﻿== CMD\_DESCRIPTION\_ACL ==

オブジェクトに権限を設定します。

== CMD\_USAGE\_ACL ==

使用方法:

cm ^acl (--^user=<ユーザー名> | --^group=<グループ名>)

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

<オブジェクト指定>

--^user ユーザー名。

--^group グループ名。

-^allowed 指定した権限を有効にします。コンマを

使用して権限を区切ります。('cm ^showpermissions'

を使用して利用できるすべての権限を表示します。)

-^denied 指定した権限を拒否します。コンマを

使用して権限を区切ります。('cm ^showpermissions'

を使用して利用できるすべての権限を表示します。)

-^overrideallowed 許可された権限をオーバーライドします。コンマを

使用して権限を区切ります。('cm ^showpermissions'

を使用して利用できるすべての権限を表示します。)

-^overridedenied 拒否された権限をオーバーライドします。コンマを

使用して権限を区切ります。('cm ^showpermissions'

を使用して利用できるすべての権限を表示します。)

オブジェクト指定 権限が設定されるオブジェクト。

このコマンドで有効なオブジェクト:

リポジトリサーバー、リポジトリ、ブランチ、変更セット、ラベル、項目、

属性。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

セキュリティで保護されたパスの特別な使用方法:

cm ^acl [(--^user=<ユーザー名> | --^group=<グループ名>)

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

[--^delete] [--^branches=[+ | -]<ブランチ>[,...]]

<指定>

--^delete セキュリティで保護されたパスを削除します。

詳細については、「備考」を参照してください。

--^branches セキュリティで保護されたパスの権限をブランチのグループに設定します。

コンマを使用してブランチを区切ります。

オプションで、各ブランチの前に + または - 記号を付けて

編集中にブランチをリストに追加する必要があるかどうかを

指定します。

詳細については、「備考」を参照してください。

指定 権限を設定するセキュリティで保護されたパス。

== CMD\_HELP\_ACL ==

権限を設定するには、Plastic SCM の仕組みについて理解しておく必要があります。

権限の仕組みについては、セキュリティガイドを参照してください。

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

備考:

このコマンドは、指定されたオブジェクト、リポジトリ、ブランチ、ラベル、

サーバーパス上のユーザーまたはグループの権限を設定します。

オブジェクト指定:

('cm ^help ^objectspec' を使用してオブジェクトの指定方法を確認できます。)

'^acl' コマンドは、特殊なタイプの指定 (セキュリティで保護されたパス) を使用します。

- セキュリティで保護されたパスの指定:

^path:サーバーパス[#タグ]

例: ^path:/src/foo.c

^path:/doc/pdf

^path:/doc/pdf#documents

権限のアクション:

-^allowed と -^denied を使用して設定する権限を指定します。

-^overrideallowed 引数と -^overridedenied 引数を使用してオーバーライドする

権限を指定します。

各アクションにはコンマで区切られた権限のリストが必要です。

権限の名前:

各権限名の前には + または - 記号が付きます。

+ 記号は権限を設定し、- 記号はそれを消去します。

オブジェクトの権限を確認するには、'cm ^showacl' コマンドを使用します。

オーバーライドされた権限:

-^overrideallowed と -^overridedenied

を使用して権限をオーバーライドすると、継承をバイパスすることができます。

これは、リポジトリまたはサーバーレベルで設定された権限をバイパスするのに

便利です。

例:

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

(ユーザー 'vio' がリポジトリレベルで拒否された権限を持っている場合でも、

リポジトリ 'test' 上のブランチ 'qa' にチェックインすることを許可します。)

サーバーパスの権限 (別名セキュリティで保護されたパス):

- 指定されたサーバーパスに権限を指定することができます。

- それらの権限はチェックイン操作中にチェックされます。

- それらの権限は更新操作中にもチェック可能で、

特定のディレクトリやファイルがワークスペースにダウンロードされるのを

回避するのに使用できます。

- チェックインするすべての項目につき、サーバーは項目パスをセキュリティで保護されたパスと

一致させようとします。一致する場合、チェックイン操作はその項目に

チェックインする権限があるかどうかをチェックします。

セキュリティで保護されたパスに定義できる権限は

次のとおりです:

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

関連するいずれの項目でも権限のチェックが成功しなかった

場合、チェックイン操作はロールバックされます。

セキュリティで保護されたパスの権限をブランチのグループに設定するには、

--^branches オプションを使用します。

例:

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

そのタグは、セキュリティで保護されたパスに関連付けられた ACL を編集するのに便利です。

例:

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

(タグがない場合、そのブランチのリストを再度指定する必要が

あります。)

セキュリティで保護されたパスのブランチのリストは編集できます。

例:

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

(リストから 'main' を削除し、'main/rel1' を追加します。)

セキュリティで保護されたパスを削除するには、--^delete 引数を使用します。

例:

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

継承:

継承は、Plastic SCM 3.0 から追加されたオプションです。

高度なオプションですが、ほぼ非推奨です。

オブジェクトにデフォルトの継承の関係性をオーバーライドしつつ、

他のあらゆるオブジェクトから権限を継承することを許可します。

オプション -^cut を使用して継承のチェーンを切ります。

オプション -^cutncpy を使用して現在の継承された権限を切って

コピーします。(これは、継承を切りながらも、実際の権限を保持できる、

Windows ファイルシステムの権限から着想を得ています。)

-^inherit オプションを使用すると、ユーザーにオブジェクト指定から継承することを許可します。

例: '-^inherit=オブジェクト指定'

例:

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

(リポジトリ 'core' 上のユーザー 'danipen' のチェックインを拒否します。)

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

(コマンドは、'main' ブランチ内の 'developers' グループに表示権限を付与し、

読み取り権限を消去して、chgperm 権限を拒否します。)

セキュリティで保護されたパスの例:

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

(コマンドはブランチ 'main' と 'main/2.0' 内で '/server' が一致するあらゆるパスについて

'devs' に対するチェックイン権限を拒否します。タグ '#rel'

が作成され、後でそれを参照できるようにします。)

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

(タグが 'rel' のセキュリティで保護されたパス '/server' を更新し、

そのセキュリティで保護されたパスが適用されるブランチグループから 'main' ブランチを削除して、

ブランチ 'main/Rel2.1' を追加します。前の例を考慮すると、

これでそのブランチのリストに 'main/Rel2.1' と 'main/2.0' が含まれるようになります。)

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

('vsanchezm' の '^read' 権限を削除し、それを '/doc' パス内でオーバーライドします。)

== CMD\_DESCRIPTION\_ACTIVATEUSER ==

ライセンスが付与されたユーザーをアクティベートします。

== CMD\_USAGE\_ACTIVATEUSER ==

使用方法:

cm ^activateuser | ^au <ユーザー名>[ ...][--^server=<リポジトリサーバー指定>]

ユーザー名 アクティベートするユーザー名。空白が含まれるユーザー名を

指定するには二重引用符 (" ") を使用します。空白を使用して

ユーザー名を区切ります。

オプション:

--^server=<リポジトリサーバー指定> 指定されたサーバー内のユーザーをアクティベートします。

サーバーが指定されていない場合は、client.conf ファイル内の

デフォルトサーバーでそのコマンドを実行します。

('cm ^help ^objectspec' を使用してリポジトリサーバー指定の

詳細を確認できます。)

== CMD\_HELP\_ACTIVATEUSER ==

備考:

ユーザーをアクティベートするには、以前にアクティベート解除されている必要があります。

デフォルトでは、ユーザーは Plastic SCM で書き込み操作を最初に実行するときに

アクティベートされます。ユーザーが自動的にアクティベートされるのは、

ユーザーの最大数が超えていない場合のみです。

Plastic SCM ユーザーのアクティベート解除の詳細については

'cm ^help ^deactivateuser' コマンドを参照してください。

例:

cm ^activateuser john

cm ^activateuser david "mary collins"

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

== CMD\_DESCRIPTION\_ADD ==

バージョン管理に項目を追加します。

== CMD\_USAGE\_ADD ==

使用方法:

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

[--^skipcontentcheck] [--^coparent] [--^filetypes=<ファイル>] [--^noinfo]

[--^format=<文字列形式>] [--^errorformat=<文字列形式>]

<項目パス>[ ...]

項目パス 追加する 1 つまたは複数の項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用して項目を区切ります。

\* を使用して現在のディレクトリのすべてのコンテンツを追加します。

オプション:

-^R -^r --^recursive 項目を再帰的に追加します。

--^silent 出力を表示しません。

--^ignorefailed 項目を追加できない場合、追加操作は

それなしで続行されます。注意: ディレクトリを追加できない

場合、そのコンテンツは追加されません。

--^skipcontentcheck 拡張子がファイルをテキストまたはバイナリとして設定するのに

十分でない場合、タイプを検出するのにコンテンツをチェックする代わりに、

バイナリとして設定されます。これは

大規模なチェックインのパフォーマンスを高めるために行われます。

--^coparent 追加される項目の親のチェックアウトを実行します。

--^filetypes 使用するファイルタイプのファイル。詳細については、次のリンクを

参照してください:

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

--^noinfo 進捗情報を出力しません。

--^format 出力メッセージを特定の形式で取得します。詳細については、

例を確認してください。

--^errorformat エラーメッセージ (ある場合) を特定の形式で

取得します。詳細については、例を確認してください。

== CMD\_HELP\_ADD ==

備考:

項目を追加するための要件:

- 追加する項目の親ディレクトリが先に追加されている必要があります。

stdin から入力を読み取る:

'^add' コマンドは stdin からパスを読み取ることができます。これを行うには、シングルダッシュ

"-" を渡します。

例: cm ^add -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用して追加するファイルを指定できます。

例:

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

(Windows で、すべての .c ファイルをワークスペースに追加します。)

例:

cm ^add file1.txt file2.txt

('file1.txt' と 'file2.txt' の項目を追加します。)

cm ^add c:\workspace\file.txt

('file.txt' の項目をパス 'c:\workspace' に追加します。)

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

('src' を再帰的に追加します。)

cm ^add -^R \*

(現在のディレクトリのすべてのコンテンツを再帰的に追加します。)

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

(現在のディレクトリのすべてのコンテンツを再帰的に追加し、各ファイルの

コンテンツを確認する代わりに、'filetypes.conf' を使用して各ファイルの拡張子に基づいて

タイプを割り当てます。)

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

('file.txt' をソース管理に追加して、'dir' のチェックアウトを実行します。)

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

(現在のディレクトリのすべてのコンテンツを再帰的に追加し、正常に

追加されたファイルについては '^ADD <項目>' を出力して、追加できなかった項目については '^ERR <項目>' を

出力します。)

== CMD\_USAGE\_ADDIGNOREPATTERN ==

使用方法:

cm ^addignorepattern <パターン>[ ...]

[--^workspace=<ワークスペースパス> | --^allworkspaces] [--^remove]

== CMD\_DESCRIPTION\_ADMIN ==

サーバー上で管理コマンドを実行します。

== CMD\_USAGE\_ADMIN ==

使用方法:

cm ^admin <コマンド> [オプション]

使用できるコマンド:

^readonly

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^admin <コマンド> --^usage

cm ^admin <コマンド> --^help

== CMD\_HELP\_ADMIN ==

備考:

サーバーの管理者のみが管理コマンドを実行できます。

例:

cm ^admin ^readonly ^enter

cm ^admin ^readonly ^status

== CMD\_DESCRIPTION\_ADMIN\_READONLY ==

サーバーの読み取り専用モードを有効/無効にします。

== CMD\_USAGE\_ADMIN\_READONLY ==

使用方法:

cm ^admin ^readonly (^enter | ^leave | ^status) [<サーバー>]

アクション:

^enter サーバーが読み取り専用モードを開始します。書き込み操作は拒否されます。

^leave サーバーが読み取り専用モードを終了します。

^status サーバーの読み取り専用モードのステータスを表示します。

オプション:

サーバー 指定されたサーバー (サーバー:ポート) でコマンドを実行します。('cm ^help ^objectspec' を

使用してサーバー指定の詳細を確認できます。)

サーバーが指定されていない場合、コマンドは現在のワークスペースの

サーバーで機能します。

現在のパスがワークスペース内にない場合、コマンドは

client.conf 設定ファイルで定義されたデフォルトのサーバーで機能します。

== CMD\_HELP\_ADMIN\_READONLY ==

備考:

サーバーの管理者のみがサーバーの読み取り専用モードを開始できます。

例:

cm ^admin ^readonly ^enter diana:8086

cm ^admin ^readonly ^leave

== CMD\_DESCRIPTION\_ANNOTATE ==

ファイルの各行が最後に変更された変更セットとその作成者を表示します。

== CMD\_USAGE\_ANNOTATE ==

使用方法:

cm ^annotate | ^blame <指定>[ ...]

[--^format=<文字列形式>]

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

[--^dateformat=<文字列の日付形式>]

[--^encoding=<名前>]

[--^stats]

[--^repository=<リポジトリ指定>]

指定 注釈を付けるファイルの指定。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

空白が含まれるパスを指定するには二重引用符 (" ") を使用します。

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^ignore 指定された比較方法を設定します。

詳細については、「備考」を参照してください。

--^dateformat 日付を出力する出力形式を設定します。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^stats 統計情報を表示します。

--^repository 注釈を計算するために使用されるリポジトリ指定を

指定します。デフォルトでは、このコマンドでは

ワークスペース内のロードされたリビジョンリポジトリが格納されたリポジトリが

使用されます。('cm ^help ^objectspec' を使用して

リポジトリ指定の詳細を確認できます。)

== CMD\_HELP\_ANNOTATE ==

備考:

バイナリファイルに注釈を付けることはできません。

--^ignore オプション:

^none 行の終わりと空白の差異を検出します。

^eol 行の終わりの差異を無視します。

^whitespaces 空白の差異を無視します。

^"eol&whitespaces" 行の終わりと空白の差異を無視します。

--^format オプション:

このコマンドの出力パラメーターは次のとおりです。

{^owner} 最後にその行を変更したユーザー。

{^rev} その行のソースリビジョン指定。

{^content} 行のコンテンツ。

{^date} その行がチェックインされた日付。

{^comment} その行のソースリビジョンのコメント。

{^changeset} その行のソースリビジョンの変更セット。

{^line} そのファイルの行番号。

{^id} 項目 ID。

{^parentid} その項目の親 ID。

{^rep} その項目のリポジトリ。

{^branch} その行のソースリビジョンのブランチ。

{^ismergerev} マージでその行のリビジョンが作成されたかどうか。

--^dateformat:

日付が出力される出力形式を指定するため。

次のページで指定されたサポートされた形式を確認してください。

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

--^repository:

リモートリポジトリからデータを取得するため。分散型のシナリオで

便利です。

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}"

(所有者フィールドに書き込み、次を空にし、日付フィールド (右揃え

) に書き込み、次を空にして、コンテンツを書き込みます。)

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

--^dateformat=yyyyMMdd

(7 つのスペースに所有者フィールド (左揃え) を書き込み、

次を空にし、その次にコメントを書き込み、その次はまた空にして、最後に形式化された日付

(例: 20170329) を書き込みます。)

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

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

(サーバーパスを使用して変更セット 73666 で始まるファイルに注釈を付けます。)

== CMD\_DESCRIPTION\_APPLY\_LOCAL ==

ローカルの変更 (ローカルでの移動、ローカルでの削除、ローカルでの変更) を確認して

それらを適用し、Plastic SCM がそれらの変更の追跡を開始できるようにします。

== CMD\_USAGE\_APPLY\_LOCAL ==

使用方法:

cm ^applylocal | ^al [--^dependencies] [<項目パス>[ ...]]

[--^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

オプション:

--^dependencies 適用する項目にローカルの変更の依存関係を

追加します。

項目パス 適用される項目。空白を使用してパスを

区切ります。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator '--^machinereadable' フラグとともに使用され、その行をどのように開始する

必要があるかを指定します。

--^endlineseparator '--^machinereadable' フラグとともに使用され、その行をどのように終了する

必要があるかを指定します。

--^fieldseparator '--^machinereadable' フラグとともに使用され、そのフィールドをどのように区切る

必要があるかを指定します。

== CMD\_HELP\_APPLY\_LOCAL ==

備考:

--^dependencies と <項目パス> が指定されていない場合、操作にはそのワークスペース内の

すべてのローカルの変更が関与します。

常に指定されたパスから再帰的に適用されます。

例:

cm ^applylocal foo.c bar.c

cm ^applylocal .

(現在のディレクトリのすべてのローカルの変更を適用します。)

cm ^applylocal

(ワークスペース内のすべてのローカルの変更を適用します。)

cm ^applylocal --^machinereadable

(ワークスペース内のすべてのローカルの変更を適用し、その結果を

簡略化された、より解析しやすい形式で出力します。)

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

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

(ワークスペース内のすべてのローカルの変更を適用し、その結果を

簡略化された、より解析しやすい形式で出力します。指定された文字列で行を開始および終了し、

フィールドを区切ります。)

== CMD\_DESCRIPTION\_ARCHIVE ==

外部ストレージにデータをアーカイブします。

== CMD\_USAGE\_ARCHIVE ==

使用方法:

cm ^archive | ^arch <リビジョン指定>[ ...][-^c | --^comment=<コメント文字列>]

[--^file=<ベースファイル>]

(リポジトリからデータを抽出し、それを外部ストレージに格納します。)

cm ^archive | ^arch <リビジョン指定>[ ...]--^restore

(以前にアーカイブ済みのリビジョンをリポジトリに復元します。)

リビジョン指定 1 つ以上のリビジョン指定。"-" モディファイアーを使用して

STDIN から読み取ることができます。('cm ^help ^objectspec' を使用して

リビジョン指定の詳細を確認できます。)

--^restore 以前にアーカイブ済みのデータを生成されたアーカイブファイルから

復元します。

オプション:

-^c | --^comment 作成するアーカイブストレージファイルにコメントを設定します。

--^file 新しいアーカイブデータファイルのプレフィックスと

パス (省略可能) に名前を付けます。

== CMD\_HELP\_ARCHIVE ==

備考:

このコマンドはリポジトリデータベースからデータを抽出し、それをデータベース領域を

節約しつつ、外部ストレージに格納します。

また、コマンドは以前にアーカイブ済みのリビジョンをリポジトリデータベース

(--^restore) に復元します。

'cm ^help ^objectspec' を使用してリビジョン指定の指定方法を確認できます。

操作を完了するには、このコマンドを実行しているユーザーが Plastic SCM サーバー管理者

(リポジトリサーバー所有者) である必要があります。

指定されたリビジョンからのすべてのデータセグメントは、--^file

引数によって定義された値で始まる名前で別個のファイルに

格納されます。この引数には、将来のアーカイブファイル用のプレフィックスが含まれる

フルパス値か、このプレフィックス値のみが含まれる場合があります。

一度アーカイブされると、指定されたリビジョンからのデータは次の 2 つの方法で

アクセスできます。

- クライアントから: クライアントはそのデータがアーカイブ済みであるかどうかを検出し、

ユーザーにファイルの場所を入力するよう求めます。

ユーザーは、アーカイブ済みのデータが置かれている場所のパスが含まれる

externaldata.conf という名前のファイルを (標準の設定ファイルの場所に、

client.conf に適用されるのと同じルールを使用して) 作成することで、

外部データの場所を設定できます。

- サーバーから: この方法では、リクエストはサーバーによって透過的に解決されるため、

ユーザーはデータがアーカイブ済みであるかどうかを知っている必要が

ありません。このようにするには、管理者がサーバーディレクトリに

externaldata.conf という名前のファイルを作成し、そこに

アーカイブ済みのボリュームがあるパスを入力します。

リビジョン (またはリビジョンのセット) のアーカイブを解除 (復元) するには、そのアーカイブ済みの

ファイルにクライアントからアクセスできる必要があります。このため、サーバー (方法 2) によって

解決されているデータのアーカイブを解除することはできません。クライアントが

それがアーカイブ済みであることを識別できないためです。方法 2 が使用された場合、

アーカイブを正常に解除するには、先に管理者が

externaldata.conf サーバーファイルを編集し、アーカイブを解除する必要があるアーカイブ済みの

ファイルへのアクセスを削除する必要があります。

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。

例:

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

(ブランチ 'main' にある 'bigfile.zip' の最新のリビジョンをアーカイブします。)

cm ^archive ^rev:myfile.pdf#^cs:2 -^c="大きな PDF ファイル" --^file=c:\arch\_files\arch

(myfile.pdf の変更セット 2 でのリビジョンを 'c:\archived\_files' フォルダーに

アーカイブします。アーカイブ済みのファイルの名前は 'arch' で始まる必要があります (例: arch\_11\_56)。)

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

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

(名前が 'volume00' で始まる、サイズが 25 Mb 以上のファイルをすべて

アーカイブします。)

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

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

(サイズが 25 Mb 以上のアーカイブ済みのファイルをすべて復元します。)

== CMD\_DESCRIPTION\_ATTRIBUTE ==

ユーザーが属性を管理できるようにします。

== CMD\_USAGE\_ATTRIBUTE ==

使用方法:

cm ^attribute | ^att <コマンド> [オプション]

コマンド:

^create | ^mk

^delete | ^rm

^set

^unset

^rename

^edit

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^attribute <コマンド> --^usage

cm ^attribute <コマンド> --^help

== CMD\_HELP\_ATTRIBUTE ==

例:

cm ^attribute ^create ステータス

cm ^attribute ^set ^att:status ^br:/main/SCM105 未処理

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

cm ^attribute ^delete ^att:status

cm ^attribute ^rename ^att:status "ビルドステータス"

cm ^attribute ^edit ^att:status "CI パイプライン内のタスクのステータス"

== CMD\_DESCRIPTION\_CHANGELIST ==

変更リスト内のグループの保留中の変更。

== CMD\_USAGE\_CHANGELIST ==

使用方法:

a) 変更リストのオブジェクトの管理:

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

(すべての変更リストを表示します。)

cm ^changelist | ^clist ^add <変更リスト名>

[<変更リスト説明>] [--^persistent | --^notpersistent] [--^symlink]

(変更リストを作成します。)

cm ^changelist | ^clist ^rm <変更リスト名> [--^symlink]

(選択された変更リストを削除します。この変更リストに保留中の変更が含まれている

場合、それらは ^default 変更リストに移動されます。)

cm ^changelist | ^clist ^edit <変更リスト名> [<アクション名> <アクション値>]

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

(選択された変更リストを編集します。)

b) 指定の変更リストのコンテンツの管理:

cm ^changelist | ^clist <変更リスト名> (^add | ^rm) <パス名>[ ...]

[--^symlink]

(指定されたパス名と一致する変更を追加 ('^add') または削除 ('^rm') することで、

選択された変更リストを追加または削除します。空白を使用して

パス名を区切ります。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。パスのステータスは、'^Added' または '^Checked-out' である必要があります。)

オプション:

変更リスト名 変更リストの名前。

変更リスト説明 変更リストの説明。

アクション名 変更リストを編集するのに '^rename' か '^description' の間から

選択します。

アクション値 変更リストを編集中に、新しい名前または新しい説明を

適用します。

--^persistent そのコンテンツがチェックインまたは元に戻されている場合でも、

変更リストはワークスペース内に残ります。

--^notpersistent (デフォルト) そのコンテンツがチェックインまたは元に戻されている

場合でも、変更リストはワークスペース内に

残りません。

--^symlink 操作をターゲットではなくシンボリックリンクに

適用します。

== CMD\_HELP\_CHANGELIST ==

備考:

'^changelist' コマンドは、ワークスペースの保留中の変更と、変更リストに

含まれている変更の両方を処理します。

例:

cm ^changelist

(現在のワークスペースの変更リストを表示します。)

cm ^changelist ^add 設定変更 "dotConf ファイル" --^persistent

(「設定変更」という名前の新しい変更リストと「dotConf

ファイル」という名前の説明を作成します。その保留中の変更リストがチェックインまたは元に戻されると、

現在のワークスペースに永続的に残ります。)

cm ^changelist ^edit 設定変更 ^rename 設定ファイル --^notpersistent

(「設定変更」という名前の変更リストを編集し、その名前を「設定ファイル」に

変更します。また、その変更リストを「非永続的」になるように変更します。)

cm ^changelist ^edit 設定変更 --^notpersistent

(「設定変更」という名前の変更リストを編集し、それを「非永続的」になるように変更します。)

cm ^changelist ^rm 設定ファイル

(保留中の変更リスト「設定ファイル」を現在のワークスペースから削除します。)

cm ^changelist 設定ファイル ^add foo.conf

(ファイル「foo.conf」を「設定ファイル」変更リストに追加します。)

cm ^changelist 設定ファイル ^rm foo.conf readme.txt

(ファイル「foo.conf」と「readme.txt」を「設定ファイル」

変更リストから削除し、それらのファイルをシステムのデフォルトの変更リストに移動します。)

== CMD\_DESCRIPTION\_CHANGESET ==

変更セットに対して高度な操作を実行します。

== CMD\_USAGE\_CHANGESET ==

使用方法:

cm ^changeset <コマンド> [オプション]

コマンド:

^move | ^mv

^delete | ^rm

^editcomment | ^edit

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^changeset <コマンド> --^usage

cm ^changeset <コマンド> --^help

== CMD\_HELP\_CHANGESET ==

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 ==

変更セットのコメントを変更します。

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

使用方法:

cm ^changeset ^editcomment | ^edit <変更セット指定> <新しいコメント>

オプション:

変更セット指定 コメントが編集されるターゲット変更セット。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

新しいコメント ターゲットの変更セットに追加される新しい

コメント。

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

備考:

- ターゲットの変更セット指定が有効である必要があります。

例:

cm ^changeset ^editcomment ^cs:15@myrepo "チェックインの詳細を追加するのを忘れていました"

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

"このコメントテキストが前のコメントに置き換わります。"

== CMD\_DESCRIPTION\_CHANGESET\_MOVE ==

変更セットとそのすべての子孫を別のブランチに移動します。

== CMD\_USAGE\_CHANGESET\_MOVE ==

使用方法:

cm ^changeset ^move | ^mv <変更セット指定> <ブランチ指定>

オプション:

変更セット指定 別のブランチに移動される最初の変更セット。同じ

ブランチ内のすべての子孫の変更セットも同様に

このコマンドのターゲットになります。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

ブランチ指定 ターゲットの変更セットが格納されるターゲット

ブランチ。空であるか存在していない必要があります。

同期先ブランチが存在しない場合は、そのコマンドによって

作成されます。

('cm ^help ^objectspec' を使用してブランチ指定の詳細を

確認できます。)

== CMD\_HELP\_CHANGESET\_MOVE ==

備考:

- ターゲットの変更セット指定が有効である必要があります。

- 同期先ブランチは空であるか存在していない必要があります。

- 同期先ブランチが存在しない場合は、作成されます。

- マージリンクはブランチの影響を受けないため、変更されません。

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 ==

変更セットをリポジトリから削除します。

== CMD\_USAGE\_CHANGESET\_DELETE ==

使用方法:

cm ^changeset ^delete | ^rm <変更セット指定>

オプション:

変更セット指定 削除されるターゲット変更セット。特定の条件の一部を

満たしている必要があります。詳細については、「備考」を参照してください。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

== CMD\_HELP\_CHANGESET\_DELETE ==

備考:

- ターゲット変更セットは、そのブランチ内で最後である必要があります。

- ターゲット変更セットは、その他の変更セットの親であってはなりません。

- ターゲット変更セットは、マージリンクのソースであっても、ソースとしての

間隔マージの一部であってもなりません。

- ターゲット変更セットに、ラベルが適用されていてはなりません。

- ターゲット変更セットは、ルート変更セット ('^cs:0') であってはなりません。

例:

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

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

== CMD\_DESCRIPTION\_CHANGEUSERPASSWORD ==

ユーザーパスワード (UP) を変更します。

== CMD\_USAGE\_CHANGEUSERPASSWORD ==

使用方法:

cm ^changepassword | ^passwd

== CMD\_HELP\_CHANGEUSERPASSWORD ==

備考:

このコマンドは、セキュリティ設定が UP (ユーザー/パスワード) のときにのみ

使用できます。詳細については、管理ガイドを確認してください。

新旧のパスワードが必要です。

例:

cm ^passwd

== CMD\_DESCRIPTION\_CHECKCONNECTION ==

サーバーへの接続を確認します。

== CMD\_USAGE\_CHECKCONNECTION ==

使用方法:

cm ^checkconnection | ^cc

== CMD\_HELP\_CHECKCONNECTION ==

備考:

- このコマンドは、設定された Plastic SCM サーバーへの有効な接続が

あるかどうかを示すメッセージを返します。

- コマンドは、設定されたユーザーが有効であるかどうかを確認します。また、

サーバーのバージョンの互換性を確認します。

== CMD\_DESCRIPTION\_CHECKDB ==

リポジトリの整合性を確認します。

== CMD\_USAGE\_CHECKDB ==

使用方法:

cm ^checkdatabase | ^chkdb [<リポジトリサーバー指定> | <リポジトリ指定>]

'cm ^help ^objectspec' を使用してリポジトリサーバー指定とリポジトリ指定の詳細を確認できます。

== CMD\_HELP\_CHECKDB ==

備考:

- リポジトリサーバー指定もリポジトリ指定も指定されていない場合、

client.conf ファイルで指定されたサーバーでチェックが行われます。

例:

cm ^checkdatabase ^repserver:localhost:8084

cm ^chkdb ^rep:default@localhost:8084

== CMD\_DESCRIPTION\_CHECKIN ==

変更をリポジトリに格納します。

== CMD\_USAGE\_CHECKIN ==

使用方法:

cm ^checkin | ^ci [<項目パス>[ ...]]

[-^c=<コメント文字列> | -^commentsfile=<コメントファイル>]

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

[--^noshowchangeset]

[--^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

オプション:

項目パス チェックインされる項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用して

項目パスを区切ります。

チェックインを現在のディレクトリに適用するには、. を使用します。

-^c 指定されたコメントをチェックイン操作で作成された

変更セットに適用します。

-^commentsfile 指定されたファイル内のコメントをチェックイン操作で

作成された変更セットに適用します。

--^all | -^a 指定されたパスでローカルに変更、移動、および削除された

項目も含めます。

--^applychanged チェックアウト済みの項目とともにワークスペースで

検出された変更済み項目にチェックイン操作を

適用します。

--^private ワークスペースで検出された非公開の項目も

含まれます。

--^update 最終的に発生する場合は、更新/マージを自動的に

処理します。

--^symlink チェックイン操作をターゲットではなくシンボリックリンクに

適用します。

--^noshowchangeset 結果の変更セットを出力しません。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator '--^machinereadable' フラグとともに使用され、その行をどのように開始する

必要があるかを指定します。

--^endlineseparator '--^machinereadable' フラグとともに使用され、その行をどのように終了する

必要があるかを指定します。

--^fieldseparator '--^machinereadable' フラグとともに使用され、そのフィールドをどのように区切る

必要があるかを指定します。

== CMD\_HELP\_CHECKIN ==

備考:

- <項目パス>が指定されていない場合、チェックインにはそのワークスペース内の

すべての保留中の変更が関与します。

- チェックイン操作は常に指定されたパスから再帰的に適用されます。

- 項目をチェックインするには:

- 項目がソースコード管理の対象になっている必要があります。

- 項目が非公開の (ソースコード管理の対象でない) 場合、チェックイン

するには --^private フラグが必須です。

- 項目がチェックアウトされている必要があります。

- 項目が変更されているもののチェックアウトされていない場合、<項目パス> が

ディレクトリであるか、ワイルドカード ('\*') が含まれている場合を除き、--^applychanged フラグは

不要です。

チェックインするにはリビジョンコンテンツが前のリビジョンと異なっている必要が

あります。

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。

stdin から入力を読み取る:

'^checkin' コマンドは stdin からパスを読み取ることができます。これを行うには、シングル

ダッシュ「-」を渡します。

例: cm ^checkin -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用してチェックインするファイルを指定できます。

例:

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

(Windows で、すべての .c ファイルをワークスペースにチェックインします。)

例:

cm ^checkin file1.txt file2.txt

('file1.txt' と 'file2.txt' のチェックアウト済みのファイルをチェックインします。)

cm ^checkin .-^commentsfile=mycomment.txt

(現在のディレクトリをチェックインし、'mycomment.txt' ファイルに

コメントを設定します。)

cm ^checkin リンク --^symlink

(ターゲットではなく「リンク」ファイルをチェックインします。UNIX 環境で

有効です。)

cm ^ci file1.txt -^c="my comment"

('file1.txt' をチェックインし、コメントを含めます。)

cm ^status --^short --^compact --^changelist=レビュー待ち | cm ^checkin -

(「レビュー待ち」という名前の変更リスト内のパスをリストし、このリストを

チェックインコマンドの入力にリダイレクトします。)

cm ^ci .--^machinereadable

(現在のディレクトリをチェックインし、その結果を簡略化された、

より解析しやすい形式で出力します。)

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

(現在のディレクトリをチェックインし、その結果を簡略化された、

より解析しやすい形式で出力します。指定された文字列で

行を開始および終了し、フィールドを区切ります。)

== CMD\_DESCRIPTION\_CHECKOUT ==

ファイルを変更準備完了としてマークします。

== CMD\_USAGE\_CHECKOUT ==

使用方法:

cm ^checkout | ^co [<項目パス>[ ...]] [-^R | -^r | --^recursive]

[--^format=<文字列形式>]

[--^errorformat=<文字列形式>] [--^resultformat=<文字列形式>]

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

[--^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

オプション:

項目パス チェックアウトされる項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用して

項目パスを区切ります。

チェックアウトを現在のディレクトリに適用するには、. を使用します。

-^R ファイルを再帰的にチェックアウトします。

--^format 出力の進捗メッセージを特定の形式で

取得します。詳細については、例を確認してください。

--^errorformat エラーメッセージ (ある場合) を特定の形式で

取得します。詳細については、例を確認してください。

--^resultformat 出力の結果メッセージを特定の形式で

取得します。詳細については、例を確認してください。

--^silent 出力を一切表示しません。

--^symlink チェックアウト操作をターゲットではなくシンボリックリンクに

適用します。

--^ignorefailed 項目をロックできない (排他的チェックアウトを実行

できない) 場合、チェックアウト操作はロックなしで

続行されます。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator '--^machinereadable' フラグとともに使用され、その行をどのように開始する

必要があるかを指定します。

--^endlineseparator '--^machinereadable' フラグとともに使用され、その行をどのように終了する

必要があるかを指定します。

--^fieldseparator '--^machinereadable' フラグとともに使用され、そのフィールドをどのように区切る

必要があるかを指定します。

== CMD\_HELP\_CHECKOUT ==

備考:

項目をチェックアウトするには:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックインされている必要があります。

サーバーでロックが設定されている (lock.conf が存在する) 場合、パスで

チェックアウトが行われるたびに、Plastic はそれがいずれかのルールに従っているかどうかを確認し、

従っている場合、パスは排他的チェックアウト (ロック) されているため、同時に

チェックアウトできるものはありません。

「cm ^lock ^list」を使用してサーバー内のすべてのロックを取得できます。

詳細については、管理者ガイドを確認してください。

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

形式文字列はプレースホルダー '{0}' をチェックアウトされる項目のパスに

置き換えます。使用方法については、例を確認してください。

stdin から入力を読み取る:

'^checkout' コマンドは stdin からパスを読み取ることができます。これを行うには、シングル

ダッシュ「-」を渡します。

例: cm ^checkout -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用してチェックアウトするファイルを指定できます。

例:

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

(Windows で、すべての .c ファイルをワークスペースにチェックアウトします。)

例:

cm ^checkout file1.txt file2.txt

('file1.txt' と 'file2.txt' のファイルをチェックアウトします。)

cm ^co \*.txt

(すべての txt ファイルをチェックアウトします。)

cm ^checkout .

