Perforce 2012.3
P4Sandbox User's Guide

May 2013

This manual copyright 2012-2013 Perforce Software.

All rights reserved.

Perforce software and documentation is available from `http://www.perforce.com`. You can download and use Perforce programs, but you can not sell or redistribute them. You can download, print, copy, edit, and redistribute the documentation, but you can not sell it, or sell any documentation derived from it. You can not modify or attempt to reverse engineer the programs.

This product is subject to U.S. export control laws and regulations including, but not limited to, the U.S. Export Administration Regulations, the International Traffic in Arms Regulation requirements, and all applicable end-use, end-user and destination restrictions. Licensee shall not permit, directly or indirectly, use of any Perforce technology in or by any U.S. embargoed country or otherwise in violation of any U.S. export control laws and regulations.

Perforce programs and documents are available from our Web site as is. No warranty or support is provided. Warranties and support, along with higher capacity servers, are sold by Perforce Software.

Perforce Software assumes no responsibility or liability for any errors or inaccuracies that might appear in this book.

By downloading and using our programs and documents you agree to these terms.

Perforce and Inter-File Branching are trademarks of Perforce Software. Perforce software includes software developed by the University of California, Berkeley and its contributors. This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (`http://www.openssl.org/`).

All other brands or product names are trademarks or registered trademarks of their respective companies or organizations.

Table of Contents

Preface About This Manual 9

Command line client versus graphical client applications 9
Getting started with Perforce 9
Perforce documentation 10
Please give us feedback 11

Chapter 1 Overview 13

Purpose 13
Features and functionality 13
P4Sandbox stream architecture and components 14
Implementation configurations and restrictions 15
Merge down and copy up paradigm 15
Local stream conflict resolution 16
Connected P4Sandbox copy restrictions 16
P4Sandbox usage 16
Task stream creation 16
Automatic and manual shelving 16
Changes and changelist submission 17
Differences between using p4 and P4V 17
Functional differences 17
Directory differences 18
P4Sandbox administration 18
See also 18

Chapter 2 Setting Up P4Sandbox 19

Installing P4Sandbox 19
P4Sandbox installer and behavior 19
P4Sandbox installation components 19
Verifying the installation directory structure 20
Creating a P4Sandbox 20
Prerequisites 20
Using p4 21

Using the configuration wizard 22
Verifying the connection 25
Using p4 27
Using a graphical client application 27
Removing a P4Sandbox 28
Issuing p4sandbox delete 28
Uninstalling a P4Sandbox 28

Chapter 3 Using P4Sandbox with the Command-Line Client 29

Overview 29
P4Sandbox-specific commands 29
Additional p4 commands 29
Modified p4 commands 30
Using P4Sandbox commands 30
TCP connection requirement 30
Prohibited mirror stream update commands 30
Invalid commands 31
Accessing help 31
Starting a P4Sandbox 31
Working in a specific client write mode 31
Merging down changes from the shared service 31
Updating P4Sandbox with shared service changes 32
Updating only the mirror stream with shared service changes 32
Copying up changes to the shared service 33
Resolving a copy up that has conflicts 33
Working with streams 34
Adding a task stream 34
Viewing stream information 34
Switching between streams 34
Copying changes between task streams 34
Deleting a task stream 35
Working with shelving 35
Shelving and unshelving in-progress work 35
Deleting a shelf 36
Viewing shelf information 36
Reviewing file and directory information 37
Determining files in your workspace 37
Stopping a P4Sandbox 37

See also 38

Chapter 4 Using P4Sandbox with the P4V Integration 39

Starting a P4Sandbox 39
Working in a specific client write mode 40
Understanding P4Sandbox Stream Graph elements 41
Verifying P4Sandbox streams workspace setting 42
Using the context menus 42
Merging down changes from the shared service 46
Copying up changes to the shared service 48
Adding a task stream 51
Copying changes among task streams 51
Merging down changes between task streams 52
Removing a task stream 52
Switching streams and shelving work 52
Resolving pending tasks 53
Closing a P4Sandbox connection 53
Stopping a P4Sandbox 53
Deleting a P4Sandbox 53
See also 54

Chapter 5 Administering P4Sandbox 55

Understanding prohibited and unnecessary tasks 55
Managing your P4Sandbox: user tasks 55
Editing the file pulling interval 56
Performing miscellaneous tasks using p4 56
  Configuring Unicode mode 56
  Managing jobspecs and jobs 56
  Supporting P4Sandbox: backup and recovery 57
Managing P4Sandbox users: Perforce Administrator tasks 58

Chapter 6 Troubleshooting P4Sandbox 59

Troubleshooting 59
Localhost connection error 59
P4Sandbox does not relaunch in P4V 59

P4Sandbox and P4V password issue 60
Cannot copy up files 60
Files shelved to an incorrect stream 61
p4 print method 61
Zip method 61

Appendix A P4Sandbox Command Reference 63

Extended p4 commands 63
p4 admin stop 64
p4 copy 65
p4 copyup 67
p4 counter 68
p4sandbox delete 69
p4sandbox init 70
p4 integrate 72
p4sandbox list 74
p4 merge 75
p4 mergedown 77
p4 populate 78
p4 pull 79
p4 remote 80
p4 remotes 82
p4 shelve 83
p4sandbox start 84
p4sandbox stop 85
p4 submit 86
p4 switch 87
Comparing Git and P4Sandbox commands 88
Command and concept equivalencies 88
Task equivalencies 89
Client mode dependent commands 94
See also 94

Appendix B Glossary 95

Terms 95

Index 101

Preface

About This Manual

This guide tells you how to use P4Sandbox (Perforce Sandbox). If you're new to version management, you don't know basic Perforce concepts, or you've never used Perforce before, read *Introducing Perforce* before reading this guide. This guide assumes a basic understanding of version management, the Perforce Visual Client (P4V), and Perforce *streams*, which are "branches with brains."

For more information about these concepts, products, and features, see the resources shown in "Perforce documentation" on page 10.

Command line client versus graphical client applications

Perforce provides many client applications that enable you to manage your files, including the Perforce Command-Line Client (`p4`) and graphical client applications such as P4V and plug-ins. The `p4` client lets you script and perform administrative tasks that are not supported by Perforce graphical client applications.

This guide contains information on using P4Sandbox with either `p4` or a graphical client application. The table below lists the pertinent chapters you should read depending on your preferred work method.

User	Chapter
All	1, 5, 6, Appendix B
`p4`	2 (`p4`-specific sections), 3, Appendix A
Graphical client application	2 (graphical client application-specific sections), 4

Getting started with Perforce

If this is your first time working with Perforce, here's how to get started:

1. Read *Introducing Perforce* to learn the basics.

   At a minimum, learn the following concepts: *changelist*, *depot*, *workspace*, *sync*, and *submit*.

2. Ask your Perforce administrator for the host name and port of your Perforce service.

   If you intend to experiment with Perforce and don't want to risk damaging your production depot, ask the Perforce administrator to start another service for test purposes. For details about installing the Perforce server, refer to the *Perforce System Administrator's Guide*.

3. Install `p4` and configure your workspace, unless your system administrator has already configured your machine. For more information, refer to the *P4 User's Guide.*
4. Learn to perform the following tasks:
   - *sync* (transfer selected files from the repository to your computer)
   - *submit* (transfer changed files from your workspace to the repository)
   - *revert* (discard changes)

   See the *P4 User's Guide,* "Managing Files and Changelists" for details.

Perforce documentation

This table provides a list of print and online documentation resources.

For specific information about...	See this documentation or URL
Perforce basics	*Introducing Perforce*
Using Perforce	*P4 User's Guide*
Installing and administering P4D, the Perforce Server; P4P, the Perforce Proxy; and security settings	*Perforce System Administrator's Guide*
`p4` command line flags and options (reference)	*Perforce Command Reference*, `p4 help` and `p4sandbox help`
P4V, the cross-platform Perforce Visual Client	*Getting Started with P4V*, P4V online help
P4Web, the browser-based Perforce client application	*How to use P4Web*, P4Web online help
Perforce plug-ins and integrations	Microsoft IDEs: *Using IDE Plug-ins* Defect trackers: *Defect Tracking Gateway Guide* Others: online help from the Perforce menu or web site
Developing Perforce client applications using the Perforce C/C++ API	*C/C++ API User's Guide*
Working with the Perforce service in Ruby, Perl, Python, and PHP	*APIs for Scripting*
Documentation on other Perforce products and older releases	`http://www.perforce.com/documentation/perforce_technical_documentation`

Please give us feedback

We are interested in receiving opinions on this guide from our users. In particular, we'd like to hear from users who have never used Perforce before. Does this guide teach the topic well? Please let us know what you think; we can be reached at `manual@perforce.com`

Chapter 1 Overview

Purpose

Using P4Sandbox lets you have the private branching and offline capabilities of a distributed version control system with the advantages of a shared version control system.

P4Sandbox lets you create and maintain private, local codelines for work such as code experimentation and bug fixes. You can optionally connect these local codelines to a Perforce shared service.

Features and functionality

P4Sandbox offers the following features and functionality:

- Connection-independent versioning (also called *always-on versioning*): P4Sandbox enables you to check changes in and out without requiring an active connection to a shared service. You connect to a shared service when merging changes or sharing your work.
- Private local codelines: You can create codelines (called *local streams*) within your P4Sandbox for specific tasks, such as bug fixes for a particular release.
- Fast context switching: When you switch between streams, P4Sandbox automatically performs the following tasks:
  - Shelves any work-in-progress from the previous stream.
  - Updates the workspace with files from the new stream.
  - Unshelves any work-in-progress in the new stream.
- Integration with other Perforce products: P4Sandbox integrates with the Perforce Command-line Client (`p4`) and graphical client applications such as P4V and P4Eclipse.

P4Sandbox stream architecture and components

P4Sandbox operates on Perforce streams. Of the four stream types, the two stream types you primarily use in P4Sandbox are:

- *Mainline:* The base or trunk of a stream system.
- *Development:* For long-term projects and new features. A development stream is less stable than its parent stream.

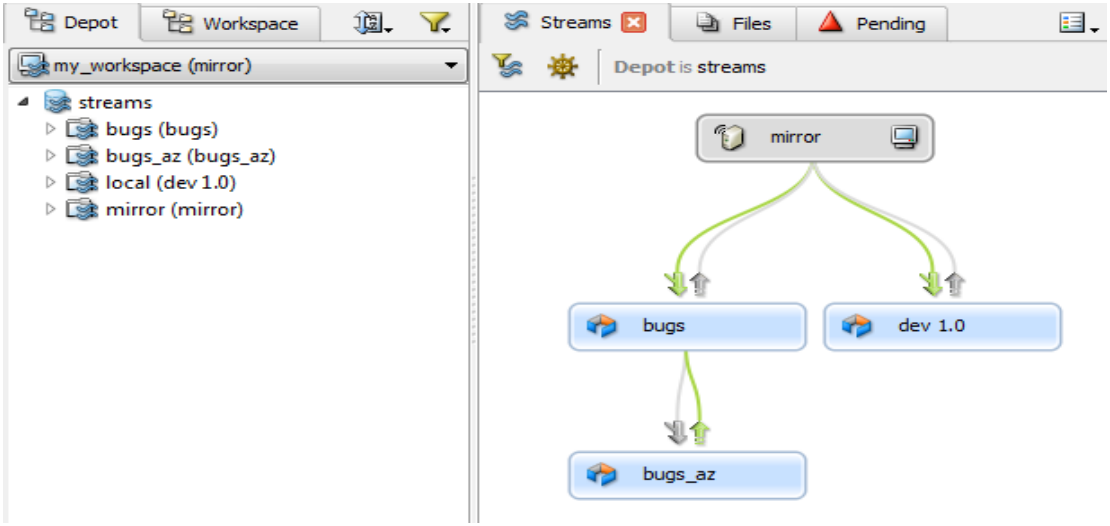
The other stream types are *release* (for bug-fixing, testing, and release distribution) and *virtual* (for filtering other stream types). For more information about Perforce streams, see the resources listed in "See also" on page 18.

P4Sandbox consists of the following components:

- *Shared service* (optional): Your enterprise's main Perforce server; the one server that everyone shares. You can use P4Sandbox with non-streams-enabled (classic) or streams-enabled shared services. This component is optional, as you can use P4Sandbox without a shared service connection.
- *Mirror stream*: A mirror stream is any mainline stream in P4Sandbox associated with a shared service. As the mirror stream mirrors the shared service, the mirror stream is the staging ground for pushes and pulls between the shared service and a local stream in P4Sandbox. Mirror streams are always mainline streams.
- *Local stream*: Any stream in P4Sandbox that is not a mirror stream. The term includes mainline streams that are not associated with shared services, as well as any development streams. Most local streams are development streams, descendants of a mirror stream. P4Sandbox automatically creates one local stream for a P4Sandbox implementation. A *task stream* is the term used in this guide for a local stream that you intend to use for a single specific task, like a bug fix.

The figure below shows P4V connected to a P4Sandbox that has a parent mirror stream (named **mirror**) with child local streams (named **bugs** and **dev 1.0**). The **bugs** local stream has a child local stream (or task stream) **bugs_az**. P4Sandbox automatically created the mirror and local streams, and the local stream was renamed to **dev 1.0**—you can see this in the **Depot** tab view, where the stream appears as **local (dev 1.0)**. The **bugs** and **bugs_az** streams were manually added.

For more information on the installed components, see "P4Sandbox installation components" on page 19.



Implementation configurations and restrictions

In general, most users use P4Sandbox in a straightforward configuration, either connected to a shared service or as a standalone implementation. However, you can also implement P4Sandbox in various configurations according to your requirements, such as multiple P4Sandboxes associated with multiple workspaces and multiple shared services.

Be aware that for an implementation that includes a shared service, P4Sandbox enforces certain behavior. We discuss these restrictions in the following three sections.

Merge down and copy up paradigm

P4Sandbox enforces a strict *merge down and copy up* paradigm, as follows:

- Changes are merged down from the more stable shared service to all affected local streams.
- Changes are copied up from less stable local streams to the shared service.
- Changes from the shared service must first be merged down before changes from the local streams can be copied up.

For more information on the Perforce stream merge down and copy up paradigm, see the articles listed in the Additional Resources section of the *Perforce Stream Adoption Guide*:

`http://www.perforce.com/product/product_features/perforce_streams`

Local stream conflict resolution

In P4Sandbox, conflict resolution cannot occur in the mirror stream. You must resolve conflicts in the local stream before submitting files to the mirror stream.

Connected P4Sandbox copy restrictions

You can only copy changes between a shared service and a P4Sandbox. You cannot copy changes between two or among multiple P4Sandboxes that are connected to a shared service.

