出。此

文件中描述的变更集将不会被导入，

因为这些变更集已在先前被导入。

--^export-marks 所导入的变更集将保存到的文件。

此文件在以后的快速导入中用于表示

已导入的变更集。

--^branchseparator Plastic SCM 使用 "/" 作为分支层次结构中的默认

分隔符。此选项允许将字符用作层次结构

分隔符，因此 main-task-sub 将在 Plastic SCM 中

映射为 /main/task/sub。

--^nodata 导出存储库，但不包括数据。这对于

检查导出功能是否可正确运行很有用。

--^from 从特定变更集导出。

--^to 导出到特定变更集。

== CMD\_HELP\_FASTEXPORT ==

备注：

- 为了将 Plastic SCM 存储库导入到 Git，请使用诸如以下命令的命令：

^cat repo.fe.00 | ^git ^fast-import --^export-marks=marks.git --^import-marks=marks.git

- 支持使用一个包含先前导入的变更集的

标记文件（'--^import-marks' 和 '--^export-marks' 文件）进行增量

导出。

这意味着将仅导出在先前快速导出中未导出的

新变更集。

示例：

cm ^fast-export repo@localhost:8087 repo.fe.00 --^import-marks=marks.cm \

--^export-marks=marks.cm

（将本地服务器中的存储库 'repo' 以 Git 快速导出格式

导出到 'repo.fe.00' 文件中，并创建标记文件以稍后

执行增量导出。）

cm ^fast-export repo@localhost:8087 repo.fe.00 --^from=20

（将本地服务器中的存储库 'repo' 以 Git 快速导出格式

从变更集 '20' 导出到 'repo.fe.00' 文件中。）

== CMD\_DESCRIPTION\_FASTIMPORT ==

将 Git 快速导出数据导入到存储库中。

== CMD\_USAGE\_FASTIMPORT ==

用法：

cm ^fast-import | ^fi <存储库规格> <快速导出文件>

[--^import-marks=<标记文件>]

[--^export-marks=<标记文件>]

[--^stats] [--^branchseparator=<分隔符>]

[--^nodata] [--^ignoremissingchangesets] [--^mastertomain]

选项：

存储库规格 要将数据导入到的

存储库。如果事先不存在存储库，

则会创建该存储库。（使用 'cm ^help ^objectspec' 可进一步

了解存储库规格。）

快速导出文件 具有存储库数据并采用 Git 快速

导出格式的文件。

--^import-marks 用于增量导入的标记文件。

此文件已在先前由

'--^export-marks' 导出。此文件中描述的

变更集将不会被导入，因为这些变更集

已在先前被导入。

--^export-marks 所导入的变更集将保存到的

文件。此文件在以后的

快速导入中用于表示

已导入的变更集。

--^stats 打印一些有关导入过程的统计信息。

--^branchseparator Plastic SCM 使用 "/" 作为分支层次结构

中的默认分隔符。此选项允许

将字符用作层次结构分隔符，因此 main-task-sub

将在 Plastic SCM 中映射为 /main/task/sub。

--^nodata 导入 Git 快速导出结果，但不包括

数据。这对于检查导入功能是否可正确

运行很有用。

--^ignoremissingchangesets 无法导入的任何变更集都将被丢弃，

然后在没有这些变更集的情况下继续

进行快速导入操作。

--^mastertomain 使用 "^main" 而不是 "^master" 进行导入。

== CMD\_HELP\_FASTIMPORT ==

备注：

- 为了导出 Git 存储库，请使用诸如以下命令的命令：

^git ^fast-export --^all -^M --^signed-tags=^strip --^tag-of-filtered-object=^drop> ..\git-fast-export.dat

-^M 选项对于检测移动的项很重要。

- 如果指定的存储库不存在，则会创建该存储库。

- 支持使用一个包含先前导入的变更集的

标记文件（'--^import-marks' 和 '--^export-marks' 文件）进行增量

导入。

这意味着将仅导入在先前快速导入中未导入的

新变更集。

示例：

cm ^fast-import mynewrepo@atenea:8084 repo.fast-export

（将导出到 'repo.fast-export' 文件中的内容导入到

服务器 'atenea:8084' 上的 'mynewrepo' 存储库中。）

cm ^fast-import repo@atenea:8084 repo.fast-export --^export-marks=rep.marks

（将导出到 'repo.fast-export' 文件中的内容导入到

服务器 'atenea:8084' 上的 'repo' 存储库中，并创建一个标记文件

以稍后执行增量导入。）

cm ^fast-import repo@server:8084 repo.fast-export --^import-marks=repo.marks \

--^export-marks=repo.marks

（导入 'repo.fast-export' 文件的内容。仅导入

不在标记文件中的新变更集。同样的这个

标记文件还用于再次保存变更集列表，以便进行

下一次增量导入。）

== CMD\_DESCRIPTION\_FILEINFO ==

检索有关工作区中的项的详细信息。

== CMD\_USAGE\_FILEINFO ==

用法：

cm ^fileinfo <项路径>[ ...][--^fields=<字段值>[,...]]

[[--^xml | -^x [=<输出文件>]] | [--^format=<格式字符串>]]

[--^symlink] [--^encoding=<名称>]

项路径 要显示的项。使用空格对各个项进行

分隔。

使用双引号 (" ") 指定包含空格的

路径。

选项：

--^fields 一串用逗号分隔的值。此字符串用于

选择要为每项打印的字段。请参阅“备注”

以了解更多信息。

--^xml | -^x 以 XML 格式将输出打印到标准输出。

可以指定输出文件。此选项

不能与 '--^format' 结合使用。

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。此选项不能与 '--^xml'

结合使用。

此 '--^format' 选项优先于 '--^fields'（如果同时

指定了这两者）。

--^symlink 将 fileinfo 操作应用于符号链接而不是

目标。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

== CMD\_HELP\_FILEINFO ==

备注：

此命令为每个所选项打印详细的属性列表。

默认情况下，每个属性都打印在新行上。

可以修改属性列表以仅显示用户所需的

属性。为实现此目的，可以使用 '--^fields=<字段列表>'，

其中会接受一串用逗号分隔的属性名称。这样就只会显示

已指明名称的那些参数。

修订头部变更集：

默认情况下会禁用此选项。请注意，检索此属性的

速度明显慢于其余属性，因此我们建议用户

将尽可能多的项组合在一起。这样可以避免

多次单独执行 'cm ^fileinfo'，从而缩短执行时间。

另外，此功能当前不适用于受控目录。

您可以在下面找到可用属性名称的完整列表。

默认情况下不会显示标有星号 ('\*') 的名称：

^ClientPath 项在磁盘上的本地路径。

^RelativePath 相对于工作区的路径。

^ServerPath 项的存储库路径。

（注意：此选项当前

不支持已变换的工作区）。

^Size 项大小。

^Hash 项哈希值总和。

^Owner 项所属的用户。

^RevisionHeadChangeset (\*) 已加载到分支头部变更集内的

修订变更集。

（请参阅上面的注意事项。）

^RevisionChangeset 当前已加载到工作区内的

修订变更集。

^RepSpec 项的存储库规格。

（使用 'cm ^help ^objectspec' 可进一步了解

存储库规格。）

^Status 工作区项状态：已添加、已签出、

已删除，等等。

^Type 修订类型（文本、二进制、目录、符号链接

或未知）。

^Changelist 项所属的更改列表（如果有）。

^IsLocked (\*) 项是否已被独占签出

锁定。

^LockedBy (\*) 将项独占签出的用户。

^LockedWhere (\*) 将项独占签出的

位置。

^IsUnderXlink 项是否位于 Xlink

之下。

^UnderXlinkTarget 项所在的 Xlink 的目标

（如果有）。

^UnderXlinkPath 通过 Xlink 链接的存储库中的项服务器路径

（如果有）。

^UnderXlinkWritable 项所属的 Xlink 是否

可写。

^UnderXlinkRelative 项所属的 Xlink 是否

是相对 Xlink。

^IsXlink 项本身是否是 Xlink。

^XlinkTarget 项指向的目标存储库（如果项

是 Xlink）。

^XlinkName 项的 Xlink 名称（如果项实际上

是 Xlink）。

^XlinkWritable Xlink 项是否为

可写 Xlink。

^XlinkRelative Xlink 项是否为

相对 Xlink。

输出格式自定义：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{^ClientPath}

{^RelativePath}

{^ServerPath}

{^Size}

{^Hash}

{^Owner}

{^RevisionHeadChangeset}

{^RevisionChangeset}

{^Status}

{^Type}

{^Changelist}

{^IsLocked}

{^LockedBy}

{^LockedWhere}

{^IsUnderXlink}

{^UnderXlinkTarget}

{^UnderXlinkPath}

{^UnderXlinkWritable}

{^UnderXlinkRelative}

{^IsXlink}

{^XlinkTarget}

{^XlinkName}

{^XlinkWritable}

{^XlinkRelative}

{^RepSpec}

请注意，'--^format' 和 '--^xml' 选项是互斥的，

因此不能同时使用。

示例：

cm ^fileinfo file1.txt file2.txt dir/

cm ^fileinfo "New Project.csproj" --^xml

cm ^fileinfo assets.art --^fields=^ServerPath,^Size,^IsLocked,^LockedBy

cm ^fileinfo proj\_specs.docx --^fields=^ServerPath,^RevisionChangeset --^xml

cm ^fileinfo samples.ogg --^format="{^ServerPath}[{^Owner}] -> {^Size}"

== CMD\_DESCRIPTION\_FIND\_QUERY ==

运行类似 SQL 的查询以查找 Plastic SCM 对象。

== CMD\_USAGE\_FIND\_QUERY ==

用法：

cm ^find <对象类型>

[^where <条件字符串>]

[^on ^repository '<存储库规格>' | ^on ^repositories '<存储库规格1>','<存储库规格2>'[,...]]

[--^format=<格式字符串>] [--^dateformat=<日期格式>]

[--^nototal] [--^file=<转储文件>] [--^xml]

[--^encoding=<名称>]

对象类型 要查找的对象类型。

（使用 'cm ^help ^showfindobjects' 可了解如何指定

这些对象。）

您还可以阅读 'cm ^find' 指南：

https://www.plasticscm.com/download/help/cmfind

选项：

条件字符串 搜索对象属性的条件。

存储库规格 搜索存储库别名或规格。

如果是 '^on ^repositories'，请使用逗号

对存储库规格字段进行分隔。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

--^format 检索特定格式的输出消息。

请阅读 'cm ^find' 指南以查看所有

可用作输出格式字符串的对象属性：

https://www.plasticscm.com/download/help/cmfind

--^dateformat 用于输出日期的格式。

--^nototal 最后不输出记录数。

--^file 转储结果的文件。

--^xml 以 XML 格式将输出打印到标准输出。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

== CMD\_HELP\_FIND\_QUERY ==

备注：

如果未指定存储库，则在工作区中配置的存储库上

进行搜索。

从命令行中使用比较运算符（>、<、>=、<=）

运行查询时，请注意，shell 将这些运算符视为 IO

重定向。因此，需要用双引号将查询

引起来。

'cm ^find' 命令接受格式字符串以显示输出。

每个输出参数由字符串进行标识，用户可以通过

在 '{' 和 '}' 花括号之间键入参数编号来引用参数。

输出参数通常对应于对象的属性。

一些有效的输出格式字符串如下：

--^format={^id}{^date}{^name}

--^format="{^item}#{^branch} ^with ^date {^date}"

XML 和编码注意事项：

指定 '--^xml' 选项后，该命令在标准输出中将命令结果

显示为 XML 文本。文本将以操作系统的

默认编码显示，因此在控制台中可能会

错误地显示非 ANSI 字符。如果将命令输出重定向到

文件，则将正确显示输出内容。当同时指定了 '--^xml' 和 '--^file'

选项时，默认编码为 utf-8。

示例：

cm ^find ^revision

cm ^find ^revision "^where ^changeset=23 ^and ^owner='maria'"

cm ^find ^branch "^on ^repository 'rep1'"

cm ^find ^label "^on ^repositories 'rep1', '^rep:default@localhost:8084'"

cm ^find ^branch "^where ^parent='^br:/main' ^on ^repository 'rep1'"

cm ^find ^revision "^where ^item='^item:.'" --^format="{^item}#{^branch}"

cm ^find ^revision "^where ^item='^item:.'" --^xml --^file=c:\queryresults\revs.xml

== CMD\_DESCRIPTION\_FINDCHANGED ==

获取已更改文件的列表。已弃用此命令，保留此命令只是为了

向后兼容。请改用 'cm ^status'。

== CMD\_USAGE\_FINDCHANGED ==

用法：

cm ^findchanged | ^fc [-^R | -^r | --^recursive] [--^checkcontent]

[--^onlychanged] [<路径>]

选项：

-^R 以递归方式在目录中查找。

--^checkcontent 按内容比较文件。

--^onlychanged 仅查找已更改的文件；无法获得

签出。

路径 （默认值：当前目录。）

查找已更改文件的初始路径。

== CMD\_HELP\_FINDCHANGED ==

备注：

如果未给出 '--^checkcontent' 选项，Plastic 将根据文件时间戳

查找更改。当指定了 '--^checkcontent' 选项时，比较的是

文件或文件夹内容，而不使用时间戳。

与 Plastic SCM 服务器断开连接的情况下，此命令对于检测

已更改的文件很有用。可以将输出通过管道传递到 checkout 命令，

从而在稍后检查更改（请参阅示例）。

示例：

cm ^findchanged .

（在当前目录中查找已更改的文件。）

cm ^findchanged -^R .| cm ^checkout -

（签出已更改的元素。）

== CMD\_DESCRIPTION\_FINDCHECKEDOUT ==

获取已签出项的列表。已弃用此命令，保留此命令只是为了

向后兼容。请改用 'cm ^status'。

== CMD\_USAGE\_FINDCHECKEDOUT ==

用法：

cm ^findcheckouts | ^fco [--^format=<格式字符串>] [--^basepath]

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^basepath 开始搜索签出的起始路径。如果

未指定，则使用当前路径。

== CMD\_HELP\_FINDCHECKEDOUT ==

备注：

此命令适合用于在一个步骤中签入项或撤销签出所有

已签出的项，并将标准输出重定向到其他命令。

请参阅示例。

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} 日期。

{1} 所有者。

{2} 工作区信息。

{3} 客户端机器名称。

{4} 项路径。

{5} 分支和存储库信息。

示例：

cm ^findcheckouts --^format="分支 {5} 上已更改的文件 {4}"

（查找已签出的项，并使用文件路径以及分支和存储库信息

来格式化输出。）

cm ^findcheckouts --^format={4} | cm ^checkin -

（签入所有已签出的项。）

cm ^findcheckouts --^format={4} | cm ^undocheckout -

（撤销签出所有已签出的项。）

== CMD\_DESCRIPTION\_FINDPRIVATE ==

获取私有项的列表。已弃用此命令，保留此命令只是为了

向后兼容。请改用 'cm ^status'。

== CMD\_USAGE\_FINDPRIVATE ==

用法：

cm ^findprivate | ^fp [-^R | -^r | --^recursive] [--^exclusions] [<路径>]

选项：

-^R 以递归方式在目录中查找。

--^exclusions 此选项允许在文件 ignore.conf 定义的

已忽略路径内禁止搜索。

路径 （默认值：当前目录。）

查找私有文件的初始路径。

== CMD\_HELP\_FINDPRIVATE ==

备注：

如果指定了任何路径，Plastic SCM 将开始从当前目录

搜索。

此命令适合用于在文件夹上添加私有项，并将输出通过管道

传递到 add 命令。请参阅示例。

示例：

cm ^findprivate .

cm ^findprivate -^R | cm ^add -

（以递归方式搜索私有项并添加这些项。）

== CMD\_DESCRIPTION\_GETCONFIG ==

获取配置信息。

== CMD\_USAGE\_GETCONFIG ==

用法：

cm ^getconfig [^setfileasreadonly] [^location] [^extensionworkingmode]

[^extensionprefix] [^defaultrepserver]

^setfileasreadonly 返回受保护文件是否保留为

只读。

^location 返回客户端配置路径。

^extensionworkingmode 返回扩展工作模式。

^extensionprefix 返回已配置的扩展前缀。

^defaultrepserver 返回默认存储库服务器的

位置。

== CMD\_HELP\_GETCONFIG ==

示例：

cm ^getconfig ^setfileasreadonly

== CMD\_DESCRIPTION\_GETFILE ==

下载给定修订的内容。

== CMD\_USAGE\_GETFILE ==

用法：

cm ^getfile | ^cat <修订规格> [--^file=<输出文件>] [--^debug]

[--^symlink] [--^raw]

修订规格 对象规格。（使用 'cm ^help ^objectspec' 可

进一步了解规格。）

选项：

--^file 用于保存输出的文件。默认情况下会打印在

标准输出中。

--^debug 使用目录规格时，该命令

将显示目录中的所有项、目录的修订 ID

和文件系统保护。

--^symlink 将操作应用于符号链接而不是

目标。

--^raw 显示文件的原始数据。

== CMD\_HELP\_GETFILE ==

示例：

cm ^cat myfile.txt#^br:/main

（在 'myfile.txt' 的分支 '^br:/main' 中获取最后一个修订。）

cm ^getfile myfile.txt#^cs:3 --^file=tmp.txt

（获取 'myfile.txt' 的变更集 3 并将该变更集写入文件 'tmp.txt'。）

cm ^cat ^serverpath:/src/foo.c#^br:/main/task003@myrepo

（在存储库 'myrepo' 中的分支 '/main/task003' 的最后一个变更集处

获取 '/src/foo.c' 的内容）

cm ^cat ^revid:1230@^rep:myrep@^repserver:myserver:8084

（获得 ID 为 1230 的修订。）

cm ^getfile ^rev:info\ --^debug

（获取 'info' 目录中的所有修订。）

== CMD\_DESCRIPTION\_GETREVISION ==

在工作区中加载修订。

== CMD\_USAGE\_GETREVISION ==

此命令会修改工作区中加载的修订，因此可能影响

以后的合并。

这是从旧版本继承的高级命令，因此请谨慎使用。

用法：

cm ^getrevision <修订规格>

修订规格 对象规格。（使用 'cm ^help ^objectspec' 可

进一步了解修订规格。）

== CMD\_HELP\_GETREVISION ==

示例：

cm ^getrevision file.txt#^cs:3

（获取 'file.txt' 的变更集 3 修订。）

== CMD\_DESCRIPTION\_GETSTATUS ==

获取项的状态。

== CMD\_USAGE\_GETSTATUS ==

这是一个自动化命令，仅用于自动执行 'cm'。

此命令的用户友好度不如预期。

用法：

cm ^getstatus | ^gs <项路径>[ ...][--^format=<格式字符串>] [--^stats]

[-^R | -^r | --^recursive]

项路径 要获取相应状态的一个或多个项。使用双引号

(" ") 指定包含空格的路径。使用

空格对各个路径进行分隔。

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^stats 打印一些有关获取状态过程的统计信息。

-^R 以递归方式显示目录中的状态。

== CMD\_HELP\_GETSTATUS ==

备注：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} 项路径。

{1} 项状态：

0 私有，

1 已签入，

2 已签出。

从 stdin 读取输入：

'^getstatus' 命令可从 stdin 读取路径。为此，请传递

一个破折号 "-"。

示例：cm ^getstatus -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要获取哪些路径的状态。

示例：

dir /S /B \*.c | cm ^getstatus --^format="路径 {0} 状态 {1}" -

（在 Windows 中，获取工作区中所有 .c 文件的状态。）

示例：

cm ^getstatus file1.txt file2.txt

（获取文件的状态。）

cm ^gs info\ -^R --^format="项 {0} 的状态为 {1}"

（获取目录及其所有项的状态，并显示

格式化的输出。）

== CMD\_DESCRIPTION\_GETTASKBRANCHES ==

获取与某个任务链接的分支。

== CMD\_USAGE\_GETTASKBRANCHES ==

这是一个自动化命令，仅用于自动执行 'cm'。

此命令的用户友好度不如预期。

用法：

cm ^gettaskbranches | ^gtb <任务名称> [--^format=<格式字符串>]

[--^dateformat=<日期格式>]

任务名称 任务标识符。

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^dateformat 用于输出日期的格式。

== CMD\_HELP\_GETTASKBRANCHES ==

备注：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

{^name} 分支名称。

{^owner} 分支的所有者。

{^date} 创建分支的日期。

{^type} 分支类型（如果是智能分支，则为 'T'，否则为 'F'）。

{^parent} 父分支。

{^comment} 分支的注释。

{^repname} 分支所在的存储库。

{^repserver} 服务器名称。

示例：

cm ^gettaskbranches 4311

cm ^gtb 4311 --^format="^br:{^name}"

cm ^gtb 4311 --^format="^br:{^name} {^date}" --^dateformat="yyyy/MM/dd HH:mm:ss"

== CMD\_DESCRIPTION\_GETWWI ==

显示有关工作区选择器的信息。

== CMD\_USAGE\_GETWWI ==

用法：

cm ^wi [<工作区路径>]

选项：

工作区路径 工作区在机器上的路径。

== CMD\_HELP\_GETWWI ==

备注：

'^wi' 命令显示工作区（存储库、分支和/或标签）

的工作配置。

示例：

cm ^wi c:\mywk

== CMD\_DESCRIPTION\_GWP ==

从路径获取工作区信息。

== CMD\_USAGE\_GWP ==

这是一个自动化命令，仅用于自动执行 'cm'。

此命令的用户友好度不如预期。

用法：

cm ^getworkspacefrompath | ^gwp <项路径> [--^format=<格式字符串>]

项路径 磁盘上的文件或文件夹。

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

== CMD\_HELP\_GWP ==

备注：

此命令显示有关路径中的工作区的信息。

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} | {^wkname} 工作区名称。

{1} | {^wkpath} 工作区路径。

{2} | {^machine} 客户端机器名称。

{3} | {^owner} 工作区所有者。

{4} | {^guid} 工作区 GUID。

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

示例：

cm ^gwp c:\myworkspace\code\file1.cpp --^format="工作区名称：{^wkname}"

== CMD\_DESCRIPTION\_HELP ==

获取有关 Plastic SCM 命令的帮助。

== CMD\_USAGE\_HELP ==

用法：

cm ^help <命令>

== CMD\_HELP\_HELP ==

== CMD\_DESCRIPTION\_IOSTATS ==

显示有关硬件的统计信息。

== CMD\_USAGE\_IOSTATS ==

用法：

cm ^iostats [<存储库服务器规格>] [<测试列表>[ ...]]

[--^nettotalmb=<值\_mb>] [--^networkiterations=<值\_迭代>]

[--^diskdatasize=<值\_大小>] [--^disktestpath=<值\_路径>]

[--^systemdisplaytime=<值\_时间>]

[--^systemdisplaytimeinterval=<值\_间隔>]

选项：

存储库服务器规格 用于执行网络测试的可用 Plastic SCM

服务器，例如 "serverUploadTest"

和/或 "serverDownloadTest"。

如果未提供服务器，该命令

将尝试与默认配置的服务器

通信。

（使用 'cm ^help ^objectspec' 可进一步了解

服务器规格。）

测试列表 可用的测试。使用空格对各个测试字段

进行分隔。

请参阅“备注”以了解更多信息。

--^nettotalmb 表示在网络测试中传输的

用户数据量（以兆字节为单位），

例如 "^serverDownloadTest" 或

"^serverUploadTest"。

值必须介于 "4" 到 "512" 之间。

（默认值：16）

--^networkiterations 表示要运行的

"^serverDownloadTest" 和/或 "^serverUploadTest"

的迭代次数。

值必须介于 "1" 到 "100" 之间。

（默认值：1）

--^diskdatasize 表示将在 "^diskTest" 上写入

并随后读取的数据量（以兆字节

为单位）。

值必须介于 "100" 到 "4096" 之间。

（默认值：512）

--^disktestpath "^diskTest" 将测试文件写入到的

路径。如果未提供此参数，

该命令将尝试使用系统临时

路径。

--^systemdisplaytime 显示系统资源使用情况的时间间隔

（单位：秒）。此选项可用于

以下测试："^systemNetworkUsage"

和 "^systemDiskUsage"。

值必须介于 "1" 到 "3600" 之间。

（默认值：5 秒）。

--^systemdisplaytimeinterval 系统性能采样的时间间隔

（单位：秒）。此选项可用于

以下测试：

"^systemNetworkUsage" 和 "^systemDiskUsage"。

值必须介于 "1" 到 "60" 之间。

（默认值：1 秒）。

== CMD\_HELP\_IOSTATS ==

备注：

此命令要求在网速测试（"^serverUploadTest" 和/或

"^serverDownloadTest"）期间使用可用的服务器。

'--^diskTestPath' 必须指向属于要测试的物理

磁盘驱动器的路径。如果未指定路径，该命令将尝试

使用系统默认的临时路径。

指定路径的磁盘驱动器必须具有足够的可用空间才能执行

测试。

在执行命令期间，执行的测试可能导致

系统性能下降。

可用的测试：

--^serveruploadtest （默认值）测量从 Plastic SCM 客户端到

服务器的数据上传速度。

--^serverdownloadtest （默认值）测量从 Plastic SCM 服务器到

客户端的数据下载速度。

--^disktest （默认值）测量磁盘读取速度和磁盘

写入速度。

--^systemnetworkusage 显示当前的系统网络资源

使用情况。

（显示 Microsoft Windows 提供的网络接口

性能计数器）。

仅在 Microsoft Windows 中可用。

--^systemdiskusage 显示当前的系统物理磁盘

使用情况。

（显示 Microsoft Windows 提供的网络接口

性能计数器）。

仅在 Microsoft Windows 中可用。

示例：

cm ^iostats MYSERVER:8087 --^serveruploadtest --^serverdownloadtest --^nettotalmb=32

== CMD\_DESCRIPTION\_ISSUETRACKER ==

在指定的问题跟踪程序中获取、更新或查找问题状态。

== CMD\_USAGE\_ISSUETRACKER ==

用法：

cm ^issuetracker <名称> ^status ^get <任务\_ID> <参数>[ ...]

cm ^issuetracker <名称> ^status ^update <任务\_ID> <状态> <参数>[ ...]

cm ^issuetracker <名称> ^status ^find <状态> <参数>[ ...]

cm ^issuetracker <名称> ^connection ^check <参数>[ ...]