(現在のディレクトリをチェックアウトします。)

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

('src' フォルダーを再帰的にチェックアウトします。)

cm ^co file.txt --^format="項目 {0} をチェックアウトしています"

--^errorformat="項目 {0} のチェックアウト中にエラーが発生しました" /

--^resultformat="項目 {0} をチェックアウトしました"

(指定された形式化文字列を使用して 'file.txt' をチェックアウトし、

進捗、結果、操作のエラーを表示します。)

cm ^checkout リンク --^symlink

(ターゲットにではなく 'リンク' ファイルをチェックアウトします。UNIX 環境で

有効です。)

cm ^checkout .-^R --^ignorefailed

(現在のフォルダーをチェックアウトします。チェックアウトできないファイルは

無視されます。)

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

(現在のディレクトリをチェックアウトし、その結果を簡略化された、

より解析しやすい形式で出力します。指定された文字列で行を開始します。)

== CMD\_DESCRIPTION\_CHECKSELECTORSYNTAX ==

セレクターの構文をチェックします。

== CMD\_USAGE\_CHECKSELECTORSYNTAX ==

使用方法:

cm ^checkselectorsyntax | ^css --^file=<セレクターファイル>

(セレクターファイルの構文をチェックします。)

^cat <セレクターファイル> | cm ^checkselectorsyntax | ^css -

(Unix。標準の入力からセレクターファイルをチェックします。)

^type <セレクターファイル> | cm ^checkselectorsyntax | ^css -

(Windows。標準の入力からセレクターファイルをチェックします。)

--^file セレクターの読み取り元のファイル。

== CMD\_HELP\_CHECKSELECTORSYNTAX ==

備考:

このコマンドはファイルまたは標準の入力のいずれかのセレクターを読み取り、

有効な構文であることをチェックします。構文チェックに合格しなかった場合、その理由が

標準出力に出力されます。

例:

cm ^checkselectorsyntax --^file=myselector.txt

('myselector.txt' ファイルの構文をチェックします。)

^cat myselector.txt | cm ^checkselectorsyntax

(標準の入力から 'myselector.txt' の構文をチェックします。)

== CMD\_DESCRIPTION\_CHGREVTYPE ==

項目のリビジョンタイプ (バイナリまたはテキスト) を変更します。

== CMD\_USAGE\_CHGREVTYPE ==

使用方法:

cm ^changerevisiontype | ^chgrevtype | ^crt <項目パス>[ ...]--^type=(^bin | ^txt)

項目パス リビジョンタイプを変更する項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用して

項目パスを区切ります。

--^type ターゲットリビジョンタイプ。'^bin' または '^txt' を選択します。

== CMD\_HELP\_CHGREVTYPE ==

備考:

このコマンドはディレクトリではなく、ファイルにのみ適用できます。

指定されたタイプはシステムでサポート対象の「^bin」または「^txt」(バイナリ

またはテキスト) である必要があります。

例:

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

(「file.txt」リビジョンタイプをテキストに変更します。)

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

(「comp.zip」と「image file.jpg」リビジョンタイプをバイナリに変更します。)

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

(すべてのファイルのリビジョンタイプをテキストに変更します。)

== CMD\_DESCRIPTION\_TRIGGER\_EDIT ==

トリガーを編集します。

== CMD\_USAGE\_TRIGGER\_EDIT ==

使用方法:

cm ^trigger | ^tr ^edit <サブタイプのタイプ> <位置番号>

[--^position=<新しい位置>]

[--^name=<新しい名前>] [--^script=<スクリプトパス>]

[--^filter=<文字列フィルター>] [--^server=<リポジトリサーバー指定>]

サブタイプのタイプ トリガー実行とトリガー操作。

トリガータイプのリストを表示するには「cm ^showtriggertypes」

と入力します。

位置番号 変更されるトリガーが占める位置。

オプション:

--^position 指定されたトリガーの新しい位置。

この位置は、同じタイプの別のトリガーによって使用中でない

必要があります。

--^name 指定されたトリガーの新しい名前。

--^script 指定されたトリガースクリプトの新しい実行パス。

スクリプトが「^webtrigger」で始まる場合、それは

ウェブトリガーと見なされます。詳細については、「備考」を

参照してください。

--^filter 指定されたフィルターに一致する項目のみをチェックします。

--^server 指定されたサーバーのトリガーを編集します。

サーバーが指定されていない場合は、クライアントに設定されている

サーバーでコマンドを実行します。

(「cm ^help ^objectspec」を使用してサーバー指定の

詳細を確認できます。)

== CMD\_HELP\_TRIGGER\_EDIT ==

備考:

ウェブトリガー: ウェブトリガーは、「^webtrigger <ターゲット URI>」をトリガーコマンド

として入力することで作成します。この場合、トリガーは指定された URI に対して

POST クエリを実行します。リクエスト本文には、JSON ディクショナリと

トリガー環境変数、および文字列の配列を指す

固定の入力キーが含まれます。

例:

cm ^trigger ^edit ^after-setselector 6 --^name="Backup2 マネージャー" --^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 ==

コードレビューを作成、編集、削除します。

== CMD\_USAGE\_CODEREVIEW ==

使用方法:

cm ^codereview <指定> <タイトル> [--^status=<ステータス名>]

[--^assignee=<ユーザー名>] [--^format=<文字列形式>]

[--^repository=<リポジトリ指定>]

(コードレビューを作成します。)

cm ^codereview -^e <ID> [--^status=<ステータス名>] [--^assignee=<ユーザー名>]

[--^repository=<リポジトリ指定>]

(コードレビューを編集します。)

cm ^codereview -^d <ID> [ ...][--^repository=<リポジトリ指定>]

(1 つ以上のコードレビューを削除します。)

指定 変更セット指定またはブランチ指定のいずれかにできます。

それが新しいコードレビューのターゲットになります。(

「cm ^help ^objectspec」を使用して変更セット指定またはブランチ指定の

詳細を確認できます。)

タイトル 新しいコードレビューのタイトルとして使用される

テキスト文字列。

ID コードレビューの識別番号。GUID を使用することも

できます。

オプション:

-^e 既存のコードレビューのパラメーターを編集します。

-^d 1 つ以上の既存のコードレビューを削除します。空白を

使用してコードレビューの ID を区切ります。

--^status コードレビューの新しいステータスを設定します。詳細については、「備考」を

参照してください。

--^assignee コードレビューの新しい担当者を設定します。

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^repository デフォルトとして使用されるリポジトリを設定します。(

「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

== CMD\_HELP\_CODEREVIEW ==

備考:

このコマンドにより、ユーザーはコードレビューを管理できます。変更セットまたはブランチの

コードレビューを作成、編集、削除します。

新しいコードレビューを作成するには、変更セット指定/ブランチ指定とタイトルが

必須です。初期ステータスと担当者も設定できます。ID (または

リクエストされた場合は GUID) が結果として返されます。

既存のコードレビューを編集または削除するには、ターゲットのコードレビューの ID

(または GUID) が必要です。エラーがない場合、メッセージは表示されません。

ステータスパラメーターは、「^Under review」

(デフォルト)、「^Reviewed」、または「^Rework required」のいずれかのみになります。

リポジトリパラメーターでは、デフォルトの作業リポジトリを

設定できます。これは、現在のワークスペースに関連付けられているサーバーとは

別のサーバーのレビューを管理するときや、現在のワークスペースが

まったくないときに便利です。

出力形式のカスタマイズ:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} ID

{1} GUID

「--^format」パラメーターは、新しいコードレビューを作成しているときにのみ有効であることに

注意してください。

例:

cm ^codereview ^cs:1856@myrepo@myserver:8084 "My code review" --^assignee=dummy

cm ^codereview ^br:/main/task001@myrepo@myserver:8084 "My code review" \

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

cm ^codereview 1367 -^e --^assignee=新しい担当者

cm ^codereview -^e 27658884-5dcc-49b7-b0ef-a5760ae740a3 --^status=レビュー済み

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

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

== CMD\_DESCRIPTION\_CRYPT ==

パスワードを暗号化します。

== CMD\_USAGE\_CRYPT ==

使用方法:

cm ^crypt <自分のパスワード>

自分のパスワード 暗号化されるパスワード。

== CMD\_HELP\_CRYPT ==

備考:

このコマンドは、引数として渡された指定のパスワードを暗号化します。

これは、設定ファイル内のパスワードを暗号化し、安全性を高めるように

設計されています。

例:

cm ^crypt dbconfpassword -> ENCRYPTED: encrypteddbconfpassword

(データベース設定ファイル「db.conf」内のパスワードを暗号化します。)

== CMD\_DESCRIPTION\_DEACTIVATEUSER ==

ライセンスが付与されたユーザーのアクティベートを解除します。

== CMD\_USAGE\_DEACTIVATEUSER ==

使用方法:

cm ^deactivateuser | ^du <ユーザー名>[ ...][--^server=<名前:ポート>]

[--^nosolveuser]

ユーザー名 アクティベートを解除するユーザーの名前。空白を使用して

ユーザー名を区切ります。

SID の場合は、「--^nosolveuser」が必須です。

オプション:

--^server 指定されたサーバー上のユーザーのアクティベートを解除します。

サーバーが指定されていない場合は、クライアントに設定されている

サーバーでコマンドを実行します。

--^nosolveuser このオプションを使用すると、コマンドはそのユーザー名が認証システム上に

存在するかどうかをチェックしません。その

<ユーザー名> はユーザー SID である必要があります。

== CMD\_HELP\_DEACTIVATEUSER ==

備考:

このコマンドはユーザーを非アクティブに設定し、そのユーザーが Plastic SCM を

使用できなくなります。

Plastic SCM ユーザーのアクティベート解除の詳細については、「cm ^activateuser」コマンドを

参照してください。

このコマンドは、そのユーザーが基盤の認証システム (例: ActiveDirectory、LDAP、ユーザー/パスワード...) 上に

存在するかどうかをチェックします。

認証システム上に存在しなくなったユーザーのアクティベート解除を

適用するには、「--^nosolveuser」オプションを使用できます。

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 ==

ファイル、変更セット、ラベル間の差分を表示します。

== CMD\_USAGE\_DIFF ==

使用方法:

cm ^diff <変更セット指定> | <ラベル指定> | <シェルブ指定> [<変更セット指定> | <ラベル指定> | <シェルブ指定>]

[<パス>]

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

[--^repositorypaths] [--^download=<ダウンロードのパス>]

[--^encoding=<名前>]

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

[--^clean]

[--^format=<文字列形式>] [--^dateformat=<文字列形式>]

「ソース」変更セットまたはシェルブセットと、「同期先」変更セット

またはシェルブセットの間の差分を表示します。変更セットは、変更セット指定

またはラベル指定のいずれかを使用して指定できます。

2 つの指定が指定される場所では、最初の指定が差分の「ソース」になり、

2 つ目の指定が「同期先」になります。

1 つの指定のみが指定された場合、その「ソース」が指定された「同期先」の

親変更セットになります。

オプションのパスが指定された場合、差分ウィンドウが起動し、

そのファイルの 2 つのリビジョン間の差分が表示されます。

cm ^diff <リビジョン指定 1> <リビジョン指定 2>

リビジョンのペア間の差分を表示します。その差分は

差分ウィンドウに表示されます。指定された最初のリビジョンが

左に表示されます。

cm ^diff <ブランチ指定> [--^added] [--^changed] [--^moved] [--^deleted]

[--^repositorypaths] [--^download=<ダウンロードのパス>]

[--^encoding=<名前>]

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

[--^clean]

[--^format=<文字列形式>] [--^dateformat=<文字列形式>]

[--^fullpaths | --^fp]

ブランチの差分を表示します。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

オプション:

--^added そのリポジトリに追加された項目で構成される差分のみを

出力します。

--^changed 変更された項目で構成される差分のみを

出力します。

--^moved 移動または名前が変更された項目で構成される差分のみを

出力します。

--^deleted 削除された項目で構成される差分のみを

出力します。

「--^added」、「--^changed」、「--^moved」、「--^deleted」のいずれも

指定されなかった場合、そのコマンドはすべての差分を出力します。

「^A」は追加された項目を意味します。

「^C」は変更された項目を意味します。

「^D」は削除された項目を意味します。

「^M」は移動された項目を意味します。左がオリジナルの項目で、

右が同期先の項目です。

--^repositorypaths ワークスペースパスの代わりにリポジトリのパスを出力します。

(このオプションは「--^fullpaths」オプションをオーバーライドします。)

--^download 指定された出力パスに差分コンテンツを

格納します。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^ignore 指定された比較方法を設定します。

詳細については、「備考」を参照してください。

--^clean マージによって生成された差分は考慮せず、

単純なチェックインによって作成された差分のみを

考慮します。

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^dateformat 日付の出力に使用される形式。

--^fullpaths、--^fp 可能であればファイルやディレクトリのフルワークスペースパスを

出力することが強制されます。

== CMD\_HELP\_DIFF ==

備考:

比較方法:

^eol 行の終わりの差分を無視します。

^whitespaces 空白の差分を無視します。

^"eol&whitespaces" 行の終わりと空白の差分を無視します。

^none 行の終わりと空白の差分を検出します。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドのパラメーターは次のとおりです。

{^path} 項目パス。

{^date} 日付/時間を変更します。

{^owner} 作成者を変更します。

{^revid} 差分で同期先として見なされるリビジョンの

リビジョン ID。

{^parentrevid} 差分で同期先として見なされるリビジョンの親の

リビジョン ID。

{^baserevid} 差分でソースとして見なされるリビジョンの

リビジョン ID。

{^srccmpath} 項目を移動する前のサーバーパス (移動操作)。

{^dstcmpath} 項目を移動した後のサーバーパス (移動操作)。

{^type} 項目のタイプ:

^D ディレクトリ、

^B バイナリファイル、

^F テキストファイル、

^S シンボリックリンク、

^X Xlink。

{^repository} その項目のリポジトリ。

{^status} 項目のステータス:

^A 追加済み、

^D 削除済み、

^M 移動済み、

^C 変更済み。

{^fsprotection} 項目の権限を表示します (Linux/Mac chmod)。

{^srcfsprotection} 親リビジョン項目の権限を表示します。

{^newline} 改行を挿入します。

「^revid」に関するメモ:

追加済みの項目については、「^baserevid」と「^parentrevid」は -1 になります。このケースでは

前のリビジョンが存在しないためです。

削除済みの項目については、「^revid」はソースリビジョンの ID になり、

「^baserevid」は同期先リビジョンがないため -1 になります。

Xlink については、「^baserevid」と「^parentrevid」は両方とも常に -1 になります。

例:

次はブランチを比較しています。

cm ^diff ^br:/main/task001

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

次は変更セットツリーを比較しています。

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

次はラベルツリーを比較しています。

cm ^diff ^lb:FirstReleaseLabel ^lb:SecondReleaseLabel

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

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

次はシェルブツリーを比較しています。

cm ^diff ^sh:2

cm ^diff ^sh:2 ^sh:4

次はリビジョン指定を比較しています。

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 ==

2 つのリビジョン間の差分の指標を表示します。

== CMD\_USAGE\_DIFFMETRICS ==

使用方法:

cm ^diffmetrics | ^dm <リビジョン指定 1> <リビジョン指定 2> [--^format=<文字列形式>]

[--^encoding=<名前>]

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

リビジョン指定 比較に使用したリビジョン。

(「cm ^help ^objectspec」を使用してリビジョン指定の詳細を確認できます。)

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^ignore 指定された比較方法を設定します。

詳細については、「備考」を参照してください。

== CMD\_HELP\_DIFFMETRICS ==

備考:

指標は、変更、追加、削除された行数です。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} 変更された行数。

{1} 追加された行数。

{2} 削除された行数。

例:

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

--^format="変更された行数は {0}、追加された行数は {1}、削除された行数は {2} です。"

(形式化された差分の指標の結果を受け取ります。)

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

== CMD\_DESCRIPTION\_FASTEXPORT ==

リポジトリを高速エクスポート形式でエクスポートします。

== CMD\_USAGE\_FASTEXPORT ==

使用方法:

cm ^fast-export | ^fe <リポジトリ指定> <高速エクスポートファイル>

[--^import-marks=<マークファイル>]

[--^export-marks=<マークファイル>]

[--^branchseparator=<文字セパレーター>]

[--^nodata] [--^from=<変更セット ID>] [--^to=<変更セット ID>]

オプション:

リポジトリ指定 データのエクスポート元のリポジトリ。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を確認できます。)

高速エクスポートファイル Git 高速エクスポート形式のリポジトリデータがある

ファイル。

--^import-marks インクリメンタルインポートに使用されるマークファイル。このファイルは

以前に「--^export-marks」によってエクスポートされています。この

ファイルに記述されている変更セットは、すでに前のインポートに

入っていたためインポートされません。

--^export-marks インポートされた変更セットが保存されるファイル。

このファイルは後の高速インポートで、すでにインポート済みの

変更セットを知らせるために使用されます。

--^branchseparator Plastic SCM はブランチ階層のデフォルトのセパレーターとして

「/」を使用します。このオプションにより、文字を階層のセパレーターとして

使用できるため、main-task-sub は Plastic

SCM に /main/task/sub としてマップされます。

--^nodata データを含まないリポジトリをエクスポートします。これは

エクスポートが正しく実行されるかどうかを確認するのに役立ちます。

--^from 特定の変更セットからエクスポートします。

--^to 特定の変更セットにエクスポートします。

== CMD\_HELP\_FASTEXPORT ==

備考:

- Plastic SCM リポジトリを Git にインポートするには、次のようなコマンドを使用します。

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

- インクリメンタルエクスポートは、以前にインポートされた変更セット

(「--^import-marks」ファイルと「--^export-marks」ファイル) が含まれるマークファイルを使用することで

サポートされます。

これは、前の高速エクスポートでエクスポートされなかった新しい変更セットのみが

エクスポートされることを意味します。

例:

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

--^export-marks=marks.cm

(ローカルサーバーにあるリポジトリ「repo」を「repo.fe.00」ファイルに Git 高速エクスポート形式で

エクスポートし、後でインクリメンタルエクスポートを実行するために

マークファイルを作成します。)

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

(ローカルサーバーにあるリポジトリ「repo」を「repo.fe.00」ファイルに

Git 高速エクスポート形式で変更セット「20」からエクスポートします。)

== CMD\_DESCRIPTION\_FASTIMPORT ==

Git 高速エクスポートデータをリポジトリにインポートします。

== CMD\_USAGE\_FASTIMPORT ==

使用方法:

cm ^fast-import | ^fi <リポジトリ指定> <高速エクスポートファイル>

[--^import-marks=<マークファイル>]

[--^export-marks=<マークファイル>]

[--^stats] [--^branchseparator=<文字セパレーター>]

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

オプション:

リポジトリ指定 データのインポート先となるリポジトリ。

前もって存在していない場合は

作成されます。(「'cm ^help ^objectspec'」を使用して

リポジトリ指定の詳細を確認できます。)

fast-export-file GIT 高速エクスポート形式のリポジトリデータ

があるファイル。

--^import-marks インクリメンタルインポートに使用されるマークファイル。

このファイルは以前に「--^export-marks'」によって

エクスポートされています。このファイルに記述されている

変更セットは、すでに前のインポートに

入っていたためインポートされません。

--^export-marks インポートされた変更セットが保存されるファイル。

このファイルは後の高速インポートで、

すでにインポート済みの変更セットを

知らせるために使用されます。

--^stats インポートプロセスに関するいくつかの統計を出力します。

--^branchseparator Plastic SCM はブランチ階層のデフォルトの

セパレーターとして「/」を使用します。このオプションにより、

文字を階層のセパレーターとして使用できるため、main-task-sub

は Plastic SCM に /main/task/sub としてマップされます。

--^nodata データを含めずに Git 高速エクスポートを

インポートします。これはインポートが正しく実行されるかどうかを

確認するのに役立ちます。

--^ignoremissingchangesets インポートできない変更セットは破棄され、

高速インポート操作はそれらの変更セット

なしで続行されます。

--^mastertomain 「^master」ではなく「^main」を使用してインポートします。

== CMD\_HELP\_FASTIMPORT ==

備考:

- git リポジトリをエクスポートするには、次のようなコマンドを使用します。

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

-^M オプションは移動された項目を検出するのに重要です。

- 指定されたリポジトリが存在しなかった場合は作成されます。

- インクリメンタルインポートは、以前にインポートされた変更セット

(「--^import-marks」ファイルと「--^export-marks」ファイル) が含まれるマークファイルを使用することで

サポートされます。

これは、前の高速インポートでインポートされなかった新しい変更セットのみが

インポートされることを意味します。

例:

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

(「repo.fast-export」ファイルにエクスポートされたコンテンツを

サーバー「atenea:8084」上の「mynewrepo」リポジトリにインポートします。)

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

(「repo.fast-export」ファイルにエクスポートされたコンテンツを、

サーバー「atenea:8084」上の「repo」リポジトリにインポートし、

後でインクリメンタルインポートを実行するためにマークファイルを作成します。)

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

--^export-marks=repo.marks

(「repo.fast-export」ファイルのコンテンツをインポートします。マークファイルになかった

新しい変更セットのみがインポートされます。次回の

インクリメンタルインポートで変更セットのリストを再度保存するのに

同じマークファイルが使用されます。

== CMD\_DESCRIPTION\_FILEINFO ==

ワークスペース内の項目に関する詳細情報を取得します。

== CMD\_USAGE\_FILEINFO ==

使用方法:

cm ^fileinfo <項目パス>[ ...][--^fields=<フィールド値>[,...]]

[[--^xml | -^x [=<出力ファイル>]] | [--^format=<文字列形式>]]

[--^symlink] [--^encoding=<名前>]

項目パス 表示する項目。空白を使用して項目を

区切ります。

空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

オプション:

--^fields コンマ区切りの値の文字列。これは、各項目で

どのフィールドを出力するかを選択します。詳細については、「備考」を

参照してください。

--^xml | -^x 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。このオプションを

「--^format」と組み合わせることはできません。

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。このオプションを「--^xml」と

組み合わせることはできません。

両方が指定された場合、この「--^format」オプションは「--^fields」よりも

優先されます。

--^symlink ファイル情報操作をターゲットではなくシンボリックリンクに

適用します。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

== CMD\_HELP\_FILEINFO ==

備考:

このコマンドは、選択された各項目の詳細な属性のリストを出力します。

各属性はデフォルトで新しい行に出力されます。

属性のリストは、ユーザーが必要とする属性のみを表示するよう

変更できます。これは、「--^fields=<フィールドリスト>」を使用することで実現できます。ここでは

コンマ区切りの属性名の文字列を受け取ります。この方法により、それらの属性のうち

名前が指定されている属性のみが表示されます。

リビジョンヘッドの変更セット:

このオプションはデフォルトでは無効になっています。この属性を取得するのは

他の残りの属性を取得するよりも大幅に時間がかかるため、ユーザーにはできるだけ多くの項目を

グループ化することをお勧めしています。これにより、

数多くの「cm ^fileinfo」を別々に実行するのを回避することで、実行時間が改善されます。

また、この機能は現在、管理対象ディレクトリでは利用できません。

以下で利用できる属性名の全一覧を確認できます。

アスタリスク (「\*」) 付きの名前はデフォルトでは表示されません。

^ClientPath その項目のディスク上のローカルパス。

^RelativePath ワークスペースに対する相対パス。

^ServerPath その項目のリポジトリパス。

(注: トランスフォームされているワークスペースは

このオプションでは現在サポートされていません。)

^Size 項目のサイズ。

^Hash 項目のハッシュの合計。

^Owner その項目が属するユーザー。

^RevisionHeadChangeset (\*) ブランチのヘッド変更セットにロードされた

リビジョンの変更セット。

(上の注記を参照。)

^RevisionChangeset ワークスペースに現在ロードされているリビジョンの

変更セット。

^RepSpec その項目のリポジトリ指定。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

^Status ワークスペース項目のステータス (追加済み、チェックアウト済み、

削除済みなど)。

^Type リビジョンタイプ (テキスト、バイナリ、ディレクトリ、シンボリックリンク、

または不明)。

^Changelist その項目が属する変更リスト (ある場合)。

^IsLocked (\*) その項目が排他的チェックアウトによって

ロックされているかどうか。

^LockedBy (\*) その項目を排他的チェックアウトしたユーザー。

^LockedWhere (\*) その項目が排他的チェックアウトされた

場所。

^IsUnderXlink その項目が Xlink の下に

あるかどうか。

^UnderXlinkTarget その項目が下にある Xlink のターゲット

(ある場合)。

^UnderXlinkPath Xlink でリンクされたリポジトリ内の項目サーバーパス

(ある場合)。

^UnderXlinkWritable その項目が属する Xlink が

書き込み可能であるかどうか。

^UnderXlinkRelative その項目が属する Xlink が

相対的であるかどうか。

^IsXlink その項目自体が Xlink であるかどうか。

^XlinkTarget その項目が示すターゲットリポジトリ (その項目が

Xlink である場合)。

^XlinkName その項目の Xlink 名 (その項目が

実際に Xlink である場合)。

^XlinkWritable その Xlink 項目が書き込み可能な Xlink で

あるかどうか。

^XlinkRelative その Xlink 項目が相対的な Xlink で

あるかどうか。

出力形式のカスタマイズ:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{^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}

「--^format」オプションと「--^xml」オプションは相互に排他的であるため、

同時には使用できないことに注意してください。

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 ==

SQL のようなクエリを実行して Plastic SCM のオブジェクトを探します。

== CMD\_USAGE\_FIND\_QUERY ==

使用方法:

cm ^find <オブジェクトタイプ>

[^where <文字列条件>]

[^on ^repository '<リポジトリ指定>' | ^on ^repositories '<リポジトリ指定 1>','<リポジトリ指定 2>'[,...]]

[--^format=<文字列形式>] [--^dateformat=<日付形式>]

[--^nototal] [--^file=<ダンプファイル>] [--^xml]

[--^encoding=<名前>]

オブジェクトタイプ 検索するオブジェクトタイプ。

「cm ^help ^showfindobjects」を使用してこれらのオブジェクトの指定方法を

確認できます。

次の「cm ^find」ガイドでも確認できます。

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

オプション:

文字列条件 オブジェクトの属性の条件を検索します。

リポジトリ指定 リポジトリのエイリアスまたは指定を検索します。

「^on ^repositories」の場合は、コンマを使用して

リポジトリ指定のフィールドを区切ります。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

--^format 出力メッセージを特定の形式で取得します。

出力形式文字列として使用できるオブジェクトの

すべての属性は、次の「cm ^find」ガイドで確認できます。

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

--^dateformat 日付の出力に使用される形式。

--^nototal 最後に記録数を出力しません。

--^file 結果をダンプするファイル。

--^xml 出力を XML 形式で標準出力に出力します。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

== CMD\_HELP\_FIND\_QUERY ==

備考:

リポジトリが指定されていない場合、検索はワークスペースに設定された

リポジトリで行われます。

コマンド行から比較演算子 (>、<、>=、<=) を使用してクエリを

実行するときは、シェルがこれらの演算子を IO リダイレクトと見なすことを

忘れないでください。そのため、クエリを二重引用符で囲む必要が

あります。

「cm ^find」コマンドは、出力を表示する形式の文字列を受け取ります。

各出力パラメーターは文字列によって識別され、ユーザーは「{」と「}」の間に

パラメーター番号を入力することでそれを参照できます。

通常、出力パラメーターはそのオブジェクトの属性に対応します。

これらは有効な出力形式文字列の一部です:

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

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

XML とエンコーディングに関する考慮事項:

「--^xml」オプションが指定されると、コマンドにコマンド結果が XML テキストとして

標準出力に表示されます。テキストの表示にオペレーションシステムのデフォルトの

エンコーディングが使用されるため、ANSI 以外の文字が間違って

コンソールに表示される可能性があります。コマンド出力をファイルにリダイレクトする場合は、

正しく表示されます。「--^xml」オプションと「--^file」オプションの両方が

指定されると、デフォルトのエンコーディングは utf-8 になります。

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 ==

変更されたファイルのリストを取得します。このコマンドは非推奨になっており、

後方互換性のためにのみ保持されています。代わりに「cm ^status」を使用してください。

== CMD\_USAGE\_FINDCHANGED ==

使用方法:

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

[--^onlychanged] [<パス>]

オプション:

-^R ディレクトリで再帰的に検出します。

--^checkcontent ファイルをコンテンツごとに比較します。

--^onlychanged 変更されたファイルのみを検出します。チェックアウトは

取得されません。

パス (デフォルト: 現在のディレクトリ。)

変更されたファイルを検出する初期パス。

== CMD\_HELP\_FINDCHANGED ==

備考:

「--^checkcontent」オプションが指定されていない場合、Plastic はファイルのタイムスタンプに基づいて

変更を検出します。「--^checkcontent」オプションが指定されている場合、タイムスタンプを

使用する代わりに、ファイルまたはフォルダーのコンテンツが比較されます。

このコマンドは、Plastic SCM サーバーから切断されている間に、変更されているファイルを

検出するのに便利です。出力をチェックアウトコマンドに送り、

後で変更をチェックできます (例を参照)。

例:

cm ^findchanged .

