



# HelixCore

---

## P4EXP User Guide

2019.3  
*December 2019*

PERFORCE

[www.perforce.com](http://www.perforce.com)



Copyright © 1999-2019 Perforce Software, Inc..

All rights reserved.

All software and documentation of Perforce Software, Inc. is available from [www.perforce.com](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.

Perforce 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.

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

# Contents

<b>How to use this guide</b> .....	<b>4</b>
Syntax conventions .....	4
Feedback .....	4
Other documentation .....	4
<b>What's new in this guide by release</b> .....	<b>5</b>
2019.2 .....	5
2019.1 .....	5
<b>P4EXP User Guide</b> .....	<b>6</b>
Connecting to Helix server .....	6
Set Connection dialog configuration .....	6
Command-line configuration .....	6
Menu options .....	8
Troubleshooting .....	10
<b>Glossary</b> .....	<b>11</b>

## How to use this guide

This guide tells you how to use P4EXP, the Helix Plugin for File Explorer. It is intended for anyone using P4EXP to perform version control management tasks with Helix server.

This section provides information on typographical conventions, feedback options, and additional documentation.

## Syntax conventions

Helix documentation uses the following syntax conventions to describe command line syntax.

Notation	Meaning
<code>literal</code>	Must be used in the command exactly as shown.
<i>italics</i>	A parameter for which you must supply specific information. For example, for a <i>serverid</i> parameter, supply the ID of the server.
<code>[-f]</code>	The enclosed elements are optional. Omit the brackets when you compose the command.
<code>...</code>	Previous argument can be repeated. <ul style="list-style-type: none"><li>▪ <code>p4 [g-opts] streamlog [ -l -L -t -m max ] stream1 ...</code> means <code>1</code> or more stream arguments separated by a space</li><li>▪ See also the use on <code>...</code> in <a href="#">Command alias syntax</a> in the <i>Helix Core P4 Command Reference</i></li></ul>
<code>element1   element2</code>	Either <i>element1</i> or <i>element2</i> is required.

### Tip

`...` has a different meaning for directories. See [Wildcards](#) in the *Helix Core P4 Command Reference*.

## Feedback

How can we improve this manual? Email us at [manual@perforce.com](mailto:manual@perforce.com).

## Other documentation

See <https://www.perforce.com/support/self-service-resources/documentation>.

## What's new in this guide by release

### 2019.2

- "Set Connection dialog configuration" on page 6 is an alternative to "Command-line configuration" on page 6.
- Support for the administrator to use Helix Authentication Service.
- For a complete list, see <https://www.perforce.com/perforce/r19.2/user/p4expnotes.txt>

### 2019.1

- Added new menu option to delete files in Helix server. For details, see [Delete in Helix server](#) option.
- Added new preference to prompt for confirmation of delete in Helix server. For details, see [Preferences](#) option.
- Added a new menu option to view pending changelists, to assist with submitting files marked for delete. For details, see [View Pending Changelists](#) option.
- For a complete list, see <https://www.perforce.com/perforce/r19.1/user/p4expnotes.txt>

## P4EXP User Guide

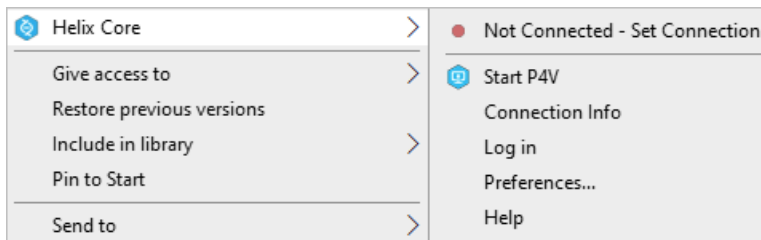
This User Guide provides an overview of P4EXP, the Helix Plugin for File Explorer, including its [menu options](#) and [troubleshooting](#). P4EXP enables you to perform Helix server tasks on files in the context of your file browser (for example, File Explorer). For information about Helix server, see the [Documentation](#) page of the [Perforce Website](#).