P4Sandbox usage

This section discusses P4Sandbox's standard functionality for task streams, shelves, and submissions.

Task stream creation

Once you have created a P4Sandbox, you can create task streams for your own work. Task streams are local streams that are small, focused containers for work such as individual features, bug fixes, and code experiments. You can quickly create and use task streams, and also quickly switch between streams because Perforce keeps track of the identical files between streams. (Generally, most files are identical, so switching streams involves changes to very few files.)

Task streams are an effective way to isolate, develop, and save work before it is ready to be shared on the shared service. This is useful in certain situations, such as your company or team having code policies on what can and cannot be pushed to the shared service. For example, if you are prohibited from checking in code that does not pass regression testing, you can save it in a task stream until it is ready to be submitted to the shared service.

Note that task streams are a concept, not a specific stream type. There is no mechanical difference between a *local stream* and a task stream; the difference is solely in usage and not in how P4Sandbox handles the stream.

Automatic and manual shelving

Shelving is the process of temporarily storing files on a Perforce server without checking in a changelist. The P4Sandbox shelving feature enables you to switch tasks and save work-in-progress without first having to check in any changes.

P4Sandbox has the following types of shelving functionality:

- Automatic: Shelving that P4Sandbox performs in the background when you switch among task streams and workspaces and have unsubmitted changes.

- Manual: Shelving that you perform when you switch among task streams and workspaces and have unsubmitted changes, or from within the mirror stream as a backup of your work.

P4Sandbox propagates any shelving action that you perform while in the mirror stream to the shared service. This means you can have pending changelists with shelved files existing in both the shared service and your P4Sandbox. Besides giving you the ability to save work-in-progress, this capability facilitates code review and pre-flight builds.

For more information on Perforce's shelving functionality, see the *P4 User's Guide,* "Shelving work in progress."

Changes and changelist submission

P4Sandbox transfers only a single submission containing a single changelist to the shared service; it does not propagate any intermediate changes made in the task stream. For example, your P4Sandbox task stream contains versions 1-10 of a change, with multiple corresponding changelists. When you submit this change to the shared service, the shared service contains only version 10 and the final submitted changelist.

If you implement P4Sandbox to work with Perforce's jobs functionality, P4Sandbox copies up job fixes to the shared service. For more information, see "p4 submit" on page 86 and "Managing jobspecs and jobs" on page 56.

Differences between using p4 and P4V

This section discusses differences when using P4Sandbox with `p4` versus a graphical client application's P4V integration.

Functional differences

`p4` supports more functionality than P4V does, especially for administrative tasks. This includes (but is not limited to) the following actions, shown in the table below with the appropriate command:

Task	Command
Connect one P4Sandbox to multiple shared services and multiple mirror streams	`p4 remote`
Configure certain internal maintenance tools	`p4 counter`
Change the TCP port	`p4 configure`
Delete a P4Sandbox	`p4sandbox delete`
Print a list of deleted and missing P4Sandboxes	`p4sandbox list`

P4V is better at presenting certain information in a more easily-understood way for most users. For example, P4V's Time Lapse View provides a more intuitive visual representation of merge and copy history than the `p4 annotate` command provides. P4V also better facilitates interactive usage, such as moving the Workspace icon in the Streams Graph view to another stream to change the active workspace.

For more information, see "P4Sandbox Command Reference" on page 63.

Directory differences

When you create P4Sandboxes using the P4Sandbox Configuration Wizard, the wizard automatically creates a `sandboxes` directory that contains necessary files on your machine's hard drive. You can edit the directory's default location during the configuration process. When you use `p4` to create P4Sandboxes, P4Sandbox does not create this directory, as it is not necessary.

P4Sandbox administration

A major difference between P4Sandbox and other Perforce products is the role of the Perforce Administrator (or superuser). With P4Sandbox, you are the primary administrator of your P4Sandbox, particularly if you configure it as a standalone implementation. A Perforce Administrator's P4Sandbox control is limited to the point of interaction—the shared service. Because Perforce Administrators have limited visibility into the interaction between a P4Sandbox and the shared service, they can only perform general tasks over P4Sandboxes, such as running reports and viewing which files users have exchanged with the shared service.

For more information, see "Administering P4Sandbox" on page 55.

See also

See the following related information and related products:

- Online information, documentation, and presentations on Perforce streams:
  `http://www.perforce.com/product/product_features/perforce_streams`
- *P4 User's Guide*, "Codelines, Branching and Streams"
- *Getting Started with P4V,* "Viewing Streams"
- *P4V Online Help*, "Working with Streams"
- Git-P4: `http://kb.perforce.com/article/1417/git-p4`

Chapter 2 Setting Up P4Sandbox

This chapter tells you how to install, configure, and remove P4Sandbox on a workstation.

For details about installing the Perforce Server, see the *Perforce System Administrator's Guide*.

For more information on `p4` and `p4sandbox` commands discussed in this chapter, see "P4Sandbox Command Reference" on page 63.

Installing P4Sandbox

For hardware and software requirements and platform-specific installation instructions, see the *Perforce P4Sandbox Release Notes* included with the release package.

P4Sandbox installer and behavior

To install P4Sandbox, download the appropriate installer or installers for your platform and client application from the Perforce downloads page:

`http://www.perforce.com/downloads/complete_list`

Note When you perform an uninstall or upgrade with active P4Sandboxes running on your machine, the installer may prompt you either to stop all active P4Sandboxes or to reboot your machine to complete the process.

P4Sandbox installation components

The P4Sandbox installation consists of the following components:

- `p4sandbox`: The broker that implements P4Sandbox functionality. All user interactions are managed through this broker.
- `p4sandbox-p4d`: The Perforce Server distribution (`p4d`) used by P4Sandbox for the local server.
- `p4sandbox-config`: A GUI configuration tool (**P4Sandbox Configuration Wizard**) for creating and editing servers. Creating a P4Sandbox server also creates a hidden directory and updates a reference file. The location of the these files and directories is dependent upon your platform.
- `.p4sandbox-list`: An XML configuration file that lists your P4Sandbox servers.
- `.p4sandbox/`: Directory containing the `p4d` databases and archive files.

- `.p4config file`: A text file containing the following three default settings and values. For more information, see "p4sandbox init" on page 70.

```
P4PORT=localhost:1666
P4USER=machine's default user
P4CLIENT=my_workspace
```

Verifying the installation directory structure

If you use P4Sandbox with a graphical client application, you must ensure that the P4Sandbox and graphical client application components reside in the same directory. The initial P4Sandbox directory location depends on your platform:

- OS X and Linux: The installation process may install P4Sandbox and graphical client application components in different directories.
- Windows: The installation process usually installs P4Sandbox in the appropriate subdirectory of the graphical client application directory.

Verify your implementation and confirm that it has the correct directory structure—with all components residing in the same directory—before you create a P4Sandbox.

Creating a P4Sandbox

You can create and configure a connected or standalone P4Sandbox using either `p4` or the P4Sandbox Configuration Wizard.

Prerequisites

This section discusses important configuration issues for you to consider.

Workspace root clobber issues

For most users, we recommend that you do not create a P4Sandbox in the shared service workspace directory (the *workspace root*), as this can result in synchronization clobber issues. For more information, see the Usage Notes section of "p4sandbox init" on page 70.

Initial content replication time

The initial content replication from the shared service to the new P4Sandbox may take several minutes (or longer), depending on the number of files to copy and network latency. Subsequent replications should be faster, as the process copies over only those files that have been updated.

Standalone to connected P4Sandbox conversion

If you install P4Sandbox as a private standalone implementation and later decide to connect it to a shared service, you must contact your Perforce Technical Support representative at `support@perforce.com` for help with this conversion.

Using p4

Use one of the following three methods to create a P4Sandbox.

p4-only method for a connected P4Sandbox

To create a P4Sandbox in the current directory, type:

```
p4sandbox init
```

In the following example, the user:

1. Sets Perforce configuration variables in the Windows registry.
2. Creates a P4Sandbox that uses the configuration variables set in step 1.
3. Connects the P4Sandbox to a shared service by use of the remote depot mechanism.
4. Branches the project's contents to the P4Sandbox's mirror stream.
5. Creates a local development area.

Example: *This example shows how to create a connected P4Sandbox.*

```
p4 set P4CONFIG=.p4config
p4sandbox init
p4 remote -p perforce:1666
p4 merge //remote/depot/stream/... //streams/mirror/...
p4 merge //streams/mirror/... //streams/local/...
```

Note the following about the above example:

- `p4 set` is a Windows-specific Perforce command. To set the appropriate environmental variables for your platform, see the "Environment Variables" section in the *Perforce Command Reference*.
- You can also exclude specific files from replication by defining `P4IGNORE` within a `P4CONFIG` file. For more information, see the "P4IGNORE" and "P4CONFIG" sections in the *Perforce Command Reference*, and the "Ignoring groups of files" section in the *P4 User's Guide.*
- For lines 4 and 5, note that you can use any of the following commands in place of `p4 merge` because (in this case) they have similar functionality:
  - `p4 copy`
  - `p4 integrate`
  - `p4 populate`

p4-only method to create a standalone P4Sandbox

To create a P4Sandbox in the current directory, type:

```
p4sandbox init
```

The command creates a standalone P4Sandbox containing only a `.p4config` file and `.p4sandbox` directory.

p4 with wizard method

The `p4sandbox-config` command launches the P4Sandbox Configuration Wizard to create a P4Sandbox.

For more information, see "Using the configuration wizard."

Using the configuration wizard

When using a graphical client application, you use the P4Sandbox Configuration Wizard to create a connected or standalone P4Sandbox.

Configure P4Sandbox

To use the P4Sandbox Configuration Wizard:

1. Launch the wizard using one of the following options:
   - Double-click the **Perforce Sandbox Configuration** icon.

     Mac OS X locations: **Applications** > **Perforce** folder

     Windows locations: **Start Menu** or **Start Menu** > **All Programs** > **Perforce** folder
   - In P4V, click **Connection** > **Configure Sandbox**.
   - From `p4`, type `p4sandbox-config`.

2. Complete the wizard pages, as applicable. The table lists pages in their usual display order.

Title	Task	Notes
Welcome to the Sandbox Setup Wizard	Create a new or edit an existing P4Sandbox.	This page does not appear if you launch the wizard directly from a client application.
Specify Central Server Settings	Specify connection to a central server or P4Sandbox.	The connection setting you specify here determines whether the P4Sandbox is a connected or standalone implementation. Consequently, this selection also affects the Configuration Wizard's subsequent page display. If you are creating a connected P4Sandbox: • Specify **Port**, **Server**, and **User** values. • Specify whether to replicate files from a **Workspace** or **Stream**, or select **None** to choose which files to replicate.
Replication Settings	Define pulling interval and (optionally) select files to copy from a central server.	This page appears as the initial page when you launch the wizard from P4V. Whether or not you can choose files to replicate depends on the options defined on the previous page. The **Central Server** field displays the specified server, user, stream, and in parentheses, the stream root information. Note: If you do not define a pulling interval, you must manually perform this action. P4Sandbox will prompt you to pull (merge down) changes, as needed, before allowing you to push (copy up) changes. Because files occupy space on your hard disk and require time to transfer across the network, we suggest you minimize the number of folders you copy to minimize disk and time consumption.

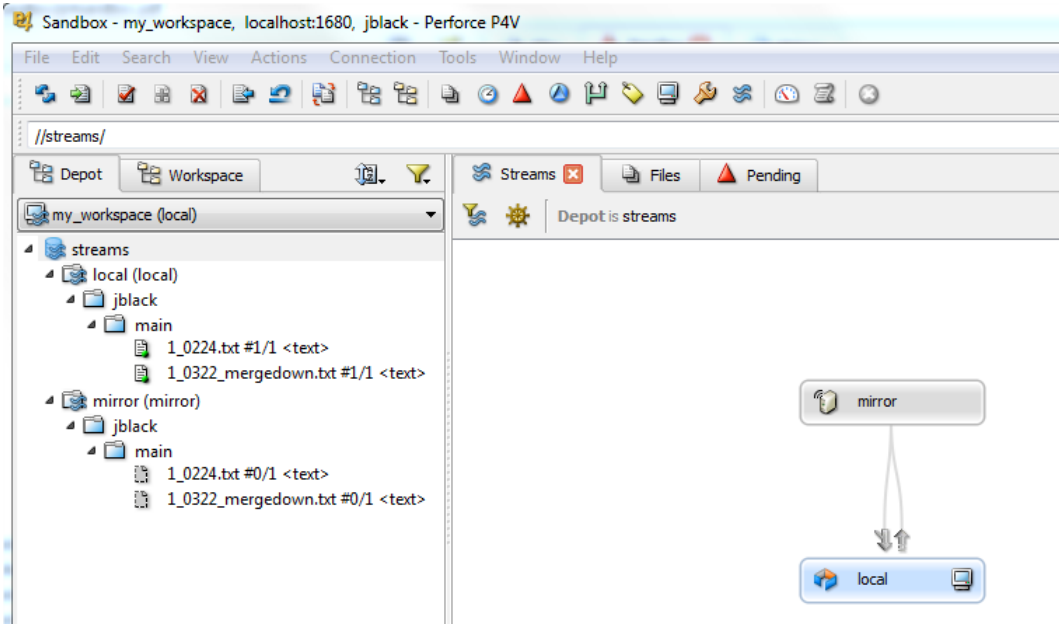
Title	Task	Notes
Specify Local Sandbox Server and Workspace Settings	Review connection settings for a P4Sandbox.	Specify the **Server**, **Port**, **User**, **Workspace Name**, **Workspace Root**, and **Charset** (character set) **for Unicode Servers** fields. When you change the **Port** or **Workspace name** value (or both), the wizard automatically updates the corresponding entry in the **Workspace Root** path. Note: The default **Workspace Root** path differs among platforms as follows: • Linux: `$home/sandboxes/sandbox-NNNN/client` • Mac OS X: `/Users/user name/sandboxes/sandboxNNNN/client` • Windows: `\Users\user name\sandboxes\sandboxNNNN/client` Carefully review the displayed fields for accuracy. Note: You cannot use the same port number for multiple P4Sandboxes. When you attempt to re-use an existing port number, the following message appears: `The specified port number is already in use by another P4Sandbox.`
Settings Summary	Review P4Sandbox settings before finalization.	Shows settings and number of files to be replicated. To edit, click **Back**. Click **Finish** to complete the configuration.

3. When you have successfully completed the configuration, P4V does one of the following:
   - For the initial P4Sandbox launch, P4V opens a dialog box indicating that it has launched a P4Sandbox connection window. This dialog box also includes links to P4Sandbox documentation and videos. Click the **Don't show again** check box to prevent the dialog box from displaying for any subsequent P4Sandboxes that you create.

- Opens a P4Sandbox connection window.