(現在のディレクトリで変更されたファイルを検出します。)

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

(変更された要素をチェックアウトします。)

== CMD\_DESCRIPTION\_FINDCHECKEDOUT ==

チェックアウトされた項目のリストを取得します。このコマンドは非推奨になっており、

後方互換性のためにのみ保持されています。代わりに「cm ^status」を使用してください。

== CMD\_USAGE\_FINDCHECKEDOUT ==

使用方法:

cm ^findcheckouts | ^fco [--^format=<文字列形式>] [--^basepath]

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^basepath チェックアウトの検索を開始するパス。指定

されていない場合は、現在のパスが使用されます。

== CMD\_HELP\_FINDCHECKEDOUT ==

備考:

このコマンドはチェックアウト済みのすべての項目を 1 回のステップでチェックインまたはチェックアウトを取り消すのに

便利で、標準出力を他のコマンドにリダイレクトします。

例を参照してください。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} 日付。

{1} 所有者。

{2} ワークスペース情報。

{3} クライアントマシン名。

{4} 項目パス。

{5} ブランチとリポジトリの情報。

例:

cm ^findcheckouts --^format="ファイル {4} がブランチ {5} で変更された"

(チェックアウト済みの項目を検出し、ファイルパスとブランチおよびリポジトリの情報を使用して

その出力を形式化します。)

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

(チェックアウト済みのすべての項目をチェックインします。)

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

(チェックアウト済みのすべての項目のチェックアウトを取り消します。)

== CMD\_DESCRIPTION\_FINDPRIVATE ==

非公開の項目のリストを取得します。このコマンドは非推奨になっており、

後方互換性のためにのみ保持されています。代わりに「cm ^status」を使用してください。

== CMD\_USAGE\_FINDPRIVATE ==

使用方法:

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

オプション:

-^R ディレクトリで再帰的に検出します。

--^exclusions このオプションは、ファイル ignore.conf によって定義された、無視されたパス内で

検索を打ち切ることを許可します。

パス (デフォルト: 現在のディレクトリ。)

非公開ファイルを検出する初期パス。

== CMD\_HELP\_FINDPRIVATE ==

備考:

任意のパスが指定されている場合、Plastic SCM は現在のディレクトリから

検索を開始します。

このコマンドはフォルダーに非公開の項目を追加するのに便利で、その出力を

追加コマンドに送ります。例を参照してください。

例:

cm ^findprivate .

cm ^findprivate -^R | cm ^add -

(非公開の項目を再帰的に検索し、それらを追加します。)

== CMD\_DESCRIPTION\_GETCONFIG ==

設定情報を取得します。

== CMD\_USAGE\_GETCONFIG ==

使用方法:

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

[^extensionprefix] [^defaultrepserver]

^setfileasreadonly 保護されたファイルが読み取り専用として

残っているかどうかを返します。

^location クライアントの設定パスを返します。

^extensionworkingmode 拡張子の動作モードを返します。

^extensionprefix 設定された拡張子のプレフィックスを返します。

^defaultrepserver デフォルトのリポジトリサーバーの場所を

返します。

== CMD\_HELP\_GETCONFIG ==

例:

cm ^getconfig ^setfileasreadonly

== CMD\_DESCRIPTION\_GETFILE ==

指定されたリビジョンのコンテンツをダウンロードします。

== CMD\_USAGE\_GETFILE ==

使用方法:

cm ^getfile | ^cat <リビジョン指定> [--^file=<出力ファイル>] [--^debug]

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

リビジョン指定 オブジェクト指定。(「cm ^help ^objectspec」を使用して指定の詳細を

確認できます。)

オプション:

--^file 出力を保存するファイル。デフォルトでは、標準出力に

出力されます。

--^debug ディレクトリ指定が使用されると、コマンドは

そのディレクトリのすべての項目、そのリビジョン ID、

ファイルシステム保護を表示します。

--^symlink 操作をターゲットではなくシンボリックリンクに

適用します。

--^raw ファイルの未加工データを表示します。

== CMD\_HELP\_GETFILE ==

例:

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

(「myfile.txt」のブランチ「^br:/main」の最新のリビジョンを取得します。)

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

(「myfile.txt」の変更セット 3 を取得し、それをファイル「tmp.txt」に書き込みます。)

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

(リポジトリ「myrepo」内のブランチ「/main/task003」の最新の変更セットにある

「/src/foo.c」のコンテンツを取得します。)

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

(ID が 1230 のリビジョンを取得します。)

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

(「info」ディレクトリ内のすべてのリビジョンを取得します。)

== CMD\_DESCRIPTION\_GETREVISION ==

ワークスペース内のリビジョンをロードします。

== CMD\_USAGE\_GETREVISION ==

このコマンドはワークスペース内にロードされたリビジョンを変更し、将来のマージに

影響を及ぼすことができるようにします。

これは古いバージョンから継承された高度なコマンドであるため、注意して使用してください。

使用方法:

cm ^getrevision <リビジョン指定>

リビジョン指定 オブジェクト指定。(「cm ^help ^objectspec」を使用して

リビジョン指定の詳細を確認できます。)

== CMD\_HELP\_GETREVISION ==

例:

cm ^getrevision file.txt#^cs:3

(「file.txt」の変更セット 3 のリビジョンを取得します。)

== CMD\_DESCRIPTION\_GETSTATUS ==

項目のステータスを取得します。

== CMD\_USAGE\_GETSTATUS ==

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

使い勝手については十分ではありません。

使用方法:

cm ^getstatus | ^gs <項目パス>[ ...][--^format=<文字列形式>] [--^stats]

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

項目パス ステータスの取得元の 1 つまたは複数の項目。空白が含まれるパスを

指定するには二重引用符 (" ") を使用します。空白を

使用してパスを区切ります。

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^stats ステータスを取得するプロセスに関するいくつかの統計を出力します。

-^R ディレクトリ内のステータスを再帰的に表示します。

== CMD\_HELP\_GETSTATUS ==

備考:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} 項目パス。

{1} 項目のステータス:

0 非公開、

1 チェックイン済み、

2 チェックアウト済み。

stdin から入力を読み取る:

「^getstatus」コマンドは stdin からパスを読み取ることができます。これを行うには、

シングルダッシュ「-」を渡します。

例: cm ^getstatus -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用してステータスを取得するパスを指定できます。

例:

dir /S /B \*.c | cm ^getstatus --^format="パス {0} ステータス {1}" -

(Windows で、ワークスペース内のすべての .c ファイルのステータスを取得します。)

例:

cm ^getstatus file1.txt file2.txt

(ファイルのステータスを取得します。)

cm ^gs info\ -^R --^format="項目 {0} のステータスは {1} です"

(ディレクトリのステータスとそのすべての項目を取得し、形式化された

出力を表示します。)

== CMD\_DESCRIPTION\_GETTASKBRANCHES ==

タスクにリンクされたブランチを取得します。

== CMD\_USAGE\_GETTASKBRANCHES ==

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

使い勝手については十分ではありません。

使用方法:

cm ^gettaskbranches | ^gtb <タスク名> [--^format=<文字列形式>]

[--^dateformat=<日付形式>]

タスク名 タスクの識別子。

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^dateformat 日付の出力に使用される形式。

== CMD\_HELP\_GETTASKBRANCHES ==

備考:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

{^name} ブランチ名。

{^owner} ブランチの所有者。

{^date} ブランチが作成された日付。

{^type} ブランチタイプ (スマートな場合は「T」、そうでない場合は「F」)。

{^parent} 親ブランチ。

{^comment} ブランチのコメント。

{^repname} ブランチが存在するリポジトリ。

{^repserver} サーバー名。

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 ==

ワークスペースセレクターに関する情報を表示します。

== CMD\_USAGE\_GETWWI ==

使用方法:

cm ^wi [<ワークスペースパス>]

オプション:

ワークスペースパス マシン上のワークスペースのパス。

== CMD\_HELP\_GETWWI ==

備考:

「^wi」コマンドはワークスペースの作業中の設定 (リポジトリ、

ブランチ、ラベル) を表示します。

例:

cm ^wi c:\mywk

== CMD\_DESCRIPTION\_GWP ==

パスからワークスペースの情報を取得します。

== CMD\_USAGE\_GWP ==

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

使い勝手については十分ではありません。

使用方法:

cm ^getworkspacefrompath | ^gwp <項目パス> [--^format=<文字列形式>]

項目パス ディスク上のファイルまたはフォルダー。

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

== CMD\_HELP\_GWP ==

備考:

このコマンドは、パス内にあるワークスペースに関する情報を表示します。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} | {^wkname} ワークスペース名。

{1} | {^wkpath} ワークスペースパス。

{2} | {^machine} クライアントマシン名。

{3} | {^owner} ワークスペースの所有者。

{4} | {^guid} ワークスペースの GUID。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

例:

cm ^gwp c:\myworkspace\code\file1.cpp --^format="ワークスペース名: {^wkname}"

== CMD\_DESCRIPTION\_HELP ==

Plastic SCM コマンドのヘルプを取得します。

== CMD\_USAGE\_HELP ==

使用方法:

cm ^help <コマンド>

== CMD\_HELP\_HELP ==

== CMD\_DESCRIPTION\_IOSTATS ==

ハードウェアに関する統計を表示します。

== CMD\_USAGE\_IOSTATS ==

使用方法:

cm ^iostats [<リポジトリサーバー指定>] [<テストのリスト>[ ...]]

[--^nettotalmb=<MB の値>] [--^networkiterations=<反復の値>]

[--^diskdatasize=<サイズの値>] [--^disktestpath=<パスの値>]

[--^systemdisplaytime=<時間の値>]

[--^systemdisplaytimeinterval=<間隔の値>]

オプション:

リポジトリサーバー指定 ネットワークテスト (「serverUploadTest」や

「serverDownloadTest」) を実行できる

Plastic SCM サーバー。

サーバーが指定されていない場合、コマンドは

デフォルトで設定されたサーバーとの通信を

試行します。

(「cm ^help ^objectspec」を使用してサーバー指定の詳細を

確認できます。)

テストのリスト 有効なテスト。空白を使用してテストフィールドを

区切ります。

詳細については、「備考」を参照してください。

--^nettotalmb 「^serverDownloadTest」や「^serverUploadTest」などの

ネットワークテスト時に転送されるユーザーデータ

(メガバイト単位) を

示します。

「4」と「512」の間の値である必要があります。

(デフォルト: 16)

--^networkiterations 実行される「^serverDownloadTest」や

「^serverUploadTest」の反復回数を

示します。

「1」と「100」の間の値である必要があります。

(デフォルト: 1)

--^diskdatasize 「^diskTest」に書き込まれ、その後読み取られる

データの量 (メガバイト単位) を

示します。

「100」と「4096」の間の値である必要があります。

(デフォルト: 512)

--^disktestpath 「^diskTest」がテストファイルを書き込む

パス。このパラメーターが指定されない場合、

コマンドはシステムの一時パスの使用を

試行します。

--^systemdisplaytime システムリソースの使用状況を示している

時間間隔 (秒単位)。このオプションは、

「^systemNetworkUsage」と「^systemDiskUsage」のテストで

利用できます。

「1」と「3600」の間の値である必要があります。

(デフォルト: 5 秒)。

--^systemdisplaytimeinterval システムパフォーマンスのサンプル間の

時間間隔 (秒単位)。このオプションは、

「^systemNetworkUsage」と

「^systemDiskUsage」のテストで利用できます。

「1」と「60」の間の値である必要があります。

(デフォルト: 1 秒)。

== CMD\_HELP\_IOSTATS ==

備考:

このコマンドでは、ネットワークの速度テスト (「^serverUploadTest」や

「^serverDownloadTest」) 中に利用できるサーバーを使用することが求められます。

「--^diskTestPath」は、テストしようとしている物理ディスクドライブに属するパスを

指定する必要があります。パスが指定されない場合、コマンドは

システムのデフォルトの一時パスの使用を試行します。

指定されたパスのディスクドライブには、テストを実行するのに十分な空き容量が

必要です。

コマンドの実行中、システムは実行されたテストが原因でパフォーマンスの低下を

経験する場合があります。

有効なテストは次のとおりです。

--^serveruploadtest (デフォルト) Plastic SCM クライアントからサーバーへの

データのアップロード速度を計測します。

--^serverdownloadtest (デフォルト) Plastic SCM サーバーからクライアントへの

データのダウンロード速度を計測します。

--^disktest (デフォルト) ディスクの読み込み速度と書き込み速度を

計測します。

--^systemnetworkusage システムのネットワークリソースの現在の使用状況を

表示します。

(Microsoft Window によって提供されるネットワークインターフェースの

パフォーマンスカウンターを表示します。)

Microsoft Windows でのみ利用できます。

--^systemdiskusage システムの物理ディスクの現在の使用状況を

表示します。

(Microsoft Window によって提供されるネットワークインターフェースの

パフォーマンスカウンターを表示します。)

Microsoft Windows でのみ利用できます。

例:

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

== CMD\_DESCRIPTION\_ISSUETRACKER ==

指定されたイシュートラッカー内のイシューのステータスを取得、更新、検出します。

== CMD\_USAGE\_ISSUETRACKER ==

使用方法:

cm ^issuetracker <名前> ^status ^get <タスク ID> <パラメーター>[ ...]

cm ^issuetracker <名前> ^status ^update <タスク ID> <ステータス> <パラメーター>[ ...]

cm ^issuetracker <名前> ^status ^find <ステータス> <パラメーター>[ ...]

cm ^issuetracker <名前> ^connection ^check <パラメーター>[ ...]

名前 接続するイシュートラッカーの名前。

現時点では Jira のみがサポートされています。

タスク ID クエリを実行または更新するイシューの数。

^status イシュートラッカー内のイシューの有効なステータス。

Jira のパラメーター (すべて必須):

--^user=<ユーザー> 認証するユーザー。

--^password=<パスワード> 認証するパスワード。

--^host=<URL> イシュートラッカーのターゲット URL。

--^projectkey=<キー> Jira プロジェクトのプロジェクトキー。

== CMD\_HELP\_ISSUETRACKER ==

例:

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

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

(「PRJ」プロジェクトのイシュー 11 のステータスを取得します。)

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

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

(「PRJ」プロジェクトのイシュー 11 のステータスを「完了」に更新します。)

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

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

(「PRJ」プロジェクトのステータスが「完了」に設定されているタスクの ID を取得します。)

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

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

(設定パラメーターが有効であるかどうかをチェックします。)

== CMD\_DESCRIPTION\_LICENSEINFO ==

ライセンス情報とライセンスの使用状況を表示します。

== CMD\_USAGE\_LICENSEINFO ==

使用方法:

cm ^licenseinfo | ^li [--^server=<リポジトリサーバー指定>] [--^inactive] [--^active]

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

オプション:

--^server 指定されたサーバーからライセンス情報を取得します。

サーバーが指定されていない場合は、クライアントに設定されている

サーバーでコマンドを実行します。

(「cm ^help ^objectspec」を使用してリポジトリサーバー指定の

詳細を確認できます。)

--^inactive 「ライセンスの使用状況」セクションにアクティブでないユーザーのみを表示します。

--^active 「ライセンスの使用状況」セクションにアクティブなユーザーのみを表示します。

--^sort 「^name」や「^status」などの指定されたいずれかの

ソートオプションでユーザーをソートします。

== CMD\_HELP\_LICENSEINFO ==

備考:

表示される情報は、有効期限日、アクティベートおよびアクティベート解除された

ユーザーなどで構成されます。

例:

cm ^licenseinfo

cm ^licenseinfo --^server=myserver:8084

cm ^licenseinfo --^sort=^name

== CMD\_DESCRIPTION\_LINKTASK ==

変更セットをタスクにリンクします。

== CMD\_USAGE\_LINKTASK ==

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

使い勝手については十分ではありません。

使用方法:

cm ^linktask | ^lt <変更セット指定> <拡張子プレフィックス> <タスク名>

変更セット指定 タスクにリンクする完全な変更セット指定。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

拡張子プレフィックス 連携するよう設定されたイシュートラッキングシステムの

拡張子のプレフィックス。

タスク名 イシュートラッキングシステム上のタスクの識別子。

== CMD\_HELP\_LINKTASK ==

例:

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

== CMD\_DESCRIPTION\_LOCK\_LIST ==

サーバー上のロックを表示します。

== CMD\_USAGE\_LOCK\_LIST ==

使用方法:

cm ^lock ^list | ^ls [<リビジョン指定> [ ...]] [--^server=<サーバー>]

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

[--^ignorecase]

リビジョン指定 1 つ以上存在する場合で、それに関連付けられている項目が

サーバー内でロックされていれば、このコマンドは指定

されたリビジョンごとに 1 行のロック行を表示します。それ以外の場合、

このコマンドはデフォルトサーバー (または「--^server」オプションで設定された

サーバー) 内のロックされたすべての項目を一覧表示します。

リビジョン指定を複数使用するときは、空白を使用して

区切ります。

(「cm ^help ^objectspec」を使用してリビジョン指定の詳細を確認できます。)

オプション:

--^server リポジトリサーバー指定。

このオプションは、現在のワークスペースまたは

client.conf ファイルから取得されたデフォルトサーバーを

オーバーライドします。

(「cm ^help ^objectspec」を使用してサーバー指定の詳細を

確認できます。)

--^onlycurrentuser 現在のユーザーによって実行されたロックのみが表示されるように

結果をフィルター処理します。

--^onlycurrentworkspace 現在のワークスペースで実行されたロックのみが表示されるように結果を

フィルター処理します (名前で照合)。

--^ignorecase サーバーパス指定が使用されたときのパスの大文字と小文字の差異を

無視します。このフラグを使用すると、コマンドはユーザーが「/sRc/fOO.c」と

記述した場合でも「/src/foo.c」として機能します。

== CMD\_HELP\_LOCK\_LIST ==

備考:

コマンドは、デフォルトサーバー内で現在ロックされている項目のリストを

表示します。また、リビジョン指定のリストも受け取ります。その場合、

選択された項目に属するロックのみが表示されます。

「--^server=<サーバー>」を使用してクエリを実行するデフォルトサーバーを設定できます。

コマンドは、指定されたサーバー内のすべてのロックの行を表示します:

- ロックされた項目の GUID。

- ロックを実行したユーザー名。

- ロックが実行されたワークスペース名。

- ロックされた項目のパス (サーバーパス形式)。

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 ==

ユーザーとグループをリストします。

== CMD\_USAGE\_LISTUSERS ==

使用方法:

cm ^listusers | ^lu <リポジトリサーバー指定> [--^onlyusers] [--^onlygroups]

[--^filter= <文字列フィルター>]

リポジトリサーバー指定 リポジトリサーバーの指定。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

オプション:

--^onlyusers ユーザーのみをリストします。

--^onlygroups グループのみをリストします。

--^filter 指定されたフィルターに一致するユーザーとグループ、または

いずれか一方のみをリストします。

== CMD\_HELP\_LISTUSERS ==

例:

cm ^lu localhost:8084

(サーバー内のすべてのユーザーをリストします。)

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

(サーバー内の「m」が含まれるユーザーのみをリストします。)

== CMD\_DESCRIPTION\_LOCATION ==

「cm」のパスを返します。

== CMD\_USAGE\_LOCATION ==

使用方法:

cm ^location

== CMD\_HELP\_LOCATION ==

== CMD\_DESCRIPTION\_LOCK ==

このコマンドはユーザーにロックの管理を許可します。

== CMD\_USAGE\_LOCK ==

使用方法:

cm ^lock <コマンド> [オプション]

コマンド:

^list | ^ls

^unlock

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^lock <コマンド> --^usage

cm ^lock <コマンド> --^help

== CMD\_HELP\_LOCK ==

例:

cm ^lock ^list

cm ^lock

(引数がない場合、「^list」は省略可能です。)

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

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

== CMD\_DESCRIPTION\_LOG ==

変更セット内のリビジョンに関する情報を取得します。

== CMD\_USAGE\_LOG ==

使用方法:

cm ^log [<変更セット指定> | <リポジトリ指定>] [--^from=<変更セット指定元>] [--^allbranches]

[--^ancestors] [--^csformat=<文字列形式>] [--^itemformat=<文字列形式>]

[--^xml[=<出力ファイル>]] [--^encoding=<名前>]

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

オプション:

変更セット指定 コマンドは指定が指定された変更セットで行われた

すべての変更を返します。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

リポジトリ指定 コマンドは指定されたリポジトリで行われたすべての変更の

リストを表示します。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

--^from 変更セット指定 [変更セット指定元] から変更セット指定 [変更セット指定] に対して

すべての変更セットで行われたすべての変更の

リストを表示します。

[変更セット指定元] 変更セットは出力に

含まれません。

リポジトリ指定が指定されると無視されます。

--^allbranches 変更セットが作成されたすべてのブランチにつき、

指定された間隔で作成されたそれらの変更セットに関する

情報を表示します。

--^ancestors 指定された変更セット ([変更セット指定]) の親リンクと

マージリンクに従って、リーチ可能な変更セットに関する

情報を表示します。元の変更セット

([変更セット指定元]) も指定されている場合、それが

すべてのパスの下限として使用されます。備考: このオプションが

使用されると、変更セットの変更は表示されません。

--^csformat 変更セットの情報を特定の形式で取得します。詳細については、

「備考」を参照してください。

--^itemformat 項目の情報を特定の形式で取得します。詳細については、

「備考」を参照してください。

--^xml 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。

--^encoding 「--^xml」オプションとともに使用され、XML 出力で使用する

エンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^fullpaths、--^fp 可能であればファイルやディレクトリのフルワークスペースパスを

出力することが強制されます。

--^repositorypaths ワークスペースパスの代わりにリポジトリのパス

(サーバーパス) を出力します。(このオプションは「--^fullpaths」オプションを

オーバーライドします)。

== CMD\_HELP\_LOG ==

備考:

- 「変更セット指定」もオプションも指定されていない場合、コマンドは

直近の 1 か月に各ブランチで作成されたすべての変更セットに関する情報を

表示します。

- オプション「--^from」のみが含まれている場合、コマンドは

その指定された変更セットから最新の変更セットまで、その変更セットが作成された

ブランチ内の、すべての変更セットに関する情報を表示します。

- オプション「--^allbranches」が間隔なしで表示される場合、コマンドは

「変更セット指定」のみが指定された場合に取得したであろう同じ情報を

取得します。

- 「--^from」が使用された場合、出力には「変更セット指定元」+1 以降の

情報が含まれます。

- 変更セットの情報を表示するために使用されるリポジトリは、そのコマンドが

実行されるパスにロードされたリポジトリです。

このコマンドは、項目 (「--^itemformat」) の形式文字列と、変更セット

(「--^csformat」) の形式文字列を受け取ります。

「--^csformat」の出力パラメーターは次のとおりです。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

{^changesetid} 変更セット番号。

{^branch} 変更セットが作成されたブランチ。

{^date} 変更セットの日付。

{^owner} 変更セットの所有者。

{^comment} 変更セットのコメント。

{^items} 変更セットに関連する項目。

{^repository} 変更セットが存在するリポジトリ。

{^repserver} サーバー名。

「--^itemformat」の出力パラメーターは次のとおりです。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

{^path} 項目パス。

{^branch} 変更セットが作成されたブランチ。

{^date} 変更セットの日付。

{^owner} 変更セットの所有者。

{^shortstatus} 短い形式を出力します。以下を参照。

{^fullstatus} 長い形式を出力します。以下を参照。

短い形式とそれに対応する長い形式は次のとおりです。

'^A' ^Added

'^D' ^Deleted

'^M' ^Moved

'^C' ^Changed

以下が有効な出力文字列です。

--^csformat="{^newline}変更セット {^changesetid} が {^date} に作成されました。{^tab} 変更された項目: {^items}。"

--^itemformat="{^newline}項目 {^path} がブランチ {^branch} で変更されました。"

例:

cm ^log

(直近の 1 か月に各ブランチで作成されたすべての変更セットに関する情報を

表示します。)

cm ^log ^cs:16

(変更セットが作成されたブランチ内の、変更セット 16 で行われた

変更に関する情報を表示します。)

cm ^log ^cs:16 --^csformat="{^newline}変更セット {^changesetid} が \

{^date} に作成されました。{^tab} 変更された項目: {^items}。"

(情報を特定の形式で表示します。)

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

(変更セット 21 から変更セット 50 の各変更セットに含まれるすべての

リビジョンに関する情報を表示します。)

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

(リポジトリの各ブランチ内の、変更セット 21 から変更セット 50 の

各変更セットに含まれるすべてのリビジョンに関する情報を

表示します。)

cm ^log ^rep:myrep@localhost:8084

(指定されたリポジトリで行われた変更に関する情報を表示します。

コマンドを実行するのにワークスペースは不要です。)

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

(変更セット 21 から各変更セットに含まれるすべてのリビジョンに関する情報を

表示します。完全な変更セット指定が指定されたため、コマンドを実行するのに

ワークスペースは不要です。)

== CMD\_DESCRIPTION\_LS ==

ツリーのコンテンツをリストします。

== CMD\_USAGE\_LS ==

使用方法:

cm ^ls | ^dir [<パス>[ ...]] [--^format=<文字列形式>] [--^symlink]

[--^selector[=<セレクター形式>]] [--^tree=<オブジェクト指定>]

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

[--^xml[=<出力ファイル>]] [--^encoding=<名前>]

オプション:

パス 表示するパスのリスト。空白を使用してパスを

区切ります。

空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

--^format 出力メッセージを特定の形式で取得します。詳細については、

「備考」を参照してください。

--^symlink 操作をターゲットではなくシンボリックリンクに

適用します。

--^selector アクティブなワークスペースセレクターからコンテンツを取得します。

セレクター形式が指定された場合は、指定された

セレクターをリストします。

セレクターは 4.x 以降は Plastic SCM の中心部分では

なくなったため、これはほぼ非推奨です。

--^tree 指定された変更セットまたはブランチ内のツリーをリストします。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

-^R 再帰的にリストします。

--^xml 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。

--^encoding 「--^xml」オプションとともに使用され、XML 出力で使用する

エンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

== CMD\_HELP\_LS ==

備考:

- パスはメタ文字で入力できます。

- リストはワークスペースセレクターに依存します。

- コマンドの出力は形式文字列を指定することで形式化できます。

- 「--^tree」オプションまたは「--^selector」オプションが指定された場合、指定された

パスはワークスペースパス: C:\Users\myuser\mywk\dir\file.txt ではなく、

サーバーパス (別名: 「cm パス」): /dir/file.txt である必要があります

- パスが指定されていない場合、現在のディレクトリが

ワークスペースパスと見なされます。「--^tree」オプションまたは「--^selector」オプションが使用された場合は、

ルートパス (「/」) が想定されます。

デフォルトの形式文字列は次のとおりです。

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

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

{^symlinktarget}"

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{^size}

{^formattedsize}

{^date}

{^type}

^dir ディレクトリ、

^txt テキストファイル、

^File ファイル。

{^location} 例: ^br:branch#cset

{^checkout}

{^name}

{^changeset}

{^path}

{^repspec}

{^owner}

{^revid}

{^parentrevid}

{^itemid}

{^brid}

{^repid}

{^server}

{^symlinktarget}

{^hash}

{^chmod}

{^wkpath} ワークスペースのルートに対する相対パス

{^branch}

{^newlocation} cset@branch

{^guid} (解決により時間がかかる)

{^itemguid}

{^transformed} トランスフォームされた項目に適用された規則を表示する

「^ls」形式設定を PLASTIC\_LS\_FORMAT 環境変数に

カスタマイズできます。

例:

cm ^ls

cm ^ls c:\workspace\src

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

(ファイル名のみ。)

cm ^ls --^symlink

(「シンボリックリンク化された」ファイルまたはディレクトリの代わりにシンボリックリンクに関する

情報を表示します。UNIX 環境で有効です。)

cm ^ls code --^selector

(現在のワークスペースセレクターから「code」サブディレクトリのコンテンツを

表示します。)

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

(指定されたセレクター上の「/code」サブディレクトリのコンテンツを表示します。

パスはサーバー形式で指定されていることに注意してください。)

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

(サーバー「denver:7070」、リポジトリ「myrep」、変更セット 44 にある「/code」サブディレクトリを

リストします。)

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

(サーバー「denver:7070」、リポジトリ「myrep」、ブランチ

「/main/scm13596」内の、最新の変更セットにある「/code」サブディレクトリをリストします。)

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

(サーバー「denver:7070」、リポジトリ「myrep」、変更セット

ae1390ed-7ce9-4ec3-a155-e5a61de0dc77 にある「/code」サブディレクトリを

リストします。)

== CMD\_DESCRIPTION\_TRIGGER\_LIST ==

サーバー上の指定されたタイプのトリガーをリストします。

== CMD\_USAGE\_TRIGGER\_LIST ==

使用方法:

cm ^trigger | ^tr ^list | ^ls [<サブタイプのタイプ>] [--^server=<リポジトリサーバー指定>]

[--^format=<文字列形式>]

オプション:

サブタイプのタイプ トリガー実行とトリガー操作。

トリガータイプのリストを表示するには「cm ^showtriggertypes」

と入力します。

--^server 指定されたサーバー上のトリガーをリストします。

サーバーが指定されていない場合は、クライアントに設定されている

サーバーでコマンドを実行します。

(「cm ^help ^objectspec」を使用してサーバー指定の

詳細を確認できます。)

--^format 出力メッセージを特定の形式で取得します。詳細については、

「備考」を参照してください。

== CMD\_HELP\_TRIGGER\_LIST ==

備考:

タイプが指定されていない場合、サーバー上のすべてのトリガーをリストします。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} トリガーの位置。

{1} トリガーの名前。

{2} トリガーのパス。

{3} トリガーの所有者。

{4} トリガーのタイプ。

{5} トリガーのフィルター。

例:

cm ^trigger list after-mklabel

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

== CMD\_DESCRIPTION\_MANIPULATESELECTOR ==

セレクターを日付に変更します。

== CMD\_USAGE\_MANIPULATESELECTOR ==

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

使い勝手については十分ではありません。

使用方法:

cm ^manipulateselector | ^ms [<ワークスペースパス> | <ワークスペース指定>] --^atdate=<セレクター日付>

ワークスペースパス ワークスペースのパス。

ワークスペース指定 ワークスペースの指定。(「cm ^help ^objectspec」を使用して

指定の詳細を確認できます。)

オプション:

--^atdate 指定された日付の構成になるようにワークスペースを

再作成するセレクターを返します。

== CMD\_HELP\_MANIPULATESELECTOR ==

備考:

パスもワークスペースも指定されていない場合、このコマンドは現在のディレクトリを

ワークスペースパスとして使用します。

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 ==

ブランチを別のブランチとマージします。

== CMD\_USAGE\_MERGE ==

使用方法:

cm ^merge <ソース指定> [--^merge] [--^cherrypicking] [--^forced]

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

[--^interval-origin=<変更セット指定> | --^ancestor=<変更セット指定>]

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

[--^automaticresolution=<競合タイプ>[;...]]