### Connecting to Helix server

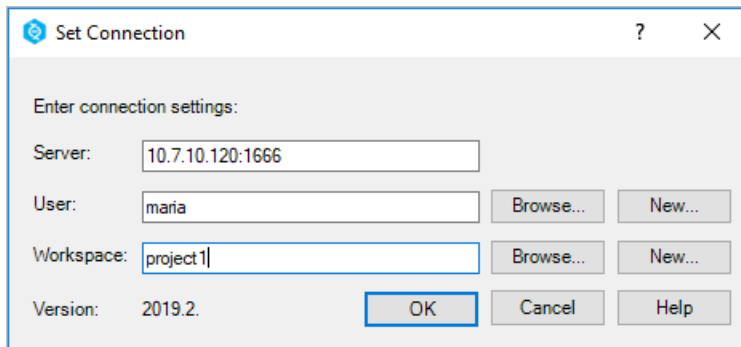
To perform Helix server version control actions on files in File Explorer, you must be connected to a Helix Core server, also referred to as Helix server, with a valid user and workspace.

### Set Connection dialog configuration

The graphic user interface provides a convenient **Set Connection** dialog.



Enter the values for your Server, user name, and workspace, then click OK.



When you are connected, you can **Log off** by using the **More...** menu.

### Command-line configuration

If you do not want to use **Set Connection** dialog, consider the command-line alternatives.

You can set these connection variables in the Windows environment, per directory with a P4CONFIG file, or a combination of both. (Note that P4CONFIG files will override Windows environment settings for the current working directory.)

**To set Helix server variables in the Windows environment:**

1. Open a command window.
2. Set the individual Helix server variables with the following commands:
  - **Server:** `p4 set P4PORT=<server:port>`
  - **User:** `p4 set P4USER=<user_name>`
  - **Workspace:** `p4 set P4CLIENT=<workspace_name>`
3. Open File Explorer and navigate to a location under the workspace root defined by the workspace named in P4CLIENT.

**To set Helix server variables with P4CONFIG files:**

1. Open a command window.
2. Set the Helix server variable for P4CONFIG with the following command:
  - **P4CONFIG:** `p4 set P4CONFIG=<config_file_name>`
3. For each workspace root location on the local disk, create a file named `<config_file_name>`; inside the file add:

```
#  
# Perforce configuration file  
#  
P4PORT=<server:port>  
P4USER=<user_name>  
P4CLIENT=<workspace_name>
```

4. Open File Explorer and navigate to a location under a workspace root containing a P4CONFIG file.

Environment variables should be an adequate way to set connection settings if you are working with a single server, user, and workspace. If you have more than one workspace that you use on a regular basis, you may benefit from the flexibility of using P4CONFIG files.

For additional information, see:

- [Windows Environment Variable Precedence](#)
- [Using Multiple Workspaces with P4EXP](#)

## Menu options

When you right-click a file in your file browser and select **Perforce**, P4EXP displays some or all of the following menu options, depending on the state of the file.

---

**Log In...** Enables you to enter your password and log in if you are not logged in to Helix server and you have a password.

### Note

If the server you are connecting to is configured with multi-factor authentication (MFA), you are prompted for another layer of verification. Depending on the setup, you may need to select a method of verification before you can enter your credentials.

For more information, see `p4 login2` in the *Helix Core P4 Command Reference*.

### Note

If the server you are connecting to is configured for authentication with Helix Authentication Service, the Identity Provider (IdP) web page opens, prompting you for the credentials registered with your Identity Provider (IdP). For details, see <https://github.com/perforce/helix-authentication-service> or contact your Helix Core server administrator.

### Note

If the server you are connecting to is configured for authentication with Helix SAML, the Helix SAML dialog opens, prompting you for the user name and password registered with your Identity Provider (IdP). For details, see "Helix SAML" in the *Helix Core Server Administrator Guide: Fundamentals* or contact your Helix Core server administrator.