名称 要连接的问题跟踪程序的名称。

目前仅支持 Jira。

任务\_ID 要查询或更新的问题的编号。

^status 问题跟踪程序中某个问题的有效状态。

Jira 参数（全部为必需参数）：

--^user=<用户> 要验证身份的用户。

--^password=<密码> 用于身份验证的密码。

--^host=<URL> 问题跟踪程序的目标 URL。

--^projectkey=<密钥> Jira 项目的项目密钥。

== CMD\_HELP\_ISSUETRACKER ==

示例：

cm ^issuetracker jira ^status ^get 11 --^user=user@mail.es --^password=pwd \

--^host=https://user.atlassian.net --^projectkey=PRJ

（获取 'PRJ' 项目的问题 11 的状态。）

cm ^issuetracker jira ^status ^update 11 "完成" --^user=user@mail.es \

--^password=pwd --^host=https://user.atlassian.net --^projectkey=PRJ

（将 'PRJ' 项目的问题 11 的状态更新为“完成”。）

cm ^issuetracker jira ^status ^find "完成" --^user=user@mail.es --^password=pwd \

--^host=https://user.atlassian.net --^projectkey=PRJ

（为 'PRJ' 项目获取相应状态设置为“完成”的任务 ID）

cm ^issuetracker jira ^connection ^check --^user=user@mail.es --^password=pwd \

--^host=https://user.atlassian.net --^projectkey=PRJ

（检查配置参数是否有效。）

== CMD\_DESCRIPTION\_LICENSEINFO ==

显示许可证信息和许可证使用情况。

== CMD\_USAGE\_LICENSEINFO ==

用法：

cm ^licenseinfo | ^li [--^server=<存储库服务器规格>] [--^inactive] [--^active]

[--^sort=(^name|^status)]

选项：

--^server 从指定的服务器获取许可证信息。

如果未指定服务器，则在客户端上配置的

服务器上执行命令。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

服务器规格。）

--^inactive 在“许可证使用情况”部分中仅显示非活动用户。

--^active 在“许可证使用情况”部分中仅显示活动用户。

--^sort 按指定的排序选项之一对用户进行排序：

'^name' 或 '^status'。

== CMD\_HELP\_LICENSEINFO ==

备注：

显示的信息包括到期日期、已激活的用户和

已停用的用户等。

示例：

cm ^licenseinfo

cm ^licenseinfo --^server=myserver:8084

cm ^licenseinfo --^sort=^name

== CMD\_DESCRIPTION\_LINKTASK ==

将变更集链接到任务。

== CMD\_USAGE\_LINKTASK ==

这是一个自动化命令，仅用于自动执行 'cm'。

此命令的用户友好度不如预期。

用法：

cm ^linktask | ^lt <变更集规格> <扩展前缀> <任务名称>

变更集规格 要链接到任务的完整变更集规格。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

扩展前缀 要使用的已配置问题跟踪系统的

扩展前缀。

任务名称 问题跟踪系统上的任务标识符。

== CMD\_HELP\_LINKTASK ==

示例：

cm ^lt ^cs:8@^rep:default@^repserver:localhost:8084 jira PRJ-1

== CMD\_DESCRIPTION\_LOCK\_LIST ==

显示服务器上的锁定情况。

== CMD\_USAGE\_LOCK\_LIST ==

用法：

cm ^lock ^list | ^ls [<修订规格 > [ ...]] [--^server=<服务器>]

[--^onlycurrentuser] [--^onlycurrentworkspace]

[--^ignorecase]

修订规格 如果存在一个或多个修订，则此命令将

为每个指定的修订显示一个锁定行

（如果在服务器中锁定了该修订的关联项）。否则，

此命令将列出默认服务器中的所有锁定项

（或者使用 '--^server' 选项来设置的已锁定项）

使用多个修订规格时，使用空格对各个修订规格

进行分隔。

（使用 'cm ^help ^objectspec' 可进一步了解修订规格。）

选项：

--^server 存储库服务器规格。

此选项将覆盖从当前工作区

或 client.conf 文件检索到的默认

服务器。

（使用 'cm ^help ^objectspec' 可进一步了解

服务器规格。）

--^onlycurrentuser 筛选结果，仅显示当前用户

执行的锁定。

--^onlycurrentworkspace 筛选结果，仅显示对当前工作区

执行的锁定（按名称进行匹配）。

--^ignorecase 使用服务器路径规格时，忽略路径中的

大小写。使用此标志后，即使用户写入 "/sRc/fOO.c"，

该命令也将对 "/src/foo.c" 有效。

== CMD\_HELP\_LOCK\_LIST ==

备注：

该命令将列出默认服务器中当前

已锁定的项。该命令还接受若干修订规格；在这种

情况下，仅显示属于所选项的锁定。

可使用 '--^server=<服务器>' 来设置要查询的默认服务器。

该命令针对指定服务器中的每个锁定都会显示一行：

- 已锁定项的 GUID。

- 执行锁定的用户名。

- 锁定操作所在的工作区的名称。

- 锁定项的路径（服务器路径格式）。

示例：

cm ^lock ^list

cm ^lock ^list --^server=myserver:8084

cm ^lock ^ls ^serverpath:/src/foo.c#^cs:99@default@localhost:8084

cm ^lock ^list ^revid:3521@default ^itemid:2381@secondary --^onlycurrentuser

cm ^lock ^ls --^onlycurrentuser

cm ^lock ^ls --^onlycurrentuser --^onlycurrentworkspace

== CMD\_DESCRIPTION\_LISTUSERS ==

列出用户和组。

== CMD\_USAGE\_LISTUSERS ==

用法：

cm ^listusers | ^lu <存储库服务器规格> [--^onlyusers] [--^onlygroups]

[--^filter= <筛选器字符串>]

存储库服务器规格 存储库服务器规格。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

选项：

--^onlyusers 仅列出用户。

--^onlygroups 仅列出组。

--^filter 仅列出匹配指定筛选器的用户

和/或组。

== CMD\_HELP\_LISTUSERS ==

示例：

cm ^lu localhost:8084

（列出服务器中的所有用户。）

cm ^listusers localhost:8084 --^onlyusers --^filter=m

（仅列出服务器中包含 "m" 的用户。）

== CMD\_DESCRIPTION\_LOCATION ==

返回 'cm' 的路径。

== CMD\_USAGE\_LOCATION ==

用法：

cm ^location

== CMD\_HELP\_LOCATION ==

== CMD\_DESCRIPTION\_LOCK ==

此命令允许用户管理锁定。

== CMD\_USAGE\_LOCK ==

用法：

cm ^lock <命令> [选项]

命令：

^list | ^ls

^unlock

要获取有关每条命令的更多信息，请运行：

cm ^lock <命令> --^usage

cm ^lock <命令> --^help

== CMD\_HELP\_LOCK ==

示例：

cm ^lock ^list

cm ^lock

（如果没有参数，则 '^list' 为可选项。）

cm ^lock ^ls ^serverpath:/src/foo.c#^cs:99@default@localhost:8084

cm ^lock ^unlock 91961b14-3dfe-4062-8c4c-f33a81d201f5

== CMD\_DESCRIPTION\_LOG ==

获取有关变更集内的修订的信息。

== CMD\_USAGE\_LOG ==

用法：

cm ^log [<变更集规格> | <存储库规格>] [--^from=<变更集规格起点>] [--^allbranches]

[--^ancestors] [--^csformat=<格式字符串>] [--^itemformat=<格式字符串>]

[--^xml[=<输出文件>]] [--^encoding=<名称>]

[--^repositorypaths | --^fullpaths | --^fp]

选项：

变更集规格 该命令将返回在相应规格的变更集内

所做的所有更改。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

存储库规格 该命令将列出在指定存储库中进行的

所有更改。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

--^from 列出从变更集规格 [变更集规格起点]

到变更集规格 [变更集规格] 的每个变更集内

进行的所有更改。

[变更集规格起点] 变更集不包含在

输出中。

提供存储库规格后会忽略此选项。

--^allbranches 对于创建了相应变更集的所有分支，

显示指定时间间隔内创建的

这些变更集的相关信息。

--^ancestors 显示可通过给定变更集（[变更集规格]）的

父链接和合并链接来访问的变更集

的相关信息。如果也提供了起点

变更集（[变更集规格起点]），此变更集将用作

所有路径的下限。备注：使用此选项时，

不会显示变更集更改。

--^csformat 检索特定格式的变更集信息。请参阅

“备注”以了解更多信息。

--^itemformat 检索特定格式的项信息。请参阅

“备注”以了解更多信息。

--^xml 以 XML 格式将输出打印到标准输出。

可以指定输出文件。

--^encoding 与 '--^xml' 选项结合使用，指定要在 XML 输出

中使用的编码（例如：utf-8）。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^fullpaths, --^fp 如果可能，强制打印文件和目录的完整

工作区路径。

--^repositorypaths 打印存储库路径（服务器路径）

而不是工作区路径。（此选项将覆盖

'--^fullpaths' 选项。）

== CMD\_HELP\_LOG ==

备注：

- 如果“变更集规格”和选项均未指定，该命令将显示

每个分支中上个月内创建的每个变更集的

相关信息。

- 如果仅包含 '--^from' 选项，该命令显示

在创建指定变更集的分支中从该变更集到最后一个

变更集的每个变更集的相关信息。

- 如果显示不带时间间隔的选项 '--^allbranches'，

该命令会检索与仅指定“变更集规格”时相同的

信息。

- 如果使用 '--^from'，输出包含从“变更集规格起点”+1 开始的

变更集的信息。

- 用于显示变更集信息的存储库就是在执行命令的

路径中加载的存储库。

此命令接受项的格式字符串 ('--^itemformat') 以及

变更集的格式字符串 ('--^csformat')。

'--^csformat' 的输出参数如下：

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

{^changesetid} 变更集编号。

{^branch} 在其中创建变更集的分支。

{^date} 变更集的日期。

{^owner} 变更集的所有者。

{^comment} 变更集的注释。

{^items} 变更集内涉及的项。

{^repository} 变更集所在的存储库。

{^repserver} 服务器名称。

'--^itemformat' 的输出参数如下：

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

{^path} 项路径。

{^branch} 在其中创建变更集的分支。

{^date} 变更集的日期。

{^owner} 变更集的所有者。

{^shortstatus} 打印短格式。请参阅下文。

{^fullstatus} 打印长格式。请参阅下文。

短格式及其对应的长格式：

'^A' ^Added

'^D' ^Deleted

'^M' ^Moved

'^C' ^Changed

这些是有效的输出字符串：

--^csformat="{^newline}变更集 {^changesetid} 创建于 {^date}；{^tab} 已更改项：{^items}。"

--^itemformat="{^newline}分支 {^branch} 中的项 {^path} 已更改。"

示例：

cm ^log

（显示每个分支中上个月创建的每个变更集的相关

信息。）

cm ^log ^cs:16

（显示在其中创建变更集的分支内的变更集 16 中

所做更改的相关信息。）

cm ^log ^cs:16 --^csformat="{^newline}变更集 {^changesetid} 创建于 \

{^date}；{^tab} 已更改项：{^items}。"

（按指定格式显示此信息。）

cm ^log --^from=^cs:20 ^cs:50

（显示从变更集 21 到变更集 50 的每个变更集中包含的

每个修订的相关信息。）

cm ^log --^from=^cs:20 ^cs:50 --^allbranches

（显示存储库的每个分支中从变更集 21 到

变更集 50 的每个变更集中包含的每个修订的

相关信息。）

cm ^log ^rep:myrep@localhost:8084

（显示在指定存储库中进行的更改的相关信息。

无需工作区也可运行此命令。）

cm ^log --^from=^cs:20@^rep:mainRep@localhost:8084

（显示从变更集 21 开始的每个变更集内包含的每个修订

的相关信息。无需工作区也可运行此命令，因为

已指定完整变更集规格。）

== CMD\_DESCRIPTION\_LS ==

列出树的内容。

== CMD\_USAGE\_LS ==

用法：

cm ^ls | ^dir [<路径>[ ...]] [--^format=<格式字符串>] [--^symlink]

[--^selector[=<选择器格式>]] [--^tree=<对象规格>]

[-^R | -^r | --^recursive]

[--^xml[=<输出文件>]] [--^encoding=<名称>]

选项：

路径 要显示的路径的列表。使用空格对各个路径进行

分隔。

使用双引号 (" ") 指定包含空格的

路径。

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^symlink 将操作应用于符号链接而不是

目标。

--^selector 从活动的工作区选择器获取内容。

如果已指定选择器格式，则列出

指定的选择器。

自 Plastic SCM 4.x 开始，选择器不再是 Plastic SCM

的重要特性，因此大多数情况下已弃用。

--^tree 列出指定变更集或分支中的树。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

-^R 以递归方式列出。

--^xml 以 XML 格式将输出打印到标准输出。

可以指定输出文件。

--^encoding 与 '--^xml' 选项结合使用，指定要在 XML 输出

中使用的编码（例如：utf-8）。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

== CMD\_HELP\_LS ==

备注：

- 可以使用元字符来键入路径。

- 此列表取决于工作区选择器。

- 可以指定格式字符串，从而将命令的输出格式化。

- 如果指定 '--^tree' 或 '--^selector' 选项，则给定的

路径必须是服务器路径（也就是：'cm path'）：/dir/file.txt，而不是

工作区路径：C:\Users\myuser\mywk\dir\file.txt

- 如果未提供路径，则认为工作区路径是当前

目录。如果使用 '--^tree' 或 '--^selector' 选项，

则采用根路径 ("/")。

默认格式字符串为：

"{^size,10} {^date:dd/MM/yyyy} {^date:HH:mm}\

{^type,-6} {^location,-12} {^checkout,-5} {^name}\

{^symlinktarget}"

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{^size}

{^formattedsize}

{^date}

{^type}

^dir 目录，

^txt 文本文件，

^File 文件。

{^location} 示例：^br:branch#cset

{^checkout}

{^name}

{^changeset}

{^path}

{^repspec}

{^owner}

{^revid}

{^parentrevid}

{^itemid}

{^brid}

{^repid}

{^server}

{^symlinktarget}

{^hash}

{^chmod}

{^wkpath} 相对于工作区根目录的路径

{^branch}

{^newlocation} cset@branch

{^guid} （将需要更长时间来解析）

{^itemguid}

{^transformed} 显示已变换项的适用规则

可以设置 PLASTIC\_LS\_FORMAT 环境变量以便自定义

'^ls' 格式。

示例：

cm ^ls

cm ^ls c:\workspace\src

cm ^ls --^format={^name}

（仅文件名。）

cm ^ls --^symlink

（显示有关符号链接的信息，而不是“符号链接”的文件或

目录。适用于 UNIX 环境。）

cm ^ls code --^selector

（显示当前工作区选择器中 'code' 子目录的

内容。）

cm ^ls /code --^selector="^rep 'myrep' ^path '/' ^branch '/^main'"

（显示指定选择器中 '/code' 子目录的内容。

请注意，该路径是按服务器格式指定的。）

cm ^ls /code --^tree=44@myrep@denver:7070

（列出服务器 'denver:7070' 存储库 'myrep' 变更集 44 的

'/code' 子目录。）

cm ^ls /code --^tree=^br:/main/scm13596@myrep@denver:7070

（列出服务器 'denver:7070' 存储库 'myrep' 分支

'/main/scm13596' 中的最新变更集的 '/code' 子目录。）

cm ^ls /code --^tree=ae1390ed-7ce9-4ec3-a155-e5a61de0dc77@myrep@denver:7070

（列出服务器 'denver:7070' 存储库 'myrep'

的变更集 ae1390ed-7ce9-4ec3-a155-e5a61de0dc77 的

'/code' 子目录。）

== CMD\_DESCRIPTION\_TRIGGER\_LIST ==

列出服务器上给定类型的触发器。

== CMD\_USAGE\_TRIGGER\_LIST ==

用法：

cm ^trigger | ^tr ^list | ^ls [<子类型-类型>] [--^server=<存储库服务器规格>]

[--^format=<格式字符串>]

选项：

子类型-类型 触发器执行和触发器操作。

键入 'cm ^showtriggertypes' 可查看触发器类型

列表。

--^server 列出指定服务器上的触发器。

如果未指定服务器，则在客户端上配置的

服务器上执行命令。

（使用 'cm ^help ^objectspec' 可进一步了解服务器

规格。）

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

== CMD\_HELP\_TRIGGER\_LIST ==

备注：

如果未指定类型，则列出服务器上的所有触发器。

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} 触发器位置。

{1} 触发器名称。

{2} 触发器路径。

{3} 触发器所有者。

{4} 触发器类型。

{5} 触发器筛选器。

示例：

cm ^trigger list after-mklabel

cm ^tr ^ls ^before-mkbranch --^server=myserver:8084

== CMD\_DESCRIPTION\_MANIPULATESELECTOR ==

将选择器更改为日期。

== CMD\_USAGE\_MANIPULATESELECTOR ==

这是一个自动化命令，仅用于自动执行 'cm'。

此命令的用户友好度不如预期。

用法：

cm ^manipulateselector | ^ms [<工作区路径> | <工作区规格>] --^atdate=<选择器日期>

工作区路径 工作区的路径。

工作区规格 工作区的规格。（使用 'cm ^help ^objectspec' 可

进一步了解规格。）

选项：

--^atdate 返回一个选择器以用于重新创建工作区，

就像在指定日期看到的工作区一样。

== CMD\_HELP\_MANIPULATESELECTOR ==

备注：

如果路径和工作区规格均未指定，则该命令将采用

当前目录作为工作区路径。

示例：

cm ^manipulateselector c:\workspace --^atdate=yyyy-MM-ddTHH:mm:ss

cm ^manipulateselector --^atdate=yyyy-MM-ddTHH:mm:ss

cm ^manipulateselector > mySelector.txt --^atdate=yyyy-MM-ddTHH:mm:ss

cm ^manipulateselector ^wk:build\_wk@BUILDER --^atdate=yyyy-MM-ddTHH:mm:ss

== CMD\_DESCRIPTION\_MERGE ==

将一个分支与另一个分支合并。

== CMD\_USAGE\_MERGE ==

用法：

cm ^merge <源规格> [--^merge] [--^cherrypicking] [--^forced]

[--^mergetype=(^onlyone|^onlysrc|^onlydst|^try|^forced)]

[--^interval-origin=<变更集规格> | --^ancestor=<变更集规格>]

[--^keepsource | --^ks] [--^keepdestination | --^kd]

[--^automaticresolution=<冲突类型>[;...]]

[--^subtractive] [--^mount] [--^printcontributors]

[--^noprintoperations] [--^silent]

[(--^to=<分支规格> | --^destination=<分支规格>)[--^shelve]]

[--^no-dst-changes]

[-^c=<注释字符串> | --^commentsfile=<注释文件>]

[--^resolveconflict --^conflict=<索引>

--^resolutionoption=(^src|^dst|(^rename --^resolutioninfo=<字符串名称>))

--^mergeresultfile=<路径> --^solvedconflictsfile=<路径>]

[--^nointeractiveresolution]

[--^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

源规格 要合并的源对象的规格：

- 分支规格：'[^br:/]br\_name'

- 标签规格：'^lb:lb\_name'

- 变更集规格：'^cs:cs\_number'

- 搁置规格：'^sh:shelve\_number'

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

选项：

--^merge 执行合并。否则，打印找到的

冲突。

--^cherrypicking 合并源变更集内包含的

变更。如果合并源规格是标签，

则不会使用此选项。

--^forced 不检查源和目标是否

已连接。

此选项仅可用于间隔合并

和挑拣。

--^mergetype 请参阅“备注”以了解更多信息。

--^interval-origin 指定将哪个变更集选为

间隔原点，以便合并仅考虑

源变更集与指定间隔原点

之间的差异。

--^ancestor 这是 --^interval-origin 的别名。

--^keepsource 对于有冲突的项，接受源参与者

的所有更改。

--^keepdestination 对于有冲突的项，保留目标参与者

的更改。

--^automaticresolution 用于解决目录冲突。此选项

可让您决定应该自动选择

源参与者还是目标参与者

来解决冲突。

使用分号对各个冲突类型进行分隔。

请参阅“备注”以了解更多信息。

--^subtractive 删除由合并引入的更改。传递给

命令的参数（源规格）用于

指定要删除哪个源的

更改。必须是变更集。如果是变更集

间隔，必须使用 '--^interval-origin' 来

定义间隔原点。为了删除更改，

系统将创建一个新的已签出修订，

该修订将具有上一个

修订的内容（不包含已删除的更改）。

--^mount 给定存储库的装入点。

--^printcontributors 打印参与者（基项、源和

目标）。

--^noprintoperations 以静默方式解决合并问题，

不显示有关解决情况的信息。

--^silent 不显示任何输出。

--^to | --^destination 对指定分支执行“合并到”操作

（通过输入分支规格）

并完全解决冲突。

“合并到”（或无工作区的合并）是在

服务器端进行的合并。正常的合并

发生在“从”分支、标签或变更集合并的

工作区上，而“合并到”完全发生

在服务器上。在正常合并中，

“目标”是工作区，而在“合并到”中，

必须始终指定目标（这就是称之为

“到”的原因）。

请查看以下链接以了解关于

“合并到”功能的更多信息：

https://www.plasticscm.com/download/help/mergeto

--^shelve 使用合并结果的更改（加上合并

可追溯性信息）来创建一个搁置，

而不是创建新的变更集。合并源为

搁置时，此选项不可用。该选项

仅可用于服务器端合并

（又称为“合并到”）。因此，需要 '--^to' 和

'--^merge' 选项。

--^no-dst-changes 确保目标参与者不具有

更改（目标变更集也是

共同上级）。目标上存在更改时，

不允许合并。

-^c 将指定的注释应用于合并操作中

创建的变更集。

--^commentsfile 将指定文件中的注释应用于合并操作中

创建的变更集。

--^resolveconflict （主要由插件使用。请参阅“备注”以了解更多信息。）

用于解决目录冲突。

--^conflict 与 '--^resolveconflict' 标志结合使用，指定

要解决的冲突的索引（从 1 开始）。

--^resolutionoption 与 '--^resolveconflict' 标志结合使用，指示

冲突解决类型。使用以下

选项之一：'^src'、'^dst'、'^rename'。

请参阅“备注”以了解更多信息。

--^resolutioninfo 与 '--^resolveconflict' 标志结合使用，提供在

'--^resolutionoption' 选项为 'rename' 时

要使用的名称。

--^mergeresultfile 与 '--^resolveconflict' 标志结合使用，将

不同调用之间的合并结果的信息

输出到文件中。指定的路径将在第一次

调用期间创建，并在每个后续

调用时更新。

--^solvedconflictsfile 与 '--^resolveconflict' 标志结合使用，

将不同调用之间已解决的冲突

的信息输出到文件中。指定的路径

将在第一次调用期间创建，并在每个

后续调用时更新。

--^nointeractiveresolution （主要由插件使用。请参阅“备注”以了解更多信息。）

避免提示用户已发生手动冲突。

这样就不会解决目录冲突。

--^machinereadable （主要由插件使用。请参阅“备注”以了解更多信息。）

以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何开头。（默认值：空字符串。）

--^endlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何结尾。（默认值：空字符串。）

--^fieldseparator 与 '--^machinereadable' 标志结合使用，指定

应如何分隔字段。（默认值：

空格。）

== CMD\_HELP\_MERGE ==

备注：

此命令用于合并两个分支之间或标签与

分支之间的更改。合并的目标必须始终为分支。

合并源将指定为参数。

目标是工作区的当前内容。

例如，要显示从分支 task001 合并到主分支

的元素，选择器必须指向主分支，

工作区必须完成更新，然后：

cm ^merge ^br:/task001

添加 '--^merge' 选项来切实执行合并：

cm ^merge ^br:/task001 --^merge

要定义合并源，可以使用以下规格：

- 分支规格：

[^br:/]br\_name

示例：^br:/main/task001

（以上示例从此分支上的最后一个变更集执行合并。）

- 标签规格：

^lb:lb\_name

示例：^lb:BL001

（从标记的变更集合并。）

- 变更集规格：

^cs:cs\_number

示例：^cs:25

（从给定的变更集内容合并。）

- 搁置规格：

^sh:shelve\_number

示例：^sh:2

（从给定的搁置内容合并。）

要自动解决目录冲突，请使用 '--^automaticresolution'

选项并指定冲突类型，然后指定在合并操作

期间必须选择的参与者（源或目标）。

（使用分号 (;) 对各个“冲突类型”-“参与者”对进行分隔。）

例如：

cm ^merge ^cs:2634 --^merge --^automaticresolution=^eviltwin-src;^changedelete-src

（自变更集 2634 开始的合并操作将会解决 "^eviltwin" 和

"^changedelete" 冲突，解决方法是在两种情况下保留源 ("-^src")

参与者）。

- 冲突类型之后的 "-^src" 后缀告诉合并命令保留

源参与者更改。

- "-^dst" 后缀将保留目标参与者更改。

以下是合并命令支持的冲突类型的列表：

"^movedeviltwin"、"^eviltwin"、"^changedelete"、"^deletechange"、"^movedelete"、

"^deletemove"、"^loadedtwice"、"^addmove"、"^moveadd"、"^divergentmove"、

"^cyclemove"、"^all"。

"^all" 值会覆盖其他选项。在以下示例中，

"^eviltwin-dst" 将被忽略：

cm ^merge ^br:/main/task062 --^merge --^automaticresolution=^all-src;^eviltwin-dst