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

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

[(--^to=<ブランチ指定> | --^destination=<ブランチ指定>)[--^shelve]]

[--^no-dst-changes]

[-^c=<コメント文字列> | --^commentsfile=<コメントファイル>]

[--^resolveconflict --^conflict=<インデックス>

--^resolutionoption=(^src|^dst|(^rename --^resolutioninfo=<文字列名>))

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

[--^nointeractiveresolution]

[--^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

ソース指定 マージ元のソースオブジェクトの指定:

- ブランチ指定: 「[^br:/]ブランチ名」

- ラベル指定: 「^lb:ラベル名」

- 変更セット指定: 「^cs:cs\_number」

- シェルブ指定: 「^sh:シェルブ番号」

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

オプション:

--^merge マージを実行します。それ以外の場合、見つかった

競合を出力します。

--^cherrypicking ソース変更セットに含まれている変更を

マージします。マージソース指定がラベルの場合、

このオプションは使用できません。

--^forced ソースと同期先がすでに接続されている場合は、

チェックされません。

このオプションは間隔マージとチェリーピックでのみ

使用できます。

--^mergetype 詳細については、「備考」を参照してください。

--^interval-origin ソース変更セットと指定した間隔の起点との間の

差分のみがマージされるようにするために、

どの変更セットを間隔の起点として選択するかを

指定します。

--^ancestor これは --^interval-origin のエイリアスです。

--^keepsource 競合がある項目についてソース共同作成者からの

すべての変更を受け取ります。

--^keepdestination 競合がある項目について同期先の共同作成者からの

すべての変更を保持します。

--^automaticresolution ディレクトリ競合を解決するために使用されます。このオプションにより、

ソースまたは同期先の共同作成者が

競合を解決するために自動的に

選択される必要があるかどうかを選択できます。

セミコロンを使用して競合タイプを区切ります。

詳細については、「備考」を参照してください。

--^subtractive マージによって導入された変更を削除します。変更を

削除するソースを指定するのに、

コマンドに渡されたパラメーター (ソース指定) が

使用されます。変更セットである必要があります。変更セットの

間隔の場合、間隔の起点を定義するのに

「--^interval-origin」を使用する必要があります。変更を

削除するために、システムは以前のリビジョンの

削除された変更以外のコンテンツがある、

チェックアウト済みの新しいリビジョンを作成します。

--^mount 指定されたリポジトリのマウントポイント。

--^printcontributors 共同作成者 (ベース、ソース、同期先) を

出力します。

--^noprintoperations その解決に関する情報を表示せずに、

サイレントにマージを解決します。

--^silent 出力を表示しません。

--^to | --^destination (ブランチ指定を入力することで)

競合が完全に解決された指定のブランチに

マージ先操作を実行します。

「マージ先」(またはワークスペースのないマージ) とは、

サーバー側で実行されるマージのことです。通常のマージは

ブランチ、ラベル、または変更セット「から」マージされる

ワークスペースで発生する一方で、マージ先は完全に

サーバー上で発生します。通常のマージでは

「同期先」はワークスペースである一方で、「マージ先」の場合、

同期先は常に指定される必要があります (これがマージ「先」と

呼ばれる理由です)。

「マージ先」機能の詳細については、次のリンクを

参照してください。

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

--^shelve 新しい変更セットを作成する代わりに、そのマージ

結果 (およびマージの追跡可能性に関する情報) が含まれる

シェルブを作成します。マージ元がシェルブのとき、

このオプションは使用できません。この

オプションは、サーバー側マージ (別名「マージ先」)

でのみ使用できます。そのため、「--^to」オプションと

「--^merge」オプションが必須です。

--^no-dst-changes 同期先の共同作成者に変更がないことを

確認します (同期先の変更セットは共通の

祖先でもあります)。同期先に変更があるとき、

マージは許可されません。

-^c 指定されたコメントをマージ操作で作成された

変更セットに適用します。

--^commentsfile 指定されたファイル内のコメントをマージ操作で

作成された変更セットに適用します。

--^resolveconflict (主にプラグインによって使用されます。詳細については、「備考」を参照してください。)

ディレクトリ競合を解決するために使用されます。

--^conflict 「--^resolveconflict」フラグとともに使用され、

解決する競合のインデックスを指定します (1 から開始)。

--^resolutionoption 「--^resolveconflict」フラグとともに使用され、

競合の解決のタイプを示します。「^src」、「^dst」、

「^rename」のいずれか 1 つのオプションを使用します。

詳細については、「備考」を参照してください。

--^resolutioninfo 「--^resolveconflict」フラグとともに使用され、

「--^resolutionoption」オプションが「rename」のときに使用する

名前を指定します。

--^mergeresultfile 「--^resolveconflict」フラグとともに使用され、

異なる呼び出し間のマージ結果の情報を

ファイルに出力します。指定されたパスは

最初の呼び出し中に作成され、次の各呼び出しで

更新されます。

--^solvedconflictsfile 「--^resolveconflict」フラグとともに使用され、

異なる呼び出し間で解決された競合に関する情報を

ファイルに出力します。指定された

パスは最初の呼び出し中に作成され、

次の各呼び出しで更新されます。

--^nointeractiveresolution (主にプラグインによって使用されます。詳細については、「備考」を参照してください。)

手動の競合についてユーザーにプロンプトを表示することを回避します。

この方法により、ディレクトリ競合が解決されなくなります。

--^machinereadable (主にプラグインによって使用されます。詳細については、「備考」を参照してください。)

結果を解析しやすい形式で出力します。

--^startlineseparator 「--^machinereadable」フラグとともに使用され、行をどのように

開始する必要があるかを指定します。(デフォルト: 空の文字列。)

--^endlineseparator 「--^machinereadable」フラグとともに使用され、行をどのように

終了する必要があるかを指定します。(デフォルト: 空の文字列。)

--^fieldseparator 「--^machinereadable」フラグとともに使用され、フィールドを

どのように区切る必要があるかを指定します。(デフォルト:

空白。)

== CMD\_HELP\_MERGE ==

備考:

このコマンドは、2 つのブランチ間またはラベルとブランチ間の変更をマージするために

使用されます。マージ先は常にブランチである必要があります。

マージ元は引数として指定されます。

同期先はワークスペースの現在のコンテンツです。

たとえば、ブランチ task001 からメインブランチにマージされる

要素を表示するには、セレクターがメインブランチに指定され、

ワークスペースが更新される必要があり、その後次のようにします。

cm ^merge ^br:/task001

マージを実際に実行するには、次のように「--^merge」オプションが追加されます。

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

マージ元を定義するには、次の指定を使用できます。

- ブランチの指定 (ブランチ指定):

[^br:/]ブランチ名

例: ^br:/main/task001

(上記はこのブランチ上の最後の変更セットからマージを実行します。)

- ラベルの指定 (ラベル指定):

^lb:ラベル名

例: ^lb:BL001

(ラベル付けされた変更セットからマージします。)

- 変更セットの指定 (変更セット指定):

^cs:cs\_number

例: ^cs:25

(指定された変更セットのコンテンツからマージします。)

- シェルブの指定 (シェルブ指定):

^sh:シェルブ番号

例: ^sh:2

(指定されたシェルブのコンテンツからマージします。)

ディレクトリの競合を自動的に解決するには、「--^automaticresolution」--^automaticresolution'

オプションを使用して競合のタイプを指定し、続けてマージ操作中に選択される必要がある

共同作成者 (ソースまたは同期先) を指定する必要があります。

(セミコロン (;) を使用して「競合のタイプ」と「共同作成者」のペアを区切ります。)

例:

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

(変更セット 2634 からのマージ操作で、両方のケースでソース (「-^src」) 共同作成者を

維持しながら、「^eviltwin」と「^changedelete」の競合を

解決します。)

- 競合タイプの後の「-^src」サフィックスは、マージコマンドにソース共同作成者の

変更を保持するよう指示します。

- 「-^dst」サフィックスは、同期先の共同作成者の変更を保持します。

マージコマンドがサポートする競合タイプのリストはこちらです:

「^movedeviltwin」、「^eviltwin」、「^changedelete」、「^deletechange」、「^movedelete」、

「^deletemove」、「^loadedtwice」、「^addmove」、「^moveadd」、「^divergentmove」、

「^cyclemove」、「^all」。

「^all」の値はその他のオプションをオーバーライドします。次の例では、

「^eviltwin-dst」が無視されます。

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

マージ競合の詳細については、次のリンクを参照してください。

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

これらが「--^mergetype」のオプションです。

^onlyone その項目を 1 人の共同作成者のみが変更した場合に自動的に

マージします。

^onlysrc その項目をソース共同作成者のみが変更した場合に自動的に

マージします。

^onlydst その項目を同期先の共同作成者のみが変更した場合に自動的に

マージします。

^try 競合を引き起こす部分のコード (各競合) を 1 人の共同作成者のみが変更した場合に

自動的にマージします。

^forced 常に自動的でないすべての競合を解決しようとします。

これらが主にプラグインや統合によって使用されるオプションです。

- 「--^resolveconflict」は、ディレクトリ競合を解決します。また、次のオプションも使用する

必要があります:

- 「--^conflict」は、解決する競合のインデックス

(1 から開始) です。

- 「--^resolutionoption」は、使用する競合の解決を

示します。これは次のいずれかになります。

- 「^src」は、ソースの変更を保持し、同期先の変更を

破棄します

- 「^dst」は、同期先の変更を保持し、ソースの変更を

破棄します

- 「^rename」(競合タイプがこの解決をサポートする

場合のみ) は、同期先を「--^resolutioninfo」オプションで指定された

指定の名前に変更します。

- 「--^resolutioninfo」は、「^rename」解決で使用する

名前を指定します

- 「--^mergeresultfile」と「--^solvedconflictsfile」は両方とも、異なる

呼び出し間のマージ情報を格納するために使用されます。

- 「--^nointeractiveresolution」は、ユーザーに手動での競合の解決を求めない

マージを示します。

- 「--^machinereadable」および「--^startlineseparator」、「--^endlineseparator」、

「--^fieldseparator」の各オプション。出力を機械が読み取り可能な方法

(より解析しやすい) で出力します。

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 同期先 ^br:/main/task --^merge

例:

cm ^merge ^br:/task001

(マージせず、単にマージされる項目を出力します。)

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

(ブランチ「task001」からマージします。)

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

(変更セットの間隔「2,5」からチェリーピックします。)

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

(競合があるそれらの要素の同期先の変更を維持しつつ、変更セット 8 から

減法マージします。)

cm ^merge ^br:/main/task001 --^to=^br:/main --^merge -^c="新しい UI 統合済み"

(ブランチ「task001」からブランチ「main」にサーバー側マージ (別名マージ先) を

実行し、コメントを設定します。)

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

(ブランチ「task001」からブランチ「main」にサーバー側マージ (別名マージ先) を実行し、

結果をシェルブに残します。)

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

(現在の「main」ヘッドから作成された場合にのみ、シェルブ 2 を「main」に

適用します。)

== CMD\_DESCRIPTION\_ATTRIBUTE\_CREATE ==

新しい属性を作成します。

== CMD\_USAGE\_ATTRIBUTE\_CREATE ==

使用方法:

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

属性名 属性の名前

== CMD\_HELP\_ATTRIBUTE\_CREATE ==

例:

cm ^attribute ^create ステータス

(属性「ステータス」を作成します。)

cm ^att ^mk 統合済み

(属性「統合済み」を作成します。)

== CMD\_DESCRIPTION\_BRANCH ==

ユーザーにブランチの管理を許可します。

== CMD\_USAGE\_BRANCH ==

使用方法:

cm ^branch | ^br <コマンド> [オプション]

コマンド:

^create | ^mk

^delete | ^rm

^rename

^history

^showmain

^showmerges

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^branch <コマンド> --^usage

cm ^branch <コマンド> --^help

== CMD\_HELP\_BRANCH ==

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 ==

新しいブランチを作成します。

== CMD\_USAGE\_BRANCH\_CREATE ==

使用方法:

cm ^branch | ^br [^create | ^mk] <ブランチ指定>

[--^changeset=<変更セット指定> | --^label=<ラベル指定>]

[-^c=<コメント文字列> | -^commentsfile=<コメントファイル>]

ブランチ指定 新しいブランチの名前または指定。

(「cm ^help ^objectspec」を使用してブランチ指定の詳細を確認できます。)

オプション:

--^changeset 新しいブランチの開始点として使用される変更セット。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を確認できます。)

--^label 新しいブランチの開始点として使用されるラベル。

(「cm ^help ^objectspec」を使用してラベル指定の詳細を確認できます。)

-^c 新しいブランチのコメントフィールドに指定されたテキストを

入力します。

-^commentsfile 新しいブランチのコメントフィールドに指定されたファイルの

コンテンツを入力します。

== CMD\_HELP\_BRANCH\_CREATE ==

備考:

最上位ブランチを作成するには、階層なしで名前を指定します。

例:

cm ^br /dev

省略可能なパラメーター「--^changeset」が指定されていない場合、新しいブランチのベースは

その親ブランチの最後の変更セットになります。新しいブランチが最上位

ブランチである場合、使用されるベース変更セットは変更セット 0 になります。

次のように「-^c」または「-^m」スイッチのいずれかを使用してコメントを指定できます。

cm ^branch /main/task001 -^c="これはコメントです"

cm ^branch /main/task001 -^m"これはコメントです"

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。PLASTICEDITOR 環境変数が設定されており、かつ

コメントが空の場合、そのエディターが自動的に起動し、

コメントを指定できるようになります。

例:

cm ^branch task001

cm ^branch ^create task001

cm ^branch ^mk task001

cm ^br ^mk task001

(最上位の「task001」ブランチを現在のワークスペースのリポジトリに

作成します。)

cm ^branch ^br:/task001/task002@

(「task002」ブランチを「task001」の子として作成します。)

cm ^br /main/task001@myrep@myserver:8084 -^c="my comment"

(「task001」ブランチをリポジトリ「myrep@myserver:8084」の

「main」の子としてコメント「my comment」付きで作成します。)

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

(「task001」ブランチを「main」の子としてベース「changeset=2837」で作成し、

コメントを「commenttask001.txt」ファイルに適用します。)

== CMD\_DESCRIPTION\_BRANCH\_DELETE ==

1 つ以上のブランチを削除します。

== CMD\_USAGE\_BRANCH\_DELETE ==

使用方法:

cm ^branch | ^br ^delete | ^rm <ブランチ指定>[ ...]

ブランチ指定 削除するブランチ。空白を使用してブランチを区切ります。

('cm ^help ^objectspec' を使用してブランチ指定の詳細を

確認できます。)

== CMD\_HELP\_BRANCH\_DELETE ==

備考:

このコマンドは、1 つ以上のブランチを削除します。

例:

cm ^branch ^delete /main/task001

(現在のワークスペースのリポジトリにある「main」の子の「task001」という

名前のブランチを削除します。)

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

(現在のワークスペースのリポジトリにあるブランチ「/main/task002」と、

リポジトリ「reptest@myserver:8084」にあるブランチ「/main/task012」を削除します。)

== CMD\_DESCRIPTION\_BRANCH\_RENAME ==

ブランチ名を変更します。

== CMD\_USAGE\_BRANCH\_RENAME ==

使用方法:

cm ^branch | ^br ^rename <ブランチ指定> <新しい名前>

ブランチ指定 名前を変更するブランチ。

(「cm ^help ^objectspec」を使用してブランチ指定の詳細を確認できます。)

新しい名前 ブランチの新しい名前。

== CMD\_HELP\_BRANCH\_RENAME ==

備考:

このコマンドは、ブランチ名を変更します。

例:

cm ^branch ^rename /main/task0 task1

(ブランチ名「/main/task0」を「/main/task1」に変更します。)

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

(リポジトリ「reptest」の「main」ブランチの名前を「secondary」に変更します。)

== CMD\_DESCRIPTION\_BRANCH\_HISTORY ==

ブランチの履歴を表示します。

== CMD\_USAGE\_BRANCH\_HISTORY ==

使用方法:

cm ^branch | ^br ^history <ブランチ指定> [--^dateformat=<日付形式>]

[--^machinereadable]

ブランチ指定 履歴を取得するブランチ指定。

(「cm ^help ^objectspec」を使用してブランチ指定の詳細を確認できます。)

オプション:

--^dateformat 日付の出力に使用される形式。

--^machinereadable 結果を解析しやすい形式で出力します。

== CMD\_HELP\_BRANCH\_HISTORY ==

例:

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

(「myserver」サーバー上の「myrepository」リポジトリの「/main/scm001」ブランチの

履歴を表示します。)

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

(現在のリポジトリの「main」ブランチの履歴を、指定された日付形式を使用した

解析しやすい形式で表示します。)

== CMD\_DESCRIPTION\_BRANCH\_SHOWMAIN ==

リポジトリのメインブランチを表示します。

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

ほとんどの場合、リポジトリのメインブランチは「/main」です。

== CMD\_USAGE\_BRANCH\_SHOWMAIN ==

使用方法:

cm ^branch | ^br ^showmain [<リポジトリ指定>] [--^encoding=<名前>]

[--^format=<形式文字列>] [--^dateformat=<日付形式>]

リポジトリ指定 メインブランチを表示するリポジトリ

指定。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を確認できます。)

オプション:

--^encoding 出力で使用するエンコーディング (utf-8 など) を

指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^format 出力メッセージを特定の形式で取得します。詳細については、

「備考」を参照してください。

--^dateformat 日付の出力に使用される形式。

== CMD\_HELP\_BRANCH\_SHOWMAIN ==

備考:

このコマンドはリポジトリのメインブランチを表示します。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{^id} ブランチ ID。

{^comment} コメント。

{^date} 日付。

{^name} 名前。

{^owner} 所有者。

{^parent} 親ブランチ名。

{^repository} リポジトリ。

{^repname} リポジトリ名。

{^repserver} サーバー。

{^changeset} ブランチのヘッド変更セット。

例:

cm ^branch ^showmain

(現在のワークスペースのリポジトリのメインブランチを表示します。)

cm ^branch ^showmain repo@server:8084

(サーバー「server:8084」内のリポジトリ「repo」のメインブランチを

表示します。)

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

(リポジトリのメインブランチを指定された日付形式で表示し、

それを utf8 で出力します。)

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

(リポジトリのメインブランチを表示し、その ID と名前のみを出力します。)

== CMD\_DESCRIPTION\_BRANCH\_SHOWMERGES ==

マージ待ちのブランチを表示します。

== CMD\_USAGE\_BRANCH\_SHOWMERGES ==

これは「cm」を自動化する目的のためだけに使用する自動化コマンドです。

使い勝手については十分ではありません。

使用方法:

cm ^branch | ^br ^showmerges <項目パス>[ ...]

[--^format=<形式文字列>]

[--^dateformat=<日付形式>]

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^dateformat 日付の出力に使用される形式。

== CMD\_HELP\_BRANCH\_SHOWMERGES ==

備考:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{^id} ブランチ ID。

{^comment} コメント。

{^date} 日付。

{^name} 名前。

{^owner} 所有者。

{^parent} 親ブランチ名。

{^parentid} 親ブランチ ID。

{^repid} リポジトリ ID。

{^repository} リポジトリ。

{^repname} リポジトリ名。

{^repserver} リポジトリサーバー。

例:

cm ^branch ^showmerges file.txt

(「file.txt」の保留中のマージに関連するブランチを表示します。)

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

(マージに関連するブランチを表示し、日付と名前のみを指定した日付形式で

出力します。)

== CMD\_DESCRIPTION\_REPOSITORY ==

ユーザーにリポジトリの管理を許可します。

== CMD\_USAGE\_REPOSITORY ==

使用方法:

cm ^repository | ^repo <コマンド> [オプション]

コマンド:

^create | ^mk

^delete | ^rm

^list | ^ls

^rename

^add

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^repository <コマンド> --^usage

cm ^repository <コマンド> --^help

== CMD\_HELP\_REPOSITORY ==

例:

cm ^repository

cm ^repository ^list

cm ^repository newrepo

cm ^repository ^create newrepo

cm ^repository ^rename oldname newname

cm ^repository ^add C:\repo\

== CMD\_DESCRIPTION\_REPOSITORY\_CREATE ==

サーバーにリポジトリを作成します。

== CMD\_USAGE\_REPOSITORY\_CREATE ==

使用方法:

cm ^repository | ^repo <リポジトリ名>

cm ^repository | ^repo <リポジトリサーバー指定> <リポジトリ名>[ ...]

cm ^repository | ^repo [^create | ^mk] <リポジトリ名>

リポジトリサーバー指定 リポジトリサーバーの指定。

(「cm ^help ^objectspec」を使用してリポジトリサーバー指定の

詳細を確認できます。)

リポジトリ名 1 つまたは複数の新しいリポジトリの名前。

空白を使用してリポジトリ名を区切ります。

== CMD\_HELP\_REPOSITORY\_CREATE ==

例:

cm ^repository MyRep

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

cm ^repo ^create Rep01

cm ^repo ^mk list

== CMD\_DESCRIPTION\_REPOSITORY\_DELETE ==

サーバーからリポジトリを削除します。

== CMD\_USAGE\_REPOSITORY\_DELETE ==

使用方法:

cm ^repository | ^repo ^delete | ^rm <リポジトリ指定>

オプション:

リポジトリ指定 リポジトリの指定。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を確認できます。)

== CMD\_HELP\_REPOSITORY\_DELETE ==

備考:

リポジトリサーバーからリポジトリを削除します。

データはデータベースバックエンドから削除されませんが、切断される

ため、アクセスできなくなります。

(データは後で再接続できます。「cm ^repository ^add」を参照してください。)

例:

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

cm ^repository ^rm myrepository@myserver:8084

cm ^repo ^rm myrepository

== CMD\_DESCRIPTION\_REPOSITORY\_LIST ==

サーバー上のリポジトリをリストします。

== CMD\_USAGE\_REPOSITORY\_LIST ==

使用方法:

cm ^repository | ^repo [^list | ^ls] [<リポジトリサーバー指定>] [--^format=<文字列形式>]

オプション:

リポジトリサーバー指定 リポジトリサーバーの指定。

(「cm ^help ^objectspec」を使用してリポジトリサーバー指定の

詳細を確認できます。)

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

== CMD\_HELP\_REPOSITORY\_LIST ==

備考:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{^repid} | {0} リポジトリ識別子。

{^repname} | {1} リポジトリ名。

{^repserver} | {2} サーバー名。

{^repowner} | {3} リポジトリ所有者。

{^repguid} | {4} リポジトリの一意識別子。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

形式パラメーター値が「^TABLE」の場合、出力は {^repid}、{^repname}、

{^repserver} のフィールドがある表形式を使用して出力されます。

例:

cm ^repository

(すべてのリポジトリをリストします。)

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

(20 個の空白内に左寄せしたリポジトリ名、次に空白、次に

リポジトリ所有者を書き込みます。)

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

(前の例と同じように書き込みます。)

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

(次のフィールドがある表形式を使用してリポジトリのリストを書き込みます:

リポジトリ ID、リポジトリ名、リポジトリサーバー名。)

== CMD\_DESCRIPTION\_REPOSITORY\_RENAME ==

リポジトリ名を変更します。

== CMD\_USAGE\_REPOSITORY\_RENAME ==

使用方法:

cm ^repository | ^repo ^rename [<リポジトリ指定>] <新しい名前>

リポジトリ指定 名前変更するリポジトリ。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

新しい名前 リポジトリの新しい名前。

== CMD\_HELP\_REPOSITORY\_RENAME ==

備考:

このコマンドは、リポジトリ名を変更します。

リポジトリ指定が指定されていない場合は、現在のリポジトリが想定されます。

例:

cm ^repository ^rename development

(現在のリポジトリの名前が「development」に変更されます。)

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

(「SERVER」上の「default」リポジトリの名前が「development」に変更されます。)

== CMD\_DESCRIPTION\_REPOSITORY\_ADD ==

データベースを追加することにより既存のリポジトリを接続します。

== CMD\_USAGE\_REPOSITORY\_ADD ==

使用方法:

cm ^repository | ^repo ^add <データベースファイル> <リポジトリ名> <リポジトリサーバー指定>

データベースファイル データベースのバックエンド上のデータベースファイルの名前。

リポジトリ名 リポジトリの名前。

リポジトリサーバー指定 リポジトリサーバーの指定。

(「cm ^help ^objectspec」を使用してリポジトリサーバー指定の

詳細を確認できます。)

== CMD\_HELP\_REPOSITORY\_ADD ==

備考:

既存のリポジトリデータベースをサーバーに再接続します。

例: 「cm ^repository ^delete」コマンドを使用した後、「^add」コマンドを使用して

リポジトリをあるサーバーから別のサーバーに移動するか、アーカイブされた

リポジトリを復元します。

例:

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

== CMD\_DESCRIPTION\_TRIGGER\_CREATE ==

サーバーに新しいトリガーを作成します。

== CMD\_USAGE\_TRIGGER\_CREATE ==

使用方法:

cm ^trigger | ^tr ^create | ^mk <サブタイプのタイプ> <新しい名前> <スクリプトパス>

[--^position=<新しい位置>]

[--^filter=<文字列フィルター>]

[--^server=<リポジトリサーバー指定>]

サブタイプのタイプ トリガー実行とトリガー操作。

トリガータイプのリストを表示するには「cm ^showtriggertypes」

と入力します。

新しい名前 新しいトリガーの名前。

スクリプトパス 実行するスクリプトが置かれているサーバー上の

ディスクパス。コマンドラインが「^webtrigger」で始まる場合、

トリガーはウェブトリガーと見なされます。詳細については

「備考」を参照してください。

オプション:

--^position 指定されたトリガーの新しい位置。

この位置は、同じタイプの別のトリガーによって使用中でない必要が

あります。

--^filter 指定されたフィルターに一致する項目のみをチェックします。

--^server 指定されたサーバーにトリガーを作成します。

サーバーが指定されていない場合は、クライアントに設定されている

サーバーでコマンドを実行します。

(「cm ^help ^objectspec」を使用してリポジトリサーバー指定の

詳細を確認できます。)

== CMD\_HELP\_TRIGGER\_CREATE ==

備考:

ウェブトリガー: ウェブトリガーは、「^webtrigger <ターゲット URI>」をトリガーコマンド

として入力することで作成します。この場合、トリガーは指定された URI に対して

POST クエリを実行します。リクエスト本文には、JSON ディクショナリと

トリガー環境変数、および文字列の配列を指す

固定の入力キーが含まれます。

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\*"

(このトリガーは、ラベル名が「LB」で開始する場合にのみ実行され、

「myRep」というリポジトリに作成されます。)

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

== CMD\_DESCRIPTION\_MOVE ==

ファイルやディレクトリを移動または名前変更します。

== CMD\_USAGE\_MOVE ==

使用方法:

cm ^move | ^mv <ソースパス> <同期先パス> [--^format=<文字列形式>]

[--^errorformat=<文字列形式>]

ソースパス ソース項目パス。

同期先パス 同期先項目パス。

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^errorformat エラーメッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

== CMD\_HELP\_MOVE ==

備考:

このコマンドは、リポジトリ内の項目を移動または名前変更します。

ローカルファイルシステム内でも変更が行われます。

ソースパスがファイルの場合、同期先パスはファイルまたはディレクトリに

することができます。最初のケースでは、ファイルが名前変更されます。それ以外の場合は

項目が移動されます。

ソースパスがディレクトリの場合、同期先パスはディレクトリにする必要があります。

移動または名前変更する項目が存在する必要があります。

形式:

{0} ソースパス (「--^format」と「--^errorformat」の両方に対して)

{1} 同期先パス (「--^format」と「--^errorformat」の両方に対して)

例:

cm ^move file.txt file.old

(項目の名前を変更します。)

cm ^mv .\file.old .\oldFiles

(「file.old」を「oldFiles」に移動します。)

cm ^move .\src .\src2

(ディレクトリ名を変更します。)

== CMD\_DESCRIPTION\_LABEL ==

ユーザーにラベルの管理を許可します。

== CMD\_USAGE\_LABEL ==

使用方法:

cm ^label | ^lb <コマンド> [オプション]

コマンド:

^create | ^mk

^delete | ^rm

^rename

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^label <コマンド> --^usage

cm ^label <コマンド> --^help

== CMD\_HELP\_LABEL ==

例:

cm ^label myNewLabel ^cs:42

(「^create」コマンドはオプションです。)

cm ^label ^rename myNewLabel newLabelName

cm ^label ^delete newLabelName

== CMD\_DESCRIPTION\_LABEL\_CREATE ==

ラベルを変更セットに適用し、必要に応じてラベルを作成します。

== CMD\_USAGE\_LABEL\_CREATE ==

使用方法:

cm ^label [^create] <ラベル指定> [<変更セット指定> | <ワークスペースパス>]

[--^allxlinkedrepositories]

[-^c=<コメント文字列> | -^commentsfile=<コメントファイル>]

ラベル指定 新しいラベル名。

(「cm ^help ^objectspec」を使用してラベル指定の詳細を

確認できます。)

変更セット指定 ラベルに対する変更セットの名前または完全な指定。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

ワークスペースパス ラベルへのワークスペースのパス。(ワークスペースが指す

変更セットにラベルが付けられます。)

オプション:

--^allxlinkedrepositories Xlink でリンクされたすべてのリポジトリに新しいラベルを作成します。

-^c 指定されたコメントを新しいラベルに適用します。

-^commentsfile 指定されたファイル内のコメントを新しいラベルに

適用します。

== CMD\_HELP\_LABEL\_CREATE ==

備考:

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。

例:

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

(変更セット 1203 に添付されたラベル「BL001」を作成し、「commentlb001.txt」

ファイルにコメントを適用します。)

cm ^label BL002 ^cs:1203 -^c="first release"

(ラベル「BL002」をコメント付きで作成し、変更セット 1203 に添付します。)

== CMD\_DESCRIPTION\_LABEL\_DELETE ==

1 つ以上のラベルを削除します。

== CMD\_USAGE\_LABEL\_DELETE ==

使用方法:

cm ^label ^delete <ラベル指定>[ ...]

ラベル指定 削除するラベル。空白を使用してラベルを区切ります。

(「cm ^help ^objectspec」を使用してラベル指定の詳細を

確認できます。)

== CMD\_HELP\_LABEL\_DELETE ==

備考:

このコマンドは、1 つ以上のラベルを削除します。

例:

cm ^label ^delete ^lb:BL001

(ラベル「BL001」を削除します。)

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

(ラベル「BL001」と「BL002」を削除します。)

== CMD\_DESCRIPTION\_LABEL\_RENAME ==

ラベル名を変更します。

== CMD\_USAGE\_LABEL\_RENAME ==

使用方法:

cm ^label ^rename <ラベル指定> <新しい名前>

ラベル指定 名前変更するラベル。

(「cm ^help ^objectspec」を使用してラベル指定の詳細を確認できます。)

新しい名前 ラベルの新しい名前。

== CMD\_HELP\_LABEL\_RENAME ==

備考:

このコマンドは、ラベル名を変更します。

例:

cm ^label ^rename ^lb:BL001 BL002

(ラベル名を「BL001」から「BL002」に変更します。)

== CMD\_DESCRIPTION\_OBLITERATE ==

非推奨

== CMD\_USAGE\_OBLITERATE ==

非推奨。

== CMD\_HELP\_OBLITERATE ==

非推奨。

== CMD\_DESCRIPTION\_OBJECTSPEC ==

オブジェクト指定の記述方法について説明します。

== CMD\_USAGE\_OBJECTSPEC ==

使用方法:

cm ^objectspec

オブジェクト指定の作成方法に関するすべての情報を取得します。

== CMD\_HELP\_OBJECTSPEC ==

いくつかの Plastic SCM コマンドは、指定されたオブジェクト (通常はブランチ、

変更セット、リポジトリなど) を参照するための入力として「オブジェクト指定」を想定します。

このドキュメントでは、使用可能なさまざまな「指定」と、その作成方法について

説明します。

各指定タイプは一意のタグで開始します。たとえば、「^rep:」や「^cs:」などです。タグは、

一般オブジェクト指定 (たとえば「cm ^setowner オブジェクト指定」) を受け取るコマンドに

対して指定する必要があります。一方、単一タイプの指定のみを受け取るコマンド (たとえば

「cm ^getfile リビジョン指定」) に対しては、多くの場合に省略できます。

-- リポジトリサーバーの指定 (リポジトリサーバー指定) --

^repserver:name:port

例:

cm ^repo ^list ^repserver:skull:8084

cm ^repo ^list skull:8084

注記:

履歴上の理由から、これを「サーバー指定」ではなく「リポジトリサーバー指定」

と呼んでいます。以前は別々のワークスペースとリポジトリサーバーが

あり、そのネーミングが残っています。

-- リポジトリの指定 (リポジトリ指定) --

^rep:rep\_name@[リポジトリサーバー指定]

例:

cm ^showowner ^rep:codice@localhost:6060

(^showowner ではリポジトリだけでなく他のタイプのオブジェクトも許可されるため

「^rep:」が必要です。そのため、ユーザーはオブジェクトタイプを

指定する必要があります。)

-- ブランチの指定 (ブランチ指定) --

^br:[/]ブランチ名[@repspec]

例:

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

(この場合、「^br:」、「^rep:」および「^repserver」は不要なので、

コマンドでははるかに短い形式:

「cm ^switch main@plastic@skull:9095」が許可されます。)

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

備考:

ブランチの最初の「/」は必須ではありません。以前はすべてのブランチを

/main、/main/task001 のように指定していました。しかし現在は、コマンドが

より簡潔になる main、main/task001 のような短い形式を選びます。

-- 変更セットの指定 (変更セット指定) --

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

変更セットの番号または GUID を指定できます。

例:

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

-- ラベルの指定 (ラベル指定) --

^lb:lb\_name[@repspec]

例:

cm ^switch ^lb:RELEASE2.0

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

-- リビジョン指定 --

さまざまなタイプのリビジョン指定があります。

^rev:item\_path[#(ブランチ指定|変更セット指定|ラベル指定)]

^rev:^serverpath:item\_path#(ブランチ指定|変更セット指定|ラベル指定)

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

^rev:^itemid:item\_id#(ブランチ指定|変更セット指定|ラベル指定)

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

-- 項目指定 --

^item:path

めったに使用されません。

例:

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

-- 属性指定 --

^att:att\_name[@repspec]

例:

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

-- シェルブ指定 --

^sh:sh\_number[@repspec]

例:

cm ^diff ^sh:2 ^sh:4

-- ワークスペース指定 --

^wk:name@clientmachine

ワークスペース関連のコマンドにのみ適用されるため、めったに使用されません。パスの代わりに

名前やマシンでワークスペースを指定する場合に役立ちます。

例:

cm ^showselector ^wk:codebase@modok

注記:

これらの指定は、一元化された方法でワークスペースメタデータを格納する

方法として「ワークスペースサーバー」が存在する古い Plastic SCM 2.x に由来します。

パフォーマンスの問題が原因で非推奨になりました。

== CMD\_DESCRIPTION\_PARTIAL ==

部分的なワークスペースでコマンドを実行します。

== CMD\_USAGE\_PARTIAL ==

使用方法:

cm ^partial <コマンド> [オプション]

コマンド:

^configure

^add

^undo

^co | ^checkout

^unco | ^undocheckout

^ci | ^checkin

^mv | ^move

^rm | ^remove

^stb | ^switch

^upd | ^update

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^partial <コマンド> --^usage

cm ^partial <コマンド> --^help

== CMD\_HELP\_PARTIAL ==

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 ==

バージョン管理に項目を追加します。

== CMD\_USAGE\_PARTIAL\_ADD ==

使用方法:

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

[--^ignorefailed] [--^skipcontentcheck] <項目パス>[ ...]

項目パス 追加する項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを区切ります。

\* を使用して現在のディレクトリのすべてのコンテンツを追加します。

オプション:

-^R 項目を再帰的に追加します。

--^silent 出力を表示しません。

--^parents 指定された項目の親ディレクトリを操作に含めます。

--^ignorefailed 項目を追加できない場合、追加操作は

それなしで続行されます。注意: ディレクトリを追加できない

場合、そのコンテンツは追加されません。

--^skipcontentcheck 拡張子がファイルをテキストまたはバイナリとして設定するのに

十分でない場合、タイプを検出するのにコンテンツをチェックする

代わりに、デフォルトでバイナリとして設定されます。

== CMD\_HELP\_PARTIAL\_ADD ==

備考:

項目を追加するための要件:

- 追加する項目の親ディレクトリが先に追加されている必要があります。

例:

cm ^partial ^add pic1.png pic2.png

(「pic1.png」と「pic2.png」の項目を追加します。)

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

(「picture.png」の項目をパス「c:\workspace」に追加します。)

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

(「src」を再帰的に追加します。)

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

(「design01.png」ファイルと「samples」親フォルダを追加します。)

cm ^partial ^add -^R \*

(現在のディレクトリのすべてのコンテンツを再帰的に追加します。)

== CMD\_DESCRIPTION\_PARTIAL\_CHECKIN ==

変更をリポジトリに格納します。

== CMD\_USAGE\_PARTIAL\_CHECKIN ==

使用方法:

cm ^partial ^checkin | ^ci [<項目パス>[ ...]]