---

**Get Latest Revision** Retrieves the most recent version of a file in the depot.

**Submit...** Checks in the default changelist. A changelist contains details about the change you made to files in the depot.

---

**Check Out** Opens the files for edit in your default changelist. A changelist logs the changes you make to files in the depot. After you have made your changes, you submit the changelist.



---

<b>Delete in Helix server</b>	Removes selected files locally and marks them for delete in Helix server. Depending on your preferences, P4EXP prompts you for confirmation before it proceeds.
	<b>Note</b> Selecting the <b>Don't confirm deletes in the future</b> check box turns off the related preference.
	When you confirm, P4EXP marks the files for delete in Helix server and displays a confirmation message with the number of files marked for delete. Files marked for delete still exist in Helix server. They are now included in a pending changelist. To complete the deletion process, you need to submit the changelist. To access the changelist, right-click and select <b>Show in P4V</b> or <b>View Pending Changelists</b> . Then, right-click the respective changelist and click <b>Submit</b> .
<b>Add to Source Control</b>	Adds the selected files to the default changelist. When you submit the changelist, the files are added to the depot. Only files that reside in directories in your workspace view can be added to a depot.
<b>Revert</b>	Discards all changes you have made to the file. Helix server restores the head revision of the file from the depot to your workspace, overwriting any changes you have made. The file is no longer checked out.
<b>Revert Unchanged Files</b>	Closes unchanged files in a changelist. The files are no longer checked out.
<b>View Pending Changelists</b>	Opens the <b>Pending Changelists</b> dialog, which lists all changelists that have not been submitted. This is helpful when you need to submit files that you marked for delete with the <b>Delete in Helix server</b> option because such files no longer show in File Explorer.
<b>Diff Against Have Revision</b>	Launches your diff utility to compare (diff) files in your workspace with files at the head revision in the depot to display any changes you have made.
<b>File History</b>	Displays the <b>History</b> dialog, which enables you to view any of the selected file's revisions, get a desired revision into your workspace, or compare two revisions of the same file.
<b>Properties</b>	Displays Helix server information about a file.
<b>Show in P4V</b>	Opens P4V with the selected file highlighted.

---

<b>Remove from Workspace</b>	Removes files from your workspace. The files remain in the depot. To restore a file to your workspace, select the file in P4V and choose <b>Get Latest Revision</b> .
<b>Refresh File State</b>	Refreshes the state of the files under Helix server control.
<b>More</b>	<p><b>Set Connection:</b> Displays the dialog for configuring a connection to your Helix server.</p> <p><b>Connection Information:</b> Displays information about your Helix server connection.</p> <p><b>Preferences:</b> Enables you to configure P4EXP as follows:</p> <ul style="list-style-type: none"> <li>■ <b>Show file state icons:</b> Select to display Helix server icon overlays on individual files to indicate file state in the file browser, as follows: <ul style="list-style-type: none"> <li>• <b>Green circle:</b> The file is synced to head.</li> <li>• <b>Yellow triangle:</b> The file is out of date.</li> <li>• <b>Red check:</b> The file is in your changelist, opened for either add, edit, or delete.</li> </ul> </li> <li>■ <b>Prompt for confirmation of delete in Helix server:</b> Clear if you do not want P4EXP to prompt you for confirmation when you try to delete a file in Helix server. By default, this option is selected.</li> <li>■ <b>Show only error messages:</b> Select to display only error messages from the server. By default, all server messages are displayed when you perform an action on a file.</li> <li>■ <b>Enable logging to file:</b> Select to log P4EXP activity to the specified file. <ul style="list-style-type: none"> <li>• <b>Name:</b> Specify the name and location of the log file.</li> <li>• <b>Size:</b> Specify the maximum size of the log file.</li> </ul> </li> <li>■ <b>Enable diagnostic logging:</b> Select to turn on verbose debug logging for actions performed from <b>Perforce</b> menu items.</li> </ul> <p><b>Log Off:</b> Logs you out if you have a password set in Helix server.</p> <p><b>Help:</b> Displays this help page.</p>

---

## Troubleshooting

If you experience issues viewing P4EXP icons, this might be related to file overlay icon limitations on Windows. For more information and a workaround, see the article [Inconsistent Icon Overlays with P4EXP](#) in the [Perforce Knowledgebase](#).