请查看以下链接以了解关于合并冲突的更多信息：

https://www.plasticscm.com/download/help/directorymerges

以下是 '--^mergetype' 的选项：

^onlyone 仅有一个参与者修改了该项的情况下，才进行

自动合并。

^onlysrc 仅有一个源参与者修改了该项的情况下，才进行

自动合并。

^onlydst 仅有一个目标参与者修改了该项的情况下，才进行

自动合并。

^try 仅有一个参与者修改了冲突代码段（每个冲突）

的情况下，才进行自动合并。

^forced 始终尝试解决所有非自动冲突。

以下是主要由插件和集成使用的选项：

- '--^resolveconflict'，用于解决目录冲突。还必须使用

以下选项：

- '--^conflict' 是要解决的冲突的索引

（从 1 开始）。

- '--^resolutionoption' 指示要使用的冲突解决

方法。选项可以是：

- '^src'，用于保留源更改而放弃

目标更改

- '^dst'，用于保留目标更改而放弃

源更改

- '^rename'（仅在冲突类型支持此解决方法

的情况下），用于将目标重命名为

通过 '--^resolutioninfo' 选项提供的给定名称。

- '--^resolutioninfo'，用于提供在 '^rename' 解决

方法中使用的名称

- '--^mergeresultfile' 和 '--^solvedconflictsfile'，这两者用于

存储不同调用之间的合并信息。

- '--^nointeractiveresolution' 指示合并操作不要求用户

进行手动冲突解决。

- '--^machinereadable' 和 '--^startlineseparator'、'--^endlineseparator'、

'--^fieldseparator' 选项，用于以机器可读的方式

（易于解析）打印输出。

示例：

cm ^merge --^machinereadable --^startlineseparator=start@\_@line \

--^endlineseparator=new@\_@line --^fieldseparator=def#\_#sep \

--^mergeresultfile=C:\Users\Borja\AppData\Local\Temp\2tmp4D6C.tmp \

--^solvedconflictsfile=C:\Users\Borja\AppData\Local\Temp\2tmp4D6D.tmp \

--^resolveconflict --^conflict=1 --^resolutionoption=rename \

--^resolutioninfo=bin\_dst ^br:/main/task --^merge

示例：

cm ^merge ^br:/task001

（不合并，仅打印要合并的项。）

cm ^merge ^br:/task001 --^merge

（从分支 'task001' 进行合并。）

cm ^merge ^cs:5 --^merge --^cherrypicking --^interval-origin=^cs:2

（从变更集间隔 (2,5] 进行挑拣。）

cm ^merge ^cs:8 --^merge --^subtractive --^keepdestination

（从变更集 8 进行减法合并，保留那些有冲突的

元素的目标更改。）

cm ^merge ^br:/main/task001 --^to=^br:/main --^merge -^c="集成的新 UI"

（进行从分支 'task001' 到分支 'main' 的服务器端合并

（又称为“合并到”），并设置注释。）

cm ^merge ^br:/main/task001 --^to=^br:/main --^merge --^shelve

（进行从分支 'task001' 到分支 'main' 的服务器端合并，

并将结果保留在搁置中。）

cm ^merge ^sh:2 --^to=^br:/main --^merge --^no-dst-changes

（仅在从当前的 'main' 头部创建时，才将搁置 2

应用于 'main'）

== CMD\_DESCRIPTION\_ATTRIBUTE\_CREATE ==

创建新的属性。

== CMD\_USAGE\_ATTRIBUTE\_CREATE ==

用法：

cm ^attribute | ^att ^create | ^mk <属性名>

属性名 属性名称

== CMD\_HELP\_ATTRIBUTE\_CREATE ==

示例：

cm ^attribute ^create 状态

（创建属性 '状态'。）

cm ^att ^mk 集成

（创建属性 '集成'。）

== CMD\_DESCRIPTION\_BRANCH ==

允许用户管理分支。

== CMD\_USAGE\_BRANCH ==

用法：

cm ^branch | ^br <命令> [选项]

命令：

^create | ^mk

^delete | ^rm

^rename

^history

^showmain

^showmerges

要获取有关每条命令的更多信息，请运行：

cm ^branch <命令> --^usage

cm ^branch <命令> --^help

== CMD\_HELP\_BRANCH ==

示例：

cm ^branch /main/scm21345

cm ^branch ^create /main/scm21345

cm ^branch ^delete /main/scm21345

cm ^branch ^rename /main/scm21345 scm21346

cm ^branch ^history /main/scm21345

cm ^branch ^showmain

cm ^branch ^showmerges file.txt

== CMD\_DESCRIPTION\_BRANCH\_CREATE ==

创建新的分支。

== CMD\_USAGE\_BRANCH\_CREATE ==

用法：

cm ^branch | ^br [^create | ^mk] <分支规格>

[--^changeset=<变更集规格> | --^label=<标签规格>]

[-^c=<注释字符串> | -^commentsfile=<注释文件>]

分支规格 新分支名称或规格。

（使用 'cm ^help ^objectspec' 可进一步了解分支规格。）

选项：

--^changeset 用作新分支的起点的变更集。

（使用 'cm ^help ^objectspec' 可进一步了解变更集规格。）

--^label 用作新分支的起点的标签。

（使用 'cm ^help ^objectspec' 可进一步了解标签规格。）

-^c 用指定的文本填充新分支的“注释”

字段。

-^commentsfile 用指定文件的内容填充新分支的“注释”

字段。

== CMD\_HELP\_BRANCH\_CREATE ==

备注：

要创建顶级分支，请指定不含任何层次结构的名称。

例如：

cm ^br /dev

如果未指定可选参数 '--^changeset'，新分支的基项

将是父分支上的最后一个变更集。如果新分支

是顶级分支，则使用的基础变更集将为变更集 0。

可以使用 '-^c' 或 '-^m' 开关来指定注释：

cm ^branch /main/task001 -^c="这是注释"

cm ^branch /main/task001 -^m "这是注释"

设置 PLASTICEDITOR 环境变量可指定用于输入注释的

编辑器。如果已设置 PLASTICEDITOR 环境变量，

并且注释为空，则会自动启动编辑器

以用于指定注释。

示例：

cm ^branch task001

cm ^branch ^create task001

cm ^branch ^mk task001

cm ^br ^mk task001

（在当前工作区的存储库中创建顶级分支

'task001'。）

cm ^branch ^br:/task001/task002@

（创建 'task002' 分支作为 'task001' 的子级。）

cm ^br /main/task001@myrep@myserver:8084 -^c="我的注释"

（在存储库 'myrep@myserver:8084' 中，创建 'task001' 分支

作为 'main' 的子级，并包含注释 '我的注释'。）

cm ^branch ^br:/main/task001 --^changeset=2837 -^commentsfile=commenttask001.txt

（创建 'task001' 分支作为 'main' 的子级，基础变更集 'changeset=2837'，

并应用 'commenttask001.txt' 文件中的注释。）

== CMD\_DESCRIPTION\_BRANCH\_DELETE ==

删除一个或多个分支。

== CMD\_USAGE\_BRANCH\_DELETE ==

用法：

cm ^branch | ^br ^delete | ^rm <分支规格>[ ...]

分支规格 要删除的分支。使用空格对各个分支进行分隔。

（使用 'cm ^help ^objectspec' 可进一步了解分支

规格。）

== CMD\_HELP\_BRANCH\_DELETE ==

备注：

此命令将删除一个或多个分支。

示例：

cm ^branch ^delete /main/task001

（在当前工作区的存储库中删除名为 'task001' 分支

（'main' 的子级）。）

cm ^br ^rm main/task002 /main/task012@reptest@myserver:8084

（删除当前工作区的存储库中的分支 '/main/task002

以及存储库 'reptest@myserver:8084' 中的分支 '/main/task012'。）

== CMD\_DESCRIPTION\_BRANCH\_RENAME ==

重命名分支。

== CMD\_USAGE\_BRANCH\_RENAME ==

用法：

cm ^branch | ^br ^rename <分支规格> <新名称>

分支规格 要重命名的分支。

（使用 'cm ^help ^objectspec' 可进一步了解分支规格。）

新名称 分支的新名称。

== CMD\_HELP\_BRANCH\_RENAME ==

备注：

此命令将重命名分支。

示例：

cm ^branch ^rename /main/task0 task1

（将分支 '/main/task0' 重命名为 '/main/task1'。）

cm ^br ^rename ^br:/main@reptest@server2:8084 secondary

（将存储库 'reptest' 的 'main' 分支重命名为 'secondary'。）

== CMD\_DESCRIPTION\_BRANCH\_HISTORY ==

显示分支的历史记录。

== CMD\_USAGE\_BRANCH\_HISTORY ==

用法：

cm ^branch | ^br ^history <分支规格> [--^dateformat=<日期格式>]

[--^machinereadable]

分支规格 用于获取历史记录的分支规格。

（使用 'cm ^help ^objectspec' 可进一步了解分支规格。）

选项：

--^dateformat 用于输出日期的格式。

--^machinereadable 以易于解析的格式输出结果。

== CMD\_HELP\_BRANCH\_HISTORY ==

示例：

cm ^branch ^history ^br:/main/scm001@myrepository@myserver:8084

（显示 'myserver' 服务器中的 'myrepository' 存储库的 '/main/scm001'

分支的历史记录。）

cm ^br ^history main --^dateformat="yyyy, dd MMMM" --^machinereadable

（以给定的日期格式和易于解析的格式，显示当前

存储库的 'main' 分支的历史记录。）

== CMD\_DESCRIPTION\_BRANCH\_SHOWMAIN ==

显示存储库的主分支。

这是一个自动化命令，仅用于自动执行 'cm'。

存储库的主分支很有可能是 '/main'。

== CMD\_USAGE\_BRANCH\_SHOWMAIN ==

用法：

cm ^branch | ^br ^showmain [<存储库规格>] [--^encoding=<名称>]

[--^format=<格式字符串>] [--^dateformat=<日期格式>]

存储库规格 要显示分支的存储库

规格。

（使用 'cm ^help ^objectspec' 可进一步了解存储库规格。）

选项：

--^encoding 指定输出中使用的编码，

如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^dateformat 用于输出日期的格式。

== CMD\_HELP\_BRANCH\_SHOWMAIN ==

备注：

此命令将显示存储库的主分支。

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{^id} 分支 ID。

{^comment} 注释。

{^date} 日期。

{^name} 名称。

{^owner} 所有者。

{^parent} 父分支名称。

{^repository} 存储库。

{^repname} 存储库名称。

{^repserver} 服务器。

{^changeset} 分支的头部变更集。

示例：

cm ^branch ^showmain

（显示当前工作区的存储库的主分支。）

cm ^branch ^showmain repo@server:8084

（显示服务器 'server:8084' 中的存储库 'repo' 的

主分支。）

cm ^br ^showmain --^dateformat="yyyy, dd MMMM" --^encoding=utf8

（以给定日期格式显示存储库的主分支，

输出编码为 utf8。）

cm ^br ^showmain --^format="{^id} - {^name}"

（显示存储库的主分支，仅打印其 ID 和名称。）

== CMD\_DESCRIPTION\_BRANCH\_SHOWMERGES ==

显示待合并的分支。

== CMD\_USAGE\_BRANCH\_SHOWMERGES ==

这是一个自动化命令，仅用于自动执行 'cm'。

此命令的用户友好度不如预期。

用法：

cm ^branch | ^br ^showmerges <项路径>[ ...]

[--^format=<格式字符串>]

[--^dateformat=<日期格式>]

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^dateformat 用于输出日期的格式。

== CMD\_HELP\_BRANCH\_SHOWMERGES ==

备注：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{^id} 分支 ID。

{^comment} 注释。

{^date} 日期。

{^name} 名称。

{^owner} 所有者。

{^parent} 父分支名称。

{^parentid} 父分支 ID。

{^repid} 存储库 ID。

{^repository} 存储库。

{^repname} 存储库名称。

{^repserver} 存储库服务器。

示例：

cm ^branch ^showmerges file.txt

（显示 'file.txt' 的待定合并中涉及的分支。）

cm ^branch ^showmerges file.txt --^format="{^date} {^name}" --^dateformat="yyMMdd"

（显示合并中涉及的分支，仅以给定的日期格式打印

日期和名称。）

== CMD\_DESCRIPTION\_REPOSITORY ==

允许用户管理存储库。

== CMD\_USAGE\_REPOSITORY ==

用法：

cm ^repository | ^repo <命令> [选项]

命令：

^create | ^mk

^delete | ^rm

^list | ^ls

^rename

^add

要获取有关每条命令的更多信息，请运行：

cm ^repository <命令> --^usage

cm ^repository <命令> --^help

== CMD\_HELP\_REPOSITORY ==

示例：

cm ^repository

cm ^repository ^list

cm ^repository 新存储库

cm ^repository ^create 新存储库

cm ^repository ^rename 旧名称 新名称

cm ^repository ^add C:\repo\

== CMD\_DESCRIPTION\_REPOSITORY\_CREATE ==

在服务器上创建存储库。

== CMD\_USAGE\_REPOSITORY\_CREATE ==

用法：

cm ^repository | ^repo <存储库名称>

cm ^repository | ^repo <存储库服务器规格> <存储库名称>[ ...]

cm ^repository | ^repo [^create | ^mk] <存储库名称>

存储库服务器规格 存储库服务器规格。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

服务器规格。）

存储库名称 一个或多个新存储库的名称。

使用空格对各个存储库名称进行分隔。

== CMD\_HELP\_REPOSITORY\_CREATE ==

示例：

cm ^repository MyRep

cm ^repo 192.168.1.140:8087 Rep01 Rep01/ModuleA Rep01/ModuleB

cm ^repo ^create Rep01

cm ^repo ^mk 列表

== CMD\_DESCRIPTION\_REPOSITORY\_DELETE ==

从服务器删除存储库。

== CMD\_USAGE\_REPOSITORY\_DELETE ==

用法：

cm ^repository | ^repo ^delete | ^rm <存储库规格>

选项：

存储库规格 存储库的规格。

（使用 'cm ^help ^objectspec' 可进一步了解存储库规格。）

== CMD\_HELP\_REPOSITORY\_DELETE ==

备注：

从存储库服务器删除存储库。

不会从数据库后端删除数据，而是会断开，

因此将无法再进行访问。

（以后可以重新连接数据，请参阅 'cm ^repository ^add'。）

示例：

cm ^repository ^delete myrepository@^repserver:myserver:8084

cm ^repository ^rm myrepository@myserver:8084

cm ^repo ^rm myrepository

== CMD\_DESCRIPTION\_REPOSITORY\_LIST ==

列出服务器上的存储库。

== CMD\_USAGE\_REPOSITORY\_LIST ==

用法：

cm ^repository | ^repo [^list | ^ls] [<存储库服务器规格>] [--^format=<格式字符串>]

选项：

存储库服务器规格 存储库服务器规格。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

服务器规格。）

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

== CMD\_HELP\_REPOSITORY\_LIST ==

备注：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{^repid} | {0} 存储库标识符。

{^repname} | {1} 存储库名称。

{^repserver} | {2} 服务器名称。

{^repowner} | {3} 存储库所有者。

{^repguid} | {4} 存储库的唯一标识符。

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

如果格式参数值为 '^TABLE'，则会使用具有 {^repid}、

{^repname} 和 {^repserver} 字段的表格式来打印输出。

示例：

cm ^repository

（列出所有存储库。）

cm ^repository ^list localhost:8084 --^format="{1, -20} {3}"

（将存储库名称写入 20 个空格位，左对齐，然后留一个空白，

再然后是存储库所有者。）

cm ^repository ^ls localhost:8084 --^format="{^repname, -20} {^repowner}"

（写入方式与前一个示例相同。）

cm ^repo ^ls localhost:8084 --^format=^TABLE

（使用具有以下字段的表格式来写入存储库列表：

存储库 ID、存储库名称和存储库服务器名称。）

== CMD\_DESCRIPTION\_REPOSITORY\_RENAME ==

重命名存储库。

== CMD\_USAGE\_REPOSITORY\_RENAME ==

用法：

cm ^repository | ^repo ^rename [<存储库规格>] <新名称>

存储库规格 要重命名的存储库。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

新名称 存储库的新名称。

== CMD\_HELP\_REPOSITORY\_RENAME ==

备注：

此命令将重命名存储库。

如果未指定存储库规格，则会采用当前存储库。

示例：

cm ^repository ^rename development

（将当前存储库重命名为 'development'。）

cm ^repo ^rename ^rep:default@SERVER:8084 development

（将 'SERVER' 上的 'default' 存储库重命名为 'development'。）

== CMD\_DESCRIPTION\_REPOSITORY\_ADD ==

通过添加现有存储库的数据库来连接此存储库。

== CMD\_USAGE\_REPOSITORY\_ADD ==

用法：

cm ^repository | ^repo ^add <数据库文件> <存储库名称> <存储库服务器规格>

数据库文件 数据库后端上的数据库文件的名称。

存储库名称 存储库的名称。

存储库服务器规格 存储库服务器规格。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

服务器规格。）

== CMD\_HELP\_REPOSITORY\_ADD ==

备注：

将现有存储库数据库重新连接到服务器。

示例：使用 'cm ^repository ^delete' 命令后，使用 '^add' 命令

将存储库从一个服务器移动到另一个服务器或还原

已存档的存储库。

示例：

cm ^repository ^add rep\_27 myrepository myserver:8084

== CMD\_DESCRIPTION\_TRIGGER\_CREATE ==

在服务器上创建新触发器。

== CMD\_USAGE\_TRIGGER\_CREATE ==

用法：

cm ^trigger | ^tr ^create | ^mk <子类型-类型> <新名称> <脚本路径>

[--^position=<新位置>]

[--^filter=<筛选器字符串>]

[--^server=<存储库服务器规格>]

子类型-类型 触发器执行和触发器操作。

键入 'cm ^showtriggertypes' 可查看触发器类型

列表。

新名称 新触发器的名称。

脚本路径 服务器上包含要执行的脚本的

磁盘路径。如果命令行以 "^webtrigger " 开头，

则认为此触发器是 Web 触发器。请参阅

“备注”以了解更多信息。

选项：

--^position 指定触发器的新位置。

此位置不能正在由相同类型的另一个

触发器使用。

--^filter 仅检查与指定筛选器匹配的项。

--^server 在指定服务器上创建触发器。

如果未指定服务器，则在客户端上配置的

服务器上执行命令。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

服务器规格。）

== CMD\_HELP\_TRIGGER\_CREATE ==

备注：

Web 触发器：创建 Web 触发器的方法是键入 "^webtrigger <目标-URI>"

作为触发器命令。在这种情况下，触发器将针对指定的 URI

执行 POST 查询，其中，请求主体包含带有触发器

环境变量的 JSON 字典，还有一个指向字符串数组的

固定 INPUT 键。

示例：

cm ^trigger ^create ^after-setselector "BackupMgr" "/path/to/script" --^position=4

cm ^tr ^mk ^before-mklabel new "/path/to/script" --^server=myserver:8084

cm ^tr ^mk ^after-mklabel Log "/path/to/script" --^filter="^rep:myRep,LB\*"

（仅当标签名称以 'LB' 开头并且标签是在名为 'myRep' 的

存储库中创建时，才会执行此触发器。）

cm ^tr ^mk ^after-checkin NotifyTeam "^webtrigger http://myserver.org/api"

== CMD\_DESCRIPTION\_MOVE ==

移动或重命名文件或目录。

== CMD\_USAGE\_MOVE ==

用法：

cm ^move | ^mv <源路径> <目标路径> [--^format=<格式字符串>]

[--^errorformat=<格式字符串>]

源路径 源项的路径。

目标路径 目标项的路径。

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^errorformat 检索特定格式的错误消息。请参阅

“备注”以了解更多信息。

== CMD\_HELP\_MOVE ==

备注：

此命令将移动或重命名存储库中的项。

还会在本地文件系统中进行更改。

如果源路径是文件，则目标路径可以是文件或

目录。在第一种情况下会重命名文件；否则会移动

相应的项。

如果源路径是目录，则目标路径必须是目录。

要移动或重命名的项必须已存在。

格式：

{0} 源路径（'--^format' 和 '--^errorformat'）

{1} 目标路径（'--^format' 和 '--^errorformat'）

示例：

cm ^move file.txt file.old

（重命名相应的项。）

cm ^mv .\file.old .\oldFiles

（将 'file.old' 移到 'oldFiles'。）

cm ^move .\src .\src2

（重命名目录。）

== CMD\_DESCRIPTION\_LABEL ==

允许用户管理标签。

== CMD\_USAGE\_LABEL ==

用法：

cm ^label | ^lb <命令> [选项]

命令：

^create | ^mk

^delete | ^rm

^rename

要获取有关每条命令的更多信息，请运行：

cm ^label <命令> --^usage

cm ^label <命令> --^help

== CMD\_HELP\_LABEL ==

示例：

cm ^label myNewLabel ^cs:42

（'^create' 命令是可选命令。）

cm ^label ^rename myNewLabel newLabelName

cm ^label ^delete newLabelName

== CMD\_DESCRIPTION\_LABEL\_CREATE ==

将标签应用于变更集，并在需要时创建标签。

== CMD\_USAGE\_LABEL\_CREATE ==

用法：

cm ^label [^create] <标签规格> [<变更集规格> | <工作区路径>]

[--^allxlinkedrepositories]

[-^c=<注释字符串> | -^commentsfile=<注释文件>]

标签规格 新标签名称。

（使用 'cm ^help ^objectspec' 可进一步了解标签

规格。）

变更集规格 要标记的变更集的名称或完整规格。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

工作区路径 要标记的工作区的路径。（工作区指向的

变更集将被标记。）

选项：

--^allxlinkedrepositories 在所有通过 Xlink 链接的存储库中创建新标签。

-^c 将指定的注释应用于新标签。

-^commentsfile 将指定文件中的注释应用于

新标签。

== CMD\_HELP\_LABEL\_CREATE ==

备注：

设置 PLASTICEDITOR 环境变量可指定用于键入注释的

编辑器。

示例：

cm ^label ^create ^lb:BL001 ^cs:1203 -^commentsfile=commentlb001.txt

（创建附加到变更集 1203 的标签 'BL001'，并应用 'commentlb001.txt'

文件中的注释。）

cm ^label BL002 ^cs:1203 -^c="首次发布"

（创建带有注释的标签 'BL002'，并附加到变更集 1203。）

== CMD\_DESCRIPTION\_LABEL\_DELETE ==

删除一个或多个标签。

== CMD\_USAGE\_LABEL\_DELETE ==

用法：

cm ^label ^delete <标签规格>[ ...]

标签规格 要删除的标签。使用空格对各个标签进行分隔。

（使用 'cm ^help ^objectspec' 可进一步了解标签

规格。）

== CMD\_HELP\_LABEL\_DELETE ==

备注：

此命令将删除一个或多个标签。

示例：

cm ^label ^delete ^lb:BL001

（删除标签 'BL001'。）

cm ^label ^delete ^lb:BL001 ^lb:BL002@reptest@server2:8084

（删除标签 'BL001' 和 'BL002'。）

== CMD\_DESCRIPTION\_LABEL\_RENAME ==

重命名标签。

== CMD\_USAGE\_LABEL\_RENAME ==

用法：

cm ^label ^rename <标签规格> <新名称>

标签规格 要重命名的标签。

（使用 'cm ^help ^objectspec' 可进一步了解标签规格。）

新名称 标签的新名称。

== CMD\_HELP\_LABEL\_RENAME ==

备注：

此命令将重命名标签。

示例：

cm ^label ^rename ^lb:BL001 BL002

（将标签 'BL001' 重命名为 'BL002'。）

== CMD\_DESCRIPTION\_OBLITERATE ==

已弃用

== CMD\_USAGE\_OBLITERATE ==

已弃用。

== CMD\_HELP\_OBLITERATE ==

已弃用。

== CMD\_DESCRIPTION\_OBJECTSPEC ==

描述如何编写对象规格。

== CMD\_USAGE\_OBJECTSPEC ==

用法：

cm ^objectspec

获取关于如何构建对象规格的所有信息。

== CMD\_HELP\_OBJECTSPEC ==

一些 Plastic SCM 命令要求将 '对象规格' 作为输入来引用

给定对象（通常是分支、变更集、存储库等）。

本文档介绍可用的不同“规格”以及如何

构建规格。

每个规格类型以唯一标记（例如 "^rep:" 或 "^cs:"）开头。必须为

采用一般对象规格的命令指定标记（例如：

"cm ^setowner 对象规格" 命令），但对于仅采用单一类型规格的命令

常常可以省略标记（例如："cm ^getfile 修订规格" 命令）。

-- 存储库服务器规格 --

^repserver:name:port

示例：

cm ^repo ^list ^repserver:skull:8084

cm ^repo ^list skull:8084

旁注：

由于历史原因，我们称之为“存储库服务器规格”，而不只是

“服务器规格”。很久以前，我们有单独的工作区服务器和

存储库服务器，命名方式沿用至今。

-- 存储库规格 --

^rep:rep\_name@[存储库服务器规格]

示例：

cm ^showowner ^rep:codice@localhost:6060

（此处的 "^rep:" 是必需的，因为 ^showowner 不仅容许存储库，

而且容许其他类型的对象。因此，需要用户指明

对象类型。）

-- 分支规格 --

^br:[/]br\_name[@repspec]

示例：

cm ^switch ^br:/main@^rep:plastic@^repserver:skull:9095

（这种情况下不需要 "^br:"、"^rep" 和 "^repserver"，所以

该命令容许简短很多的格式：

"cm ^switch main@plastic@skull:9095"。）

cm ^find ^revisions "^where ^branch='^br:/main/task001'"

备注：

分支上的初始 '/' 不是必需的。我们曾经将所有分支

指定为 /main、/main/task001，依此类推。但是现在，我们首选

更短的 main、main/task001 形式，这使命令更简洁。

-- 变更集规格 --

^cs:cs\_number|cs\_GUID[@repspec]

可以指定变更集的编号或 GUID。

示例：

cm ^ls /code --^tree=ae1390ed-7ce9-4ec3-a155-e5a61de0dc77@code@skull:7070

-- 标签规格 --

^lb:lb\_name[@repspec]

示例：

cm ^switch ^lb:RELEASE2.0

cm ^switch ^lb:RELEASE1.4@myrep@MYSERVER:8084

-- 修订规格 --

有不同类型的修订规格：

^rev:item\_path[#(分支规格|变更集规格|标签规格)]

^rev:^serverpath:item\_path#(分支规格|变更集规格|标签规格)

^rev:^revid:rev\_id[@rep\_spec]

^rev:^itemid:item\_id#(分支规格|变更集规格|标签规格)

示例：

cm ^diff ^rev:readme.txt#^cs:19 ^rev:readme.txt#^cs:20

cm ^diff ^serverpath:/doc/readme.txt#^cs:19@myrepo \

^serverpath:/doc/readme.txt#^br:/main@myrepo@localhost:8084

cm ^cat ^revid:1230@^rep:myrep@^repserver:myserver:8084

-- 项规格 --

^item:path

极少使用。

示例：

cm ^find ^revision "^where ^item='^item:.'"