The figure below shows an initial view of a sample P4Sandbox, prior to any modification with additional task streams. The word *Sandbox* appears in the title bar. This P4Sandbox includes the default **local** stream (`//streams/local`) defined on the **Specify Central Server Settings** wizard page. The **Depot** tab view shows the files contained in the **local** and **mirror** streams. You can identify the currently active stream by the following:

- The active file icons in the **local** stream (`//streams/local/jblack/main`).
- The workspace icon in the **local** stream.



Verifying the connection

This section discusses how to verify that you have successfully configured a P4Sandbox connection using either the `p4` or the graphical client application.

For more information on specifying configuration settings, contact your Perforce Administrator and also see the appropriate guide for your implementation:

- *P4Eclipse Help*, "Configuring Perforce Server Connections and Working Offline"
- *P4 User's Guide*, "Configuring P4"

- *P4V Online Help*, "Connecting to the Perforce Server"
- *Using IDE Plug-ins,* "Basic Concepts: Configuring IDEs with plug-ins"

Using p4

To verify that you have successfully configured a P4Sandbox, type `p4 info` at the command line.

If your configuration settings are correct and a server is running on the specified host and port, you receive a message similar to the following. The `Sandbox broker version:` field indicates that a P4Sandbox is running.

```
User name: jblack
Client name: my_workspace
Client host: win-jblack
Client root: C:\Users\jblack\sandboxes\sandbox1666\client
Client stream: //streams/local
Current directory: c:\Users\jblack\sandboxes\sandbox1666
Client address: 127.0.0.1:56453
Server address: win-jblack:1667
Server root: c:\Users\jblack\sandboxes\sandbox1666\.p4sandbox
Server date: 2012/01/27 11:43:48 -0800 Pacific Standard Time
Server uptime: 00:47:30
Server version: P4D/NTX86/2012.1.MAIN-TEST_ONLY/400159 (2012/01/10)
Server license: none
Case Handling: insensitive
Sandbox broker version: P4SANDBOX/NTX86/2012.1.MAIN-TEST_ONLY/400282
Sandbox broker port: localhost:1666
```

If your configuration settings are incorrect, the following error message appears:

```
Perforce client error:
        Connect to server failed; check $P4PORT.
        TCP connect to <hostname> failed.
        <hostname>: host unknown.
```

Using a graphical client application

To verify that you have successfully configured P4Sandbox:

1. Launch the graphical client application.
2. Connect to `localhost:`*nnnn* where *nnnn* is the **Local Server** port that appears on the **Settings Summary** page of the P4Sandbox Configuration Wizard.

If your configuration settings are incorrect, an error message similar to the following appears:

```
TCP connect to localhost:1666 failed.
Connect: 127.0.0.1:1666: Connection refused
```

Removing a P4Sandbox

You must use `p4` to delete a P4Sandbox; you cannot delete a P4Sandbox using a graphical client application. Either use the command procedure discussed below or contact your Perforce Administrator for assistance.

Issuing p4sandbox delete

To delete a P4Sandbox:

Example: *This example shows how to delete a specific P4Sandbox. After you issue this command, you must enter `-y` to confirm the deletion.*

```
p4sandbox delete -r .p4sandbox
```

`p4sandbox delete` performs the following actions:

- Deletes all files located in the `.p4sandbox/` directory:
  - Local `p4d` database metadata
  - Versioned files
- Removes the P4Sandbox entry from the `.p4sandbox-list` file.

After you issue the `p4sandbox delete` command, use the appropriate OS-specific process to remove the remaining files from your machine.

If you want to remove files from your workspace, you must remove them manually; `p4sandbox delete` does not affect files in your workspace.

Important! You cannot reverse the `p4sandbox delete` command. However, you can usually restore a P4Sandbox if backups are available. For more information, type `p4sandbox help delete`.

Uninstalling a P4Sandbox

`p4sandbox delete` does not uninstall a P4Sandbox. After issuing `p4sandbox delete`, use the appropriate OS-specific process to remove the program from your machine.

Note When you perform an uninstall or upgrade with active P4Sandboxes running on your machine, the installer may prompt you either to stop all active P4Sandboxes or to reboot your machine to complete the process.

Chapter 3 Using P4Sandbox with the Command-Line Client

This chapter tells you how to use the Perforce Command-Line Client (`p4`) to perform P4Sandbox tasks. To perform these tasks, you must first configure a P4Sandbox; see "Setting Up P4Sandbox" on page 19.

This chapter provides basic information about commands you use with a P4Sandbox, including command-line syntax, arguments, and flags.

For a comparison of P4Sandbox and Git terms and tasks, see "Comparing Git and P4Sandbox commands" on page 88.

Overview

P4Sandbox uses two types of commands:

- P4Sandbox-specific commands: `p4sandbox` *`command`*
- P4 standard commands: `p4` *`command`*

  This category includes two subcategories: the additional and modified commands.

P4Sandbox-specific commands

P4Sandbox adds several commands to a category `p4sandbox` (listed in general usage order):

- `p4sandbox init`
- `p4sandbox start`
- `p4sandbox stop`
- `p4sandbox list`
- `p4sandbox delete`

Additional p4 commands

P4Sandbox adds several `p4` commands that work only within a P4Sandbox context (listed in alphabetical order):

- `p4 copyup`
- `p4 mergedown`
- `p4 remote`

- `p4 remotes`
- `p4 switch`

Modified p4 commands

P4Sandbox extends the behavior of the following `p4` commands when executed within a P4Sandbox context (listed in alphabetical order):

- `p4 admin stop`
- `p4 copy`
- `p4 counter`
- `p4 integrate`(`p4 integ`)
- `p4 merge`
- `p4 populate`
- `p4 pull`
- `p4 submit`

Using P4Sandbox commands

This section provides general usage information for the P4Sandbox commands.

Note All information in this chapter specifically describes a command's behavior while operating in a P4Sandbox context. For the modified commands, see the *Perforce Command Reference* for general information about a command and its behavior in a non-P4Sandbox context.

TCP connection requirement

You are not required to have an active TCP connection when issuing commands, unless you are copying changes to and from a shared service.

Prohibited mirror stream update commands

You cannot use the following `p4` commands to update a mirror stream with changes from the local stream:

- `p4 add`
- `p4 edit`
- `p4 delete`

- `p4 move`

You must use either `p4 copy` or `p4 integrate` to update the mirror stream.

Invalid commands

Because you are normally the sole user and administrator of your P4Sandbox, commands for groups and certain commands for administrators may be essentially invalid.

Accessing help

The following commands display lists containing P4Sandbox-specific commands and extended `p4` standard commands. The enhanced standard `p4` commands include additional P4Sandbox explanatory text at the bottom of the help documentation, in a section entitled `p4sandbox extension`.

To access P4Sandbox `p4` help documentation, issue one of the following commands:

- `p4sandbox help`
- `p4 help sandbox`

You must be connected to an active P4Sandbox to use `p4 help sandbox`. See "Starting P4Sandbox" below.

Starting a P4Sandbox

You must start a P4Sandbox to use it; it is not automatically started by any other process initialization.

To start a P4Sandbox, type:

```
p4sandbox start
```

Working in a specific client write mode

The client write mode you set for your P4Sandbox determines whether files are read-only or editable, and whether you must issue the `p4 edit` command to change files.

For more commands and information about working in `allwrite` or `noallwrite` mode, see "Client mode dependent commands" on page 94.

Merging down changes from the shared service

You can perform a mergedown of shared service changes to update the entire P4Sandbox or only the mirror stream.

Updating P4Sandbox with shared service changes

To merge down from the mirror stream, type:

```
p4 mergedown
```

Resolving a mergedown that has conflicts

When you perform a mergedown that results in conflicts, you must resolve and submit all conflicts to update P4Sandbox with the latest changes from the shared service.

Example: *This example shows how to update a P4Sandbox with shared service changes when conflicts exist.*

```
p4 mergedown
(receive conflicts)
p4 resolve
p4 submit
```

Note P4Sandbox performs specific logic when merging down similar and dissimilar shelved pending changelists; see "p4 mergedown" on page 77.

Updating only the mirror stream with shared service changes

When you are connected to your company's network, use `p4 pull` to retrieve the latest changes for eventual merging into your code, even if your P4Sandbox is not connected to the network. For example, you issue the `p4 pull` command before disconnecting from the company network for a business flight. On the flight, you work in your P4Sandbox, merging down the changes from the mirror stream to the local stream before working in the local stream.

To update the mirror stream with shared service changes, type:

```
p4 pull
```

Copying up changes to the shared service

When you copy up and submit changes from your local stream to your mirror stream, the mirror stream automatically copies up and submits changes to the shared service.

Example: *This example shows how to copy up changes to the shared service.*

```
p4 copyup
p4 submit
```

Note If you create a script to copy up changes to the mirror stream automatically, be sure to include logic to first validate the synchronization status of the shared service and mirror stream before copying changes from a local or task stream to the mirror stream.

If you implement P4Sandbox connected to a shared service, be sure to periodically copy up or shelve important work to the mirror stream. This insures that you have the latest file versions available as backups on the shared service, in case of failures or accidents.

Resolving a copy up that has conflicts

When a copyup attempt results in conflicts, you must correct the conflicts in the local stream before attempting to copy up changes again to the mirror stream.

Example: *This example shows how to resolve a copyup that results in conflicts. Note that you must resolve conflicts in the local stream and issue the* `p4 submit` *command twice—once to the local stream and once to the mirror stream.*

```
p4 copyup
```

(receive conflicts)

```
p4 mergedown
p4 resolve
```

(resolve in the local stream)

```
p4 submit
```

(attempt copy again)

```
p4 copyup
p4 submit
```

Working with streams

This section discusses how to use the P4Sandbox streams functionality.

Adding a task stream

To create and switch to a new stream, type:

```
p4 copy //streams/parent/... //streams/stream/...
```

Viewing stream information

To show the currently active stream name, type:

```
p4 switch
```

To list all streams in the P4Sandbox, type:

```
p4 streams
```

Switching between streams

When you switch between streams, P4Sandbox automatically shelves any in-progress work in the currently active stream before switching to the new stream.

To switch streams, type:

```
p4 switch stream name
```

Example: *This example shows how to determine the streams in a P4Sandbox, how to determine the currently active stream, and how to switch between streams (specifically, from the* `mirror` *stream to the* `local` *stream).*

```
C:\Users\jblack\sandboxes\sandbox1666>p4 streams
Stream //streams/bugbox development //streams/local 'bugbox'
Stream //streams/local development //streams/mirror 'local'
Stream //streams/mirror mainline none 'mirror'

C:\Users\jblack\sandboxes\sandbox1666>p4 switch
On stream //streams/mirror

C:\Users\jblack\sandboxes\sandbox1666>p4 switch local
Client WIN-JBLACK switched to //streams/local.
//streams/local/jblack/main/0224.txt#1 - unshelved, opened for edit
```

Copying changes between task streams

Copy or merge changes between task streams using one of these four commands:

- `p4 copy`
- `p4 integrate`

- `p4 merge`
- `p4 populate`

P4Sandbox automatically switches to the destination stream before performing any of these commands.

Deleting a task stream

You cannot completely delete a stream because Perforce always retains a stream's history. Deleting a stream only deletes the stream specification, not any of the files that were in the stream. The files and their history remain in Perforce and continue to appear in the shared service.

Deleting a stream is possible, however, by issuing the `p4 obliterate` and `p4 change -d` commands. For more information on using these commands, see the *Perforce Command Reference*.

Working with shelving

This section discusses how to use the P4Sandbox shelving functionality.

Note P4Sandbox propagates any shelving action that you perform while in a mirror stream to the shared service. This means you can have pending changelists with shelved files existing in both the shared service and your P4Sandbox.

P4Sandbox performs specific logic when merging down similar and dissimilar shelved pending changelists; see "p4 mergedown" on page 77.

For more information about this capability, see "Automatic and manual shelving" on page 16.

Shelving and unshelving in-progress work

P4Sandbox automatically shelves and unshelves in-progress work when you switch from one task to another by switching between streams; see "Switching between streams" on page 34.

To shelve work, type:

```
p4 shelve
```

This command invokes a text editor window and displays a changelist. Enter a changelist description and save the text file to complete the command.

Example: *This example shows how to shelve work to the shared service from a local stream.*

```
p4 copyup
p4 shelve
```

To unshelve work, type:

`p4 unshelve -s changelist#`

You must include the pending changelist number that contains the shelved files to complete the command.

Deleting a shelf

To delete a shelf, do the following:

- (Optional) Use `p4 unshelve -s changelist#` to unshelve any work you want to retain before deleting the shelf and its associated changelist.
- Ensure that the changelist does not contain open files. Issue `p4 opened -c changelist#` to view this information. Resolve any open files, such as by moving them to another changelist or reverting them.

Example: *This example shows how to delete a shelf and its changelist.*

```
p4 shelve -d -c changelist#
p4 change -d changelist#
```

Viewing shelf information

To list the shelves in a P4Sandbox, type:

`p4 changes -c client -s shelved`

Note To verify your active client, type:

`p4 client`

To display shelved content, type either of the following commands:

`p4 describe -S changelist#`

`p4 files @=changelist#`

Reviewing file and directory information

This section discusses how to retrieve information about changelists, files, and directories.

Determining files in your workspace

The table below list commands for reviewing changelist and file information.

Task	Command
List changelists	`p4 changes`
List changes for a specific directory	`p4 changes dir/...`
List shelved files for a specific changelist	`p4 describe -S changelist#`
List files for a specific directory	`p4 files dir/...`
List files that have been submitted to the mirror stream	`p4 files //streams/mirror/...`
List files for a specific changelist; shows shelved files	`p4 files @=changelist#`
Show changelist information for a specific file	`p4 filelog`
Show revision history for a specific file	`p4 filelog file`
List file information	`p4 fstat file`
List files currently open for any action but not yet submitted	`p4 opened`

List modified files	`p4 status`
	or
	`p4 reconcile -n`

Stopping a P4Sandbox

To stop a P4Sandbox, issue one of the following commands:

- `p4sandbox stop`
- `p4 admin stop`

See also

For more information about these commands and their options, see “P4Sandbox Command Reference” on page 63.

You can use these commands in scripts to automate your work. For more information about using Perforce commands in scripts, see the *P4 User’s Guide*, “Scripting and Reporting.”

Chapter 4 Using P4Sandbox with the P4V Integration

This chapter discusses how to use a graphical client application for certain P4Sandbox tasks and provides examples for common scenarios. To perform these tasks, you must first configure a P4Sandbox; see “Setting Up P4Sandbox” on page 19.