# Glossary

## A

---

### **access level**

A permission assigned to a user to control which commands the user can execute. See also the 'protections' entry in this glossary and the 'p4 protect' command in the P4 Command Reference.

### **admin access**

An access level that gives the user permission to privileged commands, usually super privileges.

### **APC**

The Alternative PHP Cache, a free, open, and robust framework for caching and optimizing PHP intermediate code.

### **archive**

1. For replication, versioned files (as opposed to database metadata). 2. For the 'p4 archive' command, a special depot in which to copy the server data (versioned files and metadata).

### **atomic change transaction**

Grouping operations affecting a number of files in a single transaction. If all operations in the transaction succeed, all the files are updated. If any operation in the transaction fails, none of the files are updated.

### **avatar**

A visual representation of a Swarm user or group. Avatars are used in Swarm to show involvement in or ownership of projects, groups, changelists, reviews, comments, etc. See also the "Gravatar" entry in this glossary.

## B

---

### **base**

For files: The file revision, in conjunction with the source revision, used to help determine what integration changes should be applied to the target revision. For checked out streams: The public have version from which the checked out version is derived.

**binary file type**

A Helix server file type assigned to a non-text file. By default, the contents of each revision are stored in full, and file revision is stored in compressed format.

**branch**

(noun) A set of related files that exist at a specific location in the Perforce depot as a result of being copied to that location, as opposed to being added to that location. A group of related files is often referred to as a codeline. (verb) To create a codeline by copying another codeline with the 'p4 integrate', 'p4 copy', or 'p4 populate' command.

**branch form**

The form that appears when you use the 'p4 branch' command to create or modify a branch specification.

**branch mapping**

Specifies how a branch is to be created or integrated by defining the location, the files, and the exclusions of the original codeline and the target codeline. The branch mapping is used by the integration process to create and update branches.

**branch view**

A specification of the branching relationship between two codelines in the depot. Each branch view has a unique name and defines how files are mapped from the originating codeline to the target codeline. This is the same as branch mapping.

**broker**

Helix Broker, a server process that intercepts commands to the Helix server and is able to run scripts on the commands before sending them to the Helix server.

**C**

---

**change review**

The process of sending email to users who have registered their interest in changelists that include specified files in the depot.

**changelist**

A list of files, their version numbers, the changes made to the files, and a description of the changes made. A changelist is the basic unit of versioned work in Helix server. The changes specified in the changelist are not stored in the depot until the changelist is submitted to the depot. See also atomic change transaction and changelist number.

**changelist form**

The form that appears when you modify a changelist using the 'p4 change' command.

**changelist number**

An integer that identifies a changelist. Submitted changelist numbers are ordinal (increasing), but not necessarily consecutive. For example, 103, 105, 108, 109. A pending changelist number might be assigned a different value upon submission.

**check in**

To submit a file to the Helix server depot.

**check out**

To designate one or more files, or a stream, for edit.

**checkpoint**

A backup copy of the underlying metadata at a particular moment in time. A checkpoint can recreate db.user, db.protect, and other db.\* files. See also metadata.

**classic depot**

A repository of Helix server files that is not streams-based. The default depot name is depot. See also default depot and stream depot.

**client form**

The form you use to define a client workspace, such as with the 'p4 client' or 'p4 workspace' commands.

**client name**

A name that uniquely identifies the current client workspace. Client workspaces, labels, and branch specifications cannot share the same name.