-- 属性规格 --

^att:att\_name[@repspec]

示例：

cm ^attribute ^set ^att:merged@code@doe:8084 ^cs:25@code@doe:8084 完成

-- 搁置规格 --

^sh:sh\_number[@repspec]

示例：

cm ^diff ^sh:2 ^sh:4

-- 工作区规格 --

^wk:name@clientmachine

极少使用，因为仅适用于工作区相关的命令。适合用于

按名称和机器（而不是路径）来指定工作区。

示例：

cm ^showselector ^wk:codebase@modok

旁注：

这些规格来自早期的 Plastic SCM 2.x，当时的

“工作区服务器”是一种集中存储工作区元数据

的方式。后期由于性能问题而被弃用。

== CMD\_DESCRIPTION\_PARTIAL ==

在非完整的工作区中运行命令。

== CMD\_USAGE\_PARTIAL ==

用法：

cm ^partial <命令> [选项]

命令：

^configure

^add

^undo

^co | ^checkout

^unco | ^undocheckout

^ci | ^checkin

^mv | ^move

^rm | ^remove

^stb | ^switch

^upd | ^update

要获取有关每条命令的更多信息，请运行：

cm ^partial <命令> --^usage

cm ^partial <命令> --^help

== CMD\_HELP\_PARTIAL ==

示例：

cm ^partial ^configure +/background-blue.png

cm ^partial ^update landscape-1024.png

cm ^partial ^checkin eyes-green.png eyes-black.png

== CMD\_DESCRIPTION\_PARTIAL\_ADD ==

向版本控制中添加项。

== CMD\_USAGE\_PARTIAL\_ADD ==

用法：

cm ^partial ^add [-^R | -^r | --^recursive] [--^silent] [--^parents]

[--^ignorefailed] [--^skipcontentcheck] <项路径>[ ...]

项路径 要添加的项。使用双引号 (" ") 指定包含空格

的路径。使用空格对各个路径进行分隔。

使用 \* 添加当前目录的所有内容。

选项：

-^R 以递归方式添加项。

--^silent 不显示任何输出。

--^parents 在操作中包含指定项

的父目录。

--^ignorefailed 如果无法添加某一项，则添加操作将在没有

这一项的情况下继续进行。注意：如果无法添加

某个目录，则不会添加目录中的内容。

--^skipcontentcheck 如果不足以根据扩展名将文件设置为

文本或二进制，默认情况下会将该文件设置为二进制，

而不是检查内容来检测类型。

== CMD\_HELP\_PARTIAL\_ADD ==

备注：

添加项的要求：

- 要添加的项的父目录必须已事先添加。

示例：

cm ^partial ^add pic1.png pic2.png

（添加 'pic1.png' 和 'pic2.png' 项。）

cm ^partial ^add c:\workspace\picture.png

（添加路径 'c:\workspace' 中的 'picture.png' 项。）

cm ^partial ^add -^R c:\workspace\src

（以递归方式添加 'src'。）

cm ^partial ^add --^parents samples\design01.png

（添加 'design01.png' 文件和 'samples' 父文件夹。）

cm ^partial ^add -^R \*

（以递归方式添加当前目录的所有内容。）

== CMD\_DESCRIPTION\_PARTIAL\_CHECKIN ==

将更改存储在存储库中。

== CMD\_USAGE\_PARTIAL\_CHECKIN ==

用法：

cm ^partial ^checkin | ^ci [<项路径>[ ...]]

[-^c=<注释字符串> | -^commentsfile=<注释文件>]

[--^all | -^a] [--^applychanged] [--^keeplock]

[--^symlink] [--^ignorefailed]

选项：

项路径 要签入的项。使用双引号 (" ") 指定

包含空格的路径。使用空格对各个路径进行

分隔。

使用 . 将签入应用到当前目录。

-^c 指定在签入操作中创建的变更集

的注释。

-^commentsfile 将指定文件中的注释应用于在签入操作中

创建的变更集。

--^all | -^a 还包括在指定路径上进行了本地更改、移动

和删除的项。

--^applychanged 将签入操作应用于在工作区中

检测到的已更改项以及

签出的项。

--^keeplock 签入操作后保持锁定项的

锁定状态。

--^symlink 将签入操作应用于符号链接而不是

目标。

--^ignorefailed 无法应用的任何更改（因为无法获得

锁定状态（又称为“独占签出”）或者

因为本地更改与服务器更改冲突）

将被丢弃，然后没有这些更改的情况下

继续进行签入操作。

== CMD\_HELP\_PARTIAL\_CHECKIN ==

备注：

- 如果未指定 <项路径>，则签入操作将涉及

工作区中的所有待定更改。

- 签入操作始终以递归方式从给定路径应用。

- 要签入项，必须满足以下条件：

- 项必须受源代码管理。

- 必须签出项。

- 如果已更改但未签出项，则除非 <项路径> 是目录

或包含通配符 ('\*')，否则不需要

'--^applychanged' 标志。

修订内容必须与以前的修订不同才能

签入。

设置 PLASTICEDITOR 环境变量可指定用于键入注释的

编辑器。

从 stdin 读取输入：

'^partial ^checkin' 命令可从 stdin 读取路径。为此，请传递

一个破折号 "-"。

示例：cm ^partial ^checkin -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要签入的文件。

示例：

dir /S /B \*.c | cm ^partial ^checkin --^all -

（在 Windows 中，签入工作区中的所有 .c 文件。）

示例：

cm ^partial ^checkin figure.png landscape.png

（将签入应用于已签出的文件 'figure.png' 和 'landscape.png'。）

cm ^partial ^checkin .-^commentsfile=mycomment.txt

（将签入应用于当前目录并设置 'mycomment.txt' 文件

中的注释。）

cm ^partial ^ci background.png -^c="我的注释" --^keeplock

（将签入应用于 'background.png'，包含注释，并保持

锁定。）

cm ^partial ^checkin --^applychanged

（将签入应用于工作区中的所有待定更改。）

cm ^partial ^checkin 链接 --^symlink

（将签入应用于链接文件而非目标，适用于

UNIX 环境。）

cm ^partial ^checkin .--^ignorefailed

（将签入应用于当前目录，忽略无法应用的

更改。）

== CMD\_DESCRIPTION\_PARTIAL\_CHECKOUT ==

将文件标记为修改就绪。

== CMD\_USAGE\_PARTIAL\_CHECKOUT ==

用法：

cm ^partial ^checkout | ^co [<项路径>[ ...]] [--^resultformat=<格式字符串>]

[--^silent] [--^ignorefailed]

选项：

项路径 要签出的项。使用双引号 (" ") 指定

包含空格的路径。使用空格对各个路径进行

分隔。

使用 . 将签出应用到当前目录。

--^resultformat 检索特定格式的输出结果

消息。

--^silent 不显示任何输出。

--^ignorefailed 如果无法锁定某项（无法执行独占签出），

则签出操作将在没有这一项的情况下

继续进行。

== CMD\_HELP\_PARTIAL\_CHECKOUT ==

备注：

要签出项，必须满足以下条件：

- 项必须受源代码管理。

- 必须签入项。

如果在服务器上配置了锁（存在 lock.conf），则每次

在某条路径上进行签出时，Plastic 都会检查是否符合规则，

如果符合，该路径将处于独占签出（锁定）状态，以便任何其他人

都无法同时进行签出。

可使用 'cm ^lock ^list' 获取服务器中的所有锁。

请查看《管理员指南》以了解锁定的工作原理：

https://www.plasticscm.com/download/help/locking

示例：

cm ^partial ^checkout pic1.png pic2.png

（签出 'pic1.png' 和 'pic2.png' 文件。）

cm ^partial ^co \*.png

（签出所有 png 文件。）

cm ^partial ^checkout .

（签出当前目录。）

cm ^partial ^checkout -^R c:\workspace\src

（以递归方式签出 'src' 文件夹。）

== CMD\_DESCRIPTION\_PARTIAL\_CONFIGURE ==

可以通过在工作区中加载或卸载项来配置工作区。

== CMD\_USAGE\_PARTIAL\_CONFIGURE ==

用法：

cm ^partial ^configure <+|-路径>[ ...][--^silent] [--^ignorefailed]

[--^ignorecase] [--^restorefulldirs]

路径 要加载或卸载的路径。使用双引号 (" ")

指定包含空格的路径。使用空格对各个路径进行

分隔。

路径必须以 "/" 开头。

选项：

--^silent 不显示任何输出。

--^ignorefailed 跳过该过程中的所有错误。错误路径

不会导致命令停止。

--^ignorecase 忽略路径中的大小写。使用此标志后，

即使用户写入 "/data/teXtures"，'^configure' 也对

"/Data/Textures" 有效。

--^restorefulldirs 重置无效的目录配置（在非完整的工作区上

运行完整操作时发生）。

此列表中的目录将实现完全配置

（完全检查），这意味着这些目录将在

更新期间自动下载新内容。

此操作不会下载任何文件，只会

在非完整的工作区上还原目录

配置。

== CMD\_HELP\_PARTIAL\_CONFIGURE ==

备注：

该命令采用递归操作。

示例：

cm ^partial ^configure +/landscape\_grey.png

（加载 'landscape\_grey.png' 项。）

cm ^partial ^configure -/landscape\_black.png

（卸载 'landscape\_black.png' 项。）

cm ^partial ^configure +/soft -/soft/soft-black.png

（加载除 'soft-black.png' 以外的所有 'soft' 目录子项。）

cm ^partial ^configure -/

（卸载整个工作区。）

cm ^partial ^configure -/ +/

（加载整个工作区。）

cm ^partial ^configure -/figure-64.png --^ignorefailed

（即使已卸载 'figure-64.png' 项，也要卸载该项。）

cm ^partial ^configure +/ --^restorefulldirs

（设置所有目录来自动下载新内容。）

cm ^partial ^configure +/src/lib --^restorefulldirs

（仅设置 '/src/lib' 及其子目录来自动下载

新内容。）

== CMD\_DESCRIPTION\_PARTIAL\_MOVE ==

移动或重命名文件或目录。

== CMD\_USAGE\_PARTIAL\_MOVE ==

用法：

cm ^partial ^move | ^mv <源路径> <目标路径> [--^format=<格式字符串>]

源路径 源项的路径。

目标路径 目标项的路径。

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

== CMD\_HELP\_PARTIAL\_MOVE ==

备注：

此命令将移动或重命名存储库中的项。

还会在本地文件系统中进行更改。

如果源路径是文件，则目标路径可以是文件或

目录。在第一种情况下将重命名文件；否则将移动

相应的项。

如果源路径是目录，则目标路径必须是目录。

要移动或重命名的项必须已存在。

格式：

{0} 源路径。

{1} 目标路径。

示例：

cm ^partial ^move file.png file-blue.png

（重命名相应的项。）

cm ^partial ^mv .\file-blue.png .\blueFiles

（将 'file-blue.png' 移动到 'blueFiles'。）

cm ^partial ^move .\design .\marketing

（重命名目录。）

== CMD\_DESCRIPTION\_PARTIAL\_RM ==

从版本控制中删除一个文件或目录。

== CMD\_USAGE\_PARTIAL\_RM ==

用法：

cm ^partial ^remove | ^rm <项路径>[ ...][--^nodisk]

项路径 要删除的项路径。使用双引号 (" ")

指定包含空格的路径。使用空格对各个路径进行

分隔。

选项：

--^nodisk 从版本控制中删除相应的项，但是将该项保留在磁盘上。

== CMD\_HELP\_PARTIAL\_RM ==

备注：

磁盘中会删除项。删除的项会从源代码管理

中的父目录中被删除。

要求：

- 项必须受源代码管理。

示例：

cm ^partial ^remove src

（删除 'src'。如果 'src' 是目录，则等同于：

cm ^partial ^remove -^R src。）

cm ^partial ^remove c:\workspace\pic01.png --^nodisk

（从版本控制中删除 'pic01.png'，但将其保留在磁盘上。）

== CMD\_DESCRIPTION\_PARTIAL\_SWITCH ==

将分支设置为工作分支。

== CMD\_USAGE\_PARTIAL\_SWITCH ==

用法：

cm ^switch <分支规格> [--^report | --^silent] [--^workspace=<路径>]

（设置工作分支并更新工作区。）

cm ^switch <分支规格> --^configure <+|-路径>[ ...][--^silent]

[--^ignorefailed] [--^ignorecase] [--^workspace=<路径>]

（设置工作分支并运行工作区配置，就像 'cm

^partial ^configure' 命令一样。）

分支规格 分支的规格。（使用 'cm ^help ^objectspec' 可

进一步了解分支规格。）

路径 要加载或卸载的路径。使用双引号 (" ")

指定包含空格的路径。使用空格对各个路径

进行分隔。路径必须以 "/" 开头。

选项：

--^silent 不显示任何输出。

--^report 命令完成后，打印已应用更改

的列表。使用 '--^silent' 将会覆盖此设置。

此选项仅在未指定 '--^configure' 选项

时有效。

--^configure 更新工作分支后配置

（加载/卸载项）工作区。查看 'cm ^partial

^configure --^help' 可了解如何指定要配置的

路径。

--^ignorefailed 跳过配置过程中的所有错误。

错误路径不会导致命令停止。

--^ignorecase 忽略路径中的大小写。使用此标志后，即使

用户写入 "/data/teXtures"，选项 '--^configure' 也对

"/Data/Textures" 有效。

--^workspace=路径 工作区所在的路径。

== CMD\_HELP\_PARTIAL\_SWITCH ==

备注：

此命令允许用户更新工作分支。更新分支之后，

该命令就像 'cm ^partial ^update' 命令一样会

将工作区更新到新分支。然而，如果指定了 '--^configure'

选项，该命令就像 'cm ^partial ^configure' 命令一样会

允许使用新的分支配置来配置工作区。

示例：

cm ^switch ^br:/main/task

（将 /main/task 设置为工作分支并更新工作区。）

cm ^switch ^br:/main/task --^configure +/art/images

（将 /main/task 设置为工作分支，并配置工作区来

加载 /art/images 文件夹。）

== CMD\_DESCRIPTION\_PARTIAL\_UNCO ==

撤销项的签出。

== CMD\_USAGE\_PARTIAL\_UNCO ==

用法：

cm ^partial ^undocheckout | ^unco <项路径>[ ...][--^silent]

项路径 要应用操作的项。使用双引号 (" ")

指定包含空格的路径。使用空格对各个路径

进行分隔。

使用 . 将操作应用到当前目录。

选项：

--^silent 不显示任何输出。

== CMD\_HELP\_PARTIAL\_UNCO ==

备注：

如果已签出某个项并且不想将这个项签入，则可以

使用此命令撤销签出。文件和文件夹都可以取消

签出。项将更新为签出之前的状态。

要求：

- 项必须受源代码管理。

- 必须签出项。

示例：

cm ^partial ^undocheckout .

（撤销当前目录中的签出。）

cm ^partial ^undocheckout pic1.png pic2.png

cm ^unco c:\workspace\design01.png

（撤销签出所选文件。）

== CMD\_DESCRIPTION\_PARTIAL\_UNDO ==

撤销工作区中的更改。

== CMD\_USAGE\_PARTIAL\_UNDO ==

用法：

cm ^partial ^undo [<路径>[ ...]] [--^symlink] [-^r | --^recursive]

[<筛选器>[ ...]]

[--^silent | --^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

路径 要应用操作的文件或目录

的路径。使用双引号 (" ") 指定

包含空格的路径。使用空格对各个路径

进行分隔。

如果未指定路径，则默认情况下，

撤销操作将应用于当前目录中的所有

文件。

筛选器 将一个或多个指定筛选器应用于给定

路径。使用空格对各个筛选器进行分隔。请参阅

“筛选器”部分以了解更多信息。

选项：

--^symlink 将撤销操作应用于符号链接而不是

目标。

-^r 以递归方式执行撤销。

--^silent 不显示任何输出。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，指定

应如何分隔字段。

筛选器：

如果未指定标志，则默认情况下将撤销所有更改，但是

可以使用以下一个或多个标志来筛选路径。

如果文件或目录匹配一个或多个指定种类的更改，

则将撤销该文件或目录上的所有更改。

例如，如果同时指定了 '--^checkedout' 和 '--^moved'，则在

已签出并移动文件的情况下，两项更改都将被撤销。

--^checkedout 选择已签出的文件和目录。

--^unchanged 选择内容不变的文件。

--^changed 选择本地更改的或签出的文件和

目录。

--^deleted 选择已删除的文件和目录。

--^moved 选择已移动的文件和目录。

--^added 选择已添加的文件和目录。

== CMD\_HELP\_PARTIAL\_UNDO ==

备注：

^undo 命令有风险，因为该命令在撤销工作后将不可逆。

^undo 完成后，无法将受其影响的文件和目录

恢复到先前状态。如果参数中

未指定路径，则默认情况下，该命令将撤销

当前目录中的所有更改，但不采用递归方式。

从 /src 目录执行时，以下命令是等效的：

/src

|- file.txt

|- code.cs

\- /test

|- test\_a.py

\- test\_b.py

cm ^partial ^undo

cm ^partial ^undo \*

cm ^partial ^undo file.txt code.cs /test

cm ^partial ^undo .

cm ^partial ^undo /src file.txt code.cs

如果希望操作以递归方式执行，必须指定 '-^r' 标志。

撤销某一目录下的所有更改（包括影响目录

本身的更改）：

cm ^partial ^undo 目录路径 -^r

如果目录路径是工作区路径，则工作区中的每个更改都将

被撤销。

示例：

cm ^partial ^undo .-^r

（以递归方式撤销当前目录中的所有更改。如果从工作区的

根目录执行，则撤销整个工作区中的所有更改。）

cm ^partial ^co file.txt

cm ^partial ^undo file.txt

（撤销对 file.txt 的签出。）

^echo ^content >> file.txt

cm ^partial ^undo file.txt

（撤销对 file.txt 的本地更改。）

cm ^partial ^undo src

（撤销对 src 目录及其文件的更改。）

cm ^partial ^undo src/\*

（撤销对 src 中包含的每个文件和目录的更改，而不会

影响 src。）

cm ^partial ^undo \*.cs

（撤销对当前目录中与 \*.cs 匹配的每个文件或目录的

更改。）

cm ^partial ^undo \*.cs-^r

（撤销对当前目录以及下级每个目录中与 \*.cs 匹配的

每个文件或目录的更改。）

cm ^partial ^co file1.txt file2.txt

^echo ^content >> file1.txt

cm ^partial ^undo --^unchanged

（撤销对未更改的 file2.txt 的签出，并忽略本地更改的

file1.txt。）

^echo ^content >> file1.txt

^echo ^content >> file2.txt

cm ^partial ^co file1.txt

cm ^partial ^undo --^checkedout

（撤销已签出文件 file1.txt 中的更改，并忽略 file2.txt，

因为后者未签出。）

cm ^partial ^add file.txt

cm ^partial ^undo file.txt

（撤销添加 file.txt，使该文件再次成为私有文件。）

^rm file1.txt

^echo ^content >> file2.txt

cm ^partial ^add file3.txt

cm ^partial ^undo --^deleted --^added \*

（撤销 file1.txt 的删除和 file3.txt 的添加，并忽略 file2.txt

的更改。）

== CMD\_DESCRIPTION\_PARTIAL\_UPDATE ==

更新非完整的工作区并下载最新的更改。

== CMD\_USAGE\_PARTIAL\_UPDATE ==

用法：

cm ^partial ^update [<项路径>[ ...]] [--^changeset=<编号>]

[--^silent | --^report] [--^dontmerge]

项路径 要更新的项。使用双引号 (" ") 指定

包含空格的路径。使用空格对各个路径进行

分隔。

使用 . 将更新应用到当前目录。

如果未指定路径，则当前非完整的

工作区将完全更新。

选项：

--^changeset 将非完整的工作区更新到特定变更集。

--^silent 不显示任何输出。

--^report 命令完成后，打印已应用更改

的列表。使用 '--^silent' 将会覆盖此设置。

--^dontmerge 不合并文件冲突，只是跳过这些冲突。

其他更改会正确应用。此选项可用于

自动化，从而避免用户交互。

== CMD\_HELP\_PARTIAL\_UPDATE ==

备注：

'^partial ^update' 命令将更新已过期的文件。

该命令采用递归操作。

在使用 '--^changeset' 选项时，如果所有指定的路径都是

同一 Xlink 内的文件，则会在通过 Xlink 链接的存储库的

指定变更集内搜索要下载的版本。

示例：

cm ^partial ^update

（更新当前非完整的工作区中的所有内容。）

cm ^partial ^update .

（更新所有当前目录子项。）

cm ^partial ^update backgroud-blue.png

（更新 'backgroud-blue.png' 项。）

cm ^partial ^update soft\_black.png soft-grey.png

（更新 'soft\_black.png' 和 'soft-grey.png' 项。）

cm ^partial ^update src --^report

（更新所有 'src' 目录子项，并在最后打印已应用的

更改列表。）

cm ^partial ^update src --^changeset=4

（将所有 'src' 目录子项更新为变更集 4 中

加载的相应内容。）

cm ^partial ^update xlink/first.png --^changeset=4

（将 'xlink/first.png' 项更新为通过 Xlink 链接的存储库的变更集 4 中

加载的相应内容。）

== CMD\_DESCRIPTION\_PATCH ==

从规格生成一个补丁文件，或者将生成的补丁应用于当前

工作区。

== CMD\_USAGE\_PATCH ==

用法：

cm ^patch <源规格> [<源规格>] [--^output=<输出文件>]

[--^tool=<差异比较路径>]

生成一个补丁文件，该文件包含分支的差异差、

变更集的差异差或变更集之间的差异。还用于跟踪

文本文件和二进制文件的差异。

cm ^patch --^apply <补丁文件> [--^tool=<修补路径>]

允许应用当前工作区中生成的补丁文件的

内容。

源规格 变更集或分支的完整规格。（使用

'cm ^help ^objectspec' 可进一步了解规格。）

输出文件 用于保存补丁内容的文件。如果未指定文件，

则会在标准输出中打印补丁内容。

补丁文件 要在当前工作区中应用的补丁文件。

选项：

--^output 设置 patch 命令的输出文件。

--^tool 设置要使用的应用程序（差异比较还是修补）。

== CMD\_HELP\_PATCH ==

限制：

如果输出补丁文件已经存在，该命令不会覆盖这个文件。

应用补丁时，如果磁盘上没有已修改的文件，该命令不会

将更改应用于这些文件。

重要信息：

此命令需要差异比较工具和修补工具；可从以下网址公开获得这些工具：

http://gnuwin32.sourceforge.net/packages/patch.htm 和

http://gnuwin32.sourceforge.net/packages/diffutils.htm

安装后，建议将工具的位置添加到 PATH

环境变量。

示例：

cm ^patch ^cs:4@default@localhost:8084

（以统一格式在控制台上打印变更集 4 的差异。）

cm ^patch ^br:/main --^output=file.patch

（使用分支 "main" 的差异来生成补丁文件。）

cm ^patch ^br:/main --^output=file.patch --^tool=C:\gnu\diff.exe

（同上，使用的是自定义执行程序。）

cm ^patch ^cs:2@default ^cs:4@default

（以统一格式在控制台上打印变更集 2 和 4 之间的差异。）

cm ^patch --^apply file.patch --^tool=C:\gnu\patch.exe

（使用自定义执行程序将 'file.patch' 中的补丁应用于本地工作区。）

== CMD\_DESCRIPTION\_QUERY ==

执行 SQL 查询。需要 SQL 存储。

== CMD\_USAGE\_QUERY ==

用法：

cm ^query <SQL\_命令> [--^outputfile=<输出文件>]

[--^solveuser=<列名称>[,...]]

[--^solvepath=<列名称>[,...]]

[--^columnwidth=<值>] [--^nocolumnname]

[--^columnseparator=<分隔符>] [--^repository=<名称>]

SQL\_命令 要执行的 SQL 查询。

选项：

--^outputfile 将结果写入输出文件中。

--^solveuser 将指定的列设置为用户名列。查询

解释器将假定这些列的数据将是用户，

并将尝试解析这些数据。

--^solvepath 将指定的列设置为 itemid 列。查询

解释器将尝试将项 ID 解析为文件系统

路径。

--^columnwidth 指定每列的宽度以便格式化输出。

--^nocolumnname 不打印列名称。

--^columnseparator 使用字符作为列分隔符，而不是使用制表符 (\t)。

--^repository 要查询的存储库。

== CMD\_HELP\_QUERY ==

备注：

此命令允许用户在服务器数据库中执行 SQL 查询。

为了编写 SQL 查询，请使用以下两个预定义的函数来管理

用户和路径：

- '^SolveUser(<用户名>)'，将用户名解析为 Plastic SCM 格式。

- '^SolvePath(<路径>)'，将磁盘路径解析为项 ID。

此外，还可使用选项以便于人工阅读的形式显示查询结果。

可以使用选项 '--^solveuser=<列名称>' 和

'--^solvepath=<列名称>' 来指定必须由查询解释器

转换为易读文本的列。可以指定多个列名称，

以逗号分隔。

示例：

cm ^query "^SELECT \* ^FROM ^revision" --^columnwidth=25 --^repository=reptest

（从存储库 'reptest' 中的 '修订' 表检索数据。）

cm ^query "^SELECT b.^sname ^as 分支名称, o.^dtimestamp ^as 日期 ^from ^branch b, \

^object o, ^seid s ^where b.^iobjid=o.^iobjid ^and o.^fidowner=s.^iseidid ^and \

s.^scode='^SolveUser(john)'" --^outputfile=query.txt

（将所有者为 'john' 的分支输出到文件中。）

cm ^query "^select r.^iobjid, r.^fiditem ^as 路径, s.^scode ^as 用户名 ^FROM \

^revision r, ^object o, ^seid s ^WHERE r.^iobjid=o.^iobjid ^and \

o.^fidowner=s.^iseidid ^and o.^dtimestamp>2014/04/25" \

--^solveuser=用户名 --^solvepath=路径 --^repository=reptest@server2:9095

（从所选存储库检索选定数据。）

cm ^query "^SELECT \* ^FROM ^revision ^WHERE ^fiditem=^SolvePath(c:\mywkpath\info)"

（检索路径 'info' 的所有修订数据。）

== CMD\_DESCRIPTION\_ATTRIBUTE\_DELETE ==

删除一个或多个属性。

== CMD\_USAGE\_ATTRIBUTE\_DELETE ==

用法：

cm ^attribute | ^att ^delete | ^rm <属性规格>[ ...]