P4Sandbox tightly integrates with P4V and uses this integration to work with P4Eclipse and other IDEs. This means that after you implement P4Sandbox with P4Eclipse, you use P4V to perform your P4Sandbox tasks. Consequently, this chapter focuses on using P4Sandbox with P4V. For more information on the P4V interface, see the *P4V Online Help*.

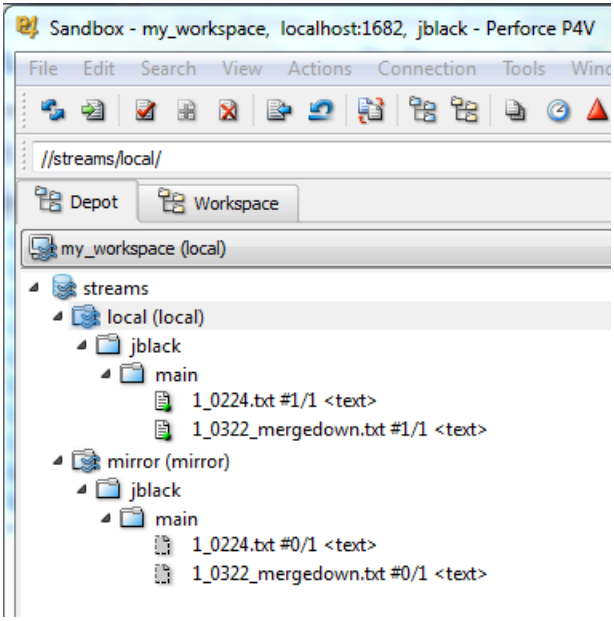
For more information on implementing P4Sandbox and P4V with other graphical client applications, see the online help that comes with your application or IDE Plug-in.

Starting a P4Sandbox

To start a P4Sandbox in P4V:

1. Launch the P4V **Open Connection** dialog.
2. Do one of the following:
   - Click **Connections** and select the P4Sandbox.
   - Enter the P4Sandbox **Server**, **User**, and **Workspace** information.
3. Click **OK**. The P4Sandbox window opens with the workspace connected to the local stream, as shown in the figure below.

Note If the **Streams** tab is active but the Stream Graph does not appear, click the **Show/hide filtering** icon.



You can also connect to your P4Sandbox while logged into a shared service. Go to **Connection > Open Recent**, and select the P4Sandbox.

Working in a specific client write mode

The client write mode determines whether files are editable (`allwrite` mode) or read-only (`noallwrite` mode). To determine a workspace's client write mode, verify the status of the `allwrite` check box in the workspace description region. If the check box is clear, the workspace is operating in `noallwrite` mode.

To view and edit a workspace's client write mode:

1. Go to **View** > **Workspaces** to open the **Workspace** tab.
2. Select the workspace.
3. Verify whether the `allwrite` check box is checked for the **Option** field of the workspace description region.
4. Click the **Edit** button to launch the **Workspace:** `<name>` window.
5. Click the **Advanced** tab.
6. Check or clear the `Allwrite` check box in the **File Options** region.

Alternately, context-click a workspace to access a context menu that includes view and edit options.

For more information, see the following:

- *P4 User's Guide*, "Configuring Workspace Options"
- *P4V Online Help*, "Managing Workspace Specifications"

Understanding P4Sandbox Stream Graph elements

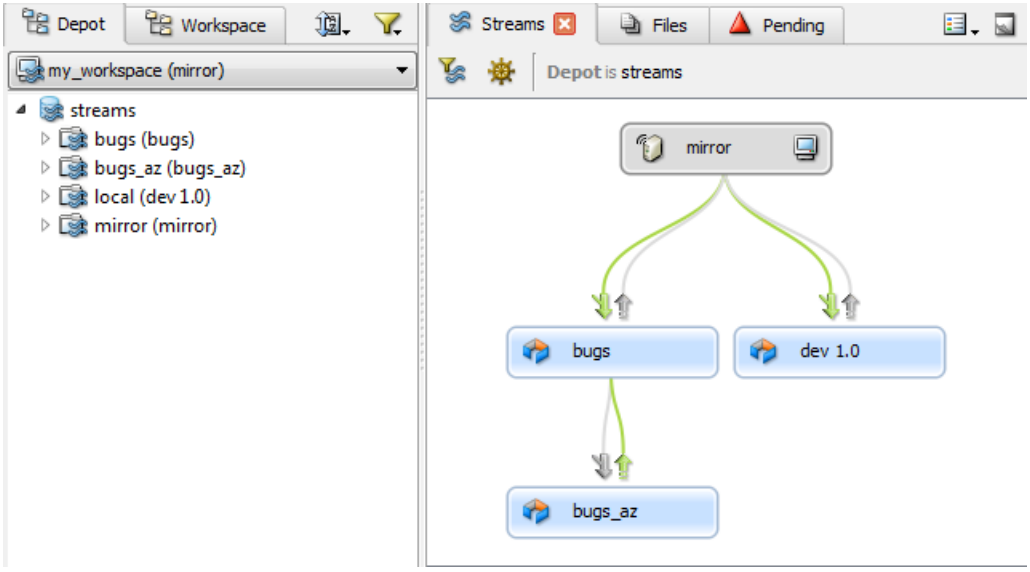
To display the mirror stream and local streams of the P4Sandbox in a P4V Stream Graph view, go to **View > Streams**.

The figure below shows a P4Sandbox Stream Graph, with the **Depot** and **Streams** views active.

A workspace icon (computer icon) in a stream indicates that the stream is tied to the current workspace. This is considered the active stream.

The arrow color indicates the following:

- Grey arrow indicates that no action is necessary.
- Red arrow indicates that conflicts exist between streams (or among multiple streams) that you must resolve.
- Green arrow indicates that there are changes between the streams.



Note To successfully copy up changes, you must first perform any necessary merge downs. For specific instructions on these tasks, see the following sections:

- "Merging down changes from the shared service" on page 46.
- "Copying up changes to the shared service" on page 48.
- "Copying changes among task streams" on page 51.
- "Merging down changes between task streams" on page 52.

In addition to the stream arrows, the **Tasks** field in the **Dashboard** tab shows pending tasks for the active stream. See "Resolving pending tasks" on page 53, below.

Verifying P4Sandbox streams workspace setting

Before you use P4Sandbox, confirm that P4V uses the same workspace when switching between streams.

To verify P4Sandbox's P4V streams workspace setting:

1. Go to **Edit > Preferences**. The **Preferences** window appears.
2. Select **Streams**.
3. Confirm that the **Use the same workspace and switch it between streams** option is chosen. If not, select it.
4. Click **OK**.

Using the context menus

Context-click a stream to access a menu of actions you can perform on a stream. A menu option's ***<name>*** field dynamically changes to indicate the active stream's name, such as ***mirror*** or ***local***. You can access P4Sandbox context menus from the following locations:

- Context-click a stream in the **Depot** tab to display a small menu of P4Sandbox actions.
- Context-click a stream in either of the following locations to display a large menu of P4Sandbox actions:
  - The **Stream Graph**.
  - The **Graph View Options** window. To display this window, click the **Show/hide filtering** icon available from the **Streams** tab.

The table below shows the available menu options in their display order. Note that the small menu includes only the **Merge/Integrate to <*name*>**, **Copy to <*name*>**, and **Branch** options.

Menu Option	Description
View Stream <*name*>	Displays the **Stream: <*name*>** window. View general stream settings and options information, such as stream name, root, and parent, owner, paths, and so on. Click **Edit** to access the **<*name*>: Edit** window (see the **Edit Stream <*name*>** description). P4V displays a warning message when you attempt to edit a mirror stream.
Pull from Central Server	Updates only the mirror stream with changes from the shared service. This action is equivalent to issuing `p4 pull` in `p4`. For more information, see "Updating only the mirror stream with shared service changes" on page 32. This menu option appears only in the mirror stream's context menu.
Merge/Integrate to <*name*>	Displays the **Merge/Integrate** window. For detailed field descriptions, see "Merging down changes from the shared service" on page 46.
Copy to <*name*>	Displays the **Copy** window. For detailed field descriptions, see "Copying up changes to the shared service" on page 48.
Branch	Displays the **Branch** window. This window is similar to the **Copy** window; however, there are only two **Branch method** specification options.
Work in this Stream	Displays only for streams currently not set as the active stream. Select to set a non-active stream as the active stream. An active stream has automatic shelving behavior. Alternately, drag the workspace icon into a stream to make it an active stream.

Menu Option	Description
New Workspace	Displays the **Workspace: New** page. Create a new workspace and define its basic and advanced settings.
Create New Stream from <*name*>	Displays the **Stream: New** page. Create a new stream and define its basic and advanced settings. See "Adding a task stream" on page 51.
Edit Stream <*name*>	Edit stream characteristics on the **Basic Settings** and **Advanced** tabs.
Delete Stream <*name*>	Displays the **Delete Stream Form** window with the stream name marked for deletion. This window is intended to confirm the stream deletion before the stream is actually removed. Click **Yes** to delete the stream. This action cannot be undone. Click **No** to cancel the deletion. Note: Deleting a stream only deletes the stream specification, not any of the files that were in the stream. The files and their history remain in Perforce and continue to appear in the Depot view. To hide deleted information in P4V, select the **Hide Deleted Depot Files** option in the **Search** > **Filter Depot** menu.
Diff Against	Displays the **Diff** window to compare differences between stream versions. Specify paths and versions to differentiate.

Menu Option	Description
Diff Against Parent	Displays the **Folder Diff** window to compare differences between files in two different folders and view detail, integration, label, and preview information for each individual file. Use the toolbar options to differentiate the files: • **Show identical file pairs** • **Show files without counterparts (unique)** • **Show file pairs with content differences** The window also provides the number of **Unique files** and **File differences.**
Label	Displays the **Label Files** window. Apply or remove a label from specified files or folders, and also set which revision to include or exclude.
Show In Depot Tree	Highlights the currently active stream in the **Depot** tab.
Print Preview Stream <*name*>	Displays **Preview of Stream <*name*>** window, with a report of general stream setting information, such as name, parent, type, description, and so on. Use the right-side **Fields** panel to set field display and default behavior.
Print Stream <*name*>	Displays the **Print** window, to print a stream report.
Refresh Stream List	Refreshes the graph with the known streams. When using P4Sandbox with both `p4` and a graphical client application, use this command to refresh the Stream Graph after creating a new stream through `p4`.
Refresh Stream <*name*>	Refreshes a specific stream to determine its integration status. When using P4Sandbox with both `p4` and a graphical client application, use this command to refresh the Stream Graph after integrating changes in an existing stream through `p4`.

Merging down changes from the shared service

You must merge down changes from the shared service before you can copy up any changes from a P4Sandbox local stream.

To merge down changes from the shared service to the ***local*** stream:

1. Context-click the ***local*** stream and select **Merge/Integrate to *'local'*.**
2. The **Merge/Integrate** window opens, with default settings. Accept or change these defaults. The table below describes the available fields.

> **Note** Because the **Merge/Integrate** window is dynamic, the available fields depend on the **Merge method** option you specify.

Menu Option	Menu Sub-option	Description
Merge method		
	Stream to stream	(Default) Select this option to enable P4V's automated logic to handle the merge details. The functionality automatically populates the **Source stream** and **Target stream** fields, as necessary.
	Specify source and target files	Select this option if you require some control over the merge details. Specify a **Source stream** from which to merge or integrate files and a **Target stream** to receive the files. Optionally, click **Save** to define and create a branch mapping specification for future integrations.
	Use branch mapping	Select this option if you require more control over the merge details; for example, if you are using a non-streams Perforce depot. Generally, most users should use the other two merge methods.
		Click the arrow between **Source** and **Target** to reverse the branch integration direction.

Menu Option	Menu Sub-option	Description
Options: Resolve and Submit		
	Add files to pending changelist	(Default) Automatically adds files to a new changelist with a standard description and uses the `Safe automatic resolve (no merging)` option. If you select the **Automatically resolve files after merging** check box, select a **Resolve option** value: • `Safe automatic resolve (no merging)` • `Automatic resolve (allow merging)` • `Accept source` • `Accept target` If you select the `default` **Pending changelist** option, you cannot enter a changelist description nor link jobs.
	Automatically submit after resolving	Automatically submits files to a new changelist with a default changelist description, and also automatically resolves files after merging. You can edit the **Resolve option** value.

Menu Option	Menu Sub-option	Description
Options: Filter		
	Revision range and files/folders	For the **Revisions to merge** field, specify either: • `All revisions` • `Revisions up to`, and limit the revisions copied by an available value. • `Revisions equal to` • `Revisions to/from`, and specify parameters You can also opt to merge specific files and folders.
	Selected changelists	You can filter to select specific changelists to use for the merge.
Options: Advanced	Available options depend on the **Merge method** you specify.	For information on these advanced merge and integration options, see the *P4V Online Help* topic "Merging Files Between Codelines."
Preview		Click to review the merge result before performing the merge.

Copying up changes to the shared service

When a stream that contains updated files that can be copied up to the mirror stream, the Stream Graph displays the following:

- A green arrow between the streams.
- A `Copy to Mirror` link in the **Tasks** field of the **Dashboard** tab.

To copy files to the mirror stream:

1. Click the `Copy to Mirror (n change)` link.

2. The **Copy** window opens, with default settings. Accept or change these defaults. The table below describes the available fields.

Note As the **Copy** window is dynamic, the available fields depend on the **Copy method** option you specify.

If you are scripting an automatic update process, be sure to include logic to first validate the synchronization status of the shared service and mirror stream before copying changes from a local or task stream to the mirror stream.

Menu Option	Menu Sub-option	Description
Copy method		
	Stream to stream	(Default) Select this option to enable P4V's automated logic to handle the copy details. The functionality automatically populates the **Source stream** and **Target stream** fields, as necessary.
	Specify source and target files	Select this option if you require some control over the copy details. Specify a **Source stream** from which to merge or integrate files and a **Target stream** to receive the files. Optionally, click **Save** to define and create a branch mapping specification for future integrations.
	Use branch mapping	Select this option if you require more control over the copy details; for example, if you are using a non-streams Perforce depot. Generally, most users should use the other two copy methods. Click the arrow between **Source** and **Target** to reverse the branch integration direction.

Menu Option	Menu Sub-option	Description
Options: Submit		
	Add files to pending changelist	If you specify a new pending changelist with this option, you can one or both of the following actions: • Enter a changelist description or accept the default. • Add previously linked jobs. If you specify the `default` changelist, you cannot do either of the above actions.
	Automatically submit copied files	(Default) Automatically submits files to a new pending changelist; includes a default changelist description. You can also opt to add previously linked jobs.
	Set as Default Settings	Saves your selected settings as the default copy options and displays them in the **Current settings** field until you define new settings.
Options: Filter		
	Revision range and files/folders	For the **Revisions to copy** field, specify either: • All revisions • Revisions up to, and limit the revisions copied by an available value. You can also opt to copy specific files and folders.
	Selected range of changelists	You can copy by filtering on specific changelists.