**client root**

The topmost (root) directory of a client workspace. If two or more client workspaces are located on one machine, they should not share a client root directory.

**client side**

The right-hand side of a mapping within a client view, specifying where the corresponding depot files are located in the client workspace.

**client workspace**

Directories on your machine where you work on file revisions that are managed by Helix server. By default, this name is set to the name of the machine on which your client workspace is located, but it can be overridden. Client workspaces, labels, and branch specifications cannot share the same name.

**code review**

A process in Helix Swarm by which other developers can see your code, provide feedback, and approve or reject your changes.

**codeline**

A set of files that evolve collectively. One codeline can be branched from another, allowing each set of files to evolve separately.

**comment**

Feedback provided in Helix Swarm on a changelist, review, job, or a file within a changelist or review.

**commit server**

A server that is part of an edge/commit system that processes submitted files (checkins), global workspaces, and promoted shelves.

**conflict**

1. A situation where two users open the same file for edit. One user submits the file, after which the other user cannot submit unless the file is resolved. 2. A resolve where the same line is changed when merging one file into another. This type of conflict occurs when the comparison of two files to a base yields different results, indicating that the files have been changed in different ways. In this case, the merge cannot be done automatically and must be resolved manually. See file conflict.

**copy up**

A Helix server best practice to copy (and not merge) changes from less stable lines to more stable lines. See also merge.

**counter**

A numeric variable used to track variables such as changelists, checkpoints, and reviews.

**CSRF**

Cross-Site Request Forgery, a form of web-based attack that exploits the trust that a site has in a user's web browser.

**D**

---

**default changelist**

The changelist used by a file add, edit, or delete, unless a numbered changelist is specified. A default pending changelist is created automatically when a file is opened for edit.

**deleted file**

In Helix server, a file with its head revision marked as deleted. Older revisions of the file are still available. In Helix server, a deleted file is simply another revision of the file.

**delta**

The differences between two files.

**depot**

A file repository hosted on the server. A depot is the top-level unit of storage for versioned files (depot files or source files) within a Helix Core server. It contains all versions of all files ever submitted to the depot. There can be multiple depots on a single installation.

**depot root**

The topmost (root) directory for a depot.

**depot side**

The left side of any client view mapping, specifying the location of files in a depot.

**depot syntax**

Helix server syntax for specifying the location of files in the depot. Depot syntax begins with: `//depot/`

**diff**

(noun) A set of lines that do not match when two files, or stream versions, are compared. A conflict is a pair of unequal diffs between each of two files and a base, or between two versions of a stream.  
(verb) To compare the contents of files or file revisions, or of stream versions. See also conflict.

**donor file**

The file from which changes are taken when propagating changes from one file to another.

**E**

---

**edge server**

A replica server that is part of an edge/commit system that is able to process most read/write commands, including 'p4 integrate', and also deliver versioned files (depot files).

**exclusionary access**

A permission that denies access to the specified files.

**exclusionary mapping**

A view mapping that excludes specific files or directories.

**extension**

Similar to a trigger, but more modern. See "Helix Core Server Administrator Guide: Fundamentals" on "Extensions".

**F**

---

**file conflict**

In a three-way file merge, a situation in which two revisions of a file differ from each other and from their base file. Also, an attempt to submit a file that is not an edit of the head revision of the file in the depot, which typically occurs when another user opens the file for edit after you have opened the file for edit.



**file pattern**

Helix server command line syntax that enables you to specify files using wildcards.

**file repository**

The master copy of all files, which is shared by all users. In Helix server, this is called the depot.

**file revision**

A specific version of a file within the depot. Each revision is assigned a number, in sequence. Any revision can be accessed in the depot by its revision number, preceded by a pound sign (#), for example testfile#3.

**file tree**

All the subdirectories and files under a given root directory.

**file type**

An attribute that determines how Helix server stores and diffs a particular file. Examples of file types are text and binary.

