Extracted from:

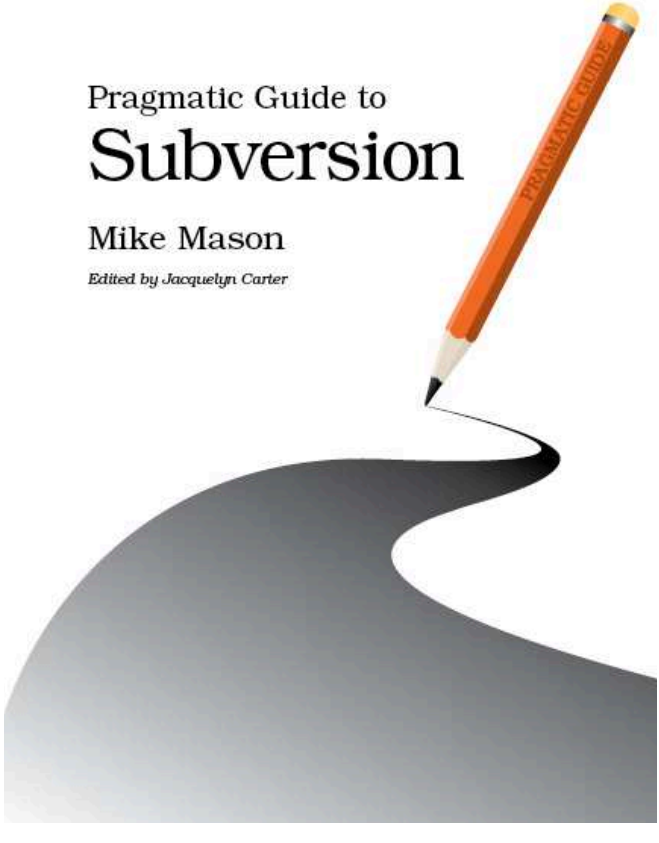
Pragmatic Guide to Subversion

The Pragmatic Programmers

Pragmatic Guide to Subversion

Mike Mason

Edited by Jacquelyn Carter



Introduction

Subversion is a wildly popular open source version control system, available for free over the Internet. Subversion is widely considered the de facto standard[1] for version control tools, even though such a thing is difficult to measure, and as a developer, you are likely to encounter Subversion as part of your work.

Subversion is a mature, fully featured system that is commonly used by both commercial and open source development teams. You can buy commercial support and consulting services to help you install, configure, and use Subversion. If you don't want the hassle of running a Subversion server, you can get a third party to do it for you, at a low cost or even for free.

Subversion is a *centralized* version control system, meaning that it uses a central server to store files and enable team collaboration. Clients can work disconnected from the network—on an airplane, for example—and need a network connection only if they actually want to commit changes to the server. This traditional centralized model assumes that development teams have reasonable network connectivity to the server. In contrast, some newer *decentralized* version control systems use a model where each user acts kind of like a server. Users can swap changes between each other without needing a central server. Most organizations will be fine with the centralized model used by Subversion, but it's worth being aware that other collaboration styles are possible.

Subversion is popular because it has all the features that programmers need and very few extra bells and whistles. It just does version control, and it does it well.

Subversion can track version information for directories and metadata, as well as files. Treating directories as first-class objects means that Subversion can track history across directory moves and renames, unlike some older version

1. Determining the exact market share for Subversion is difficult, but several online polls rate Subversion more popular than any other version control tool. Martin Fowler suggests that within the Agile/XP community, only Subversion, Git, and Mercurial would be recommended: http://martinfowler.com/bliki/VersionControlTools.html.

control systems. Every file and directory can have arbitrary metadata associated with it using Subversion *properties*.

Committing a change is *atomic*, similar to committing to a database. Either the whole commit succeeds or is rolled back; other users never see a half-finished commit. As part of the atomic commit process, Subversion groups all your changes into a *revision* (sometimes called a *changeset*) and assigns a *revision number* to the change, unlike older systems that apply a revision number to each individual file. By grouping changes to multiple files into a single logical unit, developers are able to better organize and track their changes.