Menu Option	Menu Sub-option	Description
Options: Advanced		
	Available options depend on the **Copy method** you specify.	For information on these advanced merge and integration options, see the *P4V Online Help* topic "Merging Files Between Codelines."
	Set as Default Settings	See above for **Submit**.
Preview		Click to review details before performing the copy.

Adding a task stream

To create a task stream:

1. In the Stream Graph, context-click a stream.
2. Select **Create New Stream from *'name'***. The **Stream: New** window displays.
3. Enter a name and set stream options.
4. The Stream Graph displays the new task stream with the workspace icon. When you create a new task stream, P4V automatically sets it as the active stream.

Copying changes among task streams

To copy changes:

1. Check out a file, edit, and submit.

   In the Stream Graph, green arrows display between the stream containing the edited file and all affected child and parent streams.
2. Context-click the stream and select **Copy to <*name*>**.
3. The **Copy** window opens, with default settings. Accept or change these defaults. See the table in "Copying up changes to the shared service" on page 48, which describes the available fields.

Merging down changes between task streams

To merge down changes from a source stream to a target stream:

1. Context-click the target stream and select **Merge/Integrate to <*name*>**. The **Merge/Integrate** window opens.
2. Specify all necessary merge details. See the table in "Merging down changes from the shared service" on page 46, which describes the available fields.

Removing a task stream

To delete a task stream:

1. Context-click the stream.
2. Select **Delete Stream *'name'***.

Note the following exceptions:

- You cannot delete an active task stream (one displaying a workspace icon). You must first switch the workspace to another stream.
- You cannot delete a task stream that has any number of child streams. Either first delete the child, or assign it to a new parent using one of the following methods:
  - Edit the child's **Basic Settings** by context-clicking and selecting **Edit Stream *'name'***.
  - Drag the child to a new parent.

> **Note** Deleting a stream deletes its stream specification, and the Stream Graph does not display deleted streams. However, as the files, history, and changelists that occurred on that stream still remain recorded in Perforce, the streams are still visible in the **Depot** view.

Switching streams and shelving work

When you switch streams, P4V automatically shelves any open files in the inactive stream. Conversely, for an active workspace, P4Sandbox unshelves (checks out) any previously shelved files.

You cannot directly shelve a specific file or set of files when working in a graphical client application. P4Sandbox activates the automatic shelving behavior only when you switch streams.

To switch streams: Go to the Stream Graph view, select the workspace icon and drag it to the desired local stream.

To view shelved files: Click the **Pending** tab and expand the changelist that contains the shelved file.

Note P4Sandbox performs specific logic when merging down similar and dissimilar shelved pending changelists; see "p4 mergedown" on page 77.

Resolving pending tasks

The **Dashboard** tab's **Tasks** area displays hyperlinks to pending P4Sandbox tasks, in descending order of operation. P4V displays the following general P4Sandbox task categories (listed here in alphabetical order):

- Get latest revisions
- Merge to ***stream name***
- Promote to ***stream name***
- Resolve conflicts
- Set workspace
- View jobs
- View pending changelists

Click a hyperlink to manage the task. P4V displays the appropriate windows for you to complete the necessary action.

Closing a P4Sandbox connection

To close P4Sandbox in P4V, go to **Connection** > **Close Connection**.

Stopping a P4Sandbox

You cannot stop a P4Sandbox using a graphical client application; instead, you must use `p4`. For more information, see "Stopping a P4Sandbox" on page 37.

Deleting a P4Sandbox

You cannot delete a P4Sandbox using a graphical client application; instead, you must use `p4`. For more information, see "Removing a P4Sandbox" on page 28.

See also

See the following videos on the Perforce YouTube channel:

- "Introduction to P4Sandbox": `http://youtu.be/zz_10LfTPEA`
- "P4Sandbox Private Branching": `http://youtu.be/n2aJPlWPPPA`

See also the P4V documentation at:
`http://www.perforce.com/documentation/perforce_technical_documentation`

- *Getting Started With P4V*
- *P4/P4V Cheat Sheet*
- *P4V Online Help*

Chapter 5 Administering P4Sandbox

We designed P4Sandbox to require minimal administrative oversight by both users and Perforce Administrators. Generally, if you encounter an issue while using P4Sandbox, contact your Perforce Administrator.

Understanding prohibited and unnecessary tasks

Note the following prohibited and unnecessary tasks:

- Do not run `p4 protect` to create a protections table. Issuing this command harms P4Sandbox's internal protections table.
- Do not create scripts to run triggers. Running such scripts harms P4Sandbox's internal trigger logic and behavior and may break the interoperability of a P4Sandbox workspace specification and its shared repository.

  However, you can run daemons, such as a review daemon.
- You cannot run P4Sandbox as a Windows service.
- You do not need to perform any license management tasks because P4Sandbox does not require a license file.

Managing your P4Sandbox: user tasks

To maintain your P4Sandbox, do the following:

- Avoid running an antivirus program in the P4Sandbox directory.
- Verify that P4Sandbox is secure, particularly if you work remotely and transfer proprietary information between P4Sandbox and a shared service. Your Perforce Administrator can confirm that your implementation complies with Perforce's security settings and your company's security policies and procedures.
- Check that you periodically back up P4Sandbox to an external storage device. See "Supporting P4Sandbox: Backup and Recovery" below.
- Confirm that P4Sandbox automatically creates checkpoints and journals and stores them in the `.p4sandbox` directory. Your Perforce Administrator needs these files to restore P4Sandbox.

For more information about backup recovery, checkpoints, journaling, and security, contact your Perforce Administrator.

Editing the file pulling interval

You define the periodic file pulling interval during the initial P4Sandbox configuration.

To edit this setting, use one of the following methods:

- In P4V, select a new pulling interval option on the **Replication Settings** page of the P4Sandbox Configuration Wizard; see "Using the configuration wizard" on page 22.
- In `p4`, issue `p4 counter p4sandbox_schedule_copy` and adjust the interval. See "p4 counter" on page 68.

Performing miscellaneous tasks using p4

The following administrative tasks require you to issue `p4` commands. If you do not use `p4`, contact your Perforce Administrator for help.

This section discusses:

- Configuring Unicode mode
- Managing jobspecs and jobs
- Supporting P4Sandbox: Backup and Recovery

For more information, see the following in the *Perforce System Administrator's Guide*:

- "Managing Unicode Installations"
- "Customizing Perforce: Job Specifications"
- "Supporting Perforce: Backup and Recovery"

Configuring Unicode mode

To configure P4Sandbox to operate in Unicode mode, see the Usage Notes for "p4sandbox init" on page 70.

Managing jobspecs and jobs

P4Sandbox propagates only job fixes, not jobs. You must manually copy jobs between the shared service and your P4Sandbox. Additionally, you can only submit changelists for job fixes that meet certain conditions; see "p4 submit" on page 86 for these conditions.

If your use of P4Sandbox requires jobs, do one or both of the following:

1. Pipe the job template from the shared service to your P4Sandbox using the following command:

```
p4 -p <shared service's P4PORT> jobspec -o | p4 -p <sandbox server's
P4PORT> jobspec -i
```

2. Edit the job template using `p4 jobspec`, as necessary.

Warning! Improper modifications of the Perforce job template can lead to corruption of your server's database. Before modifying any of the existing job functionality, be sure to read the chapter "Customizing Perforce: Job Specifications," in the *Perforce System Administrator's Guide*.

Supporting P4Sandbox: backup and recovery

The `.p4sandbox` folder contains all of your `db.*` metadata files and your versioned files. It also contains the three most recent checkpoint files.

P4Sandbox automatically creates checkpoint and journal files with the prefix `rolling` once per day based on the `p4sandbox_schedule_checkpoint` counter. For example: `rolling.chk.0` and `rolling.jnl.0`.

Backing up a P4Sandbox

To back up a P4Sandbox:

1. (Optional) Issue `p4sandbox stop`.

   This command ensures that the `db.*` metadata files are consistent and faster to restore from than restoring from a pair of checkpoint and journal files.

2. Copy the `.p4sandbox` folder to your backup media.

Note the following:

- You can use your existing external backup system to store P4Sandbox backups.
- Consider using scripts to automate this task.
- Run `p4 verify` to validate the backup; type:

```
$ p4sandbox start -r path to backup copy
$ p4 -p path to backup copy verify
$ p4sandbox stop -r path to backup copy
```

Restoring a P4Sandbox

To restore a P4Sandbox:

1. Copy a `.p4sandbox` folder from backup media to your machine's hard disk.
2. Issue `p4sandbox start -r path to .p4sandbox` to start your restored P4Sandbox.

Managing P4Sandbox users: Perforce Administrator tasks

Perforce Administrators have minimal oversight of P4Sandbox users, and can perform the following tasks:

- Determine who is using a P4Sandbox by issuing `p4 clients` and scanning for lines that start with `p4sandbox-`, such as the following:

  `p4sandbox-`*`username`*`-`*`computername`*

- Determine which files P4Sandbox users have accessed and when the files were checked out and checked back into a shared service by analyzing the following information from the client list's view mapping:
  - *Update time:* Shows when files were copied from the shared service to a P4Sandbox.
  - *Access time:* Shows when files were copied from a P4Sandbox to the shared service.
  - *View:* Shows which files are currently copied into a P4Sandbox.
  - *Root:* Shows the root of a P4Sandbox's workspace on the user's computer.
- Prohibit P4Sandbox implementations by adding a trigger to refuse workspaces on a global or individual basis.
- Perform standard administration, such as restoring an accidentally deleted P4Sandbox, or managing the P4Sandbox broker (`p4sandbox`) and Perforce Server (`p4sandbox-p4d`) distribution.

For more information on performing administrator tasks, see the *Perforce System Administrator's Guide* and the *Perforce Command Reference*.

Warning! Administrators must ensure that all new and existing form trigger scripts that run on shared services used by P4Sandbox do not affect the P4Sandbox workspace specification.

P4Sandbox creates a workspace specification that begin with `p4sandbox-` when creating the remote depot (see "p4 remote" on page 80). If the workspace specification is altered in any way (either the name or contents), P4Sandbox will be unable to work properly with its shared repository.

For more information, see the *Perforce System Administrator's Guide*: Scripting Perforce, Triggers and Daemons, Triggering on Forms and also the Perforce Command Reference: `p4 triggers`.

Chapter 6 Troubleshooting P4Sandbox

Troubleshooting

This chapter lists solutions to issues you may encounter when using P4Sandbox.

For help with using P4Sandbox, contact your Perforce Technical Support representative at `support@perforce.com`.

Localhost connection error

When you execute a command, P4Sandbox returns a Perforce client error message similar to the following:

```
Perforce client error:
TCP connect to localhost:1999 failed.
connect: 127.0.0.1:1999: Connection refused
```

This message that indicates P4Sandbox is disconnected and probably stopped. To restart P4Sandbox:

1. Issue a stop command to ensure that the P4Sandbox is actually stopped:
   - `p4`: No command necessary
   - P4V: **Connection > Close Connection**
2. Restart P4Sandbox:
   - `p4`: `p4sandbox start`
   - P4V: **Connection > Open Recent** > ***localhost connection to P4Sandbox***

P4Sandbox does not relaunch in P4V

P4Sandbox successfully launched during the initial configuration. But when you restart P4V, your P4Sandbox does not relaunch automatically.

This behavior indicates that P4Sandbox is disconnected from P4V. To restart P4Sandbox:

1. Click the P4Sandbox Configuration icon.
   - Mac OS X: **Applications >** `p4sandbox-config.app`
   - Windows: **Start > All Programs > Perforce**
   - P4V: **Connection > Configure Sandbox**

2. Click **Next** to go through the **P4Sandbox Configuration Wizard** pages and restart the P4Sandbox.
3. In P4V, go to **Connection > Open Recent** to select your localhost server.

P4Sandbox and P4V password issue

When you are logged into P4V and are attempting to perform local stream to mirror stream tasks, you receive the following error message:

```
Perforce password (P4PASSWD) invalid or unset.
Central server for remote depot 'remote' at address <host:port> is not
currently accessible.
```

This message indicates that you are not logged into the shared service. To log into the shared service:

1. Type the following command in `p4`:

   `p4 -p <shared service port setting> login`

   For example:

   `p4 -p play:1999 login`
2. When prompted, enter your Perforce password.

Cannot copy up files

You cannot copy up files from the P4Sandbox to the mirror stream.

Verify that you have merged down the latest changes from the shared service to the local stream. See the following sections:

- `p4` users: “Updating P4Sandbox with shared service changes” on page 32.
- graphical client application users: “Merging down changes from the shared service” on page 46.

Files shelved to an incorrect stream

You accidentally shelved files to a different stream than the intended stream.

Fix this issue using one of the following methods:

p4 print method

1. Switch to the desired stream; type:

   ```
   $ p4 switch //streams/right_stream
   ```

2. For each file that you want, type the following `p4 print` command sequence:

   ```
   p4 print -o <local file system path> <depot path>@=<shelved changelist number>
   ```

   ```
   $ p4 print -o path/to/file.cpp //streams/wrong_stream/path to file/file.cpp@=1234
   ```

   ```
   $ p4 print -o path/to/file.h   //streams/wrong_stream/path to file/file.h@=1234
   ```

3. Open these files (and any others) for edit; type:

   ```
   $ p4 reconcile
   ```

   The `p4 reconcile` command opens the following for edit:

   - All the files that you printed in step 2.
   - Any files in the directory that you have modified without running `p4 edit`.

Zip method

1. While still in the incorrect stream, make an archive of the directory that contains the files you want. Zip or duplicate the directory.
2. Switch to the desired stream.
3. Replace the directory contents with the archived content.
4. Open the appropriate files for edit; type:

   ```
   $ p4 reconcile
   ```

   The `p4 reconcile` command opens the following for edit:

   - All the archived content files.
   - Any files in the directory that you have modified without running `p4 edit`.

Appendix A P4Sandbox Command Reference

This chapter discusses P4Sandbox commands and gives specific command syntax. It also provides tables that list equivalent Git commands.

Extended p4 commands

P4Sandbox extends the functionality of the following standard `p4` commands:

- `p4 admin stop`
- `p4 copy`
- `p4 counter`
- `p4 integrate`
- `p4 merge`
- `p4 populate`
- `p4 pull`
- `p4 submit`

All information in this chapter describes a command's behavior while operating in a P4Sandbox context. See the *Perforce Command Reference* for general information about a command and its behavior in a non-P4Sandbox context.

p4 admin stop

Synopsis

Stops a connected P4Sandbox.

Syntax

```
p4 admin stop
```

Description

`p4 admin stop` requires an active TCP connection to stop a connected P4Sandbox.