**fix**

A job that has been closed in a changelist.

**form**

A screen displayed by certain Helix server commands. For example, you use the change form to enter comments about a particular changelist to verify the affected files.

**forwarding replica**

A replica server that can process read-only commands and deliver versioned files (depot files). One or more replicate servers can significantly improve performance by offloading some of the master server load. In many cases, a forwarding replica can become a disaster recovery server.

**G**

---

**Git Fusion**

A Perforce product that integrates Git with Helix, offering enterprise-ready Git repository management, and workflows that allow Git and Helix server users to collaborate on the same

projects using their preferred tools.

**graph depot**

A depot of type graph that is used to store Git repos in the Helix server. See also Helix4Git.

**group**

A feature in Helix server that makes it easier to manage permissions for multiple users.

**H**

---

**have list**

The list of file revisions currently in the client workspace.

**head revision**

The most recent revision of a file within the depot. Because file revisions are numbered sequentially, this revision is the highest-numbered revision of that file.

**Helix server**

The Helix server depot and metadata; also, the program that manages the depot and metadata, also called Helix Core server.

**Helix TeamHub**

A Perforce management platform for code and artifact repository. TeamHub offers built-in support for Git, SVN, Mercurial, Maven, and more.

**Helix4Git**

Perforce solution for teams using Git. Helix4Git offers both speed and scalability and supports hybrid environments consisting of Git repositories and 'classic' Helix server depots.

**I**

---

**iconv**

A PHP extension that performs character set conversion, and is an interface to the GNU libiconv library.

**integrate**

To compare two sets of files (for example, two codeline branches) and determine which changes in one set apply to the other, determine if the changes have already been propagated, and propagate any outstanding changes from one set to another.

**J**

---

**job**

A user-defined unit of work tracked by Helix server. The job template determines what information is tracked. The template can be modified by the Helix server system administrator. A job describes work to be done, such as a bug fix. Associating a job with a changelist records which changes fixed the bug.

**job daemon**

A program that checks the Helix server machine daily to determine if any jobs are open. If so, the daemon sends an email message to interested users, informing them the number of jobs in each category, the severity of each job, and more.

**job specification**

A form describing the fields and possible values for each job stored in the Helix server machine.

**job view**

A syntax used for searching Helix server jobs.

**journal**

A file containing a record of every change made to the Helix server's metadata since the time of the last checkpoint. This file grows as each Helix server transaction is logged. The file should be automatically truncated and renamed into a numbered journal when a checkpoint is taken.

**journal rotation**

The process of renaming the current journal to a numbered journal file.

**journaling**

The process of recording changes made to the Helix server's metadata.

## L

---

### **label**

A named list of user-specified file revisions.

### **label view**

The view that specifies which filenames in the depot can be stored in a particular label.

### **lazy copy**

A method used by Helix server to make internal copies of files without duplicating file content in the depot. A lazy copy points to the original versioned file (depot file). Lazy copies minimize the consumption of disk space by storing references to the original file instead of copies of the file.

### **license file**

A file that ensures that the number of Helix server users on your site does not exceed the number for which you have paid.

### **list access**

A protection level that enables you to run reporting commands but prevents access to the contents of files.

### **local depot**

Any depot located on the currently specified Helix server.

### **local syntax**

The syntax for specifying a filename that is specific to an operating system.

### **lock**

1. A file lock that prevents other clients from submitting the locked file. Files are unlocked with the 'p4 unlock' command or by submitting the changelist that contains the locked file. 2. A database lock that prevents another process from modifying the database db.\* file.

### **log**

Error output from the Helix server. To specify a log file, set the P4LOG environment variable or use the p4d -L flag when starting the service.

## M

---

### mapping

A single line in a view, consisting of a left side and a right side that specify the correspondences between files in the depot and files in a client, label, or branch. See also workspace view, branch view, and label view.

### MDS checksum

The method used by Helix server to verify the integrity of versioned files (depot files).