[-^c=<コメント文字列> | -^commentsfile=<コメントファイル>]

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

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

オプション:

項目パス チェックインする項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

チェックインを現在のディレクトリに適用するには、. を使用します。

-^c チェックイン操作で作成された変更セットにコメントを

指定します。

-^commentsfile 指定されたファイルからチェックイン操作で作成された

変更セットにコメントを適用します。

--^all | -^a 指定されたパスでローカルに変更、移動、および削除された

項目も含めます。

--^applychanged チェックアウト済みの項目とともにワークスペースで

検出された変更済み項目にチェックイン操作を

適用します。

--^keeplock チェックイン操作の後にロック済み項目のロックを

維持します。

--^symlink チェックイン操作をターゲットではなくシンボリックリンクに

適用します。

--^ignorefailed ロック (排他的チェックアウトとも呼ぶ) を

取得できないかローカルの変更がサーバーの

変更と競合しているために適用できない変更は破棄され、

チェックイン操作はそれらの変更なしで続行されます。

== CMD\_HELP\_PARTIAL\_CHECKIN ==

備考:

- <項目パス> が指定されていない場合、チェックインにはそのワークスペース内の

すべての保留中の変更が関与します。

- チェックイン操作は、常に指定されたパスから再帰的に適用されます。

- 項目をチェックインするには:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックアウトされている必要があります。

- 項目が変更されているもののチェックアウトされていない場合、<項目パス> が

ディレクトリであるか、ワイルドカード (「\*」) が含まれている場合を除き、「--^applychanged」フラグは

不要です。

チェックインするにはリビジョンコンテンツが前のリビジョンと異なっている必要が

あります。

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。

stdin から入力を読み取る:

「^partial ^checkin」コマンドは stdin からパスを読み取ることができます。これを行うには、

シングルダッシュ「-」を渡します。

例: cm ^partial ^checkin -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用してチェックインするファイルを指定できます。

例:

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

(Windows で、すべての .c ファイルをワークスペースにチェックインします。)

例:

cm ^partial ^checkin figure.png landscape.png

(チェックアウト済みの「figure.png」および「landscape.png」ファイルにチェックインを適用します。)

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

(現在のディレクトリにチェックインを適用し、「mycomment.txt」ファイルから

コメントを設定します。)

cm ^partial ^ci background.png -^c="my comment" --^keeplock

(チェックインを「background.png」に適用し、コメントを含め、ロックを

維持します。)

cm ^partial ^checkin --^applychanged

(ワークスペース内のすべての保留中の変更にチェックインを適用します。)

cm ^partial ^checkin リンク --^symlink

(ターゲットにではなくリンクファイルにチェックインを適用します。UNIX 環境で

有効です。)

cm ^partial ^checkin .--^ignorefailed

(現在のディレクトリにチェックインを適用します。適用できない変更は

無視します。)

== CMD\_DESCRIPTION\_PARTIAL\_CHECKOUT ==

ファイルを変更準備完了としてマークします。

== CMD\_USAGE\_PARTIAL\_CHECKOUT ==

使用方法:

cm ^partial ^checkout | ^co [<項目パス>[ ...]] [--^resultformat=<文字列形式>]

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

オプション:

項目パス チェックアウトする項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

チェックアウトを現在のディレクトリに適用するには、. を使用します。

--^resultformat 出力の結果メッセージを特定の形式で

取得します。

--^silent 出力を表示しません。

--^ignorefailed 項目をロックできない (排他的チェックアウトを実行

できない) 場合、チェックアウト操作はロックなしで

続行されます。

== CMD\_HELP\_PARTIAL\_CHECKOUT ==

備考:

項目をチェックアウトするには:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックインされている必要があります。

サーバーでロックが設定されている (lock.conf が存在する) 場合、パスで

チェックアウトが行われるたびに、Plastic はそれがいずれかのルールに従っているかどうかを確認し、

従っている場合、パスは排他的チェックアウト (ロック) されているため、同時に

チェックアウトできるものはありません。

「cm ^lock ^list」を使用してサーバー内のすべてのロックを取得できます。

ロックの仕組みについては、管理者ガイドを参照してください。

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

例:

cm ^partial ^checkout pic1.png pic2.png

(「pic1.png」および「pic2.png」ファイルをチェックアウトします。)

cm ^partial ^co \*.png

(すべての png ファイルをチェックアウトします。)

cm ^partial ^checkout .

(現在のディレクトリをチェックアウトします。)

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

(「src」フォルダーを再帰的にチェックアウトします。)

== CMD\_DESCRIPTION\_PARTIAL\_CONFIGURE ==

項目をロードまたはアンロードすることにより、ワークスペースを設定できます。

== CMD\_USAGE\_PARTIAL\_CONFIGURE ==

使用方法:

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

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

パス ロードまたはアンロードするパス。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

パスは「/」で始まる必要があります。

オプション:

--^silent 出力を表示しません。

--^ignorefailed プロセス中のすべてのエラーをスキップします。パスが正しくなくても

コマンドは停止しません。

--^ignorecase パスの大文字/小文字を無視します。このフラグを使用すると、「^configure」

はユーザーが「/data/teXtures」と記述した場合でも

「/Data/Textures」に対して動作します。

--^restorefulldirs 無効なディレクトリ設定 (部分的でない操作が部分的な

ワークスペースに対して実行された場合に発生) をリセットします。

このリストにあるディレクトリは完全に設定

(全体チェック) されるため、更新時に新規コンテンツが自動的に

ダウンロードされます。

この操作はファイルをダウンロードせず、部分的な

ワークスペースへのディレクトリ設定の復元のみを

行います。

== CMD\_HELP\_PARTIAL\_CONFIGURE ==

備考:

コマンドでは、再帰的な操作が想定されます。

例:

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

(「landscape\_grey.png」項目をロードします。)

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

(「landscape\_black.png」項目をアンロードします。)

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

(「soft-black.png」を除き、「soft」ディレクトリのすべての子項目をロードします。)

cm ^partial ^configure -/

(ワークスペース全体をアンロードします。)

cm ^partial ^configure -/ +/

(ワークスペース全体をロードします。)

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

(すでにアンロードされている場合でも、「figure-64.png」項目をアンロードします。)

cm ^partial ^configure +/ --^restorefulldirs

(新しいコンテンツを自動的にダウンロードするようにすべてのディレクトリを設定します。)

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

(新しいコンテンツを自動的にダウンロードするように「/src/lib」とそのサブディレクトリ

のみを設定します。)

== CMD\_DESCRIPTION\_PARTIAL\_MOVE ==

ファイルやディレクトリを移動または名前変更します。

== CMD\_USAGE\_PARTIAL\_MOVE ==

使用方法:

cm ^partial ^move | ^mv <ソースパス> <同期先パス> [--^format=<文字列形式>]

ソースパス ソース項目パス。

同期先パス 同期先項目パス。

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

== CMD\_HELP\_PARTIAL\_MOVE ==

備考:

このコマンドは、リポジトリ内の項目を移動または名前変更します。

ローカルファイルシステム内でも変更が行われます。

ソースパスがファイルの場合、同期先パスはファイルまたはディレクトリに

することができます。最初のケースでは、ファイルが名前変更されます。それ以外の場合は

項目が移動されます。

ソースパスがディレクトリの場合、同期先パスはディレクトリにする必要があります。

移動または名前変更する項目が存在する必要があります。

形式:

{0} ソースパス。

{1} 同期先のパス。

例:

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

(項目の名前を変更します。)

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

(「file-blue.png」を「blueFiles」に移動します。)

cm ^partial ^move .\design .\marketing

(ディレクトリを名前変更します。)

== CMD\_DESCRIPTION\_PARTIAL\_RM ==

ファイルまたはディレクトリをバージョン管理から削除します。

== CMD\_USAGE\_PARTIAL\_RM ==

使用方法:

cm ^partial ^remove | ^rm <項目パス>[ ...][--^nodisk]

項目パス 削除する項目パス。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

オプション:

--^nodisk バージョン管理から項目を削除しますが、ディスク上には保持します。

== CMD\_HELP\_PARTIAL\_RM ==

備考:

項目はディスクから削除されます。削除された項目はソースコード管理の親

ディレクトリから削除されます。

要件:

- 項目がソースコード管理の対象になっている必要があります。

例:

cm ^partial ^remove src

(「src」を削除します。「src」がディレクトリの場合、これは次と同じです:

cm ^partial ^remove -^R src。)

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

(「pic01.png」をバージョン管理から削除しますが、ディスク上には保持します。)

== CMD\_DESCRIPTION\_PARTIAL\_SWITCH ==

ブランチを作業中のブランチとして設定します。

== CMD\_USAGE\_PARTIAL\_SWITCH ==

使用方法:

cm ^switch <ブランチ指定> [--^report | --^silent] [--^workspace=<パス>]

(作業中のブランチを設定し、ワークスペースを更新します。)

cm ^switch <ブランチ指定> --^configure <+|-パス>[ ...][--^silent]

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

(作業中のブランチを設定し、ワークスペース設定を「cm

^partial ^configure」コマンドと同様に実行します。)

ブランチ指定 ブランチの指定。(「cm ^help ^objectspec」を使用して

ブランチ指定の詳細を確認できます。)

パス ロードまたはアンロードするパス。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。パスは「/」で始まる必要があります。

オプション:

--^silent 出力を表示しません。

--^report コマンドの完了時に適用された変更のリストを

出力します。「--^silent」を使用すると、この設定が上書きされます。

このオプションは、「--^configure」オプションが指定されて

いない場合にのみ動作します。

--^configure 作業中のブランチの更新後にワークスペースを設定

(項目をロード/アンロード) します。「cm ^partial

^configure --^help」を調べて、設定するパスの指定方法を

確認してください。

--^ignorefailed 設定プロセス中のすべてのエラーをスキップします。

パスが正しくなくてもコマンドは停止しません。

--^ignorecase パスの大文字/小文字を無視します。このフラグを使用すると、

オプション「--^configure」はユーザーが「/data/teXtures」と

記述した場合でも「/Data/Textures」に対して機能します。

--^workspace=パス ワークスペースが置かれているパス。

== CMD\_HELP\_PARTIAL\_SWITCH ==

備考:

このコマンドは、ユーザーに作業中のブランチの更新を許可します。ブランチの更新後、

コマンドは「cm ^partial ^update」コマンドが行うのと同様に

ワークスペースを新しいブランチに更新します。ただし、「--^configure」オプションが

指定されている場合、このコマンドは「cm ^partial ^configure」コマンドが

行うのと同様に、新しいブランチ設定を使用したワークスペースの設定を許可します。

例:

cm ^switch ^br:/main/task

(/main/task を作業中のブランチとして設定し、ワークスペースを更新します。)

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

(/main/task を作業中のブランチとして設定し、/art/images フォルダーを

ロードするようにワークスペースを設定します。)

== CMD\_DESCRIPTION\_PARTIAL\_UNCO ==

項目のチェックアウトを取り消します。

== CMD\_USAGE\_PARTIAL\_UNCO ==

使用方法:

cm ^partial ^undocheckout | ^unco <項目パス>[ ...][--^silent]

項目パス 操作を適用する項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

操作を現在のディレクトリに適用するには、. を使用します。

オプション:

--^silent 出力を表示しません。

== CMD\_HELP\_PARTIAL\_UNCO ==

備考:

項目がチェックアウト済みになっていて、それをチェックインしたくない場合は、このコマンドを

使用してチェックアウトを取り消すことができます。ファイルとフォルダーの両方のチェックアウトを

取り消すことができます。項目は、チェックアウトする前の状態に更新されます。

要件:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックアウトされている必要があります。

例:

cm ^partial ^undocheckout .

(現在のディレクトリでチェックアウトを取り消します。)

cm ^partial ^undocheckout pic1.png pic2.png

cm ^unco c:\workspace\design01.png

(選択したファイルのチェックアウトを取り消します。)

== CMD\_DESCRIPTION\_PARTIAL\_UNDO ==

ワークスペース内の変更を取り消します。

== CMD\_USAGE\_PARTIAL\_UNDO ==

使用方法:

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

[<フィルター>[ ...]]

[--^silent | --^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

パス 操作を適用するファイルまたはディレクトリの

パス。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

パスが指定されていない場合は、デフォルトで

取り消し操作が現在のディレクトリ内のすべてのファイルに

適用されます。

フィルター 指定された 1 つまたは複数のフィルターを指定されたパスに

適用します。空白を使用してフィルターを区切ります。詳細については

「フィルター」セクションを参照してください。

オプション:

--^symlink 取り消し操作をターゲットではなくシンボリックリンクに

適用します。

-^r 取り消しを再帰的に実行します。

--^silent 出力を表示しません。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator 「--^machinereadable」フラグとともに使用され、行をどのように

開始する必要があるかを指定します。

--^endlineseparator 「--^machinereadable」フラグとともに使用され、行をどのように

終了する必要があるかを指定します。

--^fieldseparator 「--^machinereadable」フラグとともに使用され、フィールドを

どのように区切る必要があるかを指定します。

フィルター:

フラグが指定されていない場合、デフォルトですべての変更が取り消されますが、

パスは下のフラグの 1 つ以上を使用してフィルター処理できます。

ファイルまたはディレクトリが指定した種類の変更の 1 つ以上と一致する場合、

指定されたファイルまたはディレクトリに対するすべての変更が取り消されます。

たとえば、「--^checkedout」と「--^moved」の両方を指定した場合に、ファイルの

チェックアウトと移動の両方が行われていると、両方の変更が取り消されます。

--^checkedout チェックアウト済みのファイルとディレクトリを選択します。

--^unchanged コンテンツが変更されていないファイルを選択します。

--^changed ローカルに変更またはチェックアウトされたファイルと

ディレクトリを選択します。

--^deleted 削除されたファイルとディレクトリを選択します。

--^moved 移動されたファイルとディレクトリを選択します。

--^added 追加されたファイルとディレクトリを選択します。

== CMD\_HELP\_PARTIAL\_UNDO ==

備考:

^undo は危険なコマンドです。作業内容が不可逆的な方法で取り消されます。

^undo が完了すると、その影響を受けたファイルとディレクトリの以前の状態を

復元することはできません。引数にパスが指定されて

いない場合は、デフォルトで現在のディレクトリでのすべての変更が

取り消されますが、再帰的ではありません。

これらは、/src ディレクトリから実行されたときと同等です。

/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

操作を再帰的にするには、「-^r」フラグを指定する必要があります。

ディレクトリ下のすべての変更を取り消すには (ディレクトリ自体に影響する

変更を含む):

cm ^partial ^undo ディレクトリパス -^r

ディレクトリパスがワークスペースパスの場合、ワークスペース内のすべての

変更が取り消されます。

例:

cm ^partial ^undo .-^r

(現在のディレクトリでのすべての変更を再帰的に取り消します。ワークスペース

のルートから実行された場合は、ワークスペース全体のすべての変更が取り消されます。)

cm ^partial ^co file.txt

cm ^partial ^undo file.txt

(file.txt に対するチェックアウトを取り消します。)

^echo ^content >> file.txt

cm ^partial ^undo file.txt

(file.txt に対するローカルの変更を取り消します。)

cm ^partial ^undo src

(src ディレクトリとそのファイルに対する変更を取り消します。)

cm ^partial ^undo src/\*

(src に含まれるすべてのファイルとディレクトリの変更を取り消します。

src には影響しません。)

cm ^partial ^undo \*.cs

(現在のディレクトリ内の、\*.cs と一致するすべてのファイルまたはディレクトリ

に対する変更を取り消します。)

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

(現在のディレクトリとその下のすべてのディレクトリ内の、\*.cs と一致する

すべてのファイルまたはディレクトリに対する変更を取り消します。)

cm ^partial ^co file1.txt file2.txt

^echo ^content >> file1.txt

cm ^partial ^undo --^unchanged

(変更されていない file2.txt のチェックアウトを取り消します。ローカルに

変更された file1.txt は無視します。)

^echo ^content >> file1.txt

^echo ^content >> file2.txt

cm ^partial ^co file1.txt

cm ^partial ^undo --^checkedout

(チェックアウト済みのファイル file1.txt 内の変更を取り消します。file2.txt は

チェックアウト済みでないため無視されます。)

cm ^partial ^add file.txt

cm ^partial ^undo file.txt

(file.txt の追加を取り消し、もう一度非公開ファイルにします。)

^rm file1.txt

^echo ^content >> file2.txt

cm ^partial ^add file3.txt

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

(file1.txt の削除と file3.txt の追加を取り消します。file2.txt の変更は

無視します。)

== CMD\_DESCRIPTION\_PARTIAL\_UPDATE ==

部分的なワークスペースを更新し、最新の変更をダウンロードします。

== CMD\_USAGE\_PARTIAL\_UPDATE ==

使用方法:

cm ^partial ^update [<項目パス>[ ...]] [--^changeset=<番号>]

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

項目パス 更新される項目。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

更新を現在のディレクトリに適用するには、. を使用します。

パスが指定されていない場合は、現在の部分的な

ワークスペースが完全に更新されます。

オプション:

--^changeset 部分的なワークスペースを特定の変更セットに更新します。

--^silent 出力を表示しません。

--^report コマンドの完了時に適用された変更のリストを

出力します。「--^silent」を使用すると、この設定が上書きされます。

--^dontmerge ファイルの競合をマージせず、スキップします。

その他の変更は正しく適用されます。このオプションは

ユーザーの対応を回避するための自動化に役立ちます。

== CMD\_HELP\_PARTIAL\_UPDATE ==

備考:

「^partial ^update」コマンドは、最新でないファイルを更新します。

コマンドでは、再帰的な操作が想定されます。

「--^changeset」オプションを使用しているときに、指定されたすべてのパスが

同じ Xlink の内部のファイルの場合、ダウンロードするバージョンは Xlink でリンクされた

リポジトリの指定された変更セット内で検索されます。

例:

cm ^partial ^update

(現在の部分的なワークスペース内のすべてを更新します。)

cm ^partial ^update .

(現在のディレクトリのすべての子項目を更新します。)

cm ^partial ^update backgroud-blue.png

(「backgroud-blue.png」項目を更新します。)

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

(「soft\_black.png」および「soft-grey.png」項目を更新します。)

cm ^partial ^update src --^report

(「src」ディレクトリのすべての子項目を更新し、適用された変更リストを

最後に出力します。)

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

(「src」ディレクトリのすべての子項目を、変更セット 4 にロードされている

コンテンツに更新します。)

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

(「xlink/first.png」項目を、Xlink でリンクされたリポジトリの変更セット 4 にロードした

コンテンツに更新します。)

== CMD\_DESCRIPTION\_PATCH ==

仕様からパッチファイルを生成するか、生成されたパッチを現在のワークスペースに

適用します。

== CMD\_USAGE\_PATCH ==

使用方法:

cm ^patch <ソース指定> [<ソース指定>] [--^output=<出力ファイル>]

[--^tool=<差分のパス>]

ブランチの差分、変更セット、または変更セット間の差分を含む

パッチファイルを生成します。テキストおよび

バイナリファイルの差分も追跡されます。

cm ^patch --^apply <パッチファイル> [--^tool=<パッチのパス>]

生成されたパッチファイルの内容を現在のワークスペースに適用することを

許可します。

ソース指定 変更セットまたはブランチの完全な指定。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

出力ファイル パッチの内容を保存するファイル。ファイルが指定されていない

場合は、パッチの内容が標準出力に出力されます。

パッチファイル 現在のワークスペースに適用するパッチファイル。

オプション:

--^output パッチコマンドの出力ファイルを設定します。

--^tool 使用するアプリケーション (差分またはパッチ) を設定します。

== CMD\_HELP\_PATCH ==

制限事項:

出力パッチファイルがすでに存在する場合、それはコマンドによって上書きされません。

パッチを適用する場合、修正されたファイルがディスク上に存在しなければ、コマンドは

変更をそのファイルに適用しません。

重要:

このコマンドでは、http://gnuwin32.sourceforge.net/packages/patch.htm

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

に公開されている差分およびパッチツールが必要です。

インストールしたら、それらの場所を PATH 環境変数に追加することを

お勧めします。

例:

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

(変更セット 4 の差分を統一された形式でコンソールに出力します。)

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

(ブランチ「main」の差分を使用してパッチファイルを生成します。)

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

(上記と同じですが、カスタムの exe を使用します。)

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

(変更セット 2 と 4 の差分を統一された形式でコンソールに出力します。)

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

(カスタムの exe を使用して「file.patch」内のパッチをローカルワークスペースに適用します。)

== CMD\_DESCRIPTION\_QUERY ==

SQL クエリを実行します。SQL ストレージが必要です。

== CMD\_USAGE\_QUERY ==

使用方法:

cm ^query <SQL コマンド> [--^outputfile=<出力ファイル>]

[--^solveuser=<列名>[,...]]

[--^solvepath=<列名>[,...]]

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

[--^columnseparator=<セパレーター>] [--^repository=<名前>]

SQL コマンド 実行する SQL クエリ。

オプション:

--^outputfile 結果を出力ファイルに書き込みます。

--^solveuser 指定された列をユーザー名列として設定します。

クエリインタープリターでは、これらの列のデータが

ユーザーであると想定され、それらの解決が試行されます。

--^solvepath 指定された列を項目 ID 列として設定します。クエリ

インタープリターでは、項目 ID のファイルシステムパスへの

解決が試行されます。

--^columnwidth 各列の幅を指定して出力を書式設定します。

--^nocolumnname 列名を出力しません。

--^columnseparator タブ (\t) ではなく文字を列セパレーターとして使用します。

--^repository クエリするリポジトリ。

== CMD\_HELP\_QUERY ==

備考:

このコマンドは、ユーザーにサーバーデータベースでの SQL クエリの実行を許可します。

SQL クエリを記述するには、次の 2 つの定義済み関数を使用してユーザーとパスを

管理します。

- ユーザー名を Plastic SCM 形式に解決する「^SolveUser(<ユーザー名>)」。

- ディスクパスを項目 ID に解決する「^SolvePath(<パス>)」。

また、クエリ結果を人間が判読可能な形式で表示するオプションを使用することもできます。

オプション「--^solveuser=<列名>」および「--^solvepath=<列名>」

を使用して、クエリインタープリターが判読可能なテキストに変換する

必要のある列を指定できます。複数の列名をコンマで区切って

指定できます。

例:

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

(リポジトリ「reptest」にある「リビジョン」テーブルからデータを取得します。)

cm ^query "^SELECT b.^sname ^as ブランチ名, o.^dtimestamp ^as 日付 ^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

(所有者が「john」のブランチをファイルに出力します。)

cm ^query "^select r.^iobjid, r.^fiditem ^as パス, s.^scode ^as ユーザー名 ^FROM \

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

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

--^solveuser=ユーザー名 --^solvepath=パス --^repository=reptest@server2:9095

(選択したデータを選択したリポジトリから取得します。)

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

(パス「info」のすべてのリビジョンデータを取得します。)

== CMD\_DESCRIPTION\_ATTRIBUTE\_DELETE ==

1 つ以上の属性を削除します。

== CMD\_USAGE\_ATTRIBUTE\_DELETE ==

使用方法:

cm ^attribute | ^att ^delete | ^rm <属性指定>[ ...]

属性指定 削除する属性。空白を使用して属性を

区切ります。

(「cm ^help ^objectspec」を使用して属性指定の詳細を

確認できます。)

== CMD\_HELP\_ATTRIBUTE\_DELETE ==

備考:

このコマンドは、1 つ以上の属性を削除します。

例:

cm ^attribute ^delete ^att:status

(属性「ステータス」を削除します。)

cm ^att ^rm ステータス ^att:integrated@reptest@server2:8084

(属性「ステータス」および「統合済み」を削除します。)

== CMD\_DESCRIPTION\_ATTRIBUTE\_UNSET ==

オブジェクトの属性を設定解除します。

== CMD\_USAGE\_ATTRIBUTE\_UNSET ==

使用方法:

cm ^attribute | ^att ^unset <属性指定> <オブジェクト指定>

属性指定 属性の指定。(「cm ^help ^objectspec」を

使用して属性指定の詳細を確認できます。)

オブジェクト指定 属性を削除するオブジェクトの指定。

属性は、ブランチ、変更セット、シェルブセット、

ラベル、項目、リビジョンに対して設定できます。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

== CMD\_HELP\_ATTRIBUTE\_UNSET ==

備考:

このコマンドは、オブジェクトに対して以前に設定された属性を設定解除します。

属性自体は削除しません。

例:

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

(属性実現「ステータス」をブランチ「main/SCM105」から削除します。)

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

(属性実現「統合済み」をリポジトリ「reptest」にある変更セット 25 から

削除します。)

== CMD\_DESCRIPTION\_ATTRIBUTE\_RENAME ==

属性の名前を変更します。

== CMD\_USAGE\_ATTRIBUTE\_RENAME ==

使用方法:

cm ^attribute | ^att ^rename <属性指定> <新しい名前>

属性指定 名前変更する属性。(「cm ^help ^objectspec」を使用して

属性指定の詳細を確認できます。)

新しい名前 属性の新しい名前。

== CMD\_HELP\_ATTRIBUTE\_RENAME ==

備考:

このコマンドは、属性名を変更します。

例:

cm ^attribute ^rename ^att:status state

(属性「status」を「state」に名前変更します。)

== CMD\_DESCRIPTION\_ATTRIBUTE\_EDIT ==

属性のコメントを編集します。

== CMD\_USAGE\_ATTRIBUTE\_EDIT ==

使用方法:

cm ^attribute | ^att ^edit <属性指定> <新しいコメント>

属性指定 コメントを変更する属性。(「cm ^help ^objectspec」を

使用して属性指定の詳細を確認できます。)

新しいコメント 属性の新しいコメント。属性値のデフォルトリストを

指定することもできます。

== CMD\_HELP\_ATTRIBUTE\_EDIT ==

備考:

このコマンドは、属性のコメントを変更します。

属性値のデフォルトリストを指定するには、次のような行を属性のコメントに

含める必要があります:

「デフォルト: 値 1, "値 2", 値 3, "最後の値"」。

例:

cm ^attribute ^edit ^att:status "CI パイプラインのブランチのステータス。"

(属性「ステータス」のコメントを編集します。)

cm ^attribute ^edit ^att:status "ブランチのステータス。デフォルト: 未処理, 解決済み, レビュー済み"

(属性「ステータス」のコメントを編集します。値のリストも指定します。

そのため、属性「ステータス」をオブジェクトに設定する際に、

次のいずれかの値を選択できます: 「未処理」、「解決済み」または「レビュー済み」。)

== CMD\_DESCRIPTION\_REPLICATE ==

警告: このコマンドは非推奨になりました。

「cm ^pull」 (「^replicate」と同等) と「cm ^push」

(「^replicate --^push」と同等) を使用してください。

== CMD\_USAGE\_REPLICATE ==

== CMD\_HELP\_REPLICATE ==

== CMD\_DESCRIPTION\_PULL ==

ブランチを別のリポジトリからプルします。

== CMD\_USAGE\_PULL ==

使用方法:

cm ^pull <ソースブランチ指定> <同期先リポジトリ指定>

[--^preview] [--^nodata] [変換オプション]

[--^user=<ユーザー名> [--^password=<パスワード>] | 認証オプション]

(サーバー間の直接レプリケーション。ブランチをリポジトリからプルします。)

cm ^pull <同期先リポジトリ指定> --^package=<パッケージファイル> [認証オプション]

(パッケージベースのレプリケーション。同期先リポジトリにパッケージをインポートします。)

cm ^pull ^hydrate <同期先ブランチ指定> [<ソースリポジトリ指定>]

[--^user=<ユーザー名> [--^password=<パスワード>] | 認証オプション]

(以前に「--^nodata」でレプリケートしたブランチのすべての変更セットの

欠落データを導入します。データを取得するリポジトリが指定されていない場合、

Plastic は「レプリケーション元」(レプリケートされたブランチの

発生元) を使用しようとします。)

cm ^pull ^hydrate <同期先変更セット指定> [<ソースリポジトリ指定>]

[--^user=<ユーザー名> [--^password=<パスワード>] | 認証オプション]

(以前に「--^nodata」でレプリケートした変更セットの欠落データを

導入します。データを取得するリポジトリが指定されていない場合、Plastic は

「レプリケーション元」を使用しようとします。)

ソースブランチ指定 リモートリポジトリからプルするブランチ。

(「cm ^help ^objectspec」を使用してブランチ指定の詳細を確認できます。)

同期先ブランチ指定 ハイドレートするブランチ。

(「cm ^help ^objectspec」を使用してブランチ指定の詳細を確認できます。)

同期先変更セット指定 ハイドレートする変更セット。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

同期先リポジトリ指定 同期先リポジトリ。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

--^package パッケージベースのレプリケーションのためにインポートする、

以前に作成したパッケージファイルを指定します。

直接ネットワーク接続がない場合にサーバー間でデータを

移動するのに役立ちます。

パッケージファイルを作成するには「cm ^push」を参照してください。

オプション:

--^preview プルされる変更に関する情報を提供しますが、

変更は実際には実行されません。このオプションは、

変更をレプリケートする前に転送されるデータを確認するのに

役立ちます。

--^nodata データをレプリケートせずにブランチの変更を

レプリケートします。このオプションは、パッケージレプリケーションでは

使用できません。

変換オプション 詳細については、「変換オプション」セクションを参照してください。

--^user、--^password 認証モードがソースと宛先で異なり、同期先を

認証するプロファイルがない場合に

使用する資格情報。

認証オプション 詳細については、「認証オプション」セクションを

参照してください。

変換オプション:

--^trmode=(^copy|^name|^table --^trtable=<変換テーブルファイル>)

ソースと同期先のリポジトリで異なる認証モードを使用できます。

「--^trmode」オプションでは、ユーザー名をソースから同期先に変換する

方法を指定します。「--^trmode」は次のいずれかの値である必要が

あります:

^copy (デフォルト)。ユーザー識別子が単純にコピーされることを意味します。

^name ユーザー識別子が名前で照合されます。

^table オプション「--^trtable」で指定された変換テーブルを使用します

(下記を参照)。

--^trtable=<変換テーブルファイル>

変換モードが「テーブル」の場合、変換テーブルは <古い名前;新しい名前>

の形式の行を (1 行あたり 1 つ) 含むファイルです。ブランチが

同期先リポジトリに書き込まれる場合、ソースリポジトリ内の「古い名前」で

識別されるユーザーは、同期先の「新しい名前」のユーザーに

設定されます。

認証オプション:

認証データは、次の 2 つのモードのいずれかを使用して指定できます:

1) 認証パラメーターを使用: --^authmode=<モード> --^authdata=<データ>

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

例:

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

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

「--^authdata」行は、client.conf ファイルと profiles.conf ファイル内の

<^SecurityConfig> エントリの内容です。profiles.conf

ファイルは、Plastic GUI の「環境設定」の下の「レプリケーションプロファイル」タブから

簡単に生成できます。

^UPWorkingMode を使用している場合は、次のように指定できます:

--^authmode=^UPWorkingMode --^user=<ユーザー> --^password=<パスワード>

2) 接続先のサーバーごとに異なるファイルがある可能性がある認証ファイルで、

そのサーバーの資格情報を含んでいます。

--^authfile=<認証ファイル>

ファイルには 2 行が含まれます:

行 1) 「--^authmode」で説明しているモード

行 2) 「--^authmode」で説明している認証データ

== CMD\_HELP\_PULL ==

備考:

「^pull」コマンドでは、ソースリポジトリと同期先リポジトリ間でブランチを

(変更セットとともに) 複製できます。

リポジトリは異なるサーバーに配置できます。

2 つのレプリケーション操作 (「^push」と「^pull」) があります。

「^pull」操作は、レプリケーション操作によって、ソースリポジトリからのデータを

同期先リポジトリに格納することが要求されることを意味します。

クライアントは同期先リポジトリに接続し、そのホストから、ソースリポジトリへの

接続を確立してターゲットデータを取得します。

プル中は、それはソースに接続される同期先サーバーです。

通常の分散シナリオでは、開発者はデータを自分のローカルサーバーから

メインサーバーにプッシュしますが、開発者が最新のリポジトリ更新をメインサーバーから

プルすることが必要な場合もあります。

レプリケーションでは、2 つのレプリケートされたリポジトリ上の同じブランチで

同時に変更が行われた状況を解決できます。

- プッシュ: 送信している変更より新しい変更があるリポジトリにデータを

プッシュしようとした場合、システムは最新の変更をプルするよう求め、

マージ操作を解決し、最後にプッシュを再試行します。

- プル: 変更セットをリモートブランチからプルするたびに、親変更セットに

正しくリンクされます。プルした変更セットが

ブランチ内の最新の変更セットの子でない場合、

マルチヘッドシナリオが出現します。ブランチには複数の「ヘッド」(ブランチ上の最後の

変更セット) が存在するようになります。再度プッシュする前に 2 つの「ヘッド」をマージ

する必要があります。

プルは 2 つのモードで動作します:

1) サーバー間の直接通信: 同期先サーバーは、ソースサーバーから

データをフェッチし、指定されたブランチのデータを

自動的に同期します。

2) プッシュと「--^package」オプションを使用して以前に生成されたパッケージをインポートします。

モード 1) では、コマンドを実行しているユーザーは、client.conf ファイル内の

デフォルト認証を使用するか、「--^authmode」および「--^authdata」修飾子

または「--^authmode」、「--^user」および「--^password」を指定して (認証

モードが ^UPWorkingMode の場合)、リモートサーバーによって認証される

必要があります。

モード 2) では、push コマンドを使用して以前に生成されたパッケージファイルを

使用する必要があります。

pull レプリケーションは間接的な方法で動作することに注意してください。実行されると、

コマンドは同期先リポジトリに対して、ソースに接続して選択されたブランチを

取得するよう求めます。

ただし、これは push コマンドを使用して直接実行できます。

これにより、コマンドは選択されたブランチをソースから同期先に

レプリケートします。

例:

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

(「main」ブランチを「remoteserver」から「myserver」にプルします。この場合、

両方のサーバーが同じ認証モードで設定されます。)

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

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

(前と同じブランチをプルしますが、今回はリモートサーバーが Active Directory で

ユーザーを認証するように設定されます。たとえば、Linux マシンから、

Active Directory 統合モードを使用するように設定されている Windows サーバーに

接続しています。自分の Active Directory ユーザーと暗号化された

パスワードを指定してサーバーに LDAP として渡します。)

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

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

(同じブランチをプルしますが、今回はユーザーがリモートサーバー上で認証され、

Plastic SCM に含まれるユーザー/パスワードデータベースを

利用します。)

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

(「main」ブランチを「remoteserver」から「myserver」にデータなしでレプリケートします。)

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

(「main」ブランチ内のすべての変更セットをハイドレートしてリモートサーバーから

データを取得します。)

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

(「myserver」内の変更セット 122169 をハイドレートしてリモートサーバー

からデータを取得します。)

== CMD\_DESCRIPTION\_PUSH ==

ブランチを別のリポジトリにプッシュします。

== CMD\_USAGE\_PUSH ==

使用方法:

cm ^push <ソースブランチ指定> <同期先リポジトリ指定>

[--^preview] [変換オプション]

[--^user=<ユーザー名> [--^password=<パスワード>] | 認証オプション]

(サーバー間の直接レプリケーション。ブランチをリポジトリからプッシュします。)

cm ^push <ソースブランチ指定> --^package=<パッケージファイル> [認証オプション]

(パッケージベースのレプリケーション。選択されたブランチでソースサーバーに

レプリケーションパッケージを作成します。

ソースブランチ指定 リモートリポジトリにプッシュするブランチ。

(「cm ^help ^objectspec」を使用してブランチ指定の詳細を確認できます。)

同期先リポジトリ指定 同期先リポジトリ。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

--^package パッケージベースのレプリケーション用のレプリケーションパッケージを

エクスポートするパスを指定します。

直接ネットワーク接続がない場合にサーバー間でデータを

移動するのに役立ちます。

オプション:

--^preview プッシュされる変更に関する情報を提供しますが、

変更は実際には実行されません。このオプションは、

変更をレプリケートする前に転送されるデータを確認するのに

役立ちます。

変換オプション 詳細については、「変換オプション」セクションを参照してください。

--^user、--^password 認証モードがソースと宛先で異なり、同期先を

認証するプロファイルがない場合に

使用する資格情報。

認証オプション 詳細については、「認証オプション」セクションを

参照してください。

変換オプション:

--^trmode=(^copy|^name|^table --^trtable=<変換テーブルファイル>)

ソースと同期先のリポジトリで異なる認証モードを使用できます。

「--^trmode」オプションでは、ユーザー名をソースから同期先に変換する

方法を指定します。「--^trmode」は次のいずれかの値である必要が

あります:

^copy (デフォルト)。ユーザー識別子が単純にコピーされることを意味します。

^name ユーザー識別子が名前で照合されます。

^table オプション「--^trtable」で指定された変換テーブルを使用します

(下記を参照)。

--^trtable=<変換テーブルファイル>

変換モードが「テーブル」の場合、変換テーブルは <古い名前;新しい名前>

の形式の行を (1 行あたり 1 つ) 含むファイルです。ブランチが

同期先リポジトリに書き込まれる場合、ソースリポジトリ内の「古い名前」で

識別されるユーザーは、同期先の「新しい名前」のユーザーに

設定されます。

認証オプション:

認証データは、次の 2 つのモードのいずれかを使用して指定できます:

1) 認証パラメーターを使用: --^authmode=<モード> --^authdata=<データ>

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

例:

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

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

「--^authdata」行は、client.conf ファイルと profiles.conf ファイル内の

<^SecurityConfig> エントリの内容です。profiles.conf

ファイルは、Plastic GUI の「環境設定」の下の「レプリケーションプロファイル」タブから

簡単に生成できます。

^UPWorkingMode を使用している場合は、次のように指定できます:

--^authmode=^UPWorkingMode --^user=<ユーザー> --^password=<パスワード>

2) 接続先のサーバーごとに異なるファイルがある可能性がある認証ファイルで、

そのサーバーの資格情報を含んでいます。

--^authfile=<認証ファイル>

ファイルには 2 行が含まれます:

行 1) 「--^authmode」で説明しているモード

行 2) 「--^authmode」で説明している認証データ

== CMD\_HELP\_PUSH ==

備考:

「^push」コマンドでは、ソースリポジトリと同期先リポジトリ間でブランチを

(変更セットとともに) 複製できます。

リポジトリは異なるサーバーに配置できます。

2 つのレプリケーション操作 (「^push」と「^pull」) があります。

「^push」操作は、レプリケーション操作によって、ソースリポジトリから

同期先リポジトリにデータが送信されることを意味します。この場合、

クライアントはソースリポジトリに接続し、複製するデータを

取得してから、同期先リポジトリに送信します。

前者 (ソース) は同期先に接続する必要がありますが、後者 (同期先) は

自身をソースに接続しません。

通常の分散シナリオでは、開発者は自分のローカルサーバーのデータを

メインサーバーにプッシュします。また、開発者は最新のリポジトリ更新を

メインサーバーからプルすることが必要な場合もあります。

レプリケーションでは、2 つの複製されたリポジトリ上の同じ

ブランチで同時に変更が行われた状況を解決できます。

- プッシュ: 送信している変更より新しい変更があるリポジトリにデータを

プッシュしようとした場合、システムは最新の変更をプルするよう求め、

マージ操作を解決し、最後にプッシュを再試行します。

- プル: 変更セットをリモートブランチからプルするたびに、親変更セットに

正しくリンクされます。プルした変更セットが

ブランチ内の最新の変更セットの子でない場合、

マルチヘッドシナリオが出現します。ブランチには複数の「ヘッド」(ブランチ上の最後の

変更セット) が存在するようになります。再度プッシュする前に 2 つの「ヘッド」をマージ

する必要があります。

プッシュは 2 つのモードで動作します:

1) サーバー間の直接通信: 起点サーバーは、データを同期先サーバーに

送信し、指定されたブランチのデータを自動的に同期します。

2) パッケージのエクスポートモード: クライアントはソースにのみ接続し、

指定されたブランチのデータとメタデータの両方を取得するレプリケーションパッケージを

生成します。「--^package」修飾子が使用されます。

どちらのモードでも、コマンドを実行しているユーザーは、client.conf

ファイルのデフォルト認証を使用するか、「--^authmode」および「--^authdata」

修飾子を指定して、サーバーによって認証される必要があります。

^push レプリケーションは直接的な方法で動作します。実行されると、コマンドは、

(プルで行われるのと同様に) ソースに接続して選択されたブランチを取得するよう

同期先リポジトリに求める代わりに、選択されたブランチをソースから同期先に

複製します。

例:

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

(「main」ブランチを「myserver」から「remoteserver」に複製します。この場合、

両方のサーバーが同じ認証モードで設定されます。)

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

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

(前と同じブランチを複製しますが、今回はリモートサーバーが Active Directory で

ユーザーを認証するように設定されます。たとえば、Linux マシンから、

Active Directory 統合モードを使用するように設定されている Windows サーバーに

接続しています。自分の Active Directory ユーザーと暗号化された

パスワードを指定してサーバーに LDAP として渡します。)

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

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

(同じブランチを複製しますが、今回はユーザーがリモートサーバー上で認証され、

Plastic SCM に含まれるユーザー/パスワードデータベースを

利用します。)

== CMD\_DESCRIPTION\_CLONE ==

リモートリポジトリのクローンを作成します。

== CMD\_USAGE\_CLONE ==

使用方法:

cm ^clone <ソースリポジトリ指定> [<同期先リポジトリ指定> | <同期先リポジトリサーバー指定>]

[--^user=<ユーザー名> [--^password=<パスワード>] | 認証オプション]

[変換オプション]

(直接リポジトリ間クローン。)

cm ^clone <ソースリポジトリ指定> --^package=<パッケージファイル>

[--^user=<ユーザー名> [--^password=<パスワード>] | 認証オプション]

(中間パッケージにクローンを作成します。これは、同期先リポジトリへのプルを

使用して後でインポートできます。)

ソースリポジトリ指定 クローン操作のソースリポジトリ。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

同期先リポジトリ指定 クローン操作の同期先リポジトリ。

存在する場合は空である必要があります。存在しない場合は

作成されます。

指定されていない場合、コマンドではユーザーの

デフォルトリポジトリサーバーが使用されます。

(「cm ^help ^objectspec」を使用してリポジトリ指定の詳細を

確認できます。)

同期先リポジトリサーバー指定 クローン操作の同期先リポジトリサーバー。

同期先リポジトリサーバーに

<ソースリポジトリ指定>と同じ名前のリポジトリがある場合、

それは空である必要があります。存在しない場合は作成されます。

指定されていない場合、コマンドではユーザーの

デフォルトリポジトリサーバーが使用されます。

(「cm ^help ^objectspec」を使用してリポジトリサーバー指定の

詳細を確認できます。)

オプション:

--^user、--^password 認証モードがソースと宛先で異なり、同期先を

認証するプロファイルがない場合に

使用する資格情報。

--^package 指定されたリポジトリをリポジトリではなくパッケージ

ファイルにエクスポートします。

直接ネットワーク接続がない場合にサーバー間でデータを

移動するのに役立ちます。

結果のパッケージは pull コマンドを使用して

インポートする必要があります。

変換オプション 詳細については、「変換オプション」セクションを参照してください。

認証オプション 詳細については、「認証オプション」セクションを

参照してください。

変換オプション:

--^trmode=(^copy|^name|^table --^trtable=<変換テーブルファイル>)

ソースと同期先のリポジトリで異なる認証モードを使用できます。

「--^trmode」オプションでは、ユーザー名をソースから同期先に変換する

方法を指定します。「--^trmode」は次のいずれかの値である必要が

あります:

^copy (デフォルト)。ユーザー識別子が単純にコピーされることを意味します。

^name ユーザー識別子が名前で照合されます。

^table オプション「--^trtable」で指定された変換テーブルを使用します

(下記を参照)。

--^trtable=<変換テーブルファイル>

変換モードが「テーブル」の場合、変換テーブルは <古い名前;新しい名前>

の形式の行を (1 行あたり 1 つ) 含むファイルです。ブランチが

同期先リポジトリに書き込まれる場合、ソースリポジトリ内の「古い名前」で

識別されるユーザーは、同期先の「新しい名前」のユーザーに

設定されます。

認証オプション:

認証データは、次の 2 つのモードのいずれかを使用して指定できます:

1) 認証パラメーターを使用: --^authmode=<モード> --^authdata=<データ>

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

例:

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

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

「--^authdata」行は、client.conf ファイルと profiles.conf ファイル内の

<^SecurityConfig> エントリの内容です。profiles.conf

ファイルは、Plastic GUI の「環境設定」の下の「レプリケーションプロファイル」タブから

簡単に生成できます。

^UPWorkingMode を使用している場合は、次のように指定できます:

--^authmode=^UPWorkingMode --^user=<ユーザー> --^password=<パスワード>

2) 接続先のサーバーごとに異なるファイルがある可能性がある認証ファイルで、

そのサーバーの資格情報を含んでいます。

--^authfile=<認証ファイル>

ファイルには 2 行が含まれます:

行 1) 「--^authmode」で説明しているモード

行 2) 「--^authmode」で説明している認証データ

== CMD\_HELP\_CLONE ==

備考:

「clone」コマンドでは、ソースリポジトリから同期先リポジトリにブランチを

(変更セット、ラベル、属性、レビューなどとともに)

複製できます。リポジトリは異なるサーバーに配置できます。

同期先リポジトリは事前に作成できますが、以前のデータが含まれている場合は

クローン操作が失敗します。

クローン操作では、リポジトリのサブモジュールや Xlink 下のリポジトリのクローンは

作成されません。

例:

cm ^clone awesomeProject@tardis@cloud

(「tardis@cloud」組織から同じ名前のローカルリポジトリに「awesomeProject」リポジトリの

クローンを作成します。)

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

(「server.home:9095」からユーザーのデフォルトリポジトリサーバーの「repo-local」に

「repo」のクローンを作成します。)

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

(「192.168.111.130:8084」から「project@192.168.111.200:9095」に

「project」リポジトリのクローンを作成します。)

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

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

(リモートリポジトリに対する認証ファイルを使用し、指定された変換テーブルに

従ってユーザーを変換して、「project」リポジトリのクローンを「ldapserver:8084」

から作成します。

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

(「project」リポジトリのクローンを「server.home:9095」から

パッケージ「project.plasticpkg」に作成します。これは後で「mordor.home:8084」にある

「project」リポジトリにプルすることでインポートされます。

== CMD\_DESCRIPTION\_REVERT ==

項目を前のリビジョンに戻します。

== CMD\_USAGE\_REVERT ==

使用方法:

cm ^revert <リビジョン指定>

リビジョン指定 コンテンツがワークスペースにロードされるリビジョンを

含む変更セットの指定。

(「cm ^help ^objectspec」を使用してリビジョン指定の詳細を

確認できます。)

== CMD\_HELP\_REVERT ==

備考:

項目がチェックインされている必要があります。

例:

cm ^revert dir#^cs:0

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

== CMD\_DESCRIPTION\_REVISION\_HISTORY ==

ファイルまたはディレクトリの履歴を表示します。

== CMD\_USAGE\_REVISION\_HISTORY ==

使用方法:

cm ^history | ^hist <項目パス>[ ...][--^long | --^format=<文字列形式>]

[--^symlink] [--^xml[=<出力ファイル>]] [--^encoding=<名前>]

項目パス 項目パス。空白を使用してパスを区切ります。空白が

含まれるパスを指定するには二重引用符 (" ") を使用します。

パスをサーバーパスリビジョンにすることもできます。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

オプション:

--^long 追加情報を表示します。

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

--^symlink 履歴操作をターゲットではなくシンボリックリンクに

適用します。

--^xml 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。

--^encoding 「--^xml」オプションとともに使用され、XML 出力で使用する

エンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

== CMD\_HELP\_REVISION\_HISTORY ==

備考:

このコマンドは、指定された項目のリビジョンのリストと、各リビジョンのラベル、

ブランチ、およびコメント情報を表示します。

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} | {^date} 日付。

{1} | {^changesetid} 変更セット番号。

{2} | {^branch} ブランチ。

{4} | {^comment} コメント。

{5} | {^owner} 所有者。

{6} | {^id} リビジョン ID。

{7} | {^repository} リポジトリ。

{8} | {^server} サーバー。

{9} | {^repspec} リポジトリ指定。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

例:

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

cm ^hist c:\workspace --^long

(すべての情報を表示します。)

cm ^history リンク --^symlink

(ターゲットにではなくリンクファイルに履歴操作を適用します。

UNIX 環境で利用できます。)

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

(指定されたブランチのサーバーパスからリビジョン履歴を取得します。)

== CMD\_DESCRIPTION\_REVISION\_TREE ==

項目のリビジョンツリーを表示します。

== CMD\_USAGE\_REVISION\_TREE ==

使用方法:

cm ^tree <パス> [--^symlink]

パス 項目パス。

オプション:

--^symlink 操作をターゲットではなくリンクファイルに適用します。

== CMD\_HELP\_REVISION\_TREE ==

例:

cm ^tree fichero1.txt

cm ^tree c:\workspace

cm ^tree リンク --^symlink

(ターゲットにではなくリンクファイルに操作を適用します。UNIX 環境で

有効です。)

== CMD\_DESCRIPTION\_RM ==

ユーザーにファイルとディレクトリの削除を許可します。

== CMD\_USAGE\_RM ==

使用方法:

cm ^remove | ^rm <コマンド> [オプション]

コマンド:

^controlled (オプション)

^private

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^remove <コマンド> --^usage

cm ^remove <コマンド> --^help

== CMD\_HELP\_RM ==

例:

cm ^remove \path\controlled\_file.txt

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

== CMD\_DESCRIPTION\_RM\_CONTROLLED ==

ファイルまたはディレクトリをバージョン管理から削除します。

== CMD\_USAGE\_RM\_CONTROLLED ==

使用方法:

cm ^remove | ^rm <項目パス>[ ...][--^format=<文字列形式>]

[--^errorformat=<文字列形式>] [--^nodisk]

項目パス 削除する項目パス。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを

区切ります。

オプション:

--^format 出力の進捗メッセージを特定の形式で

取得します。詳細については、「例」を参照してください。

--^errorformat エラーメッセージ (ある場合) を特定の形式で

取得します。詳細については、「例」を参照してください。

--^nodisk バージョン管理から項目を削除しますが、ディスク上には

保持します。

== CMD\_HELP\_RM\_CONTROLLED ==

備考:

項目はディスクから削除されます。削除された項目はソースコード管理の親

ディレクトリから削除されます。

要件:

- 項目がソースコード管理の対象になっている必要があります。

stdin から入力を読み取る:

「^remove」コマンドは stdin からパスを読み取ることができます。これを行うには、シングル

ダッシュ「-」を渡します。

例: cm ^remove -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用して削除するファイルを指定できます。

例:

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

(Windows で、ワークスペース内のすべての .c ファイルを削除します。)

例:

cm ^remove src

(「src」を削除します。src がディレクトリの場合、これは

「cm ^remove -^R src」と同じです。)

cm ^remove c:\workspace\file.txt --^format="{0} - 削除済み" \

--^errorformat="{0} - 削除エラー"

(バージョン管理とディスクから「file.txt」を削除し、

「c:\workspace\file.txt - ^REMOVED」(操作に成功した場合) または

「c:\workspace\file.txt " - ^ERROR ^REMOVING」(それ以外の場合) を書き込みます。)

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

(「file.txt」をバージョン管理から削除しますが、ディスク上には保持します。)

== CMD\_DESCRIPTION\_RM\_PRIVATE ==

非公開ファイルまたはディレクトリを削除します。

警告: コマンドを使用して削除されたファイルは完全に消去され、

復元できません。「--^dry-run」オプションを使用して、コマンドの影響を受けるファイルを

確認することをお勧めします。

== CMD\_USAGE\_RM\_PRIVATE ==

使用方法:

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

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

パス 削除するファイルまたはディレクトリのパス。空白が含まれる

パスを指定するには二重引用符 (" ") を使用します。空白を

使用してパスを区切ります。

オプション:

--^r 管理対象ディレクトリから非公開ファイルを再帰的に

削除します。

--^ignored 無視およびクロークされたファイルとディレクトリも削除します。

--^verbose 影響を受けるすべてのパスを出力します。

--^dry-run ディスクに変更を加えずにコマンドを実行します。

== CMD\_HELP\_RM\_PRIVATE ==

備考:

パスが非公開のファイルまたはディレクトリである場合は、ディスクから削除されます。

パスが管理対象ファイルの場合、コマンドは失敗します。

パスが管理対象ディレクトリの場合、「-^r」オプションを指定しない限り

コマンドは失敗します。その場合、指定されたディレクトリ下にある非公開のファイルと

ディレクトリはすべて削除されます。

例:

cm ^remove ^private private\_directory

(「private\_directory」を削除します。)

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

(「controlled\_directory」は非公開ではないため失敗します。)

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

(「controlled\_directory」下にある非公開のファイルとディレクトリをすべて削除します。)

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

(「controlled\_directory」下にある非公開ファイルの削除の影響を受ける

すべてのパスを表示します。実際には何も削除しません。)

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

(「controlled\_directory」下にある非公開ファイルの削除の影響を受けるすべての

パスを表示し、削除を実行します。)

== CMD\_DESCRIPTION\_TRIGGER\_DELETE ==

トリガーを削除します。

== CMD\_USAGE\_TRIGGER\_DELETE ==

使用方法:

cm ^trigger | ^tr ^delete | ^rm <サブタイプのタイプ> <位置番号>

[--^server=<リポジトリサーバー指定>]

サブタイプのタイプ トリガー実行とトリガー操作。

トリガータイプのリストを表示するには「cm ^showtriggertypes」

と入力します。

位置番号 作成時にトリガーに割り当てられた位置。

オプション:

--^server 指定されたサーバー上のトリガーを削除します。

サーバーが指定されていない場合は、クライアントに設定されている

サーバーでコマンドを実行します。

== CMD\_HELP\_TRIGGER\_DELETE ==

例:

cm ^trigger ^delete ^after-setselector 4

cm ^tr ^rm ^after-setselector 4

== CMD\_DESCRIPTION\_ATTRIBUTE\_SET ==

指定されたオブジェクトに属性を設定します。

== CMD\_USAGE\_ATTRIBUTE\_SET ==

使用方法:

cm ^attribute | ^att ^set <属性指定> <オブジェクト指定> <属性値>

属性指定 属性の指定。(「cm ^help ^objectspec」を

使用して属性指定の詳細を確認できます。)

オブジェクト指定 属性を設定するオブジェクトの指定。

属性は、ブランチ、変更セット、シェルブセット、

ラベル、項目、リビジョンに対して設定できます。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

属性値 オブジェクトに設定する属性値。

== CMD\_HELP\_ATTRIBUTE\_SET ==

備考:

属性をオブジェクトに設定して、このオブジェクトの追加情報を保存できます。

属性は、ブランチ、変更セット、シェルブセット、ラベル、項目、リビジョンに

対して設定できます。

例:

cm ^attribute ^set ^att:status ^br:/main/SCM105 未処理

(属性「ステータス」を「未処理」という値でブランチ「SCM105」に設定します。)

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

(属性「統合済み」をリポジトリ「reptest」のラベル「LB008」に

値「yes」で設定します。)

== CMD\_DESCRIPTION\_SETOWNER ==

オブジェクトの所有者を設定します。

== CMD\_USAGE\_SETOWNER ==

使用方法:

cm ^setowner | ^sto --^user=<ユーザー名> | --^group=<グループ> <オブジェクト指定>

--^user ユーザー名。オブジェクトの新しい所有者。

--^group グループ名。オブジェクトの新しい所有者。

オブジェクト指定 新しい所有者を設定するオブジェクトの指定。

所有者は次のオブジェクトに対して設定できます:

リポジトリサーバー、リポジトリ、ブランチ、変更セット、

ラベル、項目、リビジョン、属性。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

== CMD\_HELP\_SETOWNER ==

備考:

オブジェクトの所有者はユーザーまたはグループである可能性があります。

所有者は、リポジトリサーバー、リポジトリ、ブランチ、変更セット、

ラベル、項目、リビジョン、属性に対して設定できます。

例:

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

(リポジトリサーバーの所有者として「john」を設定します。)

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

(「development」グループを「mainRep」リポジトリの所有者として設定します。)

== CMD\_DESCRIPTION\_SETSELECTOR ==

セレクターをワークスペースに設定します。

== CMD\_USAGE\_SETSELECTOR ==

このコマンドは非推奨になりました。これは後方互換性のためにまだ存在していますが、

Plastic SCM 4.0 ではセレクターの大部分が非推奨になりました。セレクターは

作業中のブランチや変更セットを指定するためにまだ存在していますが、パスをフィルタリングするための

古いルールはサポートされなくなりました。

使用方法:

cm ^setselector | ^sts [--^file=<セレクターファイル>] [--^ignorechanges]

[--^forcedetailedprogress] [<ワークスペースパス> | <ワークスペース指定>]

オプション:

--^file セレクターのロード元のファイル。

--^ignorechanges ワークスペースの更新時に検出された保留中の変更がある

場合に表示される、保留中の変更の警告メッセージを

無視します。

--^forcedetailedprogress 標準出力がリダイレクトされた場合でも、詳細な進捗情報を

強制的に出力します。

ワークスペースパス セレクターを設定するワークスペースのパス。

ワークスペース指定 ワークスペースの指定。(「cm ^help ^objectspec」

を使用してワークスペース指定の詳細を確認できます。)

== CMD\_HELP\_SETSELECTOR ==

備考:

このコマンドは、ワークスペースのセレクターを設定します。

ワークスペースには、リポジトリからリビジョンをロードするための情報が必要です。

Plastic SCM では、この情報を取得するためにセレクターを使用します。

セレクターを使用すると、指定したブランチ、ラベル、または変更セットから

リビジョンをロードできます。

セレクターをロードするファイルが指定されていない場合は、デフォルトのオペレーティング

システムエディターが実行されます。

セレクターの例:

^repository "^default" // 作業リポジトリ

^path "/" // ルートディレクトリにルールが適用される

^branch "/^main" // ^br:/^main から最新のリビジョンを取得する

^checkout "/^main" // ブランチ ^br:/^main にチェックアウトを配置する

例:

cm ^sts

(適用される現在のセレクターファイルを開きます。)

cm ^sts ^wk:workspace\_projA@reptest

(適用される指定されたセレクターファイルを開きます。)

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

(現在のワークスペース内で指定されたセレクターファイルを設定します。)

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

(選択されたワークスペース内で指定されたセレクターファイルを設定します。)

== CMD\_DESCRIPTION\_SHELVE ==

チェックアウト済み項目のコンテンツをシェルブします。

== CMD\_USAGE\_SHELVE ==

このコマンドは非推奨になりました。代わりに 'cm ^shelveset' を使用してください。

使用方法:

cm ^shelve [<項目パス>+] [--^all] [--^dependencies]

[-^c=コメント文字列 | -^commentsfile=<コメントファイル>]

[--^encoding=名前] [--^comparisonmethod=比較方法]

(コンテンツをシェルブします。)

cm ^shelve --^apply=<シェルブ指定> [--^mount]

(格納されているシェルブセットを適用します。)

--^apply 指定されたシェルブセットのシェルブされたコンテンツを復元します。

シェルブ指定については「cm ^help ^objectspec」を確認してください。

cm ^shelve --^delete=<シェルブ指定>

(格納されているシェルブセットを削除します。)

--^delete 指定されたシェルブセットを削除します。

シェルブセット指定については「cm ^help ^objectspec」を確認してください。

オプション:

項目パス スペースで区切られた、シェルブされる項目。空白が含まれる

パスを指定するには二重引用符 (") を使用してします。

--^all 指定されたパスでローカルに変更、移動、および削除された

項目も含めます。

--^dependencies シェルブする項目にローカルの変更の依存関係を

含めます。

-^c 指定されたコメントを作成されたシェルブセットに適用します。

-^commentsfile 指定されたファイル内のコメントを作成されたシェルブセットに

適用します。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^comparisonmethod 比較方法を設定します。詳細については、「備考」を参照してください。

--^mount 指定されたリポジトリのマウントポイント。

== CMD\_HELP\_SHELVE ==

備考:

<項目パス> もオプションも指定されていない場合、シェルブにはそのワークスペース内の

すべての保留中の変更が関与します。

シェルブ操作は常に指定されたパスから再帰的に適用されます。

項目をシェルブするための要件:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックアウト済みまたは変更済みである必要があります (--^all オプションを使用する必要があります)。

比較方法:

^ignoreeol 行の終わりの差異を無視します。

^ignorewhitespaces 空白の差異を無視します。

^ignoreeolwhitespaces 行の終わりと空白の差異を無視します。

^notignore 行の終わりと空白の差異を検出します。

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。

例:

cm ^shelve -^c="my comment"

(現在のワークスペース内のすべての保留中の変更を、コメントを含めて

シェルブします。)

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

(選択された保留中の変更をシェルブし、commentshelve.txt ファイル内の

コメントを適用します。)

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

(格納されているシェルブセットを適用します。)

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

(格納されているシェルブセットを削除します。)

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

(クライアントの変更リストをシェルブします。

上のコマンドは、「pending\_to\_review」という名前の変更リスト内のパスを

リストします。このパスのリストは、シェルブコマンドの入力に

リダイレクトされます。)

== CMD\_DESCRIPTION\_SHELVESET ==

ユーザーにシェルブセットの管理を許可します。

== CMD\_USAGE\_SHELVESET ==

使用方法:

cm ^shelveset <コマンド> [オプション]

コマンド:

^create | ^mk

^delete | ^rm

^apply

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^shelveset <コマンド> --^usage

cm ^shelveset <コマンド> --^help

== CMD\_HELP\_SHELVESET ==

例:

cm ^shelveset ^create -^c="my comment"

cm ^shelveset ^delete ^sh:3

cm ^shelve ^apply ^sh:3

== CMD\_DESCRIPTION\_SHELVESET\_CREATE ==

保留中の変更をシェルブします。

== CMD\_USAGE\_SHELVESET\_CREATE ==

使用方法:

cm ^shelveset ^create | ^mk [<項目パス>[ ...]] [--^all] [--^dependencies]

[-^c=<コメント文字列> | -^commentsfile=<コメントファイル>]

オプション:

項目パス シェルブする項目。空白を使用してユーザー名を区切ります。

空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

--^all 指定されたパスでローカルに変更、移動、および削除された

項目も含めます。

--^dependencies シェルブする項目にローカルの変更の依存関係を

含めます。

-^c 指定されたコメントを作成されたシェルブに適用します。

-^commentsfile 指定されたファイル内のコメントを作成されたシェルブに

適用します。

== CMD\_HELP\_SHELVESET\_CREATE ==

「^shelveset ^create」コマンドは、チェックアウト済み項目のコンテンツをリポジトリ内に

格納します。これにより、ファイルをチェックインしなくてもコンテンツが保護

されます。

備考:

<項目パス> もオプションも指定されていない場合、シェルブセットには

そのワークスペース内のすべての保留中の変更が含まれます。

「^shelveset ^create」操作は常に指定されたパスから再帰的に

適用されます。

項目をシェルブするための要件:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックアウト済みまたは変更済みである必要があります (「--^all」オプションを使用する必要があります)。

PLASTICEDITOR 環境変数を設定して、コメントを入力するエディターを

指定します。

例:

cm ^shelveset ^create -^c="my comment"

(現在のワークスペース内のすべての保留中の変更を、コメントを含めて

シェルブします。)

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

(選択された保留中の変更をシェルブし、「commentshelve.txt」ファイル内の

コメントを適用します。なお、「^create」はデフォルトのサブコマンドです。)

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

(クライアントの変更リストをシェルブします。

上のコマンドは、「pending\_to\_review」という名前の変更リスト内の

パスをリストします。このパスのリストは、「^shelveset」コマンドの入力にリダイレクト

されます。)

== CMD\_DESCRIPTION\_SHELVESET\_DELETE ==

シェルブセットを削除します。

== CMD\_USAGE\_SHELVESET\_DELETE ==

使用方法:

cm ^shelveset ^delete | ^rm <シェルブ指定>

シェルブ指定 シェルブセットの指定。(「cm ^help ^objectspec」を使用して

シェルブセット指定の詳細を確認できます。)

== CMD\_HELP\_SHELVESET\_DELETE ==

「^shelveset ^delete」コマンドは、シェルブセットを削除します。

例:

cm ^shelveset ^delete ^sh:3

(格納されているシェルブセットを削除します。)

== CMD\_DESCRIPTION\_SHELVESET\_APPLY ==

格納されているシェルブセットを適用します。

== CMD\_USAGE\_SHELVESET\_APPLY ==

使用方法:

cm ^shelveset ^apply <シェルブ指定> [--^mount] [--^encoding=<名前>]

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

^ignoreeolwhitespaces | ^notignore)]

シェルブ指定 シェルブセットの指定。(「cm ^help ^objectspec」を使用して

シェルブセット指定の詳細を確認できます。)

オプション:

--^mount 指定されたリポジトリのマウントポイント。

--^encoding 出力のエンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^comparisonmethod 比較方法を設定します。詳細については、「備考」を参照してください。

== CMD\_HELP\_SHELVESET\_APPLY ==

「^shelveset ^apply」コマンドは、格納されているシェルブセットのコンテンツを復元します。

備考:

比較方法:

^ignoreeol 行の終わりの差異を無視します。

^ignorewhitespaces 空白の差異を無視します。

^ignoreeolwhitespaces 行の終わりと空白の差異を無視します。

^notignore 行の終わりと空白の差異を検出します。

例:

cm ^shelveset ^apply ^sh:3

(格納されているシェルブを適用します。)

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

オブジェクトと属性をリストします。

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

使用方法:

cm ^showfindobjects

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

利用可能なオブジェクトと属性:

^attribute:

次のフィールドを使用してフィルタリングすることで、属性を検索できます:

^type : 文字列。

例:

cm ^find ^attribute "^where ^type = 'ステータス'"

(タイプが「ステータス」であるすべての属性を検索します。)

^value : 文字列。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

例:

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

(今週中に適用されたすべての属性を検索します。)

^owner : ユーザー。

特別なユーザー「^me」を許可します。

例:

cm ^find ^attribute "^where ^value = '解決済み' ^and ^owner = '^me'"

(自分が適用した、「解決済み」という値を持つすべての属性を検索します。)

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^comment : 文字列。

^srcobj : オブジェクト指定: 項目パス、ブランチ、変更セット、リビジョン、またはラベル。

「cm ^help ^objectspec」を使用してこれらのオブジェクトの指定方法を確認できます。

例:

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

(項目「readme.txt」に適用された属性を検索します。)

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

(ブランチ scm23343 に適用された属性を検索します。)

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

(指定されたリビジョンに適用された属性を検索します。)

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

(指定された ID に適用された属性を検索します。)

^ID : 整数。

^attributetype:

次のフィールドを使用してフィルタリングすることで、属性タイプを検索できます:

^name : 文字列。

例:

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

(名前が「st」で始まるすべての属性を検索します。)

^value : 文字列。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

例:

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

(今日適用されたすべての属性を検索します。)

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^comment : 文字列。

例:

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

(コメントがあるすべての属性タイプを検索し、出力を XML 形式で

標準出力に出力します。)

^source : オブジェクト指定: 項目パス、ブランチ、変更セット、またはラベル。

「cm ^help ^objectspec」を使用してこれらのオブジェクトの指定方法を確認できます。

例:

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

(項目「readme.txt」内のすべての属性タイプを検索します。)

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

(変更セット「30」内のすべての属性タイプを検索します。)

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

(ラベル「v0.14.1」内のすべての属性タイプを検索します。)

^ID : 整数。

レプリケーションフィールド。下の「レプリケーション関連フィールド」を確認してください。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^branch:

次のフィールドを使用してフィルタリングすることで、ブランチを検索できます:

^name : 文字列。

例:

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

(名前が「scm23」で始まるブランチを検索します。)

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

例:

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

(先週中に作成されたブランチを検索します。)

^changesets : 日付 (ブランチ内の変更セットの)。

詳細については、このガイドの「日付定数」を確認してください。

例:

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

(今日作成された変更セットを含むブランチを検索します。)

^attribute : 文字列。

^attrvalue : 文字列。

例:

cm ^find ^branch "^where ^attribute = 'ステータス' ^and ^attrvalue = '失敗'"

(属性「ステータス」を持ち、その値が

「失敗」であるブランチを検索します。)

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^parent : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

例:

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

(自分以外によって作成され、親ブランチが

「/main」ではないブランチを検索します。)

^comment : 文字列。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^ID : 整数。

例:

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

(ID が 2029607 であるブランチを検索します。)

レプリケーションフィールド。下の「レプリケーション関連フィールド」を確認してください。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^changeset:

次のフィールドを使用してフィルタリングすることで、変更セットを検索できます:

^branch : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を

確認できます。

例:

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

(ブランチ 'scm23119' 内のすべての変更セットを検索します。)

^changesetid : 整数。

^attribute : 文字列。

例:

cm ^find ^changeset "^where ^attribute = 'ステータス'"

(属性 'ステータス' を持つ変更セットを検索します。)

^attrvalue : 文字列。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

^owner : ユーザー。

特別なユーザー「^me」を許可します。

例:

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

(作成日付が 2018/8/6 以降で、自分以外によって

作成されたすべての変更セットを検索します。)

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

例:

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

(指定された GUID を持つ変更セットを検索します。)

^comment : 文字列。

例:

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

(コメントがない変更セットを検索します。)

^onlywithrevisions : ブール値。

変更セットにリビジョンがあるかどうかをフィルタリングします。

例:

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

(リビジョンがない変更セットを検索します。)

^returnparent : ブール値。

変更セットの親を返す方法です。スクリプトに適しています。

例:

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

(変更セット 29 の親を検索します。)

^parent : 変更セット ID (整数)。

例:

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

(親が変更セット 548 であるすべての変更セットを検索します。)

^ID : 整数。

レプリケーションフィールド。下の「レプリケーション関連フィールド」を確認してください。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^label:

次のフィールドを使用してフィルタリングすることで、ラベルを検索できます:

^name : 文字列。

例:

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

(「7.0.16.」で始まる名前のラベルを検索します。)

^attribute : 文字列。

^attrvalue : 文字列。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

例:

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

^attribute = 'パブリッシュステータス' ^and ^attrvalue != 'パブリッシュ済み'"

(属性 'パブリッシュステータス' が「パブリッシュ済み」以外の値に設定されている

今月作成されたラベルを検索します。)

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^branch : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

例:

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

(メインブランチに適用されたすべてのラベルを検索します。)

^branchid : 整数。

^changeset : 変更セット ID (整数)。

例:

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

(変更セット 111733 に適用されたすべてのラベルを検索します。)

^comment : 文字列。

^ID : 整数。

レプリケーションフィールド。下の「レプリケーション関連フィールド」を確認してください。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^merge:

次のフィールドを使用してフィルタリングすることで、マージを検索できます:

^srcbranch : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

例:

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

(メインブランチからマージを検索します。)

^srcchangeset : 変更セット ID (整数)。

^dstbranch : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

^dstchangeset : 変更セット ID (整数)。

例:

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

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

(変更セット 108261 へのマージを検索し、マージ元 (ブランチと

変更セット ID)、マージ先 (ブランチと変更セット ID)、およびマージ所有者

を示す書式付き出力を出力します。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^type : 文字列。

使用可能な値は、「^merge」、「^cherrypick」、

「^cherrypicksubstractive」、「^interval」、「^intervalcherrypick」、

および「^intervalcherrypicksubstractive」です

例:

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

(自分のすべてのチェリーピックを検索します。)

^ID : 整数。

^replicationlog:

次のフィールドを使用してフィルタリングすることで、レプリケーションログを検索できます:

^branch : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

例:

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

(ブランチ「gm22358」のレプリケーションログを検索します。)

^repositoryname : 文字列。

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

^server : 文字列。

^package : ブール値。

例:

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

(サーバー名に「cloud」が含まれるパッケージから作成された

レプリケーションログを検索します。)

^ID : 整数。

^review:

次のフィールドを使用してフィルタリングすることで、コードレビューを検索できます:

^status : 文字列。

^assignee : 文字列。

例:

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

(自分のすべての保留中のレビューを検索します。)

^title : 文字列。

^target : オブジェクト指定: ブランチまたは変更セット。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

例:

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

(ブランチ「scm17932」に関連するレビューを検索します。)

^targetid : 整数。

^targettype : 文字列。

使用可能な値は「^branch」および「^changeset」です。

例:

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

(ターゲットタイプが変更セットであるレビューを検索します。)

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^ID : 整数。

^revision:

次のフィールドを使用してフィルタリングすることで、リビジョンを検索できます:

^branch : ブランチ指定。

「cm ^help ^objectspec」を使用してこのオブジェクトの指定方法を確認できます。

^changeset : 変更セット ID (整数)。

例:

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

(変更セット 111756 以降で作成されたリビジョンを

検索します。)

^item : 文字列または整数。

^itemid : 整数。

例:

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

(項目「readme.txt」と項目 ID 2250 のリビジョンを

検索します。)

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

(前の例と同じリビジョンを取得します。)

^attribute : 文字列。

^attrvalue : 文字列。

例:

cm ^find ^revision "^where ^attribute = 'ステータス' ^and ^attrvalue != '未処理'"

(属性 'ステータス' を持ち、その値が '未処理' 以外である

リビジョンを検索します。)

^archived : ブール値。

例:

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

(外部ストレージにアーカイブされたリビジョンを

検索します。)

^comment : 文字列。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^parent : リビジョン ID (整数)。

^returnparent : ブール値。

^shelve : シェルブ ID (整数)。

^size : 整数 (バイト単位)。

^type : 文字列。

使用可能な値は「^dir」、「^bin」、および「^txt」です。

例:

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

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

(自分が 2 か月前に作成し、サイズが約 3MB を超える

テキストリビジョンを検索します。)

^workspacecheckoutid : 整数。

^ID : 整数。

レプリケーションフィールド。下の「レプリケーション関連フィールド」を確認してください。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

^shelve:

次のフィールドを使用してフィルタリングすることで、シェルブを検索できます:

^owner : ユーザー。

特別なユーザー「^me」を許可します。

^date : 日付。

詳細については、このガイドの「日付定数」を確認してください。

例:

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

(昨年中に自分以外によって作成されたシェルブを

検索します。)

^attribute : 文字列。

^attrvalue : 文字列。

^comment : 文字列。

^GUID : グローバル一意識別子。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の 16 進数 ID です。

^parent : 整数。

^shelveid : 整数。

例:

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

(シェルブ 2 を検索します。)

^ID : 整数。

例:

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

(オブジェクト ID が 3848 より大きいシェルブを検索します。)

レプリケーションフィールド。下の「レプリケーション関連フィールド」を確認してください。

^ReplLogId

^ReplSrcDate

^ReplSrcId

^ReplSrcRepository

^ReplSrcServer

レプリケーション関連フィールド:

多くのオブジェクトでレプリケーションデータが追跡されます。つまり、レプリケーションが最初に作成された場所が

Plastic によって追跡されます。

次のフィールドを使用できます:

^ReplSrcServer : リポジトリ指定。「レプリケーション元サーバー」を表します。

オブジェクトのレプリケーション元のサーバーです。

例:

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

(サーバー「skull」からレプリケートされたブランチを検索します。)

^ReplSrcRepository : 文字列。「レプリケーション元リポジトリ」を表します。これは

オブジェクトのレプリケーション元のリポジトリです。

例:

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

^and ^replsrcrepository = 'codice'"

(サーバー「skull」およびリポジトリ「codice」からレプリケートされた

ブランチを検索します。)

^ReplLogId : 整数。レプリケーション操作の ID。Plastic では、

レプリカから新しいオブジェクトが作成されるたびに

新しい「replicationlog」が作成されます。

例:

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

(次のレプリカからレプリケートされたリビジョンを検索します:

2019974.)

^ReplSrcDate : 日付。これはレプリケーションが実際に行われた

日付です。

レプリケートされたオブジェクトでは、その作成日付

が保持されます。このフィールドは、特定の

時間枠内にレプリケートされたオブジェクトを検索する

場合に役立ちます。

例:

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

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

(15 日前に作成され、1 か月前にレプリケートされた

ラベルを検索します。)

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

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

(1 週間前に作成されたレプリケーションログを検索します。)

次に、レプリケートされたブランチがレプリケートされる前に

作成されたことを確認できます:

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

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

^ReplSrcId : 整数。これはレプリケーション元サーバーの ID です。

この ID を調べるには、

「cm ^find」コマンドで「^replicationsource」オブジェクトを検索します。

例:

cm ^find ^replicationsource

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

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

次に、codice@AFRODITA:8087 からレプリケートされた

変更セットを検索できます:

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

日付定数:

お使いのマシンのローカライズ設定に従った日付形式を使用できます。

たとえば、お使いのコンピューターで日付が「MM-dd-yyyy」の形式で表示される場合、

クエリで「12-31-2019」のような日付を使用できます。

クエリを簡単にするために、次の定数を使用することもできます。

'^today' : 今日の日付。

'^yesterday' : 昨日の日付。

'^this ^week' : 今週の月曜日の日付。

'^this ^month' : 今月の 1 日の日付。

'^this ^year' : 今年の 1 月 1 日の日付。

'^one ^day ^ago' : 現在の日付の 1 日前。

'^one ^week ^ago' : 現在の日付の 7 日前。

'^one ^month ^ago' : 現在の日付の 1 か月前。

'n ^days ^ago' : 現在の日付の「n」日前。

'n ^months ^ago' : 現在の日付の「n」か月前。

'n ^years ^ago' : 現在の日付の「n」年前。

次の「^where」句は、タイプ「^date」のフィールドで有効です:

'(...)^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' (...)'

「cm ^find」コマンドで特定の日付形式を強制的に使用することもできます。そのためには

--^dateformat フラグを使用します。詳細については「cm ^find --^help」を確認してください。

== CMD\_DESCRIPTION\_TRIGGER\_SHOWTYPES ==

利用可能なトリガータイプを表示します。

== CMD\_USAGE\_TRIGGER\_SHOWTYPES ==

使用方法:

cm ^trigger ^showtypes

== CMD\_DESCRIPTION\_SHOWACL ==

オブジェクトの ACL を表示します。

== CMD\_USAGE\_SHOWACL ==

使用方法:

cm ^showacl | ^sa <オブジェクト指定> [--^extended] [--^xml[=<出力ファイル>]]

[--^encoding=<名前>]

オブジェクト指定 ACL を表示するオブジェクトの指定。

このコマンドで有効なオブジェクト:

リポジトリサーバー、リポジトリ、ブランチ、変更セット、ラベル、項目、

属性。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

オプション:

--^extended ACL の階層ツリーを表示します。

--^xml 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。

--^encoding 「--^xml」オプションとともに使用され、XML 出力で使用する

エンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

== CMD\_HELP\_SHOWACL ==

例:

cm ^showacl ^repserver:PlasticServer:8084

(選択されたサーバーの ACL を表示します。)

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

(選択されたブランチ指定の ACL の階層ツリーを表示します。)

== CMD\_DESCRIPTION\_SHOWCOMMANDS ==

使用できるすべてのコマンドを表示します。

== CMD\_USAGE\_SHOWCOMMANDS ==

使用方法:

cm ^showcommands

== CMD\_HELP\_SHOWCOMMANDS ==

== CMD\_DESCRIPTION\_SHOWOWNER ==

オブジェクトの所有者を表示します。

== CMD\_USAGE\_SHOWOWNER ==

使用方法:

cm ^showowner | ^so <オブジェクト指定>

オブジェクト指定 所有者を表示するオブジェクトの指定。

オブジェクトは次のいずれかである必要があります:

リポジトリサーバー、リポジトリ、ブランチ、変更セット、

ラベル、属性、リビジョン、項目。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

== CMD\_HELP\_SHOWOWNER ==

備考:

このコマンドは、オブジェクトの所有者を表示します。所有者はユーザーまたは

グループである可能性があります。所有者は「cm ^setowner」コマンドで変更できます。

例:

cm ^showowner ^repserver:PlasticServer:8084

(選択されたサーバーの所有者を表示します。)

cm ^so ^item:samples\

(選択された項目指定の所有者を表示します。)

== CMD\_DESCRIPTION\_SHOWPERMISSIONS ==

利用できる権限をリストします。

== CMD\_USAGE\_SHOWPERMISSIONS ==

使用方法:

cm ^showpermissions | ^sp

== CMD\_HELP\_SHOWPERMISSIONS ==

例:

cm ^showpermissions

== CMD\_DESCRIPTION\_SHOWSELECTOR ==

ワークスペースセレクターを表示します。

== CMD\_USAGE\_SHOWSELECTOR ==

このコマンドは非推奨になりました。これは後方互換性のためにまだ存在していますが、

Plastic SCM 4.0 ではセレクターの大部分が非推奨になりました。セレクターは

作業中のブランチや変更セットを指定するためにまだ存在していますが、パスをフィルタリングするための

古いルールはサポートされなくなりました。

使用方法:

cm ^showselector | ^ss [<ワークスペースパス> | <ワークスペース指定>]

ワークスペースパス セレクターを表示するワークスペースのパス。

ワークスペース指定 ワークスペースの指定。(「cm ^help ^objectspec」を

使用してワークスペース指定の詳細を確認できます。)

== CMD\_HELP\_SHOWSELECTOR ==

備考:

パスもワークスペースも指定されていない場合、このコマンドは現在のディレクトリを

ワークスペースパスとして使用します。

例:

cm ^showselector c:\workspace

(選択されたワークスペースパスのセレクターを表示します。)

cm ^ss

(現在のワークスペースのセレクターを表示します。)

cm ^showselector > mySelector.txt

(現在のワークスペースのセレクターをファイルに書き込みます。)

cm ^showselector ^wk:mywk@reptest

(リポジトリ「reptest」内のワークスペース「mywk」のセレクターを表示します。)

== CMD\_DESCRIPTION\_SUPPORT ==

ユーザーにサポート関連の操作の実行を許可します。

== CMD\_USAGE\_SUPPORT ==

使用方法:

cm ^support <コマンド> [オプション]

コマンド:

^bundle

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^support <コマンド> --^usage

cm ^support <コマンド> --^help

== CMD\_HELP\_SUPPORT ==

例:

cm ^support

cm ^support ^bundle

cm ^support ^bundle c:\outputfile.zip

== CMD\_DESCRIPTION\_SUPPORT\_BUNDLE ==

関連するログを含む「サポートバンドル」パッケージを作成します。

このファイルを、問い合わせ、追加情報の要求、バグの送信時に

添付できます。

== CMD\_USAGE\_SUPPORT\_BUNDLE ==

使用方法:

cm ^support ^bundle [<出力ファイル>]

オプション:

出力ファイル 指定された場所に「サポートバンドル」パッケージを

作成します。

== CMD\_HELP\_SUPPORT\_BUNDLE ==

備考:

このコマンドは、ユーザーに「サポートバンドル」パッケージの作成を許可します。このパッケージを

問い合わせ、追加情報の要求、バグの送信時に添付できます。

ユーザーはオプションで出力ファイルの場所を指定できます。指定しない場合、

出力ファイルは temp ディレクトリに書き込まれます。

例:

cm ^support ^bundle

(temp ディレクトリに「サポートバンドル」を作成します。)

cm ^support ^bundle c:\outputfile.zip

(指定された場所に「サポートバンドル」を作成します。)

== CMD\_DESCRIPTION\_SWITCH ==

ワークスペースをブランチ、変更セット、ラベル、またはシェルブセットに切り替えます。

== CMD\_USAGE\_SWITCH ==

使用方法:

cm ^switch (<ブランチ指定> | <変更セット指定> | <ラベル指定> | <シェルブ指定>)

[--^workspace=<パス>] [--^repository=<名前>]

[--^forcedetailedprogress]

(「cm ^help ^objectspec」を使用してブランチ、変更セット、ラベル、

シェルブセット指定の詳細を確認できます。)

オプション:

--^workspace ワークスペースが置かれているパス。

--^repository 指定されたリポジトリに切り替えます。

--^forcedetailedprogress 標準出力がリダイレクトされた場合でも、詳細な

進捗情報を強制的に出力します。

== CMD\_HELP\_SWITCH ==

備考:

このコマンドは、ワークスペースツリーを指定されたオブジェクト (ブランチ、ラベル、

シェルブセット、または変更セット) のコンテンツに更新することをユーザーに許可します。

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 ==

ブランチを作業中のブランチとして設定します。

== CMD\_USAGE\_SWITCH\_TO\_BRANCH ==

このコマンドは非推奨になりました。代わりに cm switch を使用してください。

使用方法:

cm ^switchtobranch [オプション] [ブランチ指定]

ブランチ指定: ブランチの指定。

オプション:

--^label=名前 | --^changeset=番号: 指定されたラベルまたは変更セットから

リビジョンをロードします。ブランチ指定が指定されていない場合は、次のオプションの

いずれかが必要です。

--^changeset=変更セット: 指定された変更セットに切り替えます。

--^repository=リポジトリ: 指定されたリポジトリに切り替えます。

--^workspace | -wk=パス: ワークスペースが置かれているパス。

== CMD\_HELP\_SWITCH\_TO\_BRANCH ==

備考:

このコマンドは、ユーザーにブランチでの作業を許可します。

ブランチ指定が指定されていない場合は、ラベルまたは変更セットを指定する必要があります。

リポジトリが指定されていない場合は、ブランチが現在のリポジトリに設定されます。

例:

cm ^switchtobranch ^br:/main

cm ^switchtobranch ^br:/main/task001

cm ^switchtobranch --^label=BL050

(読み取り専用の設定。このコマンドは、ラベル付けされた変更セットのコンテンツを

ロードします。)

== CMD\_DESCRIPTION\_SYNC ==

Git と同期します。

== CMD\_USAGE\_SYNC ==

使用方法:

cm ^synchronize | ^sync <リポジトリ指定> ^git [<URL> [--^user=<ユーザー名> --^pwd=<パスワード>]]

[(--^txtsimilaritypercent | --^binsimilaritypercent | \

--^dirsimilaritypercent)=<値>]

[--^author] [--^skipgitlfs]

リポジトリ指定 リポジトリの指定。(「cm ^help ^objectspec」を

使用してリポジトリ指定の詳細を確認できます。)

git (デフォルト)。

オプション:

URL リモートリポジトリの URL (http(s)://、git://、または

SSH URL)。

--^user 指定された URL のユーザー名。

--^pwd 指定された URL のパスワード。

--^txtsimilaritypercent \

--^binsimilaritypercent \

--^dirsimilaritypercent

移動された項目を検出する方法は、Plastic SCM

GUI と同じです。

--^author git 作成者の名前とタイムスタンプの値を使用します。

(デフォルトでは git コミッター)

--^skipgitlfs .gitattributes ファイル内の Git LFS の設定を

無視します。これは Git LFS がサポートされていない

ように動作します。

== CMD\_HELP\_SYNC ==

備考:

git リポジトリにユーザーとパスワードが必要な場合は、「^url」、「--^user」、

および「--^pwd」オプションを使用します。

git リポジトリにユーザーとパスワードが不要な場合は、最初の同期操作で「^url」

オプションを使用します。次の同期操作では、「^url」

オプションは省略可能です。

SSH プロトコルを使用して同期を実行するには、「ssh」クライアントが PATH 環境変数に

追加され、リモートホストに接続するように正しく設定されている

必要があります (公開/非公開キーが設定されているなど)。

例:

cm ^sync default@localhost:8087 ^git git://localhost/repository

== CMD\_DESCRIPTION\_TRIGGER ==

ユーザーにトリガーの管理を許可します。

== CMD\_USAGE\_TRIGGER ==

使用方法:

cm ^trigger | ^tr <コマンド> [オプション]

コマンド:

^create | ^mk

^delete | ^rm

^edit

^list | ^ls

^showtypes

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^trigger <コマンド> --^usage

cm ^trigger <コマンド> --^help

== CMD\_HELP\_TRIGGER ==

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 ==

Plastic Tube に関連するコマンドを実行します。

== CMD\_USAGE\_TUBE ==

使用方法:

cm ^tube ^config -^u=<ユーザー> -^p=<パスワード>

(指定されたユーザーとパスワードで Plastic Tube を使用するように Plastic SCM を

設定します。)

cm ^tube ^create <リモートユーザー>

(リモートユーザーから myuser へのチューブを作成します。

ユーザー「myuser」を使用して「リモートユーザー」を「myuser」サーバーに接続できます。

「リモートユーザー」から「myuser」への接続を確立できます。

他のユーザーから現在の Tube ユーザーへのチューブのみを作成できます。)

cm ^tube ^remove <リモートユーザー>

(リモートユーザーから myuser へのチューブを削除します。)

cm ^tube ^local

(ローカルサーバーで共有されるローカルリポジトリと、共有相手のユーザーを

リストします。)

cm ^tube ^remote

(現在の Tube ユーザーと共有されている、共有リモートリポジトリを

リストします。)

cm ^tube ^share <リポジトリ指定>[ ...]-^u=<リモートユーザー> -^a=(^pull | ^push | ^pull,^push)

(ローカルリポジトリをリモートユーザーと共有し、指定されたアクセスモードを

設定します。空白を使用してリポジトリ指定を区切ります。)

cm ^tube ^unshare <リポジトリ指定>[ ...]-^u=<リモートユーザー>

(リモートユーザーとのローカルリポジトリの共有を解除します。空白を使用して

リポジトリ名を区切ります。)

cm ^tube ^connect

(Plastic SCM サーバーを Plastic Tube に接続します。)

cm ^tube ^disconnect

(Plastic SCM サーバーを Plastic Tube から切断します。)

cm ^tube ^status

(Plastic SCM サーバーが Plastic Tube に接続されているかどうかを表示します。)

オプション

-^u Tube ユーザー (plasticscm.com のユーザー)。

-^p ユーザーのパスワード。

-^a アクセスモードを設定します。

リポジトリ指定 リポジトリの指定。(「cm ^help ^objectspec」を

使用してリポジトリ指定の詳細を確認できます。)

== CMD\_HELP\_TUBE ==

備考:

Plastic Tube を管理するには、「cm ^tube」コマンドを使用します。

例:

cm ^tube ^config -^u=ruben@codicesoftware.com -^p=rubenpassword

cm ^tube ^create pablo@codicesoftware.com

(「pablo@codicesoftware.com」は現在の Plastic Tube ユーザーに接続できます)

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 ==

項目のチェックアウトを取り消します。

== CMD\_USAGE\_UNCO ==

使用方法:

cm ^undocheckout | ^unco <項目パス>[ ...][-^a | --^all] [--^symlink] [--^silent]

[--^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

項目パス 操作を適用する項目。空白を使用してパスを

区切ります。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

操作を現在のディレクトリに適用するには、. を使用します。

オプション:

-^a | --^all 指定された項目のすべての変更を取り消します。項目が

チェックアウトされている場合は、チェックアウトが

取り消されます。項目がローカルで変更されている場合は、

変更が取り消されます。

--^symlink チェックアウト取り消し操作をターゲットではなくリンクに

適用します。

--^silent 出力を表示しません。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator 「--^machinereadable」フラグとともに使用され、

行をどのように開始する必要があるかを指定します。

--^endlineseparator 「--^machinereadable」フラグとともに使用され、

行をどのように終了する必要があるかを指定します。

--^fieldseparator 「--^machinereadable」フラグとともに使用され、

フィールドをどのように区切る必要があるかを指定します。

== CMD\_HELP\_UNCO ==

備考:

項目がチェックアウト済みになっていて、それをチェックインしたくない場合は、このコマンドを

使用してチェックアウトを取り消すことができます。ファイルとフォルダーの両方のチェックアウトを

取り消すことができます。項目は、チェックアウトする前の状態に更新されます。

要件:

- 項目がソースコード管理の対象になっている必要があります。

- 項目がチェックアウトされている必要があります。

stdin から入力を読み取る:

「^undocheckout」コマンドは stdin からパスを読み取ることができます。これを行うには、

シングルダッシュ「-」を渡します。

例: cm ^undocheckout ^checkin -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用してチェックアウトを取り消すファイルを指定できます。

例:

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

(Windows で、ワークスペース内のすべての .c ファイルのチェックアウトを取り消します。)

例:

cm ^undocheckout .

(現在のディレクトリでチェックアウトを取り消します。)

cm ^undocheckout file1.txt file2.txt

cm unco c:\workspace\file.txt

(選択されたファイルのチェックアウトを取り消します。)

cm ^unco -^a file1.txt

(「file1.txt」のチェックアウトまたはローカル修正を取り消します。)

cm ^unco リンク --^symlink

(ターゲットにではなくリンクファイルにチェックアウト取り消し操作を適用します。

UNIX 環境で利用できます。)

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

(クライアントの変更リストを取り消します。

上のコマンドは、「pending\_to\_review」という名前の変更リスト内のパスを

リストします。このパスのリストは、チェックアウト取り消しコマンドの入力に

リダイレクトされます。)

cm ^unco .--^machinereadable

(現在のディレクトリでチェックアウトを取り消し、その結果を簡略化された

より解析しやすい形式で出力します。)

cm ^unco .--^machinereadable --^startlineseparator=">" --^endlineseparator="<" \

--^fieldseparator=","

(現在のディレクトリでチェックアウトを取り消し、その結果を簡略化された、

解析しやすい形式で出力します。指定された文字列で行を開始および終了し、

フィールドを区切ります。)

== CMD\_DESCRIPTION\_UNCOUNCHANGED ==

変更されていないチェックアウト済み項目を取り消します。

== CMD\_USAGE\_UNCOUNCHANGED ==

使用方法:

cm ^uncounchanged | ^unuc <項目パス>[ ...][-^R | -^r | --^recursive]

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

項目パス 操作を適用する項目。空白を使用してパスを

区切ります。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

操作を現在のディレクトリに適用するには、. を使用します。

オプション:

-^R 指定されたパス内の変更されていない項目を再帰的に取り消します。

--^symlink uncounchanged 操作をターゲットではなくリンクに

適用します。

--^silent 出力を表示しません。

== CMD\_HELP\_UNCOUNCHANGED ==

備考:

このコマンドは、ワークスペースのルートから再帰的に適用されます。

stdin から入力を読み取る:

「^uncounchanged」コマンドは stdin からパスを読み取ることができます。これを行うには、

シングルダッシュ「-」を渡します。

例: cm ^uncounchanged -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用して、チェックアウトを取り消す変更されていないファイルを

指定できます。

例:

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

(Windows で、ワークスペース内のすべての変更されていない .c ファイルの

チェックアウトを取り消します。)

例:

cm ^uncounchanged .-^R

(現在のディレクトリで、変更されていないファイルのチェックアウトを再帰的に取り消します。)

cm ^unuc /home/myuser/mywk/project/templates -^R

(選択されたディレクトリで、変更されていないファイルのチェックアウトを再帰的に取り消します。)

== CMD\_DESCRIPTION\_UNDELETE ==

特定のリビジョンを使用して項目の削除を取り消します。

== CMD\_USAGE\_UNDELETE ==

使用方法:

cm ^undelete <リビジョン指定> <パス>

リビジョン指定 コンテンツがワークスペースにロードされるリビジョンの

指定。(「cm ^help ^objectspec」を

使用してリビジョン指定の詳細を確認できます。)

パス 復元パス。

== CMD\_HELP\_UNDELETE ==

備考:

削除を取り消す項目は、ワークスペースにまだロードされていない必要があります。

Xlink では「^undelete」操作はサポートされていません。

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 ==

パスに対して変更を取り消します。

== CMD\_USAGE\_UNDOCHANGE ==

使用方法:

cm ^undochange | ^unc <項目パス>[ ...][-^R | -^r | --^recursive]

項目パス 操作を適用する項目。空白を使用してパスを

区切ります。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。

操作を現在のディレクトリに適用するには、. を使用します。

オプション:

-^R 操作を再帰的に適用します。

== CMD\_HELP\_UNDOCHANGE ==

備考:

項目がチェックアウトまたは変更されたがチェックインされておらず、チェックインしたくない

場合は、このコマンドを使用して変更を取り消すことができます。項目は

以前のコンテンツに更新されます。

stdin から入力を読み取る:

「^undochange」コマンドは stdin からパスを読み取ることができます。これを行うには、

シングルダッシュ「-」を渡します。

例: cm ^undochange -

パスは空の行が入力されるまで読み取られます。

これにより、パイプを使用して変更を取り消すファイルを指定できます。

例:

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

(Windows で、ワークスペース内のすべての .c ファイルの変更を取り消します。)

例:

cm ^unc .

(現在のディレクトリでファイルの変更を取り消します。)

cm ^undochange .-^R

(現在のディレクトリでファイルの変更を再帰的に取り消します。)

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

(選択されたファイルの変更を取り消します。)

cm ^unc c:\workspace\file.txt

(選択されたファイルの変更を取り消します。)

== CMD\_DESCRIPTION\_UNDO ==

ワークスペース内の変更を取り消します。

== CMD\_USAGE\_UNDO ==

使用方法:

cm ^undo [<パス>[ ...]] [--^symlink] [-^r | --^recursive] [<フィルター>[ ...]]

[--^silent | --^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

パス 操作を適用するファイルまたはディレクトリの

パス。空白が含まれるパスを指定するには

二重引用符 (" ") を使用します。空白を使用してパスを区切ります。

パスが指定されていない場合は、デフォルトで

取り消し操作が現在のディレクトリ内のすべてのファイルに

適用されます。

フィルター 指定された 1 つまたは複数のフィルターを指定されたパスに

適用します。空白を使用してフィルターを区切ります。詳細については

「フィルター」セクションを参照してください。

オプション:

--^symlink 取り消し操作をターゲットではなくシンボリックリンクに

適用します。

-^r 取り消しを再帰的に実行します。

--^silent 出力を表示しません。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator 「--^machinereadable」フラグとともに使用され、行をどのように

開始する必要があるかを指定します。

--^endlineseparator 「--^machinereadable」フラグとともに使用され、行をどのように

終了する必要があるかを指定します。

--^fieldseparator 「--^machinereadable」フラグとともに使用され、フィールドを

どのように区切る必要があるかを指定します。

フィルター:

フラグが指定されていない場合、デフォルトですべての変更が取り消されますが、

パスは下のフラグの 1 つ以上を使用してフィルター処理できます。

ファイルまたはディレクトリが指定した種類の変更の 1 つ以上と一致する場合、

指定されたファイルまたはディレクトリに対するすべての変更が取り消されます。

たとえば、「--^checkedout」と「--^moved」の両方を指定した場合に、ファイルの

チェックアウトと移動の両方が行われていると、両方の変更が取り消されます。

--^checkedout チェックアウト済みのファイルとディレクトリを選択します。

--^unchanged コンテンツが変更されていないファイルを選択します。

--^changed ローカルに変更またはチェックアウトされたファイルと

ディレクトリを選択します。

--^deleted 削除されたファイルとディレクトリを選択します。

--^moved 移動されたファイルとディレクトリを選択します。

--^added 追加されたファイルとディレクトリを選択します。

== CMD\_HELP\_UNDO ==

備考:

「^undo」は危険なコマンドです。作業内容が不可逆的な方法で取り消されます。

^undo が完了すると、その影響を受けたファイルとディレクトリの以前の状態を

復元することはできません。引数にパスが指定されて

いない場合は、デフォルトで現在のディレクトリでのすべての変更が

取り消されますが、再帰的ではありません。

これらは、/src ディレクトリから実行されたときと同等です。

/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

操作を再帰的にするには、「-^r」フラグを指定する必要があります。

ディレクトリ下のすべての変更を取り消すには (ディレクトリ自体に影響する

変更を含む):

cm ^undo ディレクトリパス -^r

ディレクトリパスがワークスペースパスの場合、ワークスペース内のすべての

変更が取り消されます。

削除された項目:

ファイルとディレクトリの削除を取り消すには、項目のフルパスを指定するか、

項目が含まれるディレクトリを指定して再帰的 (「-^r」)フラグを使用する必要が

あります。

例:

cm ^undo .