For complete command documentation, see the *Perforce Command Reference*, `p4 admin`.

Options

None

Related Commands

Scenario	Command
To stop a P4Sandbox that is not connected to a TCP connection	`p4sandbox stop`

p4 copy

Synopsis

Copy changes from a source stream to a target stream; also, create the target stream, as necessary.

Syntax

```
p4 copy [options] fromFile[revRange] toFile
p4 copy [options] -S stream [-P parent] [-F] [-r] [toFile[revRange] ...]
```

Description

`p4 copy` performs the following actions:

- Copies changes from a source stream to a target stream.
- Automatically creates and configures the target stream in the P4Sandbox, if the target stream does not exist
- Automatically switches your workspace to the target stream.

Note that when the source refers to files in a remote depot (that is, files in a shared service), the target must refer to a location within a new or existing mirror stream in the P4Sandbox. P4Sandbox records a mapping that links these two locations as part of the copy process.

For complete command documentation that includes global options, see the *Perforce Command Reference*, `p4 copy`. See also "Connected P4Sandbox copy restrictions" on page 16 in the Overview.

Options

Option	Meaning
fromFile[*revRange*] *toFile*	See the comprehensive description for these arguments in the *Perforce Command Reference* for `p4 integrate`.
`-S stream [-P parent] [-F] [-r] [toFile[revRange] ...]`	P4Sandbox automatically switches your workspace to the destination stream before integrating. If the target is a mirror stream, P4Sandbox automatically switches your workspace back to the source stream.

Related Commands

Scenario	Command
To retrieve recent changes from a shared service	`p4 pull`
To copy changes from the current stream to the parent stream	`p4 copyup`
To integrate files from a source to a target and automatically create target streams	`p4 integrate`
To merge changes between streams	`p4 merge`
To branch a set of files as a one-step operation	`p4 populate`

p4 copyup

Synopsis

Copy changes from the current stream to its parent stream.

Syntax

```
p4 copyup
```

Description

`p4 copyup` copies changes from the current stream to its parent.

`p4 copyup` does not submit. After issuing this command, you must execute a separate `p4 submit` command.

Options

None

Related Commands

Scenario	Command
To submit files from a mirror stream to a shared service	`p4 submit`
To merge changes from the parent stream to the current stream	`p4 mergedown`

p4 counter

Synopsis

Access, set, increment, or delete a persistent variable.

Syntax

`p4 counter p4sandbox_countername value`

Description

Use the `p4 counter` command with the provided P4Sandbox-specific options to manage default backup and pulling interval behavior.

For complete command documentation that includes global options, see the *Perforce Command Reference* for `p4 counter`.

Options

Option	Meaning
`p4sandbox_max_count_checkpoint`	Controls how many checkpoints and journals P4Sandbox retains. The default is `3`.
`p4sandbox_schedule_checkpoint`	Controls how frequently P4Sandbox runs a checkpoint and journal rotation in the background. The value is in hours and the default is `24` (that is, daily). A blank (no value) or a zero (`0`) value indicates that P4Sandbox does not perform automatic checkpoint and journal rotation. We recommend that you do not disable this feature.
`p4sandbox_schedule_copy`	Controls how frequently P4Sandbox runs a `p4 pull all` command in the background. The value is in hours and the default is blank, where a blank (no value) or a zero (`0`) indicates that P4Sandbox should not perform automatic pulls.

p4sandbox delete

Synopsis

Permanently delete a P4Sandbox.

Syntax

`p4sandbox delete [-y] [-r` *`p4dDir`* `| [-a | all] ]`

Description

`p4sandbox delete` performs the following actions:

- Permanently removes a P4Sandbox. This includes the entire *`p4dDir`* containing all metadata and versioned files.
- Permanently removes the P4Sandbox entry from the list of known P4Sandboxes contained in the platform-specific `.p4sandbox-list` file.
- Leaves any workspace files intact.

Options

Option	Meaning
`-r` *`p4dDir`*	Specify the server root directory for a P4Sandbox.
`-a`	Delete all P4Sandboxes listed in the `p4sandbox-list`. This option also removes the `p4sandbox-list` file.
`all`	Synonym for `-a`.
`-y`	Confirm deletion.

Usage Notes

- By default, `p4sandbox delete` displays a preview of the results. To execute the operation, you must specify the `-y` flag.
- You must specify one of the P4Sandboxes options; either `-r` *`p4dDir`*, `-a`, or `all`.
- When you delete a single P4Sandbox, specify the server root directory as an absolute or relative path. For example:

```
C:\Users\jblack\sandboxes> p4sandbox delete -y -r
    C:\Users\jblack\sandboxes\sandbox1666\.p4sandbox
```

Related Commands

Scenario	Command
To create and start a new P4Sandbox.	`p4sandbox init`

p4sandbox init

Synopsis

Create and start the first instance of a new P4Sandbox.

Syntax

`p4sandbox init [options]`

Description

If you do not specify options for this command, `p4sandbox init` creates a local P4Sandbox in directory `.p4sandbox`, listening on port `localhost:1666`, with a workspace root in the current directory.

If you set the `P4CONFIG` environment variable, this command also creates `.p4config` file (which may use a user-defined name) in the current directory and sets up a workspace.

Options

Option	Meaning
`-p port`	Specify the port number to which P4Sandbox listens. Default is `1666`.
`-r p4dDir`	Set the server root directory. Default is `.p4sandbox`.
`-R clientDir`	Set the path to the workspace root for the client that `p4sandbox init` creates. Default is the current working directory. Note: `-R clientDir` implies `-r clientDir/.p4sandbox` unless `-r` also specified.
`-q`	Suppress startup messages.
`-u user`	Specify the user to create, overriding the value of `$P4USER` in the environment.
`-c client`	Specify the client to create, overriding the value of `$P4CLIENT`. If `$P4PORT` is defined and P4Sandbox can read a similarly named client from that server, `p4sandbox init` copies options from the client on `$P4PORT` into the new client on the P4Sandbox.
`-w stream`	Specify the initial local stream to create. Default is `//streams/local`

Usage Notes

- Use `p4sandbox init` only for the initial P4Sandbox creation. Use `p4sandbox start` to launch an existing P4Sandbox.

- If you issue `p4sandbox init` from an existing `.p4sandbox` directory, the command fails.
- If you issue `p4sandbox init` from another Perforce workspace directory (like an existing shared service workspace directory), P4Sandbox may partially overlap or completely overwrite the existing workspace's configuration.
  - If the workspace directory originally was the shared service directory, it is now a P4Sandbox root directory, and this scenario may result in clobber issues when attempting to synchronize changes.
- To configure P4Sandbox for Unicode mode, use the `p4sandbox init` command with the appropriate options. For example, to configure P4Sandbox for UTF-32 mode, type:

  ```
  p4sandbox -C UTF32 init
  ```

  You must specify this mode during the initial P4Sandbox creation; you cannot retroactively reconfigure a P4Sandbox to operate in Unicode mode.

  For more information on specifying character sets, see the `P4CHARSET` command in the *Perforce Command Reference*.
- If you work on a Windows machine, the Windows command-line interface does not provide feedback for this command.
- For more information on `.p4config` files, see the following:
  - `P4CONFIG` in the *Perforce Command Reference*
  - *P4 User's Guide*, "Configuring P4," Using Config Files

p4 integrate

Synopsis

Integrate files from a source to a target; automatically create target streams, when necessary.

Syntax

`p4 integrate [options] fromFile[revRange] toFile`

`p4 integrate [options]-S stream [-r] [-P parent] [file[revRange] ...]`

Description

`p4 integrate` (`p4 integ`) creates the stream for the target when necessary, and then switches the workspace to the target stream.

In the case where the source is a path in a remote depot (that is, a shared service), `p4 integrate` defines a mapping from the remote depot path to a corresponding location in a mirror stream in the P4Sandbox. The target must be a path into a new or existing mirror stream.

For complete command documentation that includes global options and flags, see the *Perforce Command Reference* for this command and also `p4 resolve`. See also "Connected P4Sandbox copy restrictions" on page 16 in the Overview.

Options

Option	Meaning
fromFile[*revRange*] *toFile*	See the comprehensive description for these arguments in the *Perforce Command Reference* for `p4 integrate`.
`-S` *stream* [`-r`]`-P` *parent* *[file[revRange] ...]*	P4Sandbox automatically switches your workspace to the destination stream before integrating. If the target is a mirror stream, P4Sandbox automatically switches your workspace back to the source stream.

Usage Notes

- You can use either `p4 integ` or `p4 stream` to create new streams (such as task streams) for your P4Sandbox. We strongly recommend using `p4 integ` as this command automatically handles more of the stream creation work for you.
- A message appears during the initial integration process when there are file case conflict issues. Note that the message does not appear for subsequent integrations.

Related Commands

Scenario	Command
To copy changes from a source stream to a target stream	`p4 copy`
To copy changes from the current stream to its parent stream	`p4 copyup`
To merge changes between streams	`p4 merge`
To branch a set of files as a one-step operation	`p4 populate`

p4sandbox list

Synopsis

Show all defined P4Sandboxes.

Syntax

```
p4sandbox list [ -a | all ]
```

Description

`p4sandbox list` with no defined options displays all valid existing P4Sandboxes, excluding any that you have deleted or moved.

`p4sandbox list -a` or `all` lists all P4Sandboxes that you created, whether they are valid (currently existing) or invalid (deleted or moved).

Options

Option	Meaning
`-a`	Display all P4Sandboxes listed in the `p4sandbox-list` file, regardless of their validity.
`all`	Synonym for `-a`.

Related Commands

Scenario	Command
List all connections to shared services.	`p4 remotes`

Related Commands

Scenario	Command
To create and start a new P4Sandbox.	`p4sandbox init`
To start an existing P4Sandbox.	`p4sandbox start`

p4 merge

Synopsis

Merge changes between streams.

Syntax

```
p4 merge [options] fromFile[revRange] toFile
p4 merge [options] -S stream [-P parent] [-F] [-r] [toFile[revRange]
...]
```

Description

`p4 merge` creates the stream for the target when necessary, and then switches your workspace to the target stream.

If the source is a path in a remote depot (meaning, a shared service path), `p4 merge` defines a mapping from the remote depot path to a corresponding location in a mirror stream in the P4Sandbox. The target must be a path into a new or existing mirror stream.

If the shared service depot is a streams depot, P4Sandbox maps the entire source stream to the mirror stream and automatically updates this mirror stream mapping during each `p4 pull`.

Options

Option	Meaning
fromFile[*revRange*] *toFile*	See the comprehensive description for these arguments in the *Perforce Command Reference* for `p4 integrate`.
-S *stream* [-r]-P *parent* *[file[revRange] ...]*	P4Sandbox automatically switches your workspace to the destination stream before integrating. If the target is a mirror stream, P4Sandbox automatically switches your workspace back to the source stream.

P4Sandbox automatically switches to the destination stream before merging.

For complete command documentation that includes global options and flags, see the *Perforce Command Reference* for this command and also `p4 resolve`. See also "Connected P4Sandbox copy restrictions" on page 16 in the Overview.

Related Commands

Scenario	Command
To merge changes from the parent stream to the current stream	`p4 mergedown`
To copy changes from a source stream to a target stream	`p4 copy`
To copy changes from the current stream to its parent stream	`p4 copyup`
To branch a set of files as a one-step operation	`p4 populate`

p4 mergedown

Synopsis

Merge changes from a parent stream to a valid child stream.

Syntax

```
p4 mergedown
```

Description

`p4 mergedown` merges unintegrated changes from the current stream's parent stream to the current stream.

`p4 mergedown` internally uses `p4 resolve -am` to resolve conflicts, and then submits the files. If `p4 mergedown` encounters a conflict it cannot resolve, it stops operating on that particular issue and continues to the next conflict. You must manually resolve all remaining pending conflicts and submit them to complete the merge.

`p4 mergedown` integrates with the P4Sandbox shelving functionality to handle similar and dissimilar pending changelists. When you shelve a pending changelist to the mirror stream, this also shelves the pending changelist to the shared service. However, when another user submits or deletes the shelved pending changelist on the shared service, one of the following behaviors happens when you next merge down changes:

- If the other user does not modify the pending changelist on the shared service, `p4 mergedown` performs the appropriate action (submit or delete) to the corresponding pending changelist in the mirror stream.
- If the other user does modify the pending changelist on the shared service (for example, by adding files), `p4 mergedown` does not perform any action to the pending changelist in the mirror stream, because the changelists are now dissimilar.

Options

None

Related Commands

Scenario	Command
To copy changes from the current stream to the parent stream	`p4 copyup`

p4 populate

Synopsis

Branch a set of files as a one-step operation.

Syntax

```
p4 populate [options] fromFile[revRange] toFile
p4 populate [options] -S stream [-P parent] [-F] [-r] [toFile[rev] ...]
```

Description

For complete command documentation that includes global options and flags, see the *Perforce Command Reference*, `p4 populate`. See also "Connected P4Sandbox copy restrictions" on page 16 in the Overview.

Options

Option	Meaning
fromFile[*revRange*] *toFile*	See the comprehensive description for these arguments in the *Perforce Command Reference* for `p4 integrate`.
`-S` *stream* [`-r`]`-P` *parent* *[file[revRange] ...]*	P4Sandbox automatically switches your workspace to the destination stream before integrating. If the target is a mirror stream, P4Sandbox automatically switches your workspace back to the source stream.

Related Commands

Scenario	Command
To copy changes from a source stream to a target stream	`p4 copy`
To merge changes between streams	`p4 merge`
To integrate files from a source to a target and automatically create target streams	`p4 integrate`

p4 pull

Synopsis

Update a mirror stream with recent changes from a shared service.

Syntax

```
p4 pull
p4 pull stream
p4 pull -a
```

Description

`p4 pull` copies recent changes from a shared service to a mirror stream. Note that this command updates *only* the mirror stream, and not any associated (child) local streams.

`p4 pull` without any arguments updates the associated (parent) mirror stream of the current stream.

Requires a network connection to a shared service.

For complete command documentation about Perforce replica servers and their use of `p4 pull`, see the *Perforce Command Reference*, `p4 pull`.

Options

Option	Meaning
stream	Update a specified mirror stream
-a	Update all mirror streams

Related Commands

Scenario	Command
To copy changes from a source stream to a target stream	`p4 copy`
To copy changes from the current stream to the parent stream	`p4 copyup`
To merge changes from a parent stream to its child stream	`p4 mergedown`

p4 remote

Synopsis

Create a remote depot to and P4Sandbox clients on a shared repository.