### merge

1. To create new files from existing files, preserving their ancestry (branching). 2. To propagate changes from one set of files to another. 3. The process of combining the contents of two conflicting file revisions into a single file, typically using a merge tool like P4Merge.

### merge file

A file generated by the Helix server from two conflicting file revisions.

### metadata

The data stored by the Helix server that describes the files in the depot, the current state of client workspaces, protections, users, labels, and branches. Metadata is stored in the Perforce database and is separate from the archive files that users submit.

### modification time or modtime

The time a file was last changed.

### MPM

Multi-Processing Module, a component of the Apache web server that is responsible for binding to network ports, accepting requests, and dispatch operations to handle the request.

## N

---

### nonexistent revision

A completely empty revision of any file. Syncing to a nonexistent revision of a file removes it from your workspace. An empty file revision created by deleting a file and the #none revision specifier are

examples of nonexistent file revisions.

### **numbered changelist**

A pending changelist to which Helix server has assigned a number.

## **O**

---

### **opened file**

A file that you are changing in your client workspace that is checked out. If the file is not checked out, opening it in the file system does not mean anything to the versioning engineer.

### **owner**

The Helix server user who created a particular client, branch, or label.

## **P**

---

### **p4**

1. The Helix Core server command line program. 2. The command you issue to execute commands from the operating system command line.

### **p4d**

The program that runs the Helix server; p4d manages depot files and metadata.

### **P4PHP**

The PHP interface to the Helix API, which enables you to write PHP code that interacts with a Helix server machine.

### **PECL**

PHP Extension Community Library, a library of extensions that can be added to PHP to improve and extend its functionality.

### **pending changelist**

A changelist that has not been submitted.

**Perforce**

Perforce Software, Inc., a leading provider of enterprise-scale software solutions to technology developers and development operations (“DevOps”) teams requiring productivity, visibility, and scale during all phases of the development lifecycle.

**project**

In Helix Swarm, a group of Helix server users who are working together on a specific codebase, defined by one or more branches of code, along with options for a job filter, automated test integration, and automated deployment.

**protections**

The permissions stored in the Helix server’s protections table.

**proxy server**

A Helix server that stores versioned files. A proxy server does not perform any commands. It serves versioned files to Helix server clients.

**R**

---

**RCS format**

Revision Control System format. Used for storing revisions of text files in versioned files (depot files). RCS format uses reverse delta encoding for file storage. Helix server uses RCS format to store text files. See also reverse delta storage.

**read access**

A protection level that enables you to read the contents of files managed by Helix server but not make any changes.

**remote depot**

A depot located on another Helix server accessed by the current Helix server.

**replica**

A Helix server that contains a full or partial copy of metadata from a master Helix server. Replica servers are typically updated every second to stay synchronized with the master server.

**repo**

A graph depot contains one or more repos, and each repo contains files from Git users.

**reresolve**

The process of resolving a file after the file is resolved and before it is submitted.

**resolve**

The process you use to manage the differences between two revisions of a file, or two versions of a stream. You can choose to resolve file conflicts by selecting the source or target file to be submitted, by merging the contents of conflicting files, or by making additional changes. To resolve stream conflicts, you can choose to accept the public source, accept the checked out target, manually accept changes, or combine path fields of the public and checked out version while accepting all other changes made in the checked out version.

**reverse delta storage**

The method that Helix server uses to store revisions of text files. Helix server stores the changes between each revision and its previous revision, plus the full text of the head revision.

**revert**

To discard the changes you have made to a file in the client workspace before a submit.

**review access**

A special protections level that includes read and list accesses and grants permission to run the p4 review command.

**review daemon**

A program that periodically checks the Helix server machine to determine if any changelists have been submitted. If so, the daemon sends an email message to users who have subscribed to any of the files included in those changelists, informing them of changes in files they are interested in.

**revision number**

A number indicating which revision of the file is being referred to, typically designated with a pound sign (#).



