git-p4操作手順

Tips

"git-p4"を使用したGitリポジトリとPerforceサーバの連携手順を紹介します。

"git-p4"を使用することで、毎日のGitでの作業をPerforceへコピーすることができます。また、反対にPerforceでの作業をGitにインポートすることができます。

※クライアント環境に"git-p4"がセットアップされていない場合は、『git-p4のセットアップ手順』を参照し、セットアップを実施してください。

概要

詳細

1．推奨事項

1) インポートのファイルサイズ

PerforceからGitへインポートする際、Perforceのディポ全体をGitへインポートしないことをお勧めします。(※ディポ全体が100,000ファイル未満、500MB未満の場合を除きます)

> ⚠ **注意**
>
> Gitはリポジトリごとに1つのプロジェクトを管理するように設計されているため、リポジトリのサイズが大きくなるにつれて、Gitのパフォーマンスが低下します。

2) 非対応のファイルタイプ

Gitがサポートしていないファイルタイプ(apple,resource)のファイル、または、"+k"オプション(RCSキーワード拡張)が付与されたファイルはインポートすることができません。
インポート時にこれらのファイルが含まれているとエラーとなるため、".gitigonre"ファイルを作成し、問題ファイルをインポート対象から除外します。

3) 履歴情報

git-p4によるインポート処理において、デフォルトでは、最新リビジョンのみをPerforceサーバからインポートします。なお、履歴情報としては、この時点からのサブミット済チェンジリスト情報をインポートします。もし、全ての履歴情報が必要な場合、インポート時に"@all"を付与してコマンド実行する必要があります。

コマンド実行例：全履歴を含む場合
git p4 clone //depot/path@all　.

※末尾の" . "はカレントディレクトリを示しています

4) クライアントワークスペース仕様の定義

git-p4の処理に使用するクライアントワークスペース仕様は、シンプルに作成することをお勧めします。
具体的には、除外マッピング・オーバーレイマッピング・ワイルドカードなどを使用しないように注意します。

5) インポート時のシンタックス

インポート時のPerforceサーバのパス指定では、Perforceシンタックスである" ... "や" / "を最後に追加しないように注意します。

6) 既存Gitリポジトリに対するインポート

既にインポートされているGitリポジトリに対して、"git p4 clone"コマンドを使用することはできません。この場合は、"git p4 sync"コマンドを使用します。

7) GitとPerforceの作業ディレクトリ

Git環境の作業ディレクトリとPerforce環境の作業ディレクトリを同じ場所にしないよう注意します。

２．インポート手順

① インポート環境の作成

Perforce環境とGit環境が混在しないように、各作業のディレクトリを作成します。

■ Perforce環境の作成

※ホームディレクトリ配下に作成した例で説明します

ディレクトリ作成

```
# mkdir    git-p4-area
```

".p4config"ファイルの作成

```
# cd git-p4-area
# vi .p4config
```

".p4config"ファイルの追記例

```
P4PORT=ssl:192.168.10.1:1666
P4USER=bruno
P4CLIENT=bruno-git-p4
```

・クライアントワークスペースの作成：太字部分を必要に応じて編集してください

```
# p4 client
        Client: bruno-git-p4
        Owner:  bruno
        Root:   /Users/bruno/bruno-p4-area
        Options: noallwrite noclobber nocompress unlocked nomodtime normdir
        View:
            //depot/Test/... //bruno-git-p4/depot/Test/...
```

■ Git環境の作成

※ホームディレクトリ配下に作成した例で説明します

ディレクトリ作成

```
# mkdir    git-area
```

".p4config"ファイルのコピー

```
# cd git-area
# cp ../git-p4-area/.p4config .
```

※Git環境の作業ディレクトリにも".p4config"ファイルが必要です

② インポート

手順①で作成した環境を使用して、Perforce環境からGit環境へインポートを実施します。
※コマンドはGit環境の作業ディレクトリで実行します。

インポートコマンド例

```
# cd git-area
# git p4 clone //depot/Test@all .
```

インポート実行結果例
Importing from //depot/Test@all into . Initialized empty Git repository in /User/bruno/git-area/.git/ Import destination: refs/remotes/p4/master Importing revision 163 (100%)

3．"git-p4"コマンドのワークフロー

"git-p4"コマンドを使用してGit環境とPerforce環境を連携するワークフローは以下の通りです。

> ⚠ **注意**
>
> "git p4 submit"コマンドは、Git環境の変更内容をPerforceサーバへ登録します。
> このコマンドを実行する前に、必ず"git p4 rebase"コマンドの実行をお願いします。
> "git p4 rebase"コマンドは、Perforceサーバの変更内容をGit環境へ"rebase"します。

■ ワークフロー

① Gitリポジトリに対するファイル編集

② Gitリポジトリに対するコミット

③ git p4 rebaseコマンド実行

④ git p4 submitコマンド実行

※コマンド概要は、続く「4．"git-p4"コマンドについて」をご確認ください

4．"git-p4"コマンドについて

コマンド一覧	
git p4 clone	Perforceサーバからインポートを行い、Gitリポジトリを作成します **コマンド例** # git p4 clone //depot/main/src .
git p4 debug	p4コマンドにおけるグローバルオプション"G"と同様にPythonの辞書型オブジェクトとして情報を出力します
git p4 commit/submit	Gitリポジトリの変更情報をPerforceサーバへサブミットします
git p4 rebase	Perforceサーバの変更情報をGitリポジトリへ"rebase"します
git p4 branches	インポートを保持するgitブランチと対応するPerforceディポパスを表示します **コマンド実行結果例** p4/master <= //depot/Test/ (163)

git p4 sync	PerforceサーバからGitリポジトリへインポートを行います ※初回クローン作成後は、git p4 cloneコマンドでなく、本コマンドを使用します ※Perforceシンタックスである" ... "は使用できません。 **コマンド例** # git p4 sync //depot/main/src **コマンド例（全リビジョン情報含む場合）** # git p4 sync //depot/main/src@all **コマンド例（リビジョン情報を指定する報含）** # git p4 sync //depot/main/src@1,10