Syntax

```
p4 remote [-u user] [-B remote-depot-name] -p p4port
p4 remote -d -B remote-depot-name
```

Description

`p4 remote` performs the following functions:

- Creates a remote depot, which is a reference to a shared repository.
- Creates an inalterable client specification on the shared repository to enable data exchange between the local P4Sandbox and the shared repository.

  A P4Sandbox client specification follows the format:

  `p4sandbox-<user>-<host><seq>`

  - `<user>` is the `P4USER` (or `-u`) value.
  - `<host>` is the client's hostname.
  - `<seq>` is an optional numeric string appended to provide uniqueness, if required. (In specific cases of very long client names, P4Sandbox may truncate the trailing right-side end to provide room for the numeric string.)

After you create a remote depot connection, issue the following command to create a mirror stream:

```
p4 merge //remote/<remote depot path>/... //streams/mirror/...
```

Options

Option	Meaning
`-p`	Specifies the `$P4PORT` of the shared service, and is required when you create a remote depot connection. Specify the port using the format *`hostname:port`*.
`-u`	Specify the user account to use when connecting to a shared service. If omitted, the current user is used. Cannot differ from user specified in any previous remote depot.

Option	Meaning
`-B`	Specify the name of the remote depot. Default is `remote`.
`-d`	Delete an unused remote depot. Note that you cannot delete a used remote depot. A remote depot is considered used if you have used it as a source with any of the following commands: • `p4 copy` • `p4 integrate` • `p4 merge` • `p4 populate` Requires the `-B` specification.

Usage Notes

Warning! Client specifications that begin with `p4sandbox-` must not be altered in any way (either the name or contents), or P4Sandbox will be unable to communicate properly with the shared repository. Administrators must ensure that all new and existing form trigger scripts that run on shared services do not affect P4Sandbox clients.

For more information, see"Managing P4Sandbox users: Perforce Administrator tasks" on page 58.

Related Commands

Scenario	Command
To list all connections to shared services	`p4 remotes`
To show all defined P4Sandboxes	`p4sandbox list`

To create a mirror stream from this shared service	`p4 integrate`
	or
	`p4 merge`

p4 remotes

Synopsis

List all connections to shared services.

Syntax

```
p4 remotes
```

Description

`p4 remotes` displays the remote depot name, `$P4PORT`, user, client and branch name for each remote connection.

If a remote connection is used by multiple branches, P4Sandbox generates one separate output line for each branch.

Options

None

Related Commands

Scenario	Command
To create a remote connection to a shared service	`p4 remote`
To show all defined P4Sandboxes	`p4sandbox list`
To show all mirror streams	`p4 streams -F "Owner=p4sandbox"`

p4 shelve

Synopsis

Store files from a pending changelist in the depot, without submitting them. When issued on a mirror stream, `p4 shelve` shelves files to the shared service.

Syntax

```
p4 shelve [files]
p4 shelve -i [-f | -r]
p4 shelve -r -c changelist#
p4 shelve -c changelist# [-f] [file ...]
p4 shelve -d -c changelist# [-f] [file ...]
```

Description

When you create, update, or delete a shelf in a mirror stream, P4Sandbox performs the same action to the shelf on the shared service. This functionality enables you to push changes up to the shared service for team review and testing without having to first commit the changes to a stream.

The shelf P4Sandbox creates on the mirror stream contains the associated shelved pending change on the shared service, because P4Sandbox stores the changelist number of the shared service shelf in the description of the mirror shelf. The P4Sandbox checks incoming content changes and removes shelved content that no longer contains differences. If the shelf becomes empty, P4Sandbox deletes it and the corresponding shared service shelf.

For complete command documentation that includes global options and flags, see the *Perforce Command Reference* for `p4 shelve` and the related command `p4 unshelve`.

Related Commands

Scenario	Command
To switch to or display current stream name	`p4 switch`

p4sandbox start

Synopsis

Start one or all previously configured P4Sandbox servers.

Syntax

```
p4sandbox start all
p4sandbox start [-r p4dDir] [-q]
```

Description

`p4sandbox start all` starts all defined P4Sandbox servers.

This form of the command takes no options.

`p4sandbox start` starts one P4Sandbox.

Options

Option	Meaning
`-r`	Set the server root directory for a P4Sandbox. Default is `.p4sandbox`.
`-q`	Suppress startup messages.

Usage Notes

- If you do not specify options, `p4sandbox start` searches for a previously configured and idle P4Sandbox, either in the current directory or the nearest ancestor directory, and starts it.
- If you implement multiple sandboxes, `p4sandbox start` searches for a P4Sandbox in the current directory.
- If you work on a Windows machine, the Windows command-line interface does not provide feedback for this command.

Related Commands

Scenario	Command
To create and initiate a new P4Sandbox	`p4sandbox init`
To stop a P4Sandbox	`p4sandbox stop`

p4sandbox stop

Synopsis

Stop one or all P4Sandboxes, whether they are connected to an active TCP connection or not.

Syntax

```
p4sandbox stop all
p4sandbox stop [-r p4dDir]
```

Description

`p4sandbox stop all` stops all defined P4Sandbox servers. This form of the command takes no options.

`p4sandbox stop` stops one P4Sandbox server.

Options

Option	Meaning
`-r`	Specify the server root directory for the P4Sandbox. Default is `.p4sandbox`.

Usage Notes

- `p4sandbox stop` has the same effect as `p4 admin stop` because a user is an administrator of his or her P4Sandbox. However, `p4sandbox stop` does not require a TCP connection to the P4Sandbox.
- If you do not specify options, `p4sandbox stop` searches for a previously configured and running P4Sandbox, either in the current directory or the nearest ancestor directory, and stops it.
- If you work on a Windows machine, the Windows command-line interface does not provide feedback for this command.

Related Commands

Scenario	Command
To start an existing P4Sandbox	`p4sandbox start`
To stop a P4Sandbox that is operating on a TCP connection	`p4 admin stop`

p4 submit

Synopsis

Submit changes to a mirror stream; P4Sandbox automatically submits the same changes to the shared service.

Syntax

```
p4 submit
```

Description

`p4 submit` automatically propagates changes from a mirror stream to its defined shared service, using the same submit description as provided for the mirror stream.

The submit to the shared service also includes changelists for fixes to jobs that meet the following conditions:

- The jobs are associated with the local mirror stream's submit.
- The jobs already exist on the shared service.

For complete command documentation, see the *Perforce Command Reference*, `p4 submit`.

Related Commands

Scenario	Command
To copy changes from the current stream to its parent stream	`p4 copyup`
To merge changes from a parent stream to its child stream	`p4 mergedown`

p4 switch

Synopsis

Display current stream name, or switch stream and automatically perform related actions.

Syntax

```
p4 switch
p4 switch [-f] name
```

Description

`p4 switch` displays the current stream name.

Adding the *`name`* argument to this command activates the following behavior:

- Shelves any unsubmitted changes.
- Replaces the current client view mapping with that of the named stream.
- Synchronizes the workspace to the head revision.
- Unshelves any changes shelved by a previous `p4 switch` out of the target stream.

Options

Option	Meaning
`-f`	Forces the switch to occur when the client has a non-stream view.

Related Commands

Scenario	Command
To list all streams in a P4Sandbox	`p4 streams`
To show all mirror streams	`p4 streams -F "Owner=p4sandbox"`

Comparing Git and P4Sandbox commands

This section is provided to aid Git users who are working with P4Sandbox.

This section contains the following tables:

- "Command and concept equivalencies" on page 88
- "Task equivalencies" on page 89
- "Client mode dependent commands" on page 94

Command and concept equivalencies

The table below shows the corresponding P4Sandbox command or concept for a given Git command or concept.

Git	P4Sandbox	Comment
branch	stream	Somewhat equivalent
parent branch		
`commit`	`submit`	Equivalent
`merge`	`mergedown`	Somewhat equivalent
	`copyup`	
`pull`	`mergedown`	Somewhat equivalent
`push`	`copyup`	Somewhat equivalent
remote branch	branch on a shared service	Somewhat equivalent
remote tracking branch	mirror	Somewhat equivalent
remote repository	shared service	Somewhat equivalent
repository	P4Sandbox	Equivalent
local repository		
`stash`	`shelve`	Equivalent

Task equivalencies

The table below contains the following columns:

- ***Git Command***: Shows a command or command sequence for the corresponding Git task. This column appears in alphabetical order by command.
- ***Git Task***: Shows a task from the Git perspective; note that the actual task and result in P4Sandbox may be slightly different. Be sure to refer to the Command and Concept Equivalencies table above while reading this column.

  For example, the Git task *List branches in repository* is actually *List streams in P4Sandbox* in P4Sandbox.
- ***P4Sandbox Equivalent***: Shows the applicable P4Sandbox command or command sequence.

The table below lists commands for performing the following types of tasks:

- Setting up a local repository
- Performing branching tasks
- Integrating between branches
- Integrating between servers
- Stashing work

Note Because of table formatting limitations, certain single-line commands are truncated. These commands appear with an indented second line.

Git Command	Git Task	P4Sandbox Equivalent
Not required[1]	Start local repository after computer startup	`p4sandbox start`

Not required[1]	Stop local repository before computer shutdown[2]	`p4sandbox stop`
		or
		`p4 admin stop`

`git branch`	List branches in repository	`p4 streams`

`git branch`	Show current branch	`p4 switch`
`git branch -d` *`branch`*	Delete branch	Not supported[3]
`git checkout -b` *`branch`*	Create and switch to new branch	`p4 copy //streams/`*`parent`*`/... //streams/`*`branch`*`/...`
`git checkout` *`branch`*	Switch to existing branch	`p4 switch` *`branch`*[4]
`git clone server:/main`	Create a connection to remote repository and perform initial copy	`p4 remote -p perforce:1666` `p4 copy //remote/depot/main/... //streams/mirror/...`
`git init`	Create a new repository	`p4sandbox init`
`git log`	List history	`p4 changes`
`git log` *`dir`*	List directory history	`p4 changes` *`dir`*`/...`
`git log` *`file`*	List file history	`p4 changes` *`file`* or `p4 filelog` *`file`*

`git ls-tree HEAD` *`dir`*	List directory files	`p4 files` *`dir`*`/...`
		or
		`p4 files //streams/`*`local`*`/...`

`git ls-tree HEAD file`	List file information	`p4 fstat file`
`git merge`[5] *`branch`*	Copy up changes to a local repository that results in no conflicts	`p4 copyup` `p4 submit`

`git merge branch`
`(receive conflicts)`
`vi conflicting_files`
`git add conflicting_files`
`git commit`
	Copy up changes to a local repository that results in conflicts	`p4 copyup`
		(receive conflicts)[6]
		`p4 mergedown`
		`p4 resolve`
		(resolve in local branch)
		`p4 submit`
		`p4 copyup`
		(attempt copy again)
		`p4 submit`

`git merge parent_branch`	Merge down changes to a local repository that requires no changes to branched files	`p4 mergedown`
`git merge parent_branch`	Merge down changes to a local repository that results in no conflicts	`p4 mergedown`

`git merge parent_branch`
`vi conflicting_files`
`git add conflicting_files`
`git commit`
	Merge down changes to a local repository that results in conflicts	`p4 mergedown`
		(receive conflicts)
		`p4 resolve`
		`p4 submit`

`git pull`[5] `origin master`	Merge down changes from a remote repository that results in no changes to branched files	`p4 mergedown`

`git pull origin master`	Merge down changes from a remote repository that results in no conflicts	`p4 mergedown`

`git pull origin master`
`vi conflicting_files`
`git add conflicting_files`
`git commit`
	Merge down changes from a remote repository that results in resolving conflicts	`p4 mergedown`
		`p4 resolve`
		`p4 submit`

`git push origin master`	Copy up changes from a remote repository that results in no conflicts	`p4 copyup` `p4 submit`

`git push origin master`
(receive conflicts)
`git pull origin master`
`vi conflicting_files`
`git add conflicting_files`
`git commit`
`git push origin master`
(attempt copy again)
	Copy up changes from a remote repository that results in resolving conflicts in a development branch	`p4 copyup`
		(receive conflicts)[7]
		`p4 merge -rS`[8]
		`//streams/branch`
		`p4 resolve`
		(resolve in local branch)
		`p4 submit`
		`p4 copyup`
		(attempt copy again)
		`p4 submit`

`git stash`	Stash work in progress	`p4 shelve`[9] (edit changelist)
`git stash` `git checkout branch`	Switch to existing branch	`p4 switch branch`[4]
`git stash list`	List stashes	`p4 changes -c client` `-s shelved`

`git stash pop`	Unstash work in progress	`p4 unshelve -s` *`change_num`* `p4 shelve -d -c` *`change_num`* `p4 change -d` *`change_num`*
`git stash show [stash@{`*`n`*`}]`	Display stashed content	`p4 files @=`*`n`*

`git status`	List modified files	`p4 status`
		or
		`p4 reconcile -n`

`git status`	List files currently opened	`p4 opened`
`vi .gitignore`	Specify files to ignore	`vi .p4ignore`

1. Git does not require a command to start or stop a local repository.
2. You are not required to stop a P4Sandbox before shutting down the computer. The normal operating system shutdown will cleanly stop it.
3. Deleting a branch outright is not supported, as Perforce always retains a branch's history. It is possible, however, by using `p4 obliterate` and `p4 change -d`; for more information on using these commands, see the *Perforce Command Reference*.
4. Note that the syntax example is `p4 switch name`; `p4 switch branch` and `p4 switch name` are equivalent commands.
5. In Git, you use different commands to transmit changes to a repository, depending on whether the repository type is local (`git merge`) or remote (`git push` and `git pull`). In P4Sandbox, you use the same commands (`p4 mergedown`, `p4 copyup`) to transmit changes to a repository, regardless of the repository type.
6. The initial copy attempt results in conflicts. You must merge, resolve, and submit the conflicts in the local stream before attempting to copy up files again to the mirror stream. If the second copy up attempt is successful, perform a submit.
7. The initial copy attempt results in conflicts. You must merge, resolve, and submit the conflicts in the local stream before attempting to copy up files again to the mirror stream. If the second copy up attempt is successful, perform a submit.
8. Note that `p4 copy` and `p4 merge` can take options, while `p4 copyup` and `p4 mergedown` cannot.
9. This command invokes a text editor window and displays a changelist. Enter a changelist description to complete this command.

Client mode dependent commands

Git leaves files writable in your system, ready to modify. Perforce usually leaves files read-only and requires `p4 edit` before modification. To switch Perforce to leave files writable and no longer require `p4 edit`, set your workspace to `allwrite`. Run `p4 status` or `p4 reconcile` when you are ready to stage files for submit.

The table below lists Git commands for file administration and management tasks in alphabetical order, with the corresponding Perforce command. The appropriate Perforce command depends on your implementation's defined client mode. Note that if you set your Perforce workspace to `allwrite`, you only need to issue commands when you submit changes or move or rename a file.