**revision range**

A range of revision numbers for a specified file, specified as the low and high end of the range. For example, `myfile#5,7` specifies revisions 5 through 7 of `myfile`.

**revision specification**

A suffix to a filename that specifies a particular revision of that file. Revision specifiers can be revision numbers, a revision range, change numbers, label names, date/time specifications, or client names.

**RPM**

RPM Package Manager. A tool, and package format, for managing the installation, updates, and removal of software packages for Linux distributions such as Red Hat Enterprise Linux, the Fedora Project, and the CentOS Project.

**S**

---

**server data**

The combination of server metadata (the Helix server database) and the depot files (your organization's versioned source code and binary assets).

**server root**

The topmost directory in which `p4d` stores its metadata (`db.*` files) and all versioned files (depot files or source files). To specify the server root, set the `P4ROOT` environment variable or use the `p4d -r` flag.

**service**

In the Helix Core server, the shared versioning service that responds to requests from Helix server client applications. The Helix server (`p4d`) maintains depot files and metadata describing the files and also tracks the state of client workspaces.

**shelve**

The process of temporarily storing files in the Helix server without checking in a changelist.

**status**

For a changelist, a value that indicates whether the changelist is new, pending, or submitted. For a job, a value that indicates whether the job is open, closed, or suspended. You can customize job

statuses. For the 'p4 status' command, by default the files opened and the files that need to be reconciled.

**storage record**

An entry within the db.storage table to track references to an archive file.

**stream**

A branch with additional intelligence that determines what changes should be propagated and in what order they should be propagated.

**stream depot**

A depot used with streams and stream clients.

**submit**

To send a pending changelist into the Helix server depot for processing.

**super access**

An access level that gives the user permission to run every Helix server command, including commands that set protections, install triggers, or shut down the service for maintenance.

**symlink file type**

A Helix server file type assigned to symbolic links. On platforms that do not support symbolic links, symlink files appear as small text files.

**sync**

To copy a file revision (or set of file revisions) from the Helix server depot to a client workspace.

**T**

---

**target file**

The file that receives the changes from the donor file when you integrate changes between two codelines.

**text file type**

Helix server file type assigned to a file that contains only ASCII text, including Unicode text. See also binary file type.

**theirs**

The revision in the depot with which the client file (your file) is merged when you resolve a file conflict. When you are working with branched files, theirs is the donor file.

**three-way merge**

The process of combining three file revisions. During a three-way merge, you can identify where conflicting changes have occurred and specify how you want to resolve the conflicts.

**trigger**

A script that is automatically invoked by Helix server when various conditions are met. (See "Helix Core Server Administrator Guide: Fundamentals" on "Triggers".)

**two-way merge**

The process of combining two file revisions. In a two-way merge, you can see differences between the files.

**typemap**

A table in Helix server in which you assign file types to files.

**U**

---

**user**

The identifier that Helix server uses to determine who is performing an operation.

**V**

---

**versioned file**

Source files stored in the Helix server depot, including one or more revisions. Also known as an archive file. Versioned files typically use the naming convention 'filenamev' or '1.changelist.gz'.

**view**

A description of the relationship between two sets of files. See workspace view, label view, branch view.

**W**

---

**wildcard**

A special character used to match other characters in strings. The following wildcards are available in Helix server: \* matches anything except a slash; ... matches anything including slashes; %%0 through %%9 is used for parameter substitution in views.

**workspace**

See client workspace.

**workspace view**

A set of mappings that specifies the correspondence between file locations in the depot and the client workspace.

**write access**

A protection level that enables you to run commands that alter the contents of files in the depot. Write access includes read and list accesses.

**X**

---

**XSS**

Cross-Site Scripting, a form of web-based attack that injects malicious code into a user's web browser.

**Y**

---

**yours**

The edited version of a file in your client workspace when you resolve a file. Also, the target file when you integrate a branched file.