属性规格 要删除的属性。使用空格对各个属性进行

分隔。

（使用 'cm ^help ^objectspec' 可进一步了解属性

规格。）

== CMD\_HELP\_ATTRIBUTE\_DELETE ==

备注：

此命令将删除一个或多个属性。

示例：

cm ^attribute ^delete ^att:status

（删除属性 '状态'。）

cm ^att ^rm 状态 ^att:integrated@reptest@server2:8084

（删除属性 '状态' 和 '集成'。）

== CMD\_DESCRIPTION\_ATTRIBUTE\_UNSET ==

取消设置对象的属性。

== CMD\_USAGE\_ATTRIBUTE\_UNSET ==

用法：

cm ^attribute | ^att ^unset <属性规格> <对象规格>

属性规格 属性的规格。（使用 'cm ^help ^objectspec' 可

进一步了解属性规格。）

对象规格 要删除相应属性的对象

的规格。可以在以下对象上设置属性：分支、变更集、

搁置集、标签、项和修订。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

== CMD\_HELP\_ATTRIBUTE\_UNSET ==

备注：

该命令将取消设置先前在对象上设置的某个属性。该命令

不会删除属性对象本身。

示例：

cm ^attribute ^unset ^att:status ^br:/main/SCM105

（从分支 'main/SCM105' 删除属性实现 '状态'。）

cm ^att ^unset ^att:integrated@reptest@localhost:8084 ^cs:25@reptest@localhost:8084

（从变更集 25 删除属性实现 '集成'，全都在

存储库 'reptest' 中。）

== CMD\_DESCRIPTION\_ATTRIBUTE\_RENAME ==

为属性重命名。

== CMD\_USAGE\_ATTRIBUTE\_RENAME ==

用法：

cm ^attribute | ^att ^rename <属性规格> <新名称>

属性规格 要重命名的属性。（使用 'cm ^help ^objectspec' 可

进一步了解属性规格。）

新名称 属性的新名称。

== CMD\_HELP\_ATTRIBUTE\_RENAME ==

备注：

此命令将重命名一个属性。

示例：

cm ^attribute ^rename ^att:status state

（将属性 '状态' 重命名为 '状况'。）

== CMD\_DESCRIPTION\_ATTRIBUTE\_EDIT ==

编辑属性的注释。

== CMD\_USAGE\_ATTRIBUTE\_EDIT ==

用法：

cm ^attribute | ^att ^edit <属性规格> <新注释>

属性规格 要更改相应注释的属性。（使用 'cm ^help ^objectspec'

可进一步了解属性规格。）

新注释 属性的新注释。还可以为属性指定

一个默认值列表。

== CMD\_HELP\_ATTRIBUTE\_EDIT ==

备注：

此命令将更改属性的注释。

要指定属性的默认值列表，只需将

如下的行包含在属性注释中：

'default: 值\_一, "值二", 值3, "最终值"'。

示例：

cm ^attribute ^edit ^att:status "CI 管道中分支的状态。"

（编辑属性 '状态' 的注释。）

cm ^attribute ^edit ^att:status "分支的状态。默认值：未完成、已解决、已审查"

（编辑属性 '状态' 的注释。还指定一个值

列表。因此，当属性 '状态' 设置为对象时，可以选择

以下值之一："未完成"、"已解决" 或 "已审查"。）

== CMD\_DESCRIPTION\_REPLICATE ==

警告：已弃用此命令。

请使用 'cm ^pull'（等同于 '^replicate'）和 'cm ^push'（等同于

'^replicate --^push'）。

== CMD\_USAGE\_REPLICATE ==

== CMD\_HELP\_REPLICATE ==

== CMD\_DESCRIPTION\_PULL ==

从另一个存储库拉取分支。

== CMD\_USAGE\_PULL ==

用法：

cm ^pull <源分支规格> <目标存储库规格>

[--^preview] [--^nodata] [转换选项]

[--^user=<用户名> [--^password=<密码>] | 身份验证选项]

（服务器之间的直接复制。从存储库拉取分支。）

cm ^pull <目标存储库规格> --^package=<包文件> [身份验证选项]

（基于包的复制。将包导入目标存储库中。）

cm ^pull ^hydrate <目标分支规格> [<源存储库规格>]

[--^user=<用户名> [--^password=<密码>] | 身份验证选项]

（为分支中先前使用 '--^nodata' 复制的所有变更集

引入缺失的数据。如果未指定用于获取数据的存储库，

Plastic 会尝试使用“复制源”（复制分支

的来源））。

cm ^pull ^hydrate <目标变更集规格> [<源存储库规格>]

[--^user=<用户名> [--^password=<密码>] | 身份验证选项]

（为先前使用 '--^nodata' 复制的变更集引入缺失的

数据。如果未指定用于获取数据的存储库，Plastic 会尝试

使用“复制源”）。

源分支规格 要从远程存储库拉取的分支。

（使用 'cm ^help ^objectspec' 可进一步了解分支规格。）

目标分支规格 要进行 hydrate 的分支。

（使用 'cm ^help ^objectspec' 可进一步了解分支规格。）

目标变更集规格 要进行 hydrate 的变更集。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

目标存储库规格 目标存储库。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

--^package 指定要导入的先前创建的包文件以进行

基于包的复制。

适合在没有直接网络连接时用于在服务器

之间移动数据。

请参阅 'cm ^push' 以创建包文件。

选项：

--^preview 提供关于将拉取哪些更改的信息，

但实际上不进行任何更改。此选项

可用于检查在复制更改之前要传输

的数据。

--^nodata 复制分支更改而不复制

数据。复制包时，不允许使用

此选项。

转换选项 请参阅“转换选项”部分以了解更多信息。

--^user, --^password 源和目标的身份验证模式不同并且

没有可用于目标身份验证的配置文件

的情况下使用的凭据。

身份验证选项 请参阅“身份验证选项”部分以了解更多

信息。

转换选项：

--^trmode=(^copy|^name|^table --^trtable=<转换表文件>)

源和目标存储库可能使用不同的身份验证

模式。'--^trmode' 选项指定如何将用户名从源

转换到目标。'--^trmode' 必须为以下值

之一：

^copy （默认值）。表示只会复制用户标识符。

^name 按名称来匹配用户标识符。

^table 使用选项 '--^trtable' 中指定的转换表

（见下文）。

--^trtable=<转换表文件>

如果转换模式为 'table'，则转换表文件中包含

格式为 <旧名称;新名称> 的行（每行一个）。将分支

写入目标存储库时，源存储库中由“旧名称”

标识的用户所创建的对象将设置到目标上

具有“新名称”的用户。

身份验证选项：

可以使用以下两种模式之一来指定身份验证数据：

1) 使用身份验证参数：--^authmode=<模式> --^authdata=<数据>

--^authmode=(^NameWorkingMode|^LDAPWorkingMode|^ADWorkingMode|^UPWorkingMode)

示例：

(^LDAPWorkingMode) --^authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(^UPWorkingMode) --^authdata=dave:fPBea2rPsQaagEW3pKNveA==

'--^authdata' 行是 client.conf 文件和 profiles.conf 文件

中的 <^SecurityConfig> 条目的内容。可以从

Plastic GUI 中的“首选项”下的“复制配置文件”选项卡

轻松生成 profiles.conf 文件。

如果使用 ^UPWorkingMode，则只需指定：

--^authmode=^UPWorkingMode --^user=<用户> --^password=<密码>

2) 身份验证文件，对于连接到的每个服务器，文件可能会

有所不同，其中包含该服务器的凭据。

--^authfile=<身份验证文件>

该文件包含 2 行：

第 1 行是模式，如 '--^authmode' 中所述

第 2 行是身份验证数据，如 '--^authdata' 中所述

== CMD\_HELP\_PULL ==

备注：

'^pull' 命令可以在源存储库与目标存储库之间

复制分支（以及这些分支的变更集）。

这些存储库可以位于不同服务器中。

有两种复制操作：'^push' 和 '^pull'。

'^pull' 操作表示复制操作要求将数据从源存储库

存储到目标存储库。

客户端将连接到目标存储库，然后建立

从该主机到源存储库的连接以便检索

目标数据。在拉取期间，目标服务器将

连接到源。

尽管在典型的分布式场景中，开发者会将数据

从本地服务器推送到主服务器，但开发者可能也希望

从主服务器拉取最新的存储库更新。

复制操作可以解决在同一分支上两个复制的存储库

已进行并发更改的情况：

- 推送：如果尝试将数据推送到一个存储库，但这个存储库中的

更改比发送的更改更新，则系统会要求提取最新的更改，

解决合并操作问题，最后尝试再次推送。

- 拉取：每次从远程分支拉取变更集时，都会将这些变更集

正确链接到相应的父变更集。如果拉取的变更集不是

分支中最后一个变更集的子级，则会出现

一种多头场景。该分支将具有多个“头部”，即分支中的

最后一个变更集。此时需要先合并两个“头部”，

然后才能再次推送。

可以按两种模式进行拉取：

1) 服务器之间的直接通信：目标服务器将

从源服务器提取数据，自动为指定

分支同步数据。

2) 使用推送和 '--^package' 选项来导入先前生成的包。

模式 1) 要求远程服务器对运行命令的用户进行身份验证，

方法是使用 client.conf 文件中的默认身份验证，

或者指定 '--^authmode' 和 '--^authdata' 修饰符

或 '--^authmode'、'--^user' 和 '--^password'（如果身份验证模式为

^UPWorkingMode）。

模式 2) 要求使用先前通过推送命令生成的

包文件。

请记住，拉取复制是以间接方式进行的。执行时，

该命令要求目标存储库连接到源

并获取所选分支。

然而，可以使用推送命令来直接实现这一点。

因此，该命令会将所选分支从源复制到

目标。

示例：

cm ^pull ^br:/main@project1@remoteserver:8084 projectx@myserver:8084

（将 'main' 分支从 'remoteserver' 拉取到 'myserver'。这种情况下，

两个服务器会配置为相同的身份验证模式。）

cm ^pull ^br:/main@project1@remoteserver:8084 projectx@myserver:8084 \

--^authmode=^LDAPWorkingMode --^authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

（拉取与以前相同的分支，但现在将远程服务器配置为

使用 Active Directory 对用户进行身份验证。例如，我正在从 Linux

计算机连接到配置为使用 Active Directory 集成模式

的 Windows 服务器。我会指定自己的 Active Directory 用户名和密码，

并以 LDAP 形式将此信息传递给服务器。）

cm ^pull ^br:/main@project1@remoteserver:8084 projectx@myserver:8084 \

--^authmode=^UPWorkingMode --^user=dave --^password=mysecret

（拉取同一个分支，但现在利用 Plastic SCM 中

包含的用户名/密码数据库在远程服务器上对用户

进行身份验证。）

cm ^pull ^br:/main@project1@remoteserver:8084 projectx@myserver:8084 --^nodata

（将 'main' 分支从 'remoteserver' 复制到 'myserver' 而不复制数据。）

cm ^pull ^hydrate ^br:/main@projectx@myserver:8084 projectx@remoteserver:8084

（从远程服务器获取数据，对 'main' 分支中的所有变更集

进行 hydrate 操作。）

cm ^pull ^hydrate ^cs:122169@projectx@myserver:8084 projectx@remoteserver:8084

（从远程服务器获取数据，对 'myserver' 中的变更集 122169 进行

hydrate 操作。）

== CMD\_DESCRIPTION\_PUSH ==

将分支推送到另一个存储库。

== CMD\_USAGE\_PUSH ==

用法：

cm ^push <源分支规格> <目标存储库规格>

[--^preview] [转换选项]

[--^user=<用户名> [--^password=<密码>] | 身份验证选项]

（服务器之间的直接复制。从存储库推送分支。）

cm ^push <源分支规格> --^package=<包文件> [身份验证选项]

（基于包的复制。使用所选分支在源服务器中

创建复制包。）

源分支规格 要推送到远程存储库的分支。

（使用 'cm ^help ^objectspec' 可进一步了解分支规格。）

目标存储库规格 目标存储库。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

--^package 指定复制包的导出路径，从而进行

基于包的复制。

适合在没有直接网络连接时用于在服务器

之间移动数据。

选项：

--^preview 提供关于将推送哪些更改的信息，

但实际上不进行任何更改。此选项

可用于检查在复制更改之前要传输

的数据。

转换选项 请参阅“转换选项”部分以了解更多信息。

--^user, --^password 源和目标的身份验证模式不同并且

没有可用于目标身份验证的配置文件

的情况下使用的凭据。

身份验证选项 请参阅“身份验证选项”部分以了解更多

信息。

转换选项：

--^trmode=(^copy|^name|^table --^trtable=<转换表文件>)

源和目标存储库可能使用不同的身份验证

模式。'--^trmode' 选项指定如何将用户名从源

转换到目标。'--^trmode' 必须为以下值

之一：

^copy （默认值）。表示只会复制用户标识符。

^name 按名称来匹配用户标识符。

^table 使用选项 '--^trtable' 中指定的转换表

（见下文）。

--^trtable=<转换表文件>

如果转换模式为 'table'，则转换表文件中包含

格式为 <旧名称;新名称> 的行（每行一个）。将分支

写入目标存储库时，源存储库中由“旧名称”

标识的用户所创建的对象将设置到目标上

具有“新名称”的用户。

身份验证选项：

可以使用以下两种模式之一来指定身份验证数据：

1) 使用身份验证参数：--^authmode=<模式> --^authdata=<数据>

--^authmode=(^NameWorkingMode|^LDAPWorkingMode|^ADWorkingMode|^UPWorkingMode)

示例：

(^LDAPWorkingMode) --^authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(^UPWorkingMode) --^authdata=dave:fPBea2rPsQaagEW3pKNveA==

'--^authdata' 行是 client.conf 文件和 profiles.conf 文件

中的 <^SecurityConfig> 条目的内容。可以从

Plastic GUI 中的“首选项”下的“复制配置文件”选项卡

轻松生成 profiles.conf 文件。

如果使用 ^UPWorkingMode，则只需指定：

--^authmode=^UPWorkingMode --^user=<用户> --^password=<密码>

2) 身份验证文件，对于连接到的每个服务器，文件可能会

有所不同，其中包含该服务器的凭据。

--^authfile=<身份验证文件>

该文件包含 2 行：

第 1 行是模式，如 '--^authmode' 中所述

第 2 行是身份验证数据，如 '--^authdata' 中所述

== CMD\_HELP\_PUSH ==

备注：

'^push' 命令可以在源存储库与目标存储库之间

复制分支（以及这些分支的变更集）。

这些存储库可以位于不同服务器中。

有两种复制操作：'^push' 和 '^pull'。

'^push' 操作表示复制操作会将数据从源存储库

发送到目标存储库。在这种情况下，

客户端将连接到源存储库，获取要复制的数据，

然后将数据发送到目标存储库。虽然

前者（源）必须与目标连接，但是后者（目标）

本身不会连接到源。

在典型的分布式场景中，开发者会将数据从本地服务器

推送到主服务器。此外，开发者可能也希望从主服务器

拉取最新的存储库更新。

复制可以解决在两个复制存储库的同一分支上

已进行并发更改的情况。

- 推送：如果尝试将数据推送到一个存储库，但这个存储库中的

更改比发送的更改更新，则系统会要求提取最新的更改，

解决合并操作问题，最后尝试再次推送。

- 拉取：每次从远程分支拉取变更集时，都会将这些变更集

正确链接到相应的父变更集。如果拉取的变更集不是

分支中最后一个变更集的子级，则会出现

一种多头场景。该分支将具有多个“头部”，即分支中的

最后一个变更集。此时需要先合并两个“头部”，

然后才能再次推送。

可以按两种模式进行推送：

1) 服务器之间的直接通信：源服务器将

把数据发送到目标服务器，自动为指定

分支同步数据。

2) 导出包模式：客户端将仅连接到源

并获取指定分支的数据和元数据来生成一个

复制包。此模式下将使用 '--^package' 修饰符。

这两种模式都要求服务器对运行命令的用户进行身份验证，

方法是使用 client.conf 文件中的默认身份验证，

或者指定 '--^authmode' 和 '--^authdata' 修饰符。

^push 推送复制是以直接方式进行的。执行时，

该命令将所选分支从源复制到目标

（而不是要求目标存储库连接到源）并获取

所选分支（与拉取操作一样）。

示例：

cm ^push ^br:/main@project1@myserver:8084 projectx@remoteserver:8084

（将 'main' 分支从 'myserver' 复制到 'remoteserver'。这种情况下，

两个服务器会配置为相同的身份验证模式。）

cm ^push ^br:/main@project1@myserver:8084 projectx@remoteserver:8084 \

--^authmode=^LDAPWorkingMode --^authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

（复制与以前相同的分支，但现在将远程服务器配置为

使用 Active Directory 对用户进行身份验证。例如，我正在从 Linux

计算机连接到配置为使用 Active Directory 集成模式

的 Windows 服务器。我会指定自己的 Active Directory 用户名和密码，

并以 LDAP 形式将此信息传递给服务器。）

cm ^push ^br:/main@project1@myserver:8084 projectx@remoteserver:8084 \

--^authmode=^UPWorkingMode --^user=dave --^password=mysecret

（复制同一个分支，但现在利用 Plastic SCM 中

包含的用户名/密码数据库在远程服务器上对用户

进行身份验证。）

== CMD\_DESCRIPTION\_CLONE ==

克隆远程存储库。

== CMD\_USAGE\_CLONE ==

用法：

cm ^clone <源存储库规格> [<目标存储库规格> | <目标存储库服务器规格>]

[--^user=<用户名> [--^password=<密码>] | 身份验证选项]

[转换选项]

（存储库之间的直接克隆。）

cm ^clone <源存储库规格> --^package=<包文件>

[--^user=<用户名> [--^password=<密码>] | 身份验证选项]

（克隆到中间包，随后可以使用拉取操作将中间包

导入到目标存储库。）

源存储库规格 克隆操作的源存储库。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

目标存储库规格 克隆操作的目标存储库。如果

已存在，必须为空。如果不存在，

则会进行创建。

如果未指定，该命令将使用用户的

默认存储库服务器。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

规格。）

目标存储库服务器规格 克隆操作的目标存储库服务器。

如果存在与目标存储库服务器中的

<源存储库规格> 同名的存储库，此存储库

必须为空。如果没有，则会进行创建。

如果未指定，该命令将使用用户的

默认存储库服务器。

（使用 'cm ^help ^objectspec' 可进一步了解存储库

服务器规格。）

选项：

--^user, --^password 源和目标的身份验证模式不同并且

没有可用于目标身份验证的配置文件

的情况下使用的凭据。

--^package 将指定的存储库导出到包文件，

而不是存储库。

适合在没有直接网络连接时用于在服务器

之间移动数据。

必须使用拉取命令来导入

生成的包。

转换选项 请参阅“转换选项”部分以了解更多信息。

身份验证选项 请参阅“身份验证选项”部分以了解更多

信息。

转换选项：

--^trmode=(^copy|^name|^table --^trtable=<转换表文件>)

源和目标存储库可能使用不同的身份验证

模式。'--^trmode' 选项指定如何将用户名从源

转换到目标。'--^trmode' 必须为以下值

之一：

^copy （默认值）。表示只会复制用户标识符。

^name 按名称来匹配用户标识符。

^table 使用选项 '--^trtable' 中指定的转换表

（见下文）。

--^trtable=<转换表文件>

如果转换模式为 'table'，则转换表文件中包含

格式为 <旧名称;新名称> 的行（每行一个）。将分支

写入目标存储库时，源存储库中由“旧名称”

标识的用户所创建的对象将设置到目标上

具有“新名称”的用户。

身份验证选项：

可以使用以下两种模式之一来指定身份验证数据：

1) 使用身份验证参数：--^authmode=<模式> --^authdata=<数据>

--^authmode=(^NameWorkingMode|^LDAPWorkingMode|^ADWorkingMode|^UPWorkingMode)

示例：

(^LDAPWorkingMode) --^authdata=::0:dave:fPBea2rPsQaagEW3pKNveA

(^UPWorkingMode) --^authdata=dave:fPBea2rPsQaagEW3pKNveA==

'--^authdata' 行是 client.conf 文件和 profiles.conf 文件

中的 <^SecurityConfig> 条目的内容。可以从

Plastic GUI 中的“首选项”下的“复制配置文件”选项卡

轻松生成 profiles.conf 文件。

如果使用 ^UPWorkingMode，则只需指定：

--^authmode=^UPWorkingMode --^user=<用户> --^password=<密码>

2) 身份验证文件，对于连接到的每个服务器，文件可能会

有所不同，其中包含该服务器的凭据。

--^authfile=<身份验证文件>

该文件包含 2 行：

第 1 行是模式，如 '--^authmode' 中所述

第 2 行是身份验证数据，如 '--^authdata' 中所述

== CMD\_HELP\_CLONE ==

备注：

克隆命令可以将分支（以及这些分支的变更集、

标签、属性、审查等）从源存储库复制到

目标存储库。这些存储库可以位于不同服务器中。

可以预先创建目标存储库，但是如果此存储库包含

以前的数据，克隆操作会失败。

克隆操作不会克隆存储库子模块，也不会克隆 Xlink 下的

存储库。

示例：

cm ^clone awesomeProject@tardis@cloud

（将 'awesomeProject' 存储库从 'tardis@cloud' 组织

克隆到同名的本地存储库。）

cm ^clone repo@server.home:9095 repo-local

（将 'repo' 从 'server.home:9095' 克隆到用户的默认存储库服务器

中的 'repo-local'。）

cm ^clone project@192.168.111.130:8084 ^repserver:192.168.111.200:9095

（将 'project' 存储库从 '192.168.111.130:8084' 克隆到

'project@192.168.111.200:9095'。）

cm ^clone project@ldapserver:8084 --authfile=credentials.txt \

--^trmode=table --^trtable=table.txt

（从 'ldapserver:8084' 克隆 'project' 存储库，使用针对

远程存储库的身份验证文件，并按照指定的

转换表来转换用户。）

cm ^clone project@server.home:9095 --^package=project.plasticpkg

cm ^repository ^create project@mordor.home:8084

cm ^pull --^package=project.plasticpkg project@mordor.home:8084

（将 'project' 存储库从 'server.home:9095' 克隆到包

'project.plasticpkg'，随后通过拉取操作将这个包导入到

'mordor.home:8084' 的'project' 存储库。）

== CMD\_DESCRIPTION\_REVERT ==

将项还原到先前的修订。

== CMD\_USAGE\_REVERT ==

用法：

cm ^revert <修订规格>

修订规格 包含修订（修订的内容将加载到工作区中）

的变更集的规格。

（使用 'cm ^help ^objectspec' 可进一步了解修订

规格。）

== CMD\_HELP\_REVERT ==

备注：

必须签入项。

示例：

cm ^revert dir#^cs:0

cm ^revert C:\mywks\dir\file1.txt#23456

== CMD\_DESCRIPTION\_REVISION\_HISTORY ==

显示文件或目录的历史记录。

== CMD\_USAGE\_REVISION\_HISTORY ==

用法：

cm ^history | ^hist <项路径>[ ...][--^long | --^format=<格式字符串>]

[--^symlink] [--^xml[=<输出文件>]] [--^encoding=<名称>]

项路径 项的路径。使用空格对各个路径进行分隔。使用

双引号 (" ") 指定包含空格的路径。

路径也可以是服务器路径修订。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

选项：

--^long 显示其他信息。

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

--^symlink 将历史记录操作应用于符号链接而不是

目标。

--^xml 以 XML 格式将输出打印到标准输出。

可以指定输出文件。

--^encoding 与 '--^xml' 选项结合使用，指定要在 XML 输出

中使用的编码（例如：utf-8）。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

== CMD\_HELP\_REVISION\_HISTORY ==

备注：

此命令将显示给定项的修订列表，以及每个修订的标签、

分支和注释信息。

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} | {^date} 日期。

{1} | {^changesetid} 变更集编号。

{2} | {^branch} 分支。

{4} | {^comment} 注释。

{5} | {^owner} 所有者。

{6} | {^id} 修订 ID。

{7} | {^repository} 存储库。

{8} | {^server} 服务器。

{9} | {^repspec} 存储库规格。

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

示例：

cm ^history file1.txt "file 2.txt"

cm ^hist c:\workspace --^long

（显示所有信息。）

cm ^history 链接 --^symlink

（将历史记录操作应用于 '链接' 文件而不是目标，

适用于 UNIX 环境。）

cm ^history ^serverpath:/src/foo/bar.c#^br:/main/task001@myserver

（从给定分支中的服务器路径检索修订历史记录。）

== CMD\_DESCRIPTION\_REVISION\_TREE ==

显示项的修订树。

== CMD\_USAGE\_REVISION\_TREE ==

用法：

cm ^tree <路径> [--^symlink]

路径 项路径。

选项：

--^symlink 将操作应用于链接文件而不是目标。

== CMD\_HELP\_REVISION\_TREE ==

示例：

cm ^tree fichero1.txt

cm ^tree c:\workspace

cm ^tree 链接 --^symlink

（将操作应用于链接文件而不是目标；适用于

UNIX 环境。）

== CMD\_DESCRIPTION\_RM ==

允许用户删除文件和目录。

== CMD\_USAGE\_RM ==

用法：

cm ^remove | ^rm <命令> [选项]

命令：

^controlled（可选）

^private

要获取有关每条命令的更多信息，请运行：

cm ^remove <命令> --^usage

cm ^remove <命令> --^help

== CMD\_HELP\_RM ==

示例：

cm ^remove \path\controlled\_file.txt

cm ^remove ^private \path\private\_file.txt

== CMD\_DESCRIPTION\_RM\_CONTROLLED ==

从版本控制中删除一个文件或目录。

== CMD\_USAGE\_RM\_CONTROLLED ==

用法：

cm ^remove | ^rm <项路径>[ ...][--^format=<格式字符串>]

[--^errorformat=<格式字符串>] [--^nodisk]

项路径 要删除的项路径。使用双引号 (" ") 指定

包含空格的路径。使用空格对各个路径进行

分隔。

选项：

--^format 检索特定格式的输出进度

消息。请参阅“示例”以了解更多信息。

--^errorformat 检索特定格式的错误消息

（如果有）。请参阅“示例”以了解更多信息。

--^nodisk 从版本控制中删除相应的项，但是将该项保留在

磁盘上。

== CMD\_HELP\_RM\_CONTROLLED ==

备注：

磁盘中会删除项。删除的项会从源代码管理

中的父目录中被删除。

要求：

- 项必须受源代码管理。

从 stdin 读取输入：

'^remove' 命令可从 stdin 读取路径。为此，请传递一个

破折号 "-"。

示例：cm ^remove -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要删除的文件。

示例：

dir /S /B \*.c | cm ^remove -

（在 Windows 中，删除工作区中的所有 .c 文件。）

示例：

cm ^remove src

（删除 'src'。如果 src 是目录，则等同于

'cm ^remove -^R src'。）

cm ^remove c:\workspace\file.txt --^format="{0} - 已删除" \

--^errorformat="{0} - 删除时出错"