(現在のディレクトリ内の削除を取り消しません。)

cm ^undo .-^r

(現在のディレクトリ内のすべての削除 (およびその他の変更) を再帰的に取り消します。)

cm ^undo src/file.txt

(src/file.txt の削除 (またはその他の変更) を取り消します。)

例:

cm ^undo .-^r

(現在のディレクトリでのすべての変更を再帰的に取り消します。ワークスペース

のルートから実行された場合は、ワークスペース全体のすべての変更が取り消されます。)

cm ^co file.txt

cm ^undo file.txt

(「file.txt」のチェックアウトを取り消します。)

^echo ^content >> file.txt

cm ^undo file.txt

(「file.txt」に対するローカルの変更を取り消します。)

cm ^undo src

(src ディレクトリとそのファイルに対する変更を取り消します。)

cm ^undo src/\*

(src に含まれるすべてのファイルとディレクトリ内の変更を取り消します。

src には影響しません。)

cm ^undo \*.cs

(現在のディレクトリ内の、\*.cs と一致するすべてのファイルまたはディレクトリに対する変更を

取り消します。)

cm ^undo \*.cs -^r

(現在のディレクトリとその下のすべてのディレクトリ内の、\*.cs と一致するすべての

ファイルまたはディレクトリに対する変更を取り消します。)

cm ^co file1.txt file2.txt

^echo ^content >> file1.txt

cm ^undo --^unchanged

(変更されていない「file2.txt」のチェックアウトを取り消します。ローカルに変更された

「file1.txt」は無視します。)

^echo ^content >> file1.txt

^echo ^content >> file2.txt

cm ^co file1.txt

cm ^undo --^checkedout

(チェックアウト済みのファイル「file1.txt」内の変更を取り消します。「file2.txt」は

チェックアウト済みでないため無視されます。)

cm ^add file.txt

cm ^undo file.txt

(「file.txt」の追加を取り消し、もう一度非公開ファイルにします。)

^rm file1.txt

^echo ^content >> file2.txt

cm ^add file3.txt

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

(「file1.txt」の削除と「file3.txt」の追加を取り消します。「file2.txt」の変更は

無視します。)

== CMD\_DESCRIPTION\_LOCK\_UNLOCK ==

ロックサーバーで項目のロックを取り消します。

== CMD\_USAGE\_LOCK\_UNLOCK ==

使用方法:

cm ^lock ^unlock [<リポジトリサーバー指定>] <GUID>[ ...]

リポジトリサーバー指定 リポジトリサーバーの指定。(「cm ^help ^objectspec」

を使用してリポジトリサーバー指定の詳細を確認できます。)

GUID ロック解除する項目の GUID のリスト。空白を使用して GUID を

区切ります。

== CMD\_HELP\_LOCK\_UNLOCK ==

備考:

- このコマンドは、指定されたサーバーを使用して項目をロック解除します。

- サーバーが指定されていない場合、このコマンドは現在のワークスペースからサーバーを

取得しようとします。

- 上記の手順でサーバーが割り出されなかった場合、現在の Plastic SCM クライアントの

設定からサーバーが取得されます。

- サーバーの管理者のみが「cm ^unlock」コマンドを実行できます。

- GUID を指定するには、ハイフンで区切られた 32 桁 の形式 (オプションで

中括弧で囲むこともできます) にする必要があります:

{00000000-0000-0000-0000-000000000000}

または 00000000-0000-0000-0000-000000000000

例:

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

(選択された項目のロックを取り消します。)

cm ^lock ^unlock DIGITALIS:8084 2340b4fa-47aa-4d0e-bb00-0311af847865 \

bcb98a61-2f62-4309-9a26-e21a2685e075

(「DIGITALIS」という名前のロックサーバーで、選択された項目のロックを取り消します。)

cm ^lock ^unlock tardis@cloud 4740c4fa-56af-3dfe-de10-8711fa248635 \

71263c17-5eaf-5271-4d2c-a25f72e101d4

(「tardis」という名前のクラウドロックサーバーで、選択された項目のロックを取り消します。)

== CMD\_DESCRIPTION\_UPDATE ==

ワークスペースを更新し、最新の変更をダウンロードします。

== CMD\_USAGE\_UPDATE ==

使用方法:

cm ^update [<項目パス> | --^last]

[--^changeset=<変更セット指定>] [--^cloaked] [--^dontmerge] [--^forced]

[--^ignorechanges] [--^override] [--^recursewk] [--^skipchangedcheck]

[--^silent] [--^verbose] [--^xml[=<出力ファイル>]] [--^encoding=<名前>]

[--^machinereadable [--^startlineseparator=<セパレーター>]

[--^endlineseparator=<セパレーター>] [--^fieldseparator=<セパレーター>]]

[--^forcedetailedprogress]

項目パス 更新するパス。

更新を現在のディレクトリに適用するには、. を使用します。

パスが指定されていない場合は、現在のワークスペースが

完全に更新されます。

--^last 更新前に、ワークスペースセレクターを変更セット設定または

ラベル設定からブランチ設定に

変更します。

セレクターは、変更セットまたはラベルが属するブランチに

変更されます。

オプション:

--^changeset ワークスペースを特定の変更セットに更新します。

(「cm ^help ^objectspec」を使用して

変更セット指定の詳細を確認できます。)

--^cloaked クロークされた項目を更新操作に含めます。

このオプションが指定されていない場合、クロークされた

項目は操作で無視されます。

--^dontmerge 更新操作中に更新マージが必要な場合、それを

実行しません。

--^forced 項目をセレクターで指定されたリビジョンに強制的に

更新します。

--^ignorechanges ワークスペースの更新時に検出された保留中の変更がある

場合に表示される、保留中の変更の警告メッセージを

無視します。

--^override Plastic SCM の管理外で変更されたファイルを上書きします。

それらのファイルのコンテンツがサーバーのコンテンツで

上書きされます。

--^recursewk 現在のパス内にあるすべてのワークスペースを更新

します。特定のパスに含まれるすべてのワークスペースを更新

するのに役立ちます。

--^skipchangedcheck 更新を開始する前に、ワークスペース内にローカルの変更が

あるかどうかを確認します。常にファイルの変更前に

チェックアウトする場合は、この確認を使用して

操作をスピードアップできます。

--^silent 出力を表示しません。

--^verbose 追加情報を表示します。

--^xml 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。

--^encoding --^xml オプションとともに使用され、XML 出力で使用する

エンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator 「--^machinereadable」フラグとともに使用され、

行をどのように開始する必要があるかを指定します。

--^endlineseparator 「--^machinereadable」フラグとともに使用され、

行をどのように終了する必要があるかを指定します。

--^fieldseparator 「--^machinereadable」フラグとともに使用され、

フィールドをどのように区切る必要があるかを指定します。

--^forcedetailedprogress 標準出力がリダイレクトされた場合でも、詳細な進捗情報を

強制的に出力します。

== CMD\_HELP\_UPDATE ==

備考:

「^update」コマンドは、必要なファイルのみをダウンロードします。

コマンドでは、再帰的な操作が想定されます。

「--^last」オプションを使用する場合、パスを指定する必要はありません。

その場合、現在の作業ディレクトリが属するワークスペースが

更新されます。

(このフラグを指定すると、セレクターが変更セットまたは

ラベルを示していた場合に、ワークスペースセレクターがブランチ設定に変更される可能性があることに

注意してください。)

例:

cm ^update

(現在のワークスペース内のすべてを更新します。)

cm ^update .

(現在のディレクトリと、すべての子項目を更新します。)

cm ^update .--^forced --^verbose

(すべてのリビジョンを強制的に取得します。)

cm ^update --^last

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

(現在のディレクトリを更新し、その結果を簡略化された

より解析しやすい形式で出力します。指定された文字列で行を

開始します。)

== CMD\_DESCRIPTION\_VERSION ==

現在のクライアントバージョン番号を表示します。

== CMD\_USAGE\_VERSION ==

使用方法:

cm ^version

== CMD\_HELP\_VERSION ==

== CMD\_DESCRIPTION\_WHOAMI ==

現在の Plastic SCM ユーザーを表示します。

== CMD\_USAGE\_WHOAMI ==

使用方法:

cm ^whoami

== CMD\_HELP\_WHOAMI ==

== CMD\_USAGE\_WKTREENODESTATUS ==

使用方法:

cm ^wktreenodestatus パス 1, パス 2, ...

== CMD\_DESCRIPTION\_WORKSPACE ==

ユーザーにワークスペースの管理を許可します。

== CMD\_USAGE\_WORKSPACE ==

使用方法:

cm ^workspace | ^wk <コマンド> [オプション]

コマンド:

^list | ^ls

^create | ^mk

^delete | ^rm

^move | ^mv

^rename

各コマンドの詳細情報を取得するには、次のコマンドを実行します:

cm ^workspace <コマンド> --^usage

cm ^workspace <コマンド> --^help

== CMD\_HELP\_WORKSPACE ==

例:

cm ^workspace ^create myWorkspace ワークスペースパス

cm ^workspace ^list

cm ^workspace ^delete myWorkspace

== CMD\_DESCRIPTION\_WORKSPACE\_CREATE ==

新しいワークスペースを作成します。

== CMD\_USAGE\_WORKSPACE\_CREATE ==

使用方法:

cm ^workspace | ^wk [^create | ^mk] <ワークスペース名> <ワークスペースパス> [<リポジトリ指定>]

[--^selector[=<セレクターファイル>]

(新しいワークスペースを作成します。)

cm ^workspace | ^wk [^create | ^mk] <ワークスペース名> <ワークスペースパス> --^dynamic --^tree=[<ツリー>]

動的ワークスペースを作成します。この機能はまだ実験段階であり、Windows でのみ

使用できます。)

ワークスペース名 新しいワークスペースの名前。

ワークスペースパス 新しいワークスペースのパス。

リポジトリ指定 指定されたリポジトリで新しいワークスペースを作成します。

リポジトリ指定については「cm ^help ^objectspec」を確認してください。

オプション:

--^selector 新しいワークスペースのセレクターを編集します。

セレクターファイルが指定された場合は、指定された

ファイルから新しいワークスペースのセレクターを設定します。

--^dynamic 動的ワークスペースを作成します。この機能はまだ

実験段階であり、Windows でのみ使用できます。

このフラグを指定するには、--^tree パラメーターを使用する必要があります。

--^tree 「--^dynamic'」フラグとともに使用され、動的ワークスペースが

ロードする初期ポイントを指定します。これは

ブランチ、変更セット、またはラベル指定のいずれかにできます。

ワークスペースはその後、指定に含まれるリポジトリを使用

します。(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

== CMD\_HELP\_WORKSPACE\_CREATE ==

備考:

- ワークスペースは、ローカルファイルシステムにマップされたリポジトリのビューです。

ワークスペースセレクターは、ワークスペースのコンテンツを指定するルールを定義します。

ワークスペースセレクターを表示するには「cm ^showselector」、変更するには「cm ^setselector」を

使用します。

- リポジトリ指定も「--^selector」も指定されていない場合、ワークスペースは、

client.conf ファイルで設定されているサーバーの最初のリポジトリ

(アルファベット順) を使用するように自動的に設定されます。

- 動的ワークスペースは実験段階の機能 (Windows のみ) であり、

plasticfs.exe プログラムの実行が必要です。

例:

cm ^workspace ^create myworkspace c:\workspace

cm ^wk ^mk myworkspace /home/john/plastic\_view

(「myworkspace」ワークスペースを Windows と Linux でそれぞれ作成します。)

cm ^wk mywktest c:\wks\wktest --^selector=myselector.txt

(「myselector.txt」ファイル内のセレクターを使用して「mywktest」ワークスペースを作成します。)

cm ^wk mywkprj c:\wks\wkprj myrep@^repserver:localhost:8084

(選択されたリポジトリで「mywkprj」ワークスペースを作成します。)

cm ^wk mywkprj c:\dynwks\mywkprj --^dynamic --^tree=^br:/main@myrep@localhost:8084

(「myrep@localhost:8084」リポジトリで、最初にマウントされたときに

「^br:/main」を示す動的ワークスペース「mywkprj」を作成します。)

== CMD\_DESCRIPTION\_WORKSPACE\_DELETE ==

ワークスペースを削除します。

== CMD\_USAGE\_WORKSPACE\_DELETE ==

使用方法:

cm ^workspace | ^wk ^delete | ^rm [<ワークスペースパス> | <ワークスペース指定>] [--^keepmetadata]

ワークスペースパス 削除するワークスペースのパス。

ワークスペース指定 削除するワークスペースの指定。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

オプション:

--^keepmetadata .plastic フォルダー内のメタデータファイルを削除

しません。

== CMD\_HELP\_WORKSPACE\_DELETE ==

備考:

このコマンドは、パスまたは指定によって指定されたワークスペースを削除します。

引数が指定されていない場合は、現在のワークスペースが想定されます。

例:

cm ^workspace ^delete

(現在のワークスペースを削除します。)

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 ==

ワークスペースをリストします。

== CMD\_USAGE\_WORKSPACE\_LIST ==

使用方法:

cm ^workspace | ^wk [^list | ^ls] [--^format=<文字列形式>]

オプション:

--^format 出力メッセージを特定の形式で取得します。詳細については

「備考」を参照してください。

== CMD\_HELP\_WORKSPACE\_LIST ==

備考:

このコマンドは、出力を表示する形式の文字列を受け取ります。

このコマンドの出力パラメーターは次のとおりです。

{0} | {^wkname} ワークスペース名。

{1} | {^machine} クライアントマシン名。

{2} | {^path} ワークスペースパス。

{3} | {^wkid} ワークスペースの一意識別子。

{4} | {^wkspec} 次の形式を使用したワークスペース指定:

'wkname@machine'。

{^tab} タブスペースを挿入します。

{^newline} 改行を挿入します。

例:

cm ^wk

(すべてのワークスペースをリストします。)

cm ^workspace ^list --^format={0}#{3,40}

cm ^workspace ^list --^format={^wkname}#{^wkid,40}

(すべてのワークスペースをリストし、ワークスペース名、# 記号、ワークスペース GUID フィールドを

40 個の空白内に左寄せして表示します。)

cm ^wk --^format="パス {2} のワークスペース {0}"

cm ^wk --^format="パス {^path} のワークスペース {^wkname}"

(すべてのワークスペースをリストし、結果を形式化された文字列で表示します。)

== CMD\_DESCRIPTION\_WORKSPACE\_MOVE ==

ワークスペースを移動します。

== CMD\_USAGE\_WORKSPACE\_MOVE ==

使用方法:

cm ^workspace | ^wk ^move | ^mv [<ワークスペース指定>] <新しいパス>

オプション:

ワークスペース指定 移動するワークスペースの指定。

(「cm ^help ^objectspec」を使用して指定の詳細を確認できます。)

新しいパス ワークスペースはここに移動されます。

== CMD\_HELP\_WORKSPACE\_MOVE ==

備考:

このコマンドは、ワークスペースをディスク上の別の場所に移動することをユーザーに許可します。

例:

cm ^workspace ^move myWorkspace \new\workspaceDirectory

(「myWorkspace」を指定された場所に移動します。)

cm ^wk ^mv c:\users\maria\wkspaces\newlocation

(現在のワークスペースを新しい場所に移動します。)

== CMD\_DESCRIPTION\_WORKSPACE\_RENAME ==

ワークスペース名を変更します。

== CMD\_USAGE\_WORKSPACE\_RENAME ==

使用方法:

cm ^workspace | ^wk ^rename [<ワークスペース名>] <新しい名前>

ワークスペース名 名前変更するワークスペース。

新しい名前 ワークスペースの新しい名前。

== CMD\_HELP\_WORKSPACE\_RENAME ==

備考:

このコマンドは、ワークスペース名を変更します。

ワークスペース名が指定されていない場合は、現在のワークスペースが使用されます。

例:

cm ^workspace ^rename mywk1 wk2

(ワークスペース名を「mywk1」から「wk2」に変更します。)

cm ^wk ^rename newname

(現在のワークスペースの名前を「newname」に変更します。)

== CMD\_DESCRIPTION\_WORKSPACESTATUS ==

ワークスペース内の変更を表示します。

== CMD\_USAGE\_WORKSPACESTATUS ==

使用方法:

cm ^status [<ワークスペースパス>] [--^changelist[=<名前>] | --^changelists] [--^cutignored]

[ --^header] [ --^noheader] [ --^nomergesinfo] [ --^head]

[--^short] [--^symlink] [ --^dirwithchanges] [--^xml[=<出力ファイル>]]

[--^encoding=<名前>] [ --^wrp | --^wkrootrelativepaths]

[--^fullpaths | --^fp] [<従来のオプション>] [<検索タイプ>[ ...]]

[--^machinereadable [--^startlineseparator=セパレーター]

[--^endlineseparator=セパレーター] [--^fieldseparator=セパレーター]]

オプション:

ワークスペースパス 検索が実行されるワークスペースの

パス。

--^changelist 選択された変更リスト内の変更を表示します。

--^changelists 変更をクライアントの変更リストにグループ化して表示します。

--^cutignored 無視対象のディレクトリのコンテンツをスキップします。

「--^ignored」検索タイプが必要です。詳細については、

「検索タイプ」セクションを確認してください。

--^header ワークスペースのステータスのみを出力します。

--^noheader 変更された項目の検索結果のみを出力します。

--^nomergesinfo 変更のマージ情報を出力しません。

--^head ブランチ上の最後の変更セットのステータスを出力します。

--^short 変更が含まれるパスのみをリストします。

--^symlink 操作をターゲットではなくシンボリックリンクに

適用します。

--^dirwithchanges 変更 (内部の項目の追加、移動、削除) が含まれる

ディレクトリを表示します。

--^xml 出力を XML 形式で標準出力に出力します。

出力ファイルを指定することができます。

--^encoding --^xml オプションとともに使用され、XML 出力で使用する

エンコーディング (utf-8 など) を指定します。

サポートされるエンコーディングとその形式のテーブルを取得するには、

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

にある MSDN のドキュメントを参照してください

(ページの最後、「名前」列)。

--^wrp 現在のディレクトリの相対パスの代わりに

ワークスペースのルートの相対パスを出力します。

--^fullpaths, --^fp 他のパス出力設定を上書きして、絶対パスを強制的に

出力します。

--^machinereadable 結果を解析しやすい形式で出力します。

--^startlineseparator 「--^machinereadable」フラグとともに使用され、

行をどのように開始する必要があるかを指定します。

--^endlineseparator 「--^machinereadable」フラグとともに使用され、

行をどのように終了する必要があるかを指定します。

--^fieldseparator 「--^machinereadable」フラグとともに使用され、

フィールドをどのように区切る必要があるかを指定します。

従来のオプション:

--^cset ワークスペースのステータスを従来の形式で出力します。

--^compact ワークスペースのステータスと変更リストを従来の形式で

出力します。

--^noheaders 「--^compact」フラグとともに使用された場合、

変更リストのヘッダーが出力されません。(新しい変更リストの形式には

適用されません。)

検索タイプ:

--^added 追加された項目を出力します。

--^checkout チェックアウトされた項目を出力します。

--^changed 変更された項目を出力します。

--^copied コピーされた項目を出力します。

--^replaced 置換された項目を出力します。

--^deleted 削除された項目を出力します。

--^localdeleted ローカルで削除された項目を出力します。

--^moved 移動された項目を出力します。

--^localmoved ローカルで移動された項目を出力します。

--^percentofsimilarity=<値> 2 つのファイルを同じ項目と見なすために必要な

それらのファイル間の類似度のパーセント値。ローカルで

移動された項目の検索に使用されます。デフォルト値は

20% です。

--^txtsameext 移動された項目の検索時に、

類似度によるコンテンツ照合プロセスで

拡張子が同じであるテキストファイルのみが

考慮されます。デフォルトでは、すべての

テキストファイルが処理対象となります。

--^binanyext 移動された項目の検索時に、

類似度によるコンテンツ照合プロセスで

すべてのバイナリファイルが考慮されます。デフォルトでは、

拡張子が同じであるバイナリファイルのみが

処理対象となります。

--^private 管理対象外の項目を出力します。

--^ignored 無視対象の項目を出力します。

--^hiddenchanged 非表示の変更済み項目を出力します。(「--^changed」

を含みます。)

--^cloaked クロークされた項目を出力します。

--^controlledchanged このフラグは以下のオプションの代わりとなります:

「--^added」、「--^checkout」、「--^copied」、

「--^replaced」、「--^deleted」、「--^moved」。

--^all このフラグは以下のパラメーターの代わりとなります:

「--^controlledchanged」、「--^changed」、

「--^localdeleted」、「--^localmoved」、「--^private」。

== CMD\_HELP\_WORKSPACESTATUS ==

備考:

「^status」コマンドは、ワークスペースにロードされた変更セットを出力し、ワークスペース内の

変更された要素を取得します。

このコマンドは、ワークスペース内の保留中の変更を表示するために使用できます。

検索できる変更のタイプは、コマンドのパラメーターを使用して変更

できます。デフォルトでは、管理対象の変更もローカルの変更も含めた

すべての変更が表示されます。

類似度パラメーター「--^percentofsimilarity」(-^p) のパーセント値は、

ローカルで移動された項目の検索で、2 つの要素が同じ項目であるかどうかを判断するために使用されます。

デフォルト値は 20% ですが、この値は調整できます。

ワークスペースの変更を、クライアントの変更リスト別にグループ化して表示できます。

「^default」変更リストには、他の変更リストに含まれない変更が

含まれます。ただし、デフォルトの変更リストに表示される

変更は、指定されている検索タイプのフラグによって異なります。

変更を変更リスト別にグループ化して表示するには、管理対象の変更

(ステータスが「^added」、「^checkout」、「^copied」、

「^replaced」、「^deleted」、または「^moved」の項目) の表示も必要になります。そのため、変更リストが表示されるときは

「--^controlledchanged」オプションが自動的に有効になります。

XML 出力のデフォルトのエンコーディングは utf-8 です。

デフォルトでは、このコマンドは「--^machinereadable」または「--^short」フラグが

指定された場合を除き、現在のディレクトリの相対パスを出力します。これらの

いずれかが指定された場合、このコマンドは絶対パスを出力します。

「--^xml」フラグが指定された場合、ワークスペースのルートの相対パスが

出力されます (ただし、「--^fp」フラグも指定された場合は、

代わりに絶対パスが出力されます。)

例:

cm ^status

(作業中の変更セットと、ワークスペース内の変更されたすべての項目タイプ

(無視対象項目を除く) も出力します。)

cm ^status --^controlledchanged

(作業中の変更セットと、チェックアウト、追加、コピー、置換、削除、移動された

項目も出力します。)

cm ^status --^added

(作業中の変更セットと、ワークスペース内の追加された項目のみを出力します。)

cm ^status c:\workspaceLocation\code\client --^added

(作業中の変更セットと、指定されたパスの下にある追加された項目を

再帰的に出力します。)

cm ^status --^changelists

cm ^status --^changelist

(ワークスペースのすべての変更を、クライアントの変更リスト別にグループ化して表示します。)

cm ^status --^changelist=pending\_to\_review

(「pending\_to\_review」という名前の変更リストの変更を表示します。)

cm ^status --^changelist=default --^private

(「default」変更リスト内の変更を表示します。非公開項目と、管理対象の変更がある項目

を表示します。)

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

(変更リスト「pending\_to\_review」内の変更をチェックインします。)

cm ^status C:\workspaceLocation --^xml=output.xml

(XML 形式の、utf-8 を使用したステータス情報をファイル

output.xml に取得します。)

cm ^status --^ignored

(すべての無視対象項目を表示します。)

出力:

/main@myrepo@local (^cs:2 - ^head)

^Added

ステータス サイズ 最終変更 パス

^Ignored 0 バイト 19 秒前 out\app.exe

^Ignored 48 秒前 src

^Ignored 0 バイト 48 秒前 src\version.c

cm ^status --^ignored --^cutignored

(親ディレクトリが無視対象ではない無視対象のファイルと、無視対象のディレクトリ

を表示しますが、そのコンテンツは表示しません。)

出力:

/main@myrepo@local (^cs:2 - ^head)

^Added

ステータス サイズ 最終変更 パス

^Ignored 0 バイト 19 秒前 out\app.exe

^Ignored 48 秒前 src

== CMD\_DESCRIPTION\_XLINK ==

Xlink の詳細を作成、編集、または表示します。

== CMD\_USAGE\_XLINK ==

使用方法:

cm ^xlink [-^w] [-^rs] <Xlink パス> / (<変更セット指定> | <ラベル指定> | <ブランチ指定)>

[<拡張ルール>[ ...]]

(Xlink を作成します。)

cm ^xlink [-^rs] <Xlink パス> /<相対パス> (<変更セット指定> | <ラベル指定> | <ブランチ指定>)

[<拡張ルール>[ ...]]

(デフォルトのルート / ではなく /<相対パス> を示す読み取り専用の

部分的な Xlink を作成します。)

cm ^xlink -^e <Xlink パス> (<変更セット指定> | <ラベル指定> | <ブランチ指定>)

(Xlink を編集して、ターゲット指定を変更します。)

cm ^xlink -^s|--^show <Xlink パス>

(拡張ルールを含む Xlink 情報を表示します。)

cm ^xlink -^ar|--^addrules <Xlink パス> <拡張ルール>[ ...]

(指定された拡張ルールを Xlink に追加します。)

cm ^xlink -^dr|--^deleterules <Xlink パス> <拡張ルール>[ ...]

(指定された拡張ルールを Xlink から削除します。)

Xlink パス これは、リンクされたリポジトリがマウントされる (Xlink を作成

する場合) またはマウントされている (Xlink を編集する場合)

現在のワークスペース内のディレクトリです。

変更セット指定 リモートリポジトリ内の完全なターゲット変更セット

指定。

これにより、ワークスペースにロードされるリンクされたリポジトリの

バージョンとブランチが決定されます。

(「cm ^help ^objectspec」を使用して変更セット指定の詳細を

確認できます。)

ラベル指定 リモートリポジトリ内の完全なラベル指定。

(「cm ^help ^objectspec」を使用してラベル指定の詳細を

確認できます。)

ブランチ指定 リモートリポジトリ内の完全なブランチ指定。

これは、指定されたブランチが示している現在の

変更セットを使用します。(「cm ^help ^objectspec」を

使用してブランチ指定の詳細を確認できます。)

-^e 既存の Xlink を編集して、ターゲット変更セット指定を

変更します。

-^s | --^show 選択された Xlink に関する情報を表示します。

-^ar | --^addrules 選択された Xlink に 1 つ以上の拡張ルールを追加します。

-^dr | --^deleterules 選択された Xlink から 1 つ以上の拡張ルールを

削除します。

拡張ルール 1 つ以上の拡張ルールを指定します。各拡張ルールは

ブランチ-ターゲットブランチのペアです:

^br:/main/fix-^br:/main/develop/fix

オプション:

-^w Xlink が書き込み可能であることを示します。これは、ターゲット

リポジトリのコンテンツをブランチの自動拡張によって変更

できることを意味します。

-^rs 関連サーバー。このオプションを使用すると、リポジトリサーバーとは

別の関連する Xlink を作成できます。これにより、

別のサーバー内のレプリケートされたリポジトリに

作成された Xlink が自動的に識別されるようになります。

== CMD\_HELP\_XLINK ==

備考:

このコマンドは、指定された変更セットへの Xlink を作成します。デフォルトでは、読み取り専用の

Xlink が作成されます。これは、その Xlink 内のワークスペースにロードされたコンテンツを

変更できないことを意味します。Xlink でリンクされたコンテンツで変更を行えるように

するには、代わりに書き込み可能な Xlink を (「-^w」オプションを使用して) 作成します。

Xlink のターゲット変更セットを変更する際に、簡素化されたコマンドの構文を

使用できます。そうすると、新しいターゲット変更セットが

唯一の必須パラメーターとなります。Xlink の残りのパラメーターは

変更されません。

ブランチの自動拡張:

Xlink でリンクされた書き込み可能なリポジトリで変更が行われるときは (「-^w」オプション)、

ターゲットリポジトリに新しいブランチが作成される必要があります。新しいブランチの

名前は、最上位リポジトリで定義されたチェックアウトブランチに

基づきます。使用するブランチの名前を決定するために、次のルールが適用されます:

1) 同じ完全な名前を持つブランチがターゲットリポジトリに存在するかどうかを確認するための

チェックが行われます:

- 存在する場合は、それがチェックアウトブランチとして使用されます。

- 存在しない場合は、次のようにブランチ名が作成されます:

- Xlink でリンクされたターゲット変更セットのブランチの名前 + チェックアウトブランチの

短い名前 (最後の部分)。

- このブランチが存在する場合は、それがチェックアウトブランチとして使用されます。

- それ以外の場合は、ブランチが作成され、ブランチベースが Xlink でリンクされた

変更セットに設定されます。

2) 親リポジトリのブランチ内に、Xlink でリンクされたリポジトリ内の新しい変更セットを

示す新しいバージョンの Xlink が作成されます。

最後に、完全な Xlink 構造が、適切なバージョンの最新の変更によって

最新の状態に保たれます。

例:

cm ^xlink code\firstrepo / 1@first@localhost:8084

(現在のワークスペース内のフォルダー「firstrepo」に Xlink を作成し、

リポジトリ「first」内の変更セット「1」がマウントされるようにします。)

cm ^xlink opengl\include /includes/opengl 1627@includes@localhost:8087

(現在のワークスペース内のディレクトリ「opengl\include」に読み取り専用の

部分的な Xlink を作成し、リポジトリ「includes」内の変更セット「1627」内のパス

「/includes/opengl」がルートとしてマウントされるようにします。)これは、「/includes/opengl」

内のあらゆるものが「opengl\include」にマウントされるが、

リポジトリの残りの部分は無視されることを意味します。)

cm ^xlink -^w -^rs code\secondrepo / ^lb:LB001@second@localhost:8084

(現在のワークスペース内のフォルダー「secondrepo」に書き込み可能な

相対 Xlink を作成し、リポジトリ「second」内のラベル「LB001」がマウントされる

ようにします。)

cm ^xlink code\thirdrepo / 3@third@localhost:8087 ^br:/main-^br:/main/scm003

(現在のワークスペース内のフォルダー「thirdrepo」に Xlink を作成し、リポジトリ

「third」内の変更セット「3」がマウントされるようにします。)

cm ^xlink -^e code\secondrepo ^br:/main/task1234@second@localhost:8084

(Xlink「code\secondrepo」を編集して、リポジトリ「second」内のブランチ

「main/task1234」をリンクすることによってターゲットリポジトリを変更します。)

cm ^xlink --^show code\thirdrepo

(Xlink「code\thirdrepo」の情報を、その拡張ルール (存在する場合) も含めて

表示します)。

cm ^xlink -^ar code\secondrepo ^br:/main-^br:/main/develop ^br:/main/fix-^br:/main/develop/fix

(2 つの拡張ルールを Xlink「code\secondrepo」に追加します。)

cm ^xlink -^dr code\secondrepo ^br:/main/fix-^br:/main/develop/fix

(拡張ルールを Xlink「code\secondrepo」から削除します)。

== CMD\_USAGE\_AUTOCOMPLETE ==

使用方法:

cm ^autocomplete ^install

(「cm」コマンドの入力をシェルにインストールします。)

cm ^autocomplete ^uninstall

(「cm」コマンドの入力をシェルからアンインストールします。)

cm ^autocomplete --^line <シェル行> --^position <カーソル位置>

(「カーソル位置」に挿入する、「シェル行」に対する自動入力の候補を

返します。このコマンドは最終ユーザーが使用するためのものでは

ありませんが、選択したシェルに対する自動入力のサポートを拡張したい場合のために

説明しています。)

シェル行 自動入力がリクエストされたときにユーザーがシェルに

書き込んだ行。

Bash では、これは COMP\_LINE 環境変数の位置です。

PowerShell では、これは $wordToComplete 変数の位置です。

カーソル位置 自動入力がリクエストされたときのカーソルの

位置。

Bash では、これは COMP\_POINT 環境変数の位置です。

PowerShell では、これは $cursorPosition 変数の位置です。