Git	Task	P4Sandbox Client noallwrite mode	P4Sandbox Client allwrite mode
`git add dir`	Add new file hierarchy	`p4 reconcile -a dir`	Not required
`git add file`	Add new file	`p4 add file`	Not required
`git commit [-a]`	Submit changes	`p4 submit`	`p4 reconcile -de` `p4 submit`
`git mv src dest`	Move/Rename file	`p4 edit src` `p4 move src dest`	`p4 edit src` `p4 move src dest`
`git rm file`	Delete file	`p4 delete file`	Not required
Not required	Edit file	`p4 edit file`	Not required

See also

For more information about:

- Perforce command syntax, see the *Perforce Command Reference*.
- Perforce command usage, see the *P4 User's Guide*, "Issuing P4 Commands."
- Perforce commands that require administrator or superuser permission, see the *Perforce System Administrator's Guide.*

Appendix B Glossary

This glossary describes P4Sandbox-specific terms. For Perforce-specific and general version management terms used in Perforce documentation, see the *Perforce User's Guide,* "Appendix A: Glossary."

Terms

Term	Definition
branch	See the *P4 User's Guide's* Glossary for the following terms: • *branch* • *branch form* • *branch mapping* • *branch view*
shared service	A company's or team's main Perforce server or depot; the server that you connect to a P4Sandbox to which you merge down and copy up changes (see *merge down and copy up paradigm*). Also called a *central server*.
connection-independent versioning	P4Sandbox's functionality that enables you to check changes in and out without requiring an active connection to a shared service. You connect to a shared service through the *remote depot* when merging down and copying up changes.
development stream	One of the four Perforce stream types. Sometimes referred to as *dev stream*. Stream for long-term projects and new features. A development stream is less stable than its parent stream. See also *local stream*.
local stream	Any P4Sandbox stream that is not a *mirror stream*. This includes *mainline streams* that are not associated with a *shared service*, as well as any development streams. Most local streams are development streams, descendants of a *mirror stream*. See also *development stream*.

Term	Definition
mainline stream	One of the four Perforce stream types. The base or trunk of a stream system.
merge down and copy up paradigm	The paradigm and process for managing change through a hierarchy of codelines. Codelines, represented within Perforce as streams, exist in a hierarchy of stability, from more stable to less stable. You merge down changes from more stable streams to less stable streams, and copy up changes from less stable streams to more stable streams. In P4Sandbox, you must merge down changes from the *mirror stream* before you copy up changes from any *local stream*.

mirror stream	A mirror stream is any *mainline stream* associated with a *shared service* through a branch mapping.
	A valid mirror stream meets all the following criteria:
	• Type: `mainline`
	• Owner: `p4sandbox`
	• Branch form: A branch form exists and includes the following:
	  • The branch form name generally matches the stream name convention: "`$depot_$streamshortname`"
	  For example, the stream "`//streams/foo`" has a corresponding branch form named "`streams_foo`".
	  • The branch form maps paths from a single remote depot into the *mainline stream*.

`p4`	The Perforce Command-Line Client.
P4Sandbox	The P4Sandbox server; also means the local implementation of P4Sandbox.
`p4sandbox`	The broker that implements P4Sandbox functionality. All user interactions are managed through this broker.
`p4sandbox command`	The P4Sandbox command convention.
`p4sandbox-p4d`	The `p4d` distribution used by P4Sandbox for the local server.

Term	Definition
`p4sandbox-config`	A GUI configuration tool (**P4Sandbox Configuration Wizard**) for creating and editing a P4Sandbox implementation. Creating a P4Sandbox also creates a hidden directory and updates a reference file. The location of the these files and directories is dependent upon your platform.
`.p4sandbox-list`	An XML configuration file that lists your P4Sandbox servers.
`.p4sandbox/`	Directory containing the `p4d` databases and archive files.

`.p4config file`	A text file containing the following three default settings and values:
	`P4PORT=localhost:1666`
	`P4USER=`*`machine's default user`*
	`P4CLIENT=my_workspace`
	For more information, see "p4sandbox init" on page 70.

P4Sandbox installation	The collection of files installed by the P4Sandbox installer; specifically, the `p4`, `p4sandbox`, and `p4sandbox-p4d` executable files. It does not contain the `db.*` and versioned files included in a *P4Sandbox server*.
P4Sandbox server	The `.p4sandbox` folder, all the `db.*` and versioned files it contains, and the process that listens on the configured `P4PORT`. See also *P4Sandbox*.
private local branching	The ability to create streams (called *local streams*) within P4Sandbox for specific tasks, such as bug fixes for a particular release, and determine codeline management and policies on these streams.
release stream	One of the four Perforce stream types. Stream for bug-fixing, testing, and release distribution.
remote depot	The mechanism by which P4Sandbox connects to the *shared service*. If you are using P4Sandbox with `p4` use the `p4 remote` and `p4 remotes` commands. If using P4Sandbox with a graphical client application or plug-in, the P4Sandbox Configuration Wizard automatically establishes this connection during the setup process. See "p4 remote" on page 80 and "p4 remotes" on page 82.

Term	Definition
sandbox	Term not used to indicate a P4Sandbox implementation. See *P4Sandbox*.

shelving	Shelving is the process of temporarily storing files on a Perforce server without checking in a changelist.
	P4Sandbox has the following types of shelving functionality:
	• Automatic: Shelving that P4Sandbox performs in the background when you switch among *task streams* and workspaces and have unsubmitted changes.
	• User-directed:
	  • Shelving that you perform when you switch among task streams and workspaces and have unsubmitted changes.
	  • Shelving that you perform to the *mirror stream* as a backup of your work.
	You can perform user-directed shelving from either `p4` by issuing the `p4 shelve` command or from P4V by using the shelving commands available from the context menus.

stream	A stream is a Perforce database object that describes information about a branch; in short, a "branch with brains." There are four stream types: *development*, *mainline*, *release*, and *virtual*.
stream switching	See *task switching*.
task stream	A *local stream* intended for a single specific task, and then abandoned once the task is complete; for example, a bug fix. Task streams are a concept, not a specific stream type. There is no mechanical difference between a *local stream* and a task stream; the difference is solely in usage and not in how P4Sandbox handles the stream.

Term	Definition
task switching	To switch from one task to another by switching from one stream to another. If you are using the P4V Stream Graph, you can switch tasks by moving the workspace icon between streams. When you switch between streams, P4Sandbox automatically performs the following tasks: • Shelves any work-in-progress changes from the previous stream. • Updates the workspace with files from the new stream. • Unshelves any work-in-progress changes on the new stream.
versioned files	The formatted files that store a file's history of content changes. For more information about versioned files, see the *Perforce System Administrator's Guide*, "Supporting Perforce: Backup and Recovery."
virtual stream	One of the four Perforce stream types. Stream for submitting changes directly to a parent stream.
workspace	Also known as *client*. See also the *P4 User's Guide's* Glossary for the following terms: • *client form* • *client name* • *client root* • *client side* • *client workspace view* • *client workspace*
workspace specifications	Also known as *client form*. Term means the entire form, both form name and contents.

Index

A

administrators
- determining file access and usage 58
- determining P4Sandbox users 58
- performing standard tasks 58
- prohibiting implementation 58

affected files of P4Sandbox delete 28
allwrite mode 31, 94
always-on versioning 13
antivirus programs 55
automatic
- checkpoint and journaling 55
- task switching 99

automating tasks 38

B

backup 55
branch 95

C

central server 95
checkpoint and journaling functionality 57
client write mode 31
- allwrite 94
- noallwrite 94

closing 53
command conventions 29
commands 37
- operating in P4Sandbox context 63
- p4 add 30
- p4 admin stop 30, 37
- p4 change -d 35
- p4 changes 37
- p4 changes -c client -s shelved 36
- p4 changes dir/... 37
- p4 changes file 37
- p4 clients 58
- p4 copy 31, 34
- p4 copyup 29, 33
- p4 counter 30
- p4 delete 30
- p4 edit 30
- p4 filelog file 37
- p4 files @=changelist# 36
- p4 files dir/... 37
- p4 fstat file 37
- p4 help sandbox 31
- p4 info 27
- p4 integ 34
- p4 integrate 21, 30, 31
- p4 jobspec 57
- p4 merge 30, 35
- p4 mergedown 32
- p4 move 31
- p4 obliterate 35
- p4 opened 37
- p4 opened -c changelist# 36
- p4 populate 21, 30, 35
- p4 protect 55
- p4 pull 30, 32
- p4 reconcile -n 37
- p4 remote 29
- p4 remotes 30
- p4 shelve 35
- p4 streams 34
- p4 submit 33, 56
- p4 switch 34
- p4 unshelve -s changelist# 36
- p4 verify 57
- P4CHARSET 71
- p4sandbox delete 28, 29
- p4sandbox init 21, 22, 29
- p4sandbox list 29
- p4sandbox start 29, 31
- p4sandbox stop 37, 57
- TCP connection not required 85
- TCP connection required 64

configuration methods
- P4Sandbox using configuration wizard 22

using p4 21, 22
wizard using p4 22
configuration verification 25
incorrect settings 27
p4 method 27
connected implementation 15
connection-independent versioning 13, 95
converting standalone implementation 20
copying up changes 33, 48, 51
copyup to shared service 60
D
daemons 55
delete P4Sandbox 28, 53
development stream 14
disconnected P4Sandbox 59
downloads 19
E
editable files 31
equivalent Git and P4Sandbox commands 63
external backup system 57
F
fast context switching 13
fixing errors
copy up issues 60
incorrectly shelved files 61
localhost connection error 59
P4V password issue 60
relaunch error 59
G
Git commands
git branch 89
git branch -d branch 90
git checkout -b branch 90
git checkout branch 90
git clone server main 90
git init 90
git log 90
git log dir 90
git log file 90
git ls-tree HEAD dir 90
git ls-tree HEAD file 91
git merge branch 91
git merge parent_branch 91
git pull origin master 92
git push origin master 92
git stash 92
git stash list 92
git stash pop 93
git stash show 93
git status 93
vi .gitignore 93
Git terms
branch 88
commit 88
local repository 88
merge 88
parent branch 88
pull 88
push 88
remote branch 88
remote repository 88
remote tracking branch 88
repository 88
stash 88
I
implementation
options 15
restrictions 15, 16
implementation prerequisites 20
incorrectly shelved files 61
initial content replication 20
Initial content replication time 20
installation components
.p4config file 20
.p4sandbox 19
.p4sandbox/ 19
.p4sandbox-config 19
.p4sandbox-list 19
.p4sandbox-p4d 19
installation directory verification 20
installed components 14
installer 19
invalid P4V password issue 60
J
jobspecs and jobs 56

L
large context menu 43
licensing 55
local stream 14, 95
M
mainline stream 14, 96
merge down and copy up 15, 96
merging down changes 31, 46, 52
mirror stream 14, 96
    allowed commands 31
    prohibited commands 30
N
noallwrite mode 31, 94
P
p4 9, 13, 29
p4 add 30
p4 admin stop 30, 37
p4 and P4V difference
    directory 18
p4 change -d 35
p4 change -d changelist# 36
p4 changes 37
p4 changes -c client -s shelved 36
p4 changes dir/... 37
p4 changes file 37
p4 client 36
p4 clients 58
p4 command 29
p4 copy 21, 31, 34
p4 copyup 29, 33
p4 counter 30
p4 delete 30
p4 edit 30, 31
p4 filelog file 37
p4 files @=changelist# 36
p4 files dir/... 37
p4 fstat file 37
p4 help sandbox 31
p4 info 27
p4 integ 30, 34
p4 integrate 21, 30, 31
p4 jobspec 57
p4 merge 30, 35
p4 mergedown 29, 32
p4 move 31
p4 obliterate 35
p4 opened 37
p4 opened -c changelist#-s changelist# 36
p4 populate 21, 30, 35
p4 print Method 61
p4 protect 55
p4 pull 30, 32
p4 reconcile -n 37
p4 remote 29
p4 remotes 30
p4 shelve 35
p4 shelve -d -c changelist 36
p4 status 37
p4 streams 34
p4 submit 33, 56
p4 switch 34
p4 switch stream name 34
p4 unshelve 83
p4 unshelve -s changelist# 36
p4 verify 57
p4config file 20, 22, 97
P4Eclipse 13
p4ignore 21
p4sandbox 19, 96
P4Sandbox broker version 27
p4sandbox command 29, 96
P4Sandbox Configuration Wizard 22
    icon location 22
    page descriptions 23
P4Sandbox Configuration Wizard, see also p4sandbox-config 97
p4sandbox delete 28, 29
p4sandbox directory 22
p4sandbox folder 57
p4sandbox help 31
P4Sandbox implementation conversion 20
p4sandbox init 21, 22, 29
P4Sandbox installation 97
p4sandbox list 29
P4Sandbox server 97
p4sandbox start 29, 31

p4sandbox stop 29, 37, 57
p4sandbox/ 97
p4sandbox_max_count_checkpoint 68
p4sandbox_schedule_checkpoint 68
p4sandbox_schedule_checkpoint counter 57
p4sandbox_schedule_copy 68
p4sandbox-config 19, 22, 97
p4sandbox-list 19, 97
p4sandbox-p4d 19, 96
P4V 13
P4V initial P4Sandbox behaviors 24
pending tasks, resolving 53
Perforce Command-Line Client 9, 13, 29
Perforce downloads page 19
Perforce Server distribution 19
piping job template between servers 57
private local branching 13, 97
product integration 13
R
read-only files 31, 94
release stream 14, 97
remote depot 97
restarting P4Sandbox 59
rolling prefix 57
S
scripts 38
security 55
shared service 14, 95
shelving 98
  automatic 98
  in p4 35
  unshelving 36
  user-directed 98
  using the Stream Graph 52
  viewing in P4V 53
small context menu 43
standalone implementation 15
starting 31, 39
stopping 53
stream 98
Stream Graph
  Copy menu 49
  large context menu 43
  Merge/Integrate menu 46
  small context menu 43
  stream context menu 43
stream switching 98
stream types 14
streams
  adding 34, 44
  copying 43
  copying changes 34
  deleting 35, 44
  diff against 44
  editing 44
  printing 45
  switching 52
  switching between 34
  viewing 34, 43, 45
T
task stream 14
task streams
  adding 51
  copying 51
  deleting 52
  merging 52
  removing 52
task switching 99
TCP connection 30
triggers 55
U
unaffected files of P4Sandbox delete 28
Unicode mode configuration 56, 71
updating only the mirror stream 32, 43
V
validating backup 57
versioned files 99
viewing information in p4 37
virtual stream 14, 99
W
Windows service 55
Workspace root clobber 20
writable files 94
Z
Zip Method 61