（从版本控制中和磁盘上删除 'file.txt'，

如果操作成功，则写入 "c:\workspace\file.txt - ^REMOVED"，

否则写入 "c:\workspace\file.txt - ^ERROR ^REMOVING"。）

cm ^remove c:\workspace\file.txt --^nodisk

（从版本控制中删除 'file.txt'，但将其保留在磁盘上。）

== CMD\_DESCRIPTION\_RM\_PRIVATE ==

删除一个私有文件或目录。

警告：使用该命令删除的文件将被永久擦除而无法

恢复。建议使用 '--^dry-run' 选项来检查

哪些文件将受到该命令的影响。

== CMD\_USAGE\_RM\_PRIVATE ==

用法：

cm ^remove | ^rm ^private <路径>[ ...][-^R | -^r | --^recursive] [--^ignored]

[--^verbose] [--^dry-run]

路径 要删除的文件或目录的路径。使用双

引号 (" ") 指定包含空格的路径。使用

空格对各个路径进行分隔。

选项：

--^r 以递归方式从受控目录内删除私有

文件。

--^ignored 还删除已忽略和掩蔽的文件和目录。

--^verbose 打印所有受影响的路径。

--^dry-run 运行此命令而不在磁盘上进行任何更改。

== CMD\_HELP\_RM\_PRIVATE ==

备注：

如果路径是私有文件或目录，则会从磁盘上将其删除。

如果路径是受控文件，该命令会失败。

如果路径是受控目录，则除非指定 '-^r' 选项

，否则该命令将失败，在这种情况下，它将删除指定目录下的

所有私有文件和目录。

示例：

cm ^remove ^private private\_directory

（删除 'private\_directory'。）

cm ^remove ^private c:\workspace\controlled\_directory

（失败，因为 'controlled\_directory' 并非私有目录。）

cm ^remove ^private -^r c:\workspace\controlled\_directory

（删除 'controlled\_directory' 下的所有私有文件和目录。）

cm ^rm ^private --^dry-run --^verbose c:\workspace\controlled\_directory -^r

（显示 'controlled\_directory' 下的私有文件被删除所影响的

所有路径，实际上不删除任何内容。）

cm ^rm ^private --^verbose c:\workspace\controlled\_directory -^r

（显示 'controlled\_directory' 下的私有文件被删除所影响的

所有路径，并执行删除。）

== CMD\_DESCRIPTION\_TRIGGER\_DELETE ==

删除触发器。

== CMD\_USAGE\_TRIGGER\_DELETE ==

用法：

cm ^trigger | ^tr ^delete | ^rm <子类型-类型> <位置编号>

[--^server=<存储库服务器规格>]

子类型-类型 触发器执行和触发器操作。

键入 'cm ^showtriggertypes' 可查看触发器类型

列表。

位置编号 创建触发器时分配给触发器的位置。

选项：

--^server 删除指定服务器上的触发器。

如果未指定服务器，则在客户端上配置的

服务器上执行命令。

== CMD\_HELP\_TRIGGER\_DELETE ==

示例：

cm ^trigger ^delete ^after-setselector 4

cm ^tr ^rm ^after-setselector 4

== CMD\_DESCRIPTION\_ATTRIBUTE\_SET ==

在给定对象上设置属性。

== CMD\_USAGE\_ATTRIBUTE\_SET ==

用法：

cm ^attribute | ^att ^set <属性规格> <对象规格> <属性值>

属性规格 属性的规格。（使用 'cm ^help ^objectspec' 可

进一步了解属性规格。）

对象规格 要设置相应属性的对象的规格。

可以在以下对象上设置属性：分支、变更集、

搁置集、标签、项和修订。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

属性值 要为对象设置的属性值。

== CMD\_HELP\_ATTRIBUTE\_SET ==

备注：

可以在一个对象上设置属性来保存此对象的

其他信息。

可以在以下对象上设置属性：分支、变更集、

搁置集、标签、项和修订。

示例：

cm ^attribute ^set ^att:status ^br:/main/SCM105 未完成

（将属性 '状态' 设置为分支 'SCM105'，值为 '未完成'。）

cm ^att ^set ^att:integrated@reptest@server2:8084 ^lb:LB008@reptest@server2:8084 是

（在存储库 'reptest' 中将属性 '集成' 设置为标签 'LB008'，

值为 '是'。）

== CMD\_DESCRIPTION\_SETOWNER ==

设置对象的所有者。

== CMD\_USAGE\_SETOWNER ==

用法：

cm ^setowner | ^sto --^user=<用户名> | --^group=<组> <对象规格>

--^user 用户名。对象的新所有者。

--^group 组名称。对象的新所有者。

对象规格 要设置新所有者的对象的规格。

可以在以下对象上设置所有者：

存储库服务器、存储库、分支、变更集、

标签、项、修订和属性。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

== CMD\_HELP\_SETOWNER ==

备注：

对象的所有者可以是用户或组。

可以在以下对象上设置所有者：存储库服务器、

存储库、分支、变更集、标签、项、修订和属性。

示例：

cm ^setowner --^user=john ^repserver:localhost:8084

（将 'john' 设置为存储库服务器所有者。）

cm ^sto --^group=development ^rep:mainRep@PlasticServer:8084

（将 'development' 组设置为 'mainRep' 存储库的所有者。）

== CMD\_DESCRIPTION\_SETSELECTOR ==

将选择器设置为工作区。

== CMD\_USAGE\_SETSELECTOR ==

已弃用此命令。为了向后兼容，此命令仍然存在，

但是在 Plastic SCM 4.0 中，已在很大程度上弃用选择器。仍然存在的选择器

是为了指定工作分支或变更集，但不再支持

用于筛选路径的旧规则。

用法：

cm ^setselector | ^sts [--^file=<选择器文件>] [--^ignorechanges]

[--^forcedetailedprogress] [<工作区路径> | <工作区规格>]

选项：

--^file 要从中加载选择器的文件。

--^ignorechanges 忽略在更新工作区时

检测到待定更改的情况下显示

的待定更改警告消息。

--^forcedetailedprogress 即使在标准输出被重定向的情况下也要强制显示

详细进度。

工作区路径 用于设置选择器的工作区的路径。

工作区规格 工作区的规格。（使用 'cm ^help ^objectspec'

可进一步了解工作区规格。）

== CMD\_HELP\_SETSELECTOR ==

备注：

此命令将设置工作区的选择器。

工作区需要某些信息才能从存储库加载修订。

为了获得这些信息，Plastic SCM 需要使用选择器。

通过使用选择器，可以从给定的分支、标签或

变更集加载修订。

如果未指定用于加载选择器的文件，则将执行默认的

操作系统编辑器。

示例选择器：

^repository "^default" // 工作存储库

^path "/" // 规则将应用于根目录

^branch "/^main" // 从 ^br:/^main 获取最新修订

^checkout "/^main" // 将签出内容放在 ^br:/^main 分支中

示例：

cm ^sts

（打开要应用的当前选择器文件。）

cm ^sts ^wk:workspace\_projA@reptest

（打开要应用的指定选择器文件。）

cm ^setselector --^file=c:\selectors\sel.xml

（在当前工作区中设置指定的选择器文件。）

cm ^setselector --^file=c:\selectors\sel.xml ^wk:MyWorkspace

（在选定工作区中设置指定的选择器文件。）

== CMD\_DESCRIPTION\_SHELVE ==

搁置已签出项的内容。

== CMD\_USAGE\_SHELVE ==

已弃用此命令。请改用 'cm ^shelveset'。

用法：

cm ^shelve [<项路径>+] [--^all] [--^dependencies]

[-^c=注释字符串 | -^commentsfile=<注释文件>]

[--^encoding=名称] [--^comparisonmethod=比较方法]

（搁置内容。）

cm ^shelve --^apply=<搁置规格> [--^mount]

（应用存储的搁置集。）

--^apply 还原指定搁置集的已搁置内容。

搁置规格：查看 'cm ^help ^objectspec'。

cm ^shelve --^delete=<搁置规格>

（删除存储的搁置集。）

--^delete 删除指定的搁置集。

搁置集规格：查看 'cm ^help ^objectspec'。

选项：

项路径 要搁置的项（以空格分隔）。可以使用引号 (")

来指定包含空格的路径。

--^all 还包括在给定路径上进行了本地更改、

移动和删除的项。

--^dependencies 在要搁置的项中包含本地更改

依赖项。

-^c 将指定的注释应用于创建的搁置集。

-^commentsfile 将指定文件中的注释应用于创建的

搁置集。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^comparisonmethod 设置比较方法。请参阅“备注”以了解更多信息。

--^mount 给定存储库的装入点。

== CMD\_HELP\_SHELVE ==

备注：

如果既没有指定 <项路径>，也没有指定任何其他选项，则搁置将涉及

工作区中的所有待定更改。

搁置操作始终以递归方式从给定路径应用。

搁置项的要求：

- 项必须受源代码管理。

- 必须签出或更改项（必须使用 --^all 选项）。

比较方法：

^ignoreeol 忽略行尾差异。

^ignorewhitespaces 忽略空格差异。

^ignoreeolwhitespaces 忽略行尾和空格差异。

^notignore 检测行尾和空格差异。

设置 PLASTICEDITOR 环境变量可指定用于键入注释的

编辑器。

示例：

cm ^shelve -^c="我的注释"

（搁置当前工作区中的所有待定更改，包括

注释。）

cm ^shelve file1.txt "file 2.txt" -^commentsfile=commentshelve.txt

（搁置选定的待定更改并应用 commentshelve.txt

文件中的注释。）

cm ^shelve --^apply=^sh:3

（应用存储的搁置集。）

cm ^shelve --^delete=^sh:3

（删除存储的搁置集。）

cm ^status --^short --^changelist=pending\_to\_review | cm ^shelve -

（搁置客户端更改列表。

以上命令将列出名为 'pending\_to\_review'

的更改列表中的路径，并且路径列表将重定向到

shelve 命令的输入。）

== CMD\_DESCRIPTION\_SHELVESET ==

允许用户管理搁置集。

== CMD\_USAGE\_SHELVESET ==

用法：

cm ^shelveset <命令> [选项]

命令：

^create | ^mk

^delete | ^rm

^apply

要获取有关每条命令的更多信息，请运行：

cm ^shelveset <命令> --^usage

cm ^shelveset <命令> --^help

== CMD\_HELP\_SHELVESET ==

示例：

cm ^shelveset ^create -^c="我的注释"

cm ^shelveset ^delete ^sh:3

cm ^shelve ^apply ^sh:3

== CMD\_DESCRIPTION\_SHELVESET\_CREATE ==

搁置待定更改。

== CMD\_USAGE\_SHELVESET\_CREATE ==

用法：

cm ^shelveset ^create | ^mk [<项路径>[ ...]] [--^all] [--^dependencies]

[-^c=<注释字符串> | -^commentsfile=<注释文件>]

选项：

项路径 要搁置的项。使用空格对各个用户名进行分隔。

使用双引号 (" ") 指定包含空格的

路径。

--^all 还包括在给定路径上进行了本地更改、

移动和删除的项。

--^dependencies 在要搁置的项中包含本地更改

依赖项。

-^c 将指定的注释应用于创建的搁置。

-^commentsfile 将指定文件中的注释应用于创建的

搁置。

== CMD\_HELP\_SHELVESET\_CREATE ==

'^shelveset ^create' 命令可将已签出项的内容存储在

存储库中。这样，无需签入文件便可以

保护内容。

备注：

如果既没有指定 <项路径>，也没有指定任何其他选项，则搁置集将

包含工作区中的所有待定更改。

'^shelveset ^create' 操作始终从给定路径以递归方式

应用。

搁置项的要求：

- 项必须受源代码管理。

- 必须签出或更改项（必须使用 '--^all' 选项）。

设置 PLASTICEDITOR 环境变量可指定用于键入注释的

编辑器。

示例：

cm ^shelveset ^create -^c="我的注释"

（搁置当前工作区中的所有待定更改，包括

注释。）

cm ^shelveset file1.txt "file 2.txt" -^commentsfile=commentshelve.txt

（搁置选定的待定更改并应用 'commentshelve.txt'

文件中的注释。注意，'^create' 是默认的子命令。）

cm ^status --^short --^changelist=pending\_to\_review | cm ^shelveset -

（搁置客户端更改列表。

以上命令将列出名为 'pending\_to\_review'

的更改列表中的路径，并且路径列表将重定向到

'^shelveset' 命令的输入。）

== CMD\_DESCRIPTION\_SHELVESET\_DELETE ==

删除搁置集。

== CMD\_USAGE\_SHELVESET\_DELETE ==

用法：

cm ^shelveset ^delete | ^rm <搁置规格>

搁置规格 搁置集规格。（使用 'cm ^help ^objectspec' 可

进一步了解搁置集规格。）

== CMD\_HELP\_SHELVESET\_DELETE ==

'^shelveset ^delete' 命令可删除搁置集。

示例：

cm ^shelveset ^delete ^sh:3

（删除存储的搁置集。）

== CMD\_DESCRIPTION\_SHELVESET\_APPLY ==

应用存储的搁置集。

== CMD\_USAGE\_SHELVESET\_APPLY ==

用法：

cm ^shelveset ^apply <搁置规格> [--^mount] [--^encoding=<名称>]

[--^comparisonmethod=(^ignoreeol | ^ignorewhitespaces| \

^ignoreeolwhitespaces | ^notignore)]

搁置规格 搁置集规格。（使用 'cm ^help ^objectspec' 可

进一步了解搁置集规格。）

选项：

--^mount 给定存储库的装入点。

--^encoding 指定输出编码，如：utf-8。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^comparisonmethod 设置比较方法。请参阅“备注”以了解更多信息。

== CMD\_HELP\_SHELVESET\_APPLY ==

'^shelveset ^apply' 命令可还原存储的搁置集的内容。

备注：

比较方法：

^ignoreeol 忽略行尾差异。

^ignorewhitespaces 忽略空格差异。

^ignoreeolwhitespaces 忽略行尾和空格差异。

^notignore 检测行尾和空格差异。

示例：

cm ^shelveset ^apply ^sh:3

（应用存储的搁置。）

== CMD\_DESCRIPTION\_SHOW\_FIND\_OBJECTS ==

列出对象和属性。

== CMD\_USAGE\_SHOW\_FIND\_OBJECTS ==

用法：

cm ^showfindobjects

== CMD\_HELP\_SHOW\_FIND\_OBJECTS ==

可用对象和属性：

^attribute:

可以通过使用以下字段进行筛选来查找属性：

^type ：字符串。

示例：

cm ^find ^attribute "^where ^type = '状态'"

（查找 '状态' 类型的所有属性。）

^value ：字符串。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

示例：

cm ^find ^attribute "^where ^date > '^this ^week'"

（查找本周内应用的所有属性。）

^owner ：用户。

容许特殊用户 '^me'。

示例：

cm ^find ^attribute "^where ^value = '已解决' ^and ^owner = '^me'"

（查找具有 '已解决' 值并由我应用的所有属性。）

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^comment ：字符串。

^srcobj ：对象规格：项路径、分支、变更集、修订或标签。

使用 'cm ^help ^objectspec' 可了解如何指定这些对象。

示例：

cm ^find ^attribute "^where ^srcobj = '^item:readme.txt'"

（查找应用于项 'readme.txt' 的属性。）

cm ^find ^attribute "^where ^srcobj = '^br:/main/scm23343'"

（查找应用于分支 scm23343 的属性。）

cm ^find ^attribute "^where ^srcobj = '^rev:readme.txt#^br:/main/task002'"

（查找应用于指定修订的属性。）

cm ^find ^attribute "^where ^srcobj = '^rev:^revid:1126'"

（查找应用于指定修订 ID 的属性。）

^ID ：整数。

^attributetype:

可以通过使用以下字段进行筛选来查找属性类型：

^name ：字符串。

示例：

cm ^find ^attributetype "^where ^name ^like 'st%'"

（查找名称以 'st' 开头的所有属性。）

^value ：字符串。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

示例：

cm ^find ^attribute "^where ^date > '^today'"

（查找今天应用的所有属性。）

^owner ：用户。

容许特殊用户 '^me'。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^comment ：字符串。

示例：

cm ^find ^attributetype "^where ^comment != ''" --^xml

（查找具有某个注释的所有属性类型，并且以 XML 格式

将输出打印到标准输出。）

^source ：对象规格：项路径、分支、变更集或标签。

使用 'cm ^help ^objectspec' 可了解如何指定这些对象。

示例：

cm ^find ^attributetype "^where ^source = '^item:readme.txt'"

（查找项 'readme.txt' 中的所有属性类型。）

cm ^find ^attributetype "^where ^source = '^cs:30'"

（查找变更集 '30' 中的所有属性类型。）

cm ^find ^attributetype "^where ^source = '^lb:v0.14.1'"

（查找标签 'v0.14.1' 中的所有属性类型。）

^ID ：整数。

复制字段。查看下文的“与复制相关的字段”。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^branch:

可以通过使用以下字段进行筛选来查找分支：

^name ：字符串。

示例：

cm ^find ^branch "^where ^name ^like 'scm23%'"

（查找名称以 'scm23' 开头的分支。）

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

示例：

cm ^find ^branch "^where ^date > '^one ^week ^ago'"

（查找上周创建的分支。）

^changesets ：日期（分支中变更集的日期）。

请在本指南中查看“日期常量”以了解更多信息。

示例：

cm ^find ^branch "^where ^changesets >= '^today'"

（查找包含今天创建的变更集的分支。）

^attribute ：字符串。

^attrvalue ：字符串。

示例：

cm ^find ^branch "^where ^attribute = '状态' ^and ^attrvalue = '已失败'"

（查找具有属性 '状态' 且属性值为 '已失败'

的分支。）

^owner ：用户。

容许特殊用户 '^me'。

^parent ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

示例：

cm ^find ^branch "^where ^owner != '^me' ^and ^parent != '^br:/main'"

（查找由我以外的人创建且父分支

不是 '/main' 的分支。）

^comment ：字符串。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^ID ：整数。

示例：

cm ^find ^branch "^where ^id = 2029607"

（查找 ID 为 2029607 的分支。）

复制字段。查看下文的“与复制相关的字段”。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^changeset:

可以通过使用以下字段进行筛选来查找变更集：

^branch ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此

对象。

示例：

cm ^find ^changeset "^where ^branch = '/main/scm23119'"

（查找分支 'scm23119' 中的所有变更集。）

^changesetid ：整数。

^attribute ：字符串。

示例：

cm ^find ^changeset "^where ^attribute = '状态'"

（查找包含属性 '状态' 的变更集。）

^attrvalue ：字符串。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

^owner ：用户。

容许特殊用户 '^me'。

示例：

cm ^find ^changeset "^where ^date >= '2018/8/6' ^and ^owner != '^me'"

（查找创建日期等于或晚于 2018/8/6

并且由我以外的人创建的所有变更集。）

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

示例：

cm ^find ^changeset "^where ^guid = '1b30674f-14cc-4fd7-962b-676c8a6f5cb6'"

（查找具有指定 GUID 的变更集。）

^comment ：字符串。

示例：

cm ^find ^changeset "^where ^comment = ''"

（查找不含注释的变更集。）

^onlywithrevisions ：布尔值。

筛选变更集是否具有修订。

示例：

cm ^find ^changeset "^where ^onlywithrevisions = 'false'"

（查找不含修订的变更集。）

^returnparent ：布尔值。

一种返回变更集父级的方法。非常适合用于编写脚本。

示例：

cm ^find ^changeset "^where ^changesetid = 29 ^and ^returnparent = 'true'"

（查找变更集 29 的父级。）

^parent ：变更集 ID（整数）。

示例：

cm ^find ^changeset "^where ^parent = 548"

（查找父级为变更集 548 的所有变更集。）

^ID ：整数。

复制字段。查看下文的“与复制相关的字段”。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^label:

可以通过使用以下字段进行筛选来查找标签：

^name ：字符串。

示例：

cm ^find ^label "^where ^name ^like '7.0.16.%'"

（查找名称以 '7.0.16.' 开头的标签。）

^attribute ：字符串。

^attrvalue ：字符串。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

示例：

cm ^find ^label "^where ^date >= '^this ^month' ^and \

^attribute = '发布状态' ^and ^attrvalue != '已发布'"

（查找本月创建并且属性 '发布状态' 设置为

'已发布' 以外的值的标签。）

^owner ：用户。

容许特殊用户 '^me'。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^branch ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

示例：

cm ^find ^label "^where ^branch = '/main'"

（查找应用于主分支的所有标签。）

^branchid ：整数。

^changeset ：变更集 ID（整数）。

示例：

cm ^find ^label "^where ^changeset = 111733"

（查找应用于变更集 111733 的标签。）

^comment ：字符串。

^ID ：整数。

复制字段。查看下文的“与复制相关的字段”。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^merge:

可以通过使用以下字段进行筛选来查找合并：

^srcbranch ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

示例：

cm ^find ^merge "^where ^srcbranch = '^br:/main'"

（从主分支中查找合并。）

^srcchangeset ：变更集 ID（整数）。

^dstbranch ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

^dstchangeset ：变更集 ID（整数）。

示例：

cm ^find ^merge "^where ^dstchangeset = 108261" \

--^format="{^srcbranch} {^srcchangeset} {^dstbranch} {^dstchangeset} {^owner}"

（查找与变更集 108261 的合并，并打印

格式化的输出，从而显示源（分支和变更集 ID）、

目标（分支和变更集 ID）和合并所有者。）

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

^owner ：用户。

容许特殊用户 '^me'。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^type ：字符串。

可能的值有 '^merge'、'^cherrypick'、

'^cherrypicksubstractive'、'^interval'、'^intervalcherrypick'

和 '^intervalcherrypicksubstractive'

示例：

cm ^find ^merge "^where ^type = '^cherrypick' ^and ^owner = '^me'"

（查找我所有的挑拣。）

^ID ：整数。

^replicationlog:

可以通过使用以下字段进行筛选来查找复制日志：

^branch ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

示例：

cm ^find ^replicationlog "^where ^branch = '/main/gm22358'"

（查找分支 'gm22358' 的复制日志。）

^repositoryname ：字符串。

^owner ：用户。

容许特殊用户 '^me'。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

^server ：字符串。

^package ：布尔值。

示例：

cm ^find ^replicationlog "^where ^package = 'T' ^and ^server ^like '%cloud%'"

（查找从服务器名称中包含 'cloud' 的包

创建的复制日志。）

^ID ：整数。

^review:

可以通过使用以下字段进行筛选来查找代码审查：

^status ：字符串。

^assignee ：字符串。

示例：

cm ^find ^review "^where ^status = '待定' ^and ^assignee = '^me'"

（查找我所有的待定审查。）

^title ：字符串。

^target ：对象规格：分支或变更集。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

示例：

cm ^find ^review "^where ^target = '^br:/main/scm17932'"

（查找与分支 'scm17932' 相关的审查。）

^targetid ：整数。

^targettype ：字符串。

可能的值有 '^branch' 和 '^changeset'。

示例：

cm ^find ^review "^where ^targettype = '^changeset'"

（查找目标类型为变更集的审查。）

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

^owner ：用户。

容许特殊用户 '^me'。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^ID ：整数。

^revision:

可以通过使用以下字段进行筛选来查找修订：

^branch ：分支规格。

使用 'cm ^help ^objectspec' 可了解如何指定此对象。

^changeset ：变更集 ID（整数）。

示例：

cm ^find ^revision "^where ^changeset >= 111756"

（查找在变更集 111756 以及更高版本中

创建的修订。）

^item ：字符串或整数。

^itemid ：整数。

示例：

cm ^find ^revision "^where ^item = 'readme.txt' ^or ^itemid = 2250"

（查找项 'readme.txt' 和项 ID 2250

的修订。）

cm ^find ^revision "^where ^item = 'readme.txt' ^or ^item = 2250"

（获取与先前示例相同的修订。）

^attribute ：字符串。

^attrvalue ：字符串。

示例：

cm ^find ^revision "^where ^attribute = '状态' ^and ^attrvalue != '未完成'"

（查找包含属性 '状态' 且属性值

不是 '未完成' 的修订。）

^archived ：布尔值。

示例：

cm ^find ^revision "^where ^archived = 'true'"

（查找在外部存储中存档

的修订。）

^comment ：字符串。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^owner ：用户。

容许特殊用户 '^me'。

^parent ：修订 ID（整数）。

^returnparent ：布尔值。

^shelve ：搁置 ID（整数）。

^size ：整数（以字节为单位）。

^type ：字符串。

可能的值有 '^dir'、'^bin' 和 '^txt'。

示例：

cm ^find ^revision "^where ^type = '^txt' and \

^size > 300000 ^and ^owner = '^me' and ^date >= '2 ^months ^ago'"

（查找我两个月前创建且大小

大于 3MB 的文本修订。）

^workspacecheckoutid ：整数。

^ID ：整数。

复制字段。查看下文的“与复制相关的字段”。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^shelve:

可以通过使用以下字段进行筛选来查找搁置：

^owner ：用户。

容许特殊用户 '^me'。

^date ：日期。

请在本指南中查看“日期常量”以了解更多信息。

示例：

cm ^find ^shelve "^where ^owner != '^me' ^and ^date >= '^1 ^years ^ago'"

（查找去年我以外的人创建的

搁置。）

^attribute ：字符串。

^attrvalue ：字符串。

^comment ：字符串。

^GUID ：全局唯一标识符。

十六进制 ID，格式为 xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

^parent ：整数。

^shelveid ：整数。

示例：

cm ^find ^shelve "^where ^shelveid = 2"

（查找名称为 2 的搁置。）

^ID ：整数。

示例：

cm ^find ^shelve "^where ^id >= 3848"

（查找对象 ID 大于 3848 的搁置。）

复制字段。查看下文的“与复制相关的字段”。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

与复制相关的字段：

许多对象会跟踪复制数据，这意味着 Plastic 会跟踪最初创建复制

的位置。

可以使用的字段包括：

^ReplSrcServer ：存储库规格。表示“复制源服务器”。

从中复制对象的服务器。

示例：

cm ^find ^branch "^where ^replsrcserver='skull.codicefactory.com:9095'"

（查找从服务器 'skull' 复制的分支。）

^ReplSrcRepository ：字符串。表示“复制源存储库”。这是

从中复制对象的存储库。

示例：

cm ^find ^branch "^where ^replsrcserver = 'skull.codicefactory.com:9095' \

^and ^replsrcrepository = 'codice'"

（查找从服务器 'skull' 和存储库 'codice'

复制的分支。）

^ReplLogId ：整数。复制操作的 ID。在 Plastic 中，

每次从复本创建新对象时，

都会创建一个新的 'replicationlog'。

示例：

cm ^find ^revision "^where ^repllogid = 2019974"

（查找从复本 2019974 复制

的修订。）

^ReplSrcDate ：日期。这是实际发生复制的

日期。

复制的对象将保留其原始创建

日期，如果要查找在特定时间

范围内复制的对象，则此字段

很有用。

示例：

cm ^find ^label "^where ^replsrcdate >= '^one ^month ^ago' \

^and ^date >= '15 ^days ^ago'"

（查找 15 天前创建并在一个月前

复制的标签。）

cm ^find ^replicationlog "^where ^date > '^one ^week ^ago'"

8780433 27/09/2018 8:49:38 codice@BACKYARD:8087 F mbarriosc

（查找一周前创建的复制日志。）

现在可以检查复制的分支是否是

在复制之前创建的：

cm ^find ^branch "^where ^repllogid = 8780433"

8780443 26/09/2018 12:20:55 /main/scm23078 maria codice T

^ReplSrcId ：整数。这是复制源服务器的 ID。

可以使用 'cm ^find' 命令搜索

'^replicationsource' 对象来发现此 ID。

示例：

cm ^find ^replicationsource

7860739 codice@AFRODITA:8087 d9c4372a-dc55-4fdc-ad3d-baeb2e975f27

8175854 codice@BACKYARD:8087 66700d3a-036b-4b9a-a26f-adfc336b14f9

现在可以查找从 codice@AFRODITA:8087

复制的变更集：

cm ^find ^changesets "^where ^replsrcid = 7860739"

日期常量：

可以使用遵循计算机本地化设置的日期格式。

例如，如果计算机以 'MM-dd-yyyy' 格式显示日期，

则可以在查询中使用 '12-31-2019' 之类的日期。

还可以使用以下常量来简化查询：

'^today' ：今天的日期。

'^yesterday' ：昨天的日期。

'^this ^week' ：本周星期一的日期。

'^this ^month' ：本月第一天的日期。

'^this ^year' ：本年 1 月 1 日。

'^one ^day ^ago' ：当前日期前一天。

'^one ^week ^ago' ：当前日期前七天。

'^one ^month ^ago' ：当前日期前一个月。

'n ^days ^ago' ：当前日期前 'n' 天。

'n ^months ^ago' ：当前日期前 'n' 个月。

'n ^years ^ago' ：当前日期前 'n' 年。

以下 '^where' 子句对于类型为 '^date' 的字段有效：

'(...)^where ^date > '^today' (...)'

'(...)^where ^date < '^yesterday' (...)'

'(...)^where ^date > '^this ^week' (...)'

'(...)^where ^date > '^this ^month' (...)'

'(...)^where ^date < '^one ^day ^ago' ^and ^date > '3 ^days ^ago' (...)'

'(...)^where ^date < '^one ^week ^ago' ^and ^date > '3 ^weeks ^ago' (...)'

'(...)^where ^date < '^one ^month ^ago' ^and ^date > '3 ^months ^ago' (...)'

'(...)^where ^date > '1 ^year ^ago' (...)'

还可以在 'cm ^find' 命令中强制使用特定日期格式，方法是使用

--^dateformat 标志。请查看 'cm ^find --^help' 以了解更多详细信息。

== CMD\_DESCRIPTION\_TRIGGER\_SHOWTYPES ==

显示可用的触发器类型。

== CMD\_USAGE\_TRIGGER\_SHOWTYPES ==

用法：

cm ^trigger ^showtypes

== CMD\_DESCRIPTION\_SHOWACL ==

显示对象的 ACL。

== CMD\_USAGE\_SHOWACL ==

用法：

cm ^showacl | ^sa <对象规格> [--^extended] [--^xml[=<输出文件>]]

[--^encoding=<名称>]

对象规格 要显示相应 ACL 的对象的规格。

此命令的有效对象为：

存储库服务器、存储库、分支、变更集、标签、项

和属性。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

选项：

--^extended 显示 ACL 层次结构树。

--^xml 以 XML 格式将输出打印到标准输出。

可以指定输出文件。

--^encoding 与 '--^xml' 选项结合使用，指定要在 XML 输出

中使用的编码（例如：utf-8）。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

== CMD\_HELP\_SHOWACL ==

示例：

cm ^showacl ^repserver:PlasticServer:8084

（显示所选服务器的 ACL。）

cm ^sa ^br:/main --^extended

（显示所选分支规格的 ACL 层次结构树。）

== CMD\_DESCRIPTION\_SHOWCOMMANDS ==

显示所有可用命令。

== CMD\_USAGE\_SHOWCOMMANDS ==

用法：

cm ^showcommands

== CMD\_HELP\_SHOWCOMMANDS ==

== CMD\_DESCRIPTION\_SHOWOWNER ==

显示对象的所有者。

== CMD\_USAGE\_SHOWOWNER ==

用法：

cm ^showowner | ^so <对象规格>

对象规格 要显示相应所有者的对象的规格。

对象必须为下列其中之一：

存储库服务器、存储库、分支、变更集、

标签、属性、修订和项。

（使用 'cm ^help ^objectspec' 可进一步了解规格。）

== CMD\_HELP\_SHOWOWNER ==

备注：

此命令将显示对象的所有者。所有者可以是用户或

组。可以使用 'cm ^setowner' 命令修改所有者。

示例：

cm ^showowner ^repserver:PlasticServer:8084

（显示所选服务器的所有者。）

cm ^so ^item:samples\

（显示所选项规格的所有者。）

== CMD\_DESCRIPTION\_SHOWPERMISSIONS ==

列出可用的权限。

== CMD\_USAGE\_SHOWPERMISSIONS ==

用法：

cm ^showpermissions | ^sp

== CMD\_HELP\_SHOWPERMISSIONS ==

示例：

cm ^showpermissions

== CMD\_DESCRIPTION\_SHOWSELECTOR ==

显示工作区选择器。

== CMD\_USAGE\_SHOWSELECTOR ==

已弃用此命令。为了向后兼容，此命令仍然存在，

但是在 Plastic SCM 4.0 中，已在很大程度上弃用选择器。仍然存在的选择器

是为了指定工作分支或变更集，但不再支持

用于筛选路径的旧规则。

用法：

cm ^showselector | ^ss [<工作区路径> | <工作区规格>]

工作区路径 用于显示选择器的工作区的路径。

工作区规格 工作区的规格。（使用 'cm ^help ^objectspec' 可

可进一步了解工作区规格。）

== CMD\_HELP\_SHOWSELECTOR ==

备注：

如果路径和工作区规格均未指定，则该命令将采用

当前目录作为工作区路径。

示例：

cm ^showselector c:\workspace

（显示所选工作区路径的选择器。）

cm ^ss

（显示当前工作区的选择器。）

cm ^showselector > mySelector.txt

（将当前工作区的选择器写入文件。）

cm ^showselector ^wk:mywk@reptest

（显示存储库 'reptest' 中的工作区 'mywk' 的选择器。）

== CMD\_DESCRIPTION\_SUPPORT ==

允许用户执行与支持相关的操作。

== CMD\_USAGE\_SUPPORT ==

用法：

cm ^support <命令> [选项]

命令：

^bundle

要获取有关每条命令的更多信息，请运行：

cm ^support <命令> --^usage

cm ^support <命令> --^help

== CMD\_HELP\_SUPPORT ==

示例：

cm ^support

cm ^support ^bundle

cm ^support ^bundle c:\outputfile.zip

== CMD\_DESCRIPTION\_SUPPORT\_BUNDLE ==

创建带有相关日志的“支持捆绑包”包。

可以在请求帮助、询问其他信息或者提交 bug 时

附加该文件。

== CMD\_USAGE\_SUPPORT\_BUNDLE ==

用法：

cm ^support ^bundle [<输出文件>]

选项：

输出文件 在指定的位置创建“支持捆绑包”

包。

== CMD\_HELP\_SUPPORT\_BUNDLE ==

备注：

此命令允许用户创建一个“支持捆绑包”包，然后便可在请求帮助、

询问其他信息或提交 bug 时附加这个包。

用户可以选择指定输出文件的位置；否则，

输出文件将写入到临时目录。

示例：

cm ^support ^bundle

（在临时目录中创建“支持捆绑包”。）

cm ^support ^bundle c:\outputfile.zip

（在指定位置创建“支持捆绑包”。）

== CMD\_DESCRIPTION\_SWITCH ==

将工作区切换到分支、变更集、标签或搁置集。

== CMD\_USAGE\_SWITCH ==

用法：

cm ^switch (<分支规格> | <变更集规格> | <标签规格> | <搁置规格>)

[--^workspace=<路径>] [--^repository=<名称>]

[--^forcedetailedprogress]

（使用 'cm ^help ^objectspec' 可进一步了解分支、变更集、标签、

和搁置集规格。）

选项：

--^workspace 工作区所在的路径。

--^repository 切换到指定的存储库。

--^forcedetailedprogress 即使在标准输出被重定向的情况下也要强制

显示详细进度。

== CMD\_HELP\_SWITCH ==

备注：

此命令允许用户将工作区树更新为指定对象

（分支、标签、搁置集或变更集）的内容。

示例：

cm ^switch ^br:/main

cm ^switch ^lb:Rel1.1

cm ^switch ^br:/main/scm002 --^repository=rep2

cm ^switch ^cs:4375

cm ^switch ^sh:2

== CMD\_DESCRIPTION\_SWITCH\_TO\_BRANCH ==

将分支设置为工作分支。

== CMD\_USAGE\_SWITCH\_TO\_BRANCH ==

已弃用此命令。请改用 cm switch。

用法：

cm ^switchtobranch [选项] [分支规格]

分支规格：分支的规格。

选项：

--^label=名称 | --^changeset=数字：从指定标签或变更集

加载修订。如果未提供任何分支规格，则需要这些选项

之一。

--^changeset=变更集：切换到指定的变更集。

--^repository=存储库：切换到指定的存储库。

--^workspace | -wk=路径：工作区所在的路径。

== CMD\_HELP\_SWITCH\_TO\_BRANCH ==

备注：

此命令允许用户在分支中工作。

如果未指定分支规格，则必须指定标签或变更集。

如果未指定存储库，则分支将设置为当前存储库。

示例：

cm ^switchtobranch ^br:/main

cm ^switchtobranch ^br:/main/task001

cm ^switchtobranch --^label=BL050

（只读配置。该命令将加载已标记的变更集

的内容。）

== CMD\_DESCRIPTION\_SYNC ==

与 Git 同步。

== CMD\_USAGE\_SYNC ==

用法：

cm ^synchronize | ^sync <存储库规格> ^git [<url> [--^user=<用户名> --^pwd=<密码>]]

[(--^txtsimilaritypercent | --^binsimilaritypercent | \

--^dirsimilaritypercent)=<值>]

[--^author] [--^skipgitlfs]

存储库规格 存储库的规格。（使用 'cm ^help ^objectspec' 可

进一步了解存储库规格。）

git （默认值）。

选项：

url 远程存储库 URL（http(s):// 或 git:// 或

SSH URL）。

--^user 指定的 URL 的用户名。

--^pwd 指定的 URL 的密码。

--^txtsimilaritypercent \

--^binsimilaritypercent \

--^dirsimilaritypercent

要检测已移动的项，与 Plastic SCM GUI 的

方法相同。

--^author 使用 Git 作者提供的名称和时间戳值。

（默认为 Git 提交者）

--^skipgitlfs 忽略 .gitattributes 文件中的

Git LFS 配置。行为类似于不支持

Git LFS。

== CMD\_HELP\_SYNC ==

备注：

如果 Git 存储库需要用户名和密码，则应使用 '^url'、'--^user'、

和 '--^pwd' 选项。

如果 Git 存储库不需要用户名和密码，则应在第一次同步

操作中使用 '^url' 选项。在后续的同步操作中，'^url'

选项是可选的。

要使用 SSH 协议执行同步，必须将 'ssh' 客户端

添加到 PATH 环境变量，并正确配置为连接到

远程主机（即配置私钥/公钥）。

示例：

cm ^sync default@localhost:8087 ^git git://localhost/repository

== CMD\_DESCRIPTION\_TRIGGER ==

允许用户管理触发器。

== CMD\_USAGE\_TRIGGER ==

用法：

cm ^trigger | ^tr <命令> [选项]

命令：

^create | ^mk

^delete | ^rm

^edit

^list | ^ls

^showtypes

要获取有关每条命令的更多信息，请运行：

cm ^trigger <命令> --^usage

cm ^trigger <命令> --^help

== CMD\_HELP\_TRIGGER ==

示例：

cm ^tr ^mk ^before-mklabel new "/path/to/script" --^server=myserver:8084

cm ^tr ^edit ^before-mklabel 7 --^position=4 --^server=myserver:8084

cm ^tr ^ls ^before-mkbranch --^server=myserver:8084

cm ^tr ^rm ^after-setselector 4

cm ^tr ^showtypes

== CMD\_DESCRIPTION\_TUBE ==

运行与 Plastic Tube 有关的命令。

== CMD\_USAGE\_TUBE ==

用法：

cm ^tube ^config -^u=<用户> -^p=<密码>

（配置 Plastic SCM 以将 Plastic Tube 与指定的用户和密码

配合使用。）

cm ^tube ^create <远程用户>

（创建 Tube“远程用户 -> 我的用户”。

用户“我的用户”允许“远程用户”连接到“我的用户”服务器。

可以建立从“远程用户”到“我的用户”的连接。

只能创建其他用户到当前 Tube 用户的 Tube。）

cm ^tube ^remove <远程用户>

（删除 Tube“远程用户 -> 我的用户”。）

cm ^tube ^local

（列出在本地服务器上共享的本地存储库以及共享到

的用户。）

cm ^tube ^remote

（列出与当前 Tube 用户共享的共享远程

存储库。）

cm ^tube ^share <存储库规格>[ ...]-^u=<远程用户> -^a=(^pull | ^push | ^pull,^push)

（与远程用户共享本地存储库并设置指定的

访问模式。使用空格对各个存储库规格进行分隔。）

cm ^tube ^unshare <存储库规格>[ ...]-^u=<远程用户>

（取消与远程用户共享本地存储库。使用空格

对各个存储库规格进行分隔。）

cm ^tube ^connect

（将 Plastic SCM 服务器连接到 Plastic Tube。）

cm ^tube ^disconnect

（断开 Plastic SCM 服务器与 Plastic Tube 的连接。）

cm ^tube ^status

（显示 Plastic SCM 服务器是否已连接到 Plastic Tube。）

选项

-^u Tube 用户（plasticscm.com 用户）。

-^p 用户密码。

-^a 设置访问模式。

存储库规格 存储库的规格。（使用 'cm ^help ^objectspec' 可

进一步了解存储库规格。）

== CMD\_HELP\_TUBE ==

备注：

使用 'cm ^tube' 命令来管理 Plastic Tube。

示例：

cm ^tube ^config -^u=ruben@codicesoftware.com -^p=rubenpassword

cm ^tube ^create pablo@codicesoftware.com

（'pablo@codicesoftware.com' 可以连接到当前的 Plastic Tube 用户）

cm ^tube ^remove pablo@codicesoftware.com

cm ^tube ^local

cm ^tube ^remote

cm ^tube ^share repo@server:8087 -^u=pablo@codicesoftware.com -^a=^pull,^push

cm ^tube ^share repo@server:8087 doc@server:8087 -^u=pablo@codicesoftware.com -^a=^push

cm ^tube ^unshare repo@server:8087 -^u=pablo@codicesoftware.com

cm ^tube ^connect

cm ^tube ^disconnect

cm ^tube ^status

== CMD\_DESCRIPTION\_UNCO ==

撤销项的签出。

== CMD\_USAGE\_UNCO ==

用法：

cm ^undocheckout | ^unco <项路径>[ ...][-^a | --^all] [--^symlink] [--^silent]

[--^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

项路径 要应用操作的项。使用空格对各个路径进行

分隔。使用双引号 (" ") 指定包含空格

的路径。

使用 . 将操作应用到当前目录。

选项：

-^a | --^all 撤销指定项中的所有更改。如果

已签出项，则签出将被

还原。如果项是在本地修改的，则

修改将被还原。

--^symlink 将 undocheckout 操作应用于链接而不是

目标。

--^silent 不显示任何输出。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，

指定行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，

指定行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，

指定应如何分隔字段。

== CMD\_HELP\_UNCO ==

备注：

如果已签出某个项并且不想将这个项签入，则可以

使用此命令撤销签出。文件和文件夹都可以取消

签出。项将更新为签出之前的状态。

要求：

- 项必须受源代码管理。

- 必须签出项。

从 stdin 读取输入：

'^undocheckout' 命令可从 stdin 读取路径。为此，请传递

一个破折号 "-"。

示例：cm ^undocheckout ^checkin -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要撤销签出的文件。

示例：

dir /S /B \*.c | cm ^undocheckout --^all -

（在 Windows 中，撤销对工作区中所有 .c 文件的签出。）

示例：

cm ^undocheckout .

（撤销当前目录中的签出。）

cm ^undocheckout file1.txt file2.txt

cm unco c:\workspace\file.txt

（撤销签出所选文件。）

cm ^unco -^a file1.txt

（撤销对 'file1.txt' 的签出或本地修改）

cm ^unco 链接 --^symlink

（将 undocheckout 操作应用于链接文件而不是目标，

适用于 UNIX 环境。）

cm ^status --^short --^changelist=pending\_to\_review | cm ^undocheckout -

（撤销客户端更改列表。

以上命令将列出名为 'pending\_to\_review'

的更改列表中的路径，并且路径列表将重定向到

undocheckout 命令的输入。）

cm ^unco .--^machinereadable

（撤销当前目录中的签出，并以易于解析的

简化格式打印结果。）

cm ^unco .--^machinereadable --^startlineseparator=">" --^endlineseparator="<" \

--^fieldseparator=","

（撤销当前目录中的签出，并以易于解析的简化

格式打印结果，以指定的字符串作为行的开头和结尾以及

对各字段进行分隔。）

== CMD\_DESCRIPTION\_UNCOUNCHANGED ==

撤销未更改的已签出项。

== CMD\_USAGE\_UNCOUNCHANGED ==

用法：

cm ^uncounchanged | ^unuc <项路径>[ ...][-^R | -^r | --^recursive]

[--^symlink] [--^silent]

项路径 要应用操作的项。使用空格对各个路径进行

分隔。使用双引号 (" ") 指定包含空格

的路径。

使用 . 将操作应用到当前目录。

选项：

-^R 在指定路径中以递归方式撤销未更改的项。

--^symlink 将 uncounchanged 操作应用于链接而不是

目标。

--^silent 不显示任何输出。

== CMD\_HELP\_UNCOUNCHANGED ==

备注：

此命令是从工作区根目录以递归方式应用的。

从 stdin 读取输入：

'^uncounchanged' 命令可从 stdin 读取路径。为此，请传递

一个破折号 "-"。

示例：cm ^uncounchanged -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要撤销签出的

未更改文件。

示例：

dir /S /B \*.c | cm ^uncounchanged -

（在 Windows 中，撤销对工作区中所有未更改的 .c 文件

的签出。）

示例：

cm ^uncounchanged .-^R

（在当前目录中以递归方式撤销未更改文件的签出。）

cm ^unuc /home/myuser/mywk/project/templates -^R

（在所选目录中以递归方式撤销未更改文件的签出。）

== CMD\_DESCRIPTION\_UNDELETE ==

使用特定修订来取消删除某个项。

== CMD\_USAGE\_UNDELETE ==

用法：

cm ^undelete <修订规格> <路径>

修订规格 所含内容将加载到工作区中的

修订的规格。（使用 'cm ^help ^objectspec' 可

进一步了解修订规格。）

路径 还原路径。

== CMD\_HELP\_UNDELETE ==

备注：

要取消删除的项不应加载到工作区中。

Xlink 不支持 '^undelete' 操作。

示例：

cm ^undelete ^revid:756 C:\mywks\src\foo.c

cm ^undelete ^itemid:68#^cs:2 C:\mywks\dir\myfile.pdf

cm ^undelete ^serverpath:/src#^br:/main C:\mywks\Dir

== CMD\_DESCRIPTION\_UNDOCHANGE ==

撤销路径中的更改。

== CMD\_USAGE\_UNDOCHANGE ==

用法：

cm ^undochange | ^unc <项路径>[ ...][-^R | -^r | --^recursive]

项路径 要应用操作的项。使用空格对各个路径进行

分隔。使用双引号 (" ") 指定包含空格

的路径。

使用 . 将操作应用到当前目录。

选项：

-^R 以递归方式应用操作。

== CMD\_HELP\_UNDOCHANGE ==

备注：

如果已签出某个项或已修改但未签入这个项，并且不想

签入这个项，则可以使用此命令撤销更改。项将

更新为之前的内容。

从 stdin 读取输入：

'^undochange' 命令可从 stdin 读取路径。为此，请传递

一个破折号 "-"。

示例：cm ^undochange -

路径将一直读取到有空行输入为止。

这种情况下允许使用竖线指定要撤销更改的文件。

示例：

dir /S /B \*.c | cm ^undochange -

（在 Windows 中，撤销对工作区中所有 .c 文件的更改。）

示例：

cm ^unc .

（撤销当前目录中的文件更改。）

cm ^undochange .-^R

（以递归方式撤销当前目录中的文件更改。）

cm ^unc file1.txt "file 2.txt"

（撤销所选文件的更改。）

cm ^unc c:\workspace\file.txt

（撤销所选文件的更改。）

== CMD\_DESCRIPTION\_UNDO ==

撤销工作区中的更改。

== CMD\_USAGE\_UNDO ==

用法：

cm ^undo [<路径>[ ...]] [--^symlink] [-^r | --^recursive] [<筛选器>[ ...]]

[--^silent | --^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

路径 要应用操作的文件或目录

的路径。使用双引号 (" ") 指定包含空格的

路径。使用空格对各个路径进行分隔。

如果未指定路径，则默认情况下，

撤销操作将应用于当前目录中的所有

文件。

筛选器 将一个或多个指定筛选器应用于给定

路径。使用空格对各个筛选器进行分隔。请参阅

“筛选器”部分以了解更多信息。

选项：

--^symlink 将撤销操作应用于符号链接而不是

目标。

-^r 以递归方式执行撤销。

--^silent 不显示任何输出。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，指定

行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，指定

应如何分隔字段。

筛选器：

如果未指定标志，则默认情况下将撤销所有更改，但是

可以使用以下一个或多个标志来筛选路径。

如果文件或目录匹配一个或多个指定种类的更改，

则将撤销该文件或目录上的所有更改。

例如，如果同时指定了 '--^checkedout' 和 '--^moved'，则在

已签出并移动文件的情况下，两项更改都将被撤销。

--^checkedout 选择已签出的文件和目录。

--^unchanged 选择内容不变的文件。

--^changed 选择本地更改的或签出的文件和

目录。

--^deleted 选择已删除的文件和目录。

--^moved 选择已移动的文件和目录。

--^added 选择已添加的文件和目录。

== CMD\_HELP\_UNDO ==

备注：

'^undo' 命令有风险，因为该命令在撤销工作后将不可逆。

^undo 完成后，无法将受其影响的文件和目录

恢复到先前状态。如果参数中

未指定路径，则默认情况下，该命令将撤销

当前目录中的所有更改，但不采用递归方式。

从 /src 目录执行时，以下命令是等效的：

/src

|- file.txt

|- code.cs

\- /test

|- test\_a.py

\- test\_b.py

cm ^undo

cm ^undo \*

cm ^undo file.txt code.cs /test

cm ^undo .

cm ^undo /src file.txt code.cs

如果希望操作以递归方式执行，必须指定 '-^r' 标志。

撤销某一目录下的所有更改（包括影响目录

本身的更改）：

cm ^undo 目录路径 -^r

如果目录路径是工作区路径，则工作区中的每个更改都将

被撤销。

已删除项：

要撤销对文件和目录的删除，必须指定项的

完整路径，或者指定所属目录并使用递归 ('-^r')

标志。

例如：

cm ^undo .

（不撤销当前目录中的删除。）

cm ^undo .-^r

（以递归方式撤销当前目录中的所有删除（和其他更改）。）

cm ^undo src/file.txt

（撤销对 src/file.txt 的删除（或其他更改）。）

示例：

cm ^undo .-^r

（以递归方式撤销当前目录中的所有更改。如果从工作区的

根目录执行，则撤销整个工作区中的所有更改。）

cm ^co file.txt

cm ^undo file.txt

（撤销对 'file.txt' 的签出。）

^echo ^content >> file.txt

cm ^undo file.txt

（撤销对 'file.txt' 的本地更改。）

cm ^undo src

（撤销对 src 目录及其文件的更改。）

cm ^undo src/\*

（撤销对 src 中包含的每个文件和目录的更改，而不会

影响 src。）

cm ^undo \*.cs

（撤销对当前目录中与 \*.cs 匹配的每个文件或目录的

更改。）

cm ^undo \*.cs -^r

（撤销对当前目录中以及下级所有目录中与 \*.cs 匹配的

每个文件或目录的更改。）

cm ^co file1.txt file2.txt

^echo ^content >> file1.txt

cm ^undo --^unchanged

（撤销对未更改的 'file2.txt' 的签出，并忽略本地更改的

'file1.txt'。）

^echo ^content >> file1.txt

^echo ^content >> file2.txt

cm ^co file1.txt

cm ^undo --^checkedout

（撤销已签出文件 'file1.txt' 中的更改，并忽略 'file2.txt'，

因为后者未签出。）

cm ^add file.txt

cm ^undo file.txt

（撤销添加 'file.txt'，使该文件再次成为私有文件。）

^rm file1.txt

^echo ^content >> file2.txt

cm ^add file3.txt

cm ^undo --^deleted --^added \*

（撤销 'file1.txt' 的删除和 'file3.txt' 的添加，并忽略 'file2.txt'

的更改。）

== CMD\_DESCRIPTION\_LOCK\_UNLOCK ==

撤销锁定服务器上的项锁定。

== CMD\_USAGE\_LOCK\_UNLOCK ==

用法：

cm ^lock ^unlock [<存储库服务器规格>] <GUID>[ ...]

存储库服务器规格 存储库服务器规格。（使用 'cm ^help ^objectspec'

可进一步了解存储库服务器规格。）

GUID 要解锁的项 GUID 的列表。使用空格对各个 GUID

进行分隔。

== CMD\_HELP\_LOCK\_UNLOCK ==

备注：

- 该命令使用指定的服务器来解锁项。

- 如果未指定服务器，则该命令会尝试从当前工作区

获取服务器。

- 如果在先前的步骤中未计算服务器，则会从当前的

Plastic SCM 客户端配置中获取服务器。

- 只有服务器的管理员才能运行 'cm ^unlock' 命令。

- 要指定 GUID，格式应为 32 位数字，并用

短横线分隔（也可以选择用花括号括起来）：

{00000000-0000-0000-0000-000000000000}

或 00000000-0000-0000-0000-000000000000

示例：

cm ^lock ^unlock 91961b14-3dfe-4062-8c4c-f33a81d201f5

（撤销所选项锁定。）

cm ^lock ^unlock DIGITALIS:8084 2340b4fa-47aa-4d0e-bb00-0311af847865 \

bcb98a61-2f62-4309-9a26-e21a2685e075

（撤销名为 'DIGITALIS' 的锁定服务器上的所选项锁定。）

cm ^lock ^unlock tardis@cloud 4740c4fa-56af-3dfe-de10-8711fa248635 \

71263c17-5eaf-5271-4d2c-a25f72e101d4

（撤销名为 'tardis' 的 Cloud 锁定服务器上的所选项锁定。）

== CMD\_DESCRIPTION\_UPDATE ==

更新工作区并下载最新的更改。

== CMD\_USAGE\_UPDATE ==

用法：

cm ^update [<项路径> | --^last]

[--^changeset=<变更集规格>] [--^cloaked] [--^dontmerge] [--^forced]

[--^ignorechanges] [--^override] [--^recursewk] [--^skipchangedcheck]

[--^silent] [--^verbose] [--^xml[=<输出文件>]] [--^encoding=<名称>]

[--^machinereadable [--^startlineseparator=<分隔符>]

[--^endlineseparator=<分隔符>] [--^fieldseparator=<分隔符>]]

[--^forcedetailedprogress]

项路径 要更新的路径。

使用 . 将更新应用到当前目录。

如果未指定路径，则当前工作区

将完全更新。

--^last 在更新之前，将工作区选择器

从变更集配置或标签配置更改为

分支配置。

选择器将更改为变更集或标签

所属的分支。

选项：

--^changeset 将工作区更新为特定变更集。

（使用 'cm ^help ^objectspec' 可进一步了解

变更集规格。）

--^cloaked 在更新操作中包含掩蔽的项。

如果未指定此选项，则在操作中

将忽略掩蔽的项。

--^dontmerge 如果在更新操作期间需要

合并更新，请不要执行合并。

--^forced 强制将项更新为选择器中指定的

修订。

--^ignorechanges 忽略在更新工作区时

检测到待定更改的情况下显示

的待定更改警告消息。

--^override 覆盖不受 Plastic SCM 控制的已更改文件。

这些文件的内容将被服务器内容

覆盖。

--^recursewk 更新在当前路径中找到的所有

工作区。适合用于更新特定路径中包含的

所有工作区。

--^skipchangedcheck 该更新在开始之前会检查工作区中是否

存在本地更改。如果始终在修改

文件之前签出，则可以进行此检查并

加快操作速度。

--^silent 不显示任何输出。

--^verbose 显示其他信息。

--^xml 以 XML 格式将输出打印到标准输出。

可以指定输出文件。

--^encoding 与 --^xml 选项结合使用，指定要在 XML 输出

中使用的编码（例如：utf-8）。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，

指定行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，

指定行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，

指定应如何分隔字段。

--^forcedetailedprogress 即使在标准输出被重定向的情况下也要强制显示

详细进度。

== CMD\_HELP\_UPDATE ==

备注：

'^update' 命令仅下载所需的文件。

该命令采用递归操作。

使用 --^last' 选项时，无需指定路径。

在这种情况下，当前工作目录所属的工作区

将更新。

（切记，如果工作区选择器先前指向某个

变更集或标签，则指定该标志可能导致

工作区选择器更改为分支配置。）

示例：

cm ^update

（更新当前工作区中的所有项。）

cm ^update .

（更新当前目录以及所有子项。）

cm ^update .--^forced --^verbose

（强制检索所有修订。）

cm ^update --^last

cm ^update .--^machinereadable --^startlineseparator=">"

（更新当前目录，并以易于解析的简化格式

打印结果，以指定的字符串作为行的

开头。）

== CMD\_DESCRIPTION\_VERSION ==

显示当前的客户端版本号。

== CMD\_USAGE\_VERSION ==

用法：

cm ^version

== CMD\_HELP\_VERSION ==

== CMD\_DESCRIPTION\_WHOAMI ==

显示当前的 Plastic SCM 用户。

== CMD\_USAGE\_WHOAMI ==

用法：

cm ^whoami

== CMD\_HELP\_WHOAMI ==

== CMD\_USAGE\_WKTREENODESTATUS ==

用法：

cm ^wktreenodestatus 路径1, 路径2, ...

== CMD\_DESCRIPTION\_WORKSPACE ==

允许用户管理工作区。

== CMD\_USAGE\_WORKSPACE ==

用法：

cm ^workspace | ^wk <命令> [选项]

命令：

^list | ^ls

^create | ^mk

^delete | ^rm

^move | ^mv

^rename

要获取有关每条命令的更多信息，请运行：

cm ^workspace <命令> --^usage

cm ^workspace <命令> --^help

== CMD\_HELP\_WORKSPACE ==

示例：

cm ^workspace ^create myWorkspace 工作区路径

cm ^workspace ^list

cm ^workspace ^delete myWorkspace

== CMD\_DESCRIPTION\_WORKSPACE\_CREATE ==

创建新的工作区。

== CMD\_USAGE\_WORKSPACE\_CREATE ==

用法：

cm ^workspace | ^wk [^create | ^mk] <工作区名称> <工作区路径> [<存储库规格>]

[--^selector[=<选择器文件>]

（创建新的工作区。）

cm ^workspace | ^wk [^create | ^mk] <工作区名称> <工作区路径> --^dynamic --^tree=[<树>]

（创建动态工作区。此功能仍处于实验阶段，

仅适用于 Windows。）

工作区名称 新工作区的名称。

工作区路径 新工作区的路径。

存储库规格 使用指定的存储库创建新的工作区。

存储库规格：查看 'cm ^help ^objectspec'。

选项：

--^selector 编辑新工作区的选择器。

如果指定了选择器文件，则会从指定文件

为新工作区设置选择器。

--^dynamic 创建动态工作区。此功能仍处于

实验阶段，仅适用于 Windows。

指定此标志需要使用 --^tree 参数。

--^tree 与 '--^dynamic' 标志结合使用，指定要

加载动态工作区的初始点。此选项

可以是分支、变更集或标签规格。

工作区稍后将使用规格中的

存储库。（使用 'cm ^help ^objectspec' 可进一步了解规格。）

== CMD\_HELP\_WORKSPACE\_CREATE ==

备注：

- 工作区是映射到本地文件系统的存储库的视图。

工作区选择器可以定义用于指定工作区内容的规则。

使用 'cm ^showselector' 显示工作区选择器，或使用 'cm ^setselector' 修改

工作区选择器。

- 如果存储库规格和 '--^selector' 均未指定，则工作区

将自动配置为使用 client.conf 文件中配置的

服务器的第一个存储库（按字母顺序）。

- 动态工作区是一项实验性功能（仅适用于 Windows），

并需要运行 plasticfs.exe 程序。

示例：

cm ^workspace ^create myworkspace c:\workspace

cm ^wk ^mk myworkspace /home/john/plastic\_view

（分别在 Windows 和 Linux 中创建 'myworkspace' 工作区。）

cm ^wk mywktest c:\wks\wktest --^selector=myselector.txt

（使用 'myselector.txt' 文件中的选择器创建 'mywktest' 工作区。）

cm ^wk mywkprj c:\wks\wkprj myrep@^repserver:localhost:8084

（使用所选存储库创建 'mywkprj' 工作区。）

cm ^wk mywkprj c:\dynwks\mywkprj --^dynamic --^tree=^br:/main@myrep@localhost:8084

（使用 'myrep@localhost:8084' 存储库创建动态的 'mywkprj'

工作区，并在首次装入该工作区时指向 '^br:/main'。）

== CMD\_DESCRIPTION\_WORKSPACE\_DELETE ==

删除工作区。

== CMD\_USAGE\_WORKSPACE\_DELETE ==

用法：

cm ^workspace | ^wk ^delete | ^rm [<工作区路径> | <工作区规格>] [--^keepmetadata]

工作区路径 要删除的工作区的路径。

工作区规格 要删除的工作区的规格。（使用

'cm ^help ^objectspec' 可进一步了解规格。）

选项：

--^keepmetadata 不删除 .plastic 文件夹中的元数据

文件。

== CMD\_HELP\_WORKSPACE\_DELETE ==

备注：

此命令将删除由路径或规格指定的工作区。

如果未指定任何参数，则将采用当前工作区。

示例：

cm ^workspace ^delete

（删除当前工作区。）

cm ^wk ^delete c:\workspace

cm ^workspace rm /home/danipen/wks

cm ^wk ^rm ^wk:MiWorkspace

cm ^wk ^rm ^wk:MiWorkspace@DIGITALIS

== CMD\_DESCRIPTION\_WORKSPACE\_LIST ==

列出工作区。

== CMD\_USAGE\_WORKSPACE\_LIST ==

用法：

cm ^workspace | ^wk [^list | ^ls] [--^format=<格式字符串>]

选项：

--^format 检索特定格式的输出消息。请参阅

“备注”以了解更多信息。

== CMD\_HELP\_WORKSPACE\_LIST ==

备注：

此命令接受格式字符串以显示输出。

此命令的输出参数如下：

{0} | {^wkname} 工作区名称。

{1} | {^machine} 客户端机器名称。

{2} | {^path} 工作区路径。

{3} | {^wkid} 工作区唯一标识符。

{4} | {^wkspec} 使用以下格式的工作区规格：

'wkname@machine'。

{^tab} 插入一个制表符空格位。

{^newline} 插入一个新行。

示例：

cm ^wk

（列出所有工作区。）

cm ^workspace ^list --^format={0}#{3,40}

cm ^workspace ^list --^format={^wkname}#{^wkid,40}

（列出所有工作区，并在 40 个空格位中显示工作区名称、# 符号

和工作区 GUID 字段（左对齐）。）

cm ^wk --^format="路径 {2} 中的工作区 {0}"

cm ^wk --^format="路径 {^path} 中的工作区 {^wkname}"

（列出所有工作区并将结果显示为格式化的字符串。）

== CMD\_DESCRIPTION\_WORKSPACE\_MOVE ==

移动工作区。

== CMD\_USAGE\_WORKSPACE\_MOVE ==

用法：

cm ^workspace | ^wk ^move | ^mv [<工作区规格>] <新路径>

选项：

工作区规格 要移动的工作区的规格。（使用

'cm ^help ^objectspec' 可进一步了解规格。）

新路径 工作区将移动到此处。

== CMD\_HELP\_WORKSPACE\_MOVE ==

备注：

此命令允许用户将工作区移动到磁盘上的另一个位置。

示例：

cm ^workspace ^move myWorkspace \new\workspaceDirectory

（将 'myWorkspace' 移动到指定位置。）

cm ^wk ^mv c:\users\maria\wkspaces\newlocation

（将当前工作区移动到新位置。）

== CMD\_DESCRIPTION\_WORKSPACE\_RENAME ==

重命名工作区。

== CMD\_USAGE\_WORKSPACE\_RENAME ==

用法：

cm ^workspace | ^wk ^rename [<工作区名称>] <新名称>

工作区名称 要重命名的工作区。

新名称 工作区的新名称。

== CMD\_HELP\_WORKSPACE\_RENAME ==

备注：

此命令将重命名工作区。

如果未提供工作区名称，则将使用当前工作区。

示例：

cm ^workspace ^rename mywk1 wk2

（将工作区 'mywk1' 重命名为 'wk2'。）

cm ^wk ^rename newname

（将当前工作区重命名为 'newname'。）

== CMD\_DESCRIPTION\_WORKSPACESTATUS ==

显示工作区中的更改。

== CMD\_USAGE\_WORKSPACESTATUS ==

用法：

cm ^status [<工作区路径>] [--^changelist[=<名称>] | --^changelists] [--^cutignored]

[ --^header] [ --^noheader] [ --^nomergesinfo] [ --^head]

[--^short] [--^symlink] [ --^dirwithchanges] [--^xml[=<输出文件>]]

[--^encoding=<名称>] [ --^wrp | --^wkrootrelativepaths]

[--^fullpaths | --^fp] [<旧选项>] [<搜索类型>[ ...]]

[--^machinereadable [--^startlineseparator=分隔符]

[--^endlineseparator=分隔符] [--^fieldseparator=分隔符]]

选项：

工作区路径 要作为搜索范围的工作区的

路径。

--^changelist 显示所选更改列表中的更改。

--^changelists 显示按客户端更改列表进行分组的更改。

--^cutignored 跳过已忽略目录的内容。

需要 '--^ignored' 搜索类型。请参阅

“搜索类型”部分以了解更多信息。

--^header 仅打印工作区状态。

--^noheader 仅打印修改后的项搜索结果。

--^nomergesinfo 不打印更改的合并信息。

--^head 打印分支上最后一个变更集的状态。

--^short 仅列出包含更改的路径。

--^symlink 将操作应用于符号链接而不是

目标。

--^dirwithchanges 显示含有更改的目录

（含有已添加、移动、删除的项）。

--^xml 以 XML 格式将输出打印到标准输出。

可以指定输出文件。

--^encoding 与 --^xml 选项结合使用，指定要在 XML 输出

中使用的编码（例如：utf-8）。

请参阅位于以下网址的 MSDN 文档：

http://msdn.microsoft.com/en-us/library/system.text.encoding.aspx

以查看包含受支持编码及其格式的表格

（位于页面末尾的“名称”列中）。

--^wrp 打印工作区根目录的相对路径，而不是

当前目录的相对路径。

--^fullpaths, --^fp 强制打印绝对路径，覆盖所有其他

路径打印设置。

--^machinereadable 以易于解析的格式输出结果。

--^startlineseparator 与 '--^machinereadable' 标志结合使用，

指定行应如何开头。

--^endlineseparator 与 '--^machinereadable' 标志结合使用，

指定行应如何结尾。

--^fieldseparator 与 '--^machinereadable' 标志结合使用，

指定应如何分隔字段。

旧选项：

--^cset 以旧格式打印工作区状态。

--^compact 以旧格式打印工作区状态和

更改列表。

--^noheaders 与 '--^compact' 标志结合使用时，不会

打印更改列表标题。（不适用于

新更改列表格式。）

搜索类型：

--^added 打印已添加的项。

--^checkout 打印已签出的项。

--^changed 打印已更改的项。

--^copied 打印已复制的项。

--^replaced 打印已替换的项。

--^deleted 打印已删除的项。

--^localdeleted 打印已在本地删除的项。

--^moved 打印已移动的项。

--^localmoved 打印已在本地移动的项。

--^percentofsimilarity=<值> 为了将两个文件视为同一个项而需要

两者之间达到的相似程度百分比。用于

搜索已在本地移动的项。默认值

为 20%。

--^txtsameext 在搜索已移动的项期间，

相似性内容匹配过程将

仅考虑具有相同扩展名的

文本文件。默认情况下，

将处理任何文本文件。

--^binanyext 在搜索已移动的项期间，

相似性内容匹配过程将

考虑任何二进制文件。默认情况下，

只会处理具有相同扩展名的

二进制文件。

--^private 打印非受控项。

--^ignored 打印已忽略的项。

--^hiddenchanged 打印隐藏的已更改项。（包括

'--^changed'）

--^cloaked 打印掩蔽的项。

--^controlledchanged 此标志替代以下选项：

'--^added'、'--^checkout'、'--^copied'、

'--^replaced'、'--^deleted'、'--^moved'。

--^all 此标志替换以下参数：

'--^controlledchanged'、'--^changed'、

'--^localdeleted'、'--^localmoved'、'--^private'。

== CMD\_HELP\_WORKSPACESTATUS ==

备注：

'^status' 命令将打印工作区中已加载的变更集，并在工作区

内获取已更改的元素。

此命令可用于显示工作区中的待定更改；

可以使用命令参数来修改可以搜索的

更改类型。默认情况下将显示所有更改，无论是

受控更改还是本地更改。

搜索已在本地移动的项时会使用相似性参数百分比

'--^percentofsimilarity' (-^p) 来确定两个元素是否为同一个项。

默认值为 20%，但可以调整。

可以显示按客户端更改列表分组的工作区更改。

'^default' 更改列表包括其他更改列表中

未包含的更改。因此，默认更改列表将显示的更改

取决于指定的搜索类型标志。

要显示按更改列表分组的更改，还需要显示受控

的更改（状态为 '^added'、'^checkout'、'^copied'、

、'^replaced'、'^deleted' 或 '^moved' 的项）。因此，当显示更改列表时，

'--^controlledchanged' 选项将自动启用。

XML 输出的默认编码为 utf-8。

默认情况下，此命令将打印当前目录的相对路径，

除非指定了 '--^machinereadable' 或 '--^short' 标志。如果

指定了其中任何一个标志，该命令将打印绝对路径。

如果指定了 '--^xml' 标志，则将打印工作区根目录的

相对路径（除非也指定了 '--^fp' 标志，那么将

打印绝对路径）。

示例：

cm ^status

（打印工作变更集以及工作区中所有已更改的

项类型，但已忽略的项除外。）

cm ^status --^controlledchanged

（打印工作变更集以及已签出、添加、复制、替换、

删除和移动的项。）

cm ^status --^added

（仅打印工作变更集和工作区中添加的项。）

cm ^status c:\workspaceLocation\code\client --^added

（以递归方式打印指定路径下的工作变更集和

添加的项。）

cm ^status --^changelists

cm ^status --^changelist

（显示按客户端更改列表分组的所有工作区更改。）

cm ^status --^changelist=pending\_to\_review

（显示名为 'pending\_to\_review' 的更改列表中的更改。）

cm ^status --^changelist=default --^private

（显示 'default' 更改列表中的更改，包括私有项以及

具有受控更改的项（如果有）。）

cm ^status --^short --^changelist=pending\_to\_review | cm ^checkin -

（签入更改列表 'pending\_to\_review' 中的更改。）

cm ^status C:\workspaceLocation --^xml=output.xml

（获取 XML 格式的状态信息，并在文件 output.xml 中

使用 utf-8。）

cm ^status --^ignored

（显示所有已忽略的项。）

输出：

/main@myrepo@local (^cs:2 - ^head)

^Added

状态 大小 上次修改时间 路径

^Ignored 0 字节 19 秒前 out\app.exe

^Ignored 48 秒前 src

^Ignored 0 字节 48 秒前 src\version.c

cm ^status --^ignored --^cutignored

（显示父目录未被忽略的已忽略文件，并显示

内容未被忽略的已忽略目录。）

输出：

/main@myrepo@local (^cs:2 - ^head)

^Added

状态 大小 上次修改时间 路径

^Ignored 0 字节 19 秒前 out\app.exe

^Ignored 48 秒前 src

== CMD\_DESCRIPTION\_XLINK ==

创建、编辑或显示 Xlink 的详细信息。

== CMD\_USAGE\_XLINK ==

用法：

cm ^xlink [-^w] [-^rs] <Xlink\_路径> / (<变更集规格> | <标签规格> | <分支规格)>

[<扩展规则>[ ...]]

（创建一个 Xlink。）

cm ^xlink [-^rs] <Xlink\_路径> /<相对路径> (<变更集规格> | <标签规格> | <分支规格>)

[<扩展规则>[ ...]]

（创建指向 /<相对路径>（而不是指向默认根目录 /）的只读

部分 Xlink。）

cm ^xlink -^e <Xlink\_路径> (<变更集规格> | <标签规格> | <分支规格>)

（编辑 Xlink 以更改目标规格。）

cm ^xlink -^s|--^show <Xlink\_路径>

（显示 Xlink 信息，包括扩展规则。）

cm ^xlink -^ar|--^addrules <Xlink\_路径> <扩展规则>[ ...]

（将给定的扩展规则添加到 Xlink。）

cm ^xlink -^dr|--^deleterules <Xlink\_路径> <扩展规则>[ ...]

（从 Xlink 中删除给定的扩展规则。）

Xlink\_路径 这是当前工作区中将装入（创建 Xlink 时）

或已经装入（编辑 Xlink 时）

已链接的存储库的目录。

变更集规格 远程存储库中的完整目标变更集

规格。

此规格确定了已链接的存储库的工作区中

加载了哪个版本和分支。

（使用 'cm ^help ^objectspec' 可进一步了解变更集

规格。）

标签规格 远程存储库中的完整标签规格。

（使用 'cm ^help ^objectspec' 可进一步了解标签

规格。）

分支规格 远程存储库中的完整分支规格。

此规格使用指定的分支所指向的

当前变更集。（使用 'cm ^help ^objectspec' 可

可进一步了解分支规格。）

-^e 编辑现有的 Xlink 以更改目标变更集

规格。

-^s | --^show 显示有关所选 Xlink 的信息。

-^ar | --^addrules 将一个或多个扩展规则添加到所选 Xlink。

-^dr | --^deleterules 从所选 Xlink 中删除一个或多个扩展

规则。

扩展规则 指定一个或多个扩展规则。每个扩展

规则都是一对分支/目标分支：

^br:/main/fix-^br:/main/develop/fix

选项：

-^w 表示 Xlink 是可写的。这意味着

可以通过分支自动扩展功能来修改

目标存储库的内容。

-^rs 相对服务器。此选项允许创建独立于

存储库服务器的相对 Xlink。这样，

在不同服务器的复制存储库中

创建的 Xlink 将被自动识别。

== CMD\_HELP\_XLINK ==

备注：

此命令将创建指向给定变更集的 Xlink。默认情况下将创建一个

只读 Xlink。这意味着无法修改 Xlink 内部工作区中

加载的内容。为了能够更改通过 Xlink 链接的内容，

请创建可写的 Xlink（使用 '-^w' 选项）。

在编辑 Xlink 的目标变更集时，可以使用

该命令的简化语法。这样，唯一需要的参数是

新的目标变更集。Xlink 的其余参数将不会

被修改。

分支自动扩展：

在任何可写 Xlink 链接的存储库（'-^w' 选项）中进行更改时，

需要在目标存储库中创建一个新分支。新分支的

名称基于顶级存储库中定义的签出

分支。为了确定要使用的分支的名称，遵循以下规则：

1) 系统会检查目标存储库中是否存在具有

相同全名的分支：

- 如果存在，则将此分支用作签出分支。

- 如果不存在，则通过以下方式构建分支名称：

- 通过 Xlink 链接的目标变更集的分支名称 + 签出分支

的简短名称（最后一部分）。

- 如果存在此分支，则将此分支用作签出分支。

- 否则，将创建分支并将分支基项设置为

通过 Xlink 链接的变更集。

2) 在父存储库的分支中创建 Xlink 的新版本，

指向通过 Xlink 链接的存储库中的新变更集。

最后，完整的 Xlink 结构会与正确版本中的

最新更改保持同步。

示例：

cm ^xlink code\firstrepo / 1@first@localhost:8084

（在当前工作区的文件夹 'firstrepo' 中创建一个 Xlink，

在其中将装入存储库 'first' 中的变更集 '1'。）

cm ^xlink opengl\include /includes/opengl 1627@includes@localhost:8087

（在当前工作区的目录 'opengl\include' 中创建只读

部分 Xlink，在其中，存储库 'includes' 的变更集 '1627' 中的路径

'/includes/opengl' 将作为根目录装入。这意味着 '/includes/opengl'

中的任何内容都将装入到 'opengl\include' 中，

而存储库的其余部分将被忽略。）

cm ^xlink -^w -^rs code\secondrepo / ^lb:LB001@second@localhost:8084

（在当前工作区的文件夹 'secondrepo' 中创建一个

可写的相对 Xlink，在其中将装入存储库 'second' 中的

标签 'LB001'。）

cm ^xlink code\thirdrepo / 3@third@localhost:8087 ^br:/main-^br:/main/scm003

（在当前工作区的文件夹 'thirdrepo' 中创建一个 Xlink，

在其中将装入存储库 'third' 中的变更集 '3'。）

cm ^xlink -^e code\secondrepo ^br:/main/task1234@second@localhost:8084

（通过链接存储库 'second' 中的分支 'main/task1234'，

编辑 Xlink 'code\secondrepo' 以更改目标存储库。）

cm ^xlink --^show code\thirdrepo

（显示 Xlink 'code\thirdrepo' 的信息，包括其扩展规则

（如果存在））。

cm ^xlink -^ar code\secondrepo ^br:/main-^br:/main/develop ^br:/main/fix-^br:/main/develop/fix

（将两个扩展规则添加到 Xlink 'code\secondrepo'。）

cm ^xlink -^dr code\secondrepo ^br:/main/fix-^br:/main/develop/fix

（从 Xlink 'code\secondrepo' 中删除扩展规则）。

== CMD\_USAGE\_AUTOCOMPLETE ==

用法：

cm ^autocomplete ^install

（在 shell 中安装 'cm' 命令补齐功能。）

cm ^autocomplete ^uninstall

（从 shell 中卸载 'cm' 命令补齐功能。）

cm ^autocomplete --^line <shell\_行> --^position <光标位置>

（返回自动补齐建议，以便将“shell\_行”插入

“光标位置”。此命令并非供最终用户使用，但这里记载

此命令是考虑到您可能需要扩展对所选 shell 的

自动补齐支持。）

shell\_行 当请求自动补齐时由用户

写入 shell 的行。

在 Bash 中，此值位于 COMP\_LINE 环境变量中。

在 PowerShell 中，则位于 $wordToComplete 变量中。

光标位置 请求自动补齐时的

光标位置。

在 Bash 中，此值位于 COMP\_POINT 环境变量中。

在 PowerShell 中，则位于 $cursorPosition 变量中。