Subversion has cheap *branches* and *tags* that can be created almost instantly. Branches are used to distinguish different lines of development, most commonly to separate code that is in production vs. code that is being actively developed. Tags are used to "mark" the state of the code at a particular point in time so that state can be re-created later. Subversion also supports *merge tracking*, which helps automatically merge changes between branches.

Subversion is a truly multiplatform tool. You can run Subversion on Windows, Linux, OS X, and many other flavors of Unix. Each of these operating systems is considered a first-class platform by the Subversion developers, and you can run a production-strength server on any of them. A Subversion client can talk to a Subversion server even if the client and server are running on different operating systems. This is good news for anyone trying to fit Subversion into their existing infrastructure. For those evaluating Subversion, the wide choice of operating system makes things much easier since you can run a server on pretty much any spare machine.

Who Is This Book For?

Most developers have at least some experience with a source control tool and are expected to fluidly switch between tools depending on where they are working. This book was written to bridge the gap between knowing something about version control and knowing about Subversion specifically.

Pragmatic Guide to Subversion will quickly get you up to speed on Subversion. We don't spend a lot of time covering the philosophy of version control or trying to persuade you it's a good idea to store your files somewhere safe. If you are interested in a broader discussion of version control concepts and some of the reasoning behind what we do, check out *Pragmatic Version Control Using Subversion* [Mas06],[2] my previous book.

2. http://pragprog.com/titles/svn2/

How to Read This Book

This book is organized into parts that each cover a portion of the Subversion software management "life cycle." Each part of the book contains some introductory pages discussing how Subversion handles particular concepts. You should read the introductions to get a feel for the overall concepts and how everything ties together, but after that, feel free to jump straight to a particular task. If you are new to source control, it's perfectly OK to read the book in order—everything will make sense and give you a good grounding in Subversion concepts.

The book is laid out as double-page spreads for each task, with a discussion of the task on the left page and the actual steps to achieve the task on the right page. This works naturally for the printed book, but many readers will be reading a digital version. If you have the screen space for it, try setting your reader to show two pages side by side to enhance your reading experience.

The parts of the book are organized as follows:

- *Part I: Getting Started* covers core Subversion concepts such as the client, server, repository, and working copies. You will learn how to choose and install a Subversion client, how to set up a local repository, and how to import your existing code into Subversion.
- *Part II: Working with Subversion* discusses daily workflow when using Subversion. You'll learn how to check out from a repository, examine or undo your changes, and commit to the repository.
- *Part III: Working with a Team* covers how to use Subversion in a team setting, how to stay in sync with your team, and how to resolve conflicts.
- *Part IV: Using the History* shows you Subversion's powerful history tools so you can understand changes made to your source tree and who made them. In some cases, you might want to undo changes that have been committed to the repository—this part shows you how.
- *Part V: Branching, Merging, and Tagging* tackles one of the more complex topics in source control. Using branches and tags, you can reliably release software to production and support it going forward.
- *Part VI: File Locking* covers Subversion's optional file locking features, which is useful if your repository contains unmergeable files such as spreadsheets or graphics.
- *Part VII: Setting Up a Server* shows you how to install a Subversion server on Linux or Windows, including securing and backing up the server. If you'd like to use third-party hosting instead of running your own server, we discuss how to do this too.

- *Part VIII: Advanced Topics* discusses Subversion features that you might not need every day but that will be important maybe once or twice when you set up a project. Included here is information on how to store multiple projects in a single Subversion repository and how to store third-party code in your own repository.

Subversion Versions

Subversion is developed by a team of programmers collaborating over the Internet. It's open source, and the Subversion team regularly releases new versions. Major versions are given numbers such as 1.6 or 1.7, with patch and bug fix releases getting numbers like 1.6.3, 1.7.1, and so on.

In general, you should always use the newest release of Subversion, because new features, bug fixes, and performance improvements are continually being made by the Subversion developers. New major versions are always backward compatible with older servers and working copies—so client version 1.5.*x* will work with server version 1.6.*x*—but the opposite is not always true.

Always upgrade all the clients on a computer at the same time. For example, if you have both the command-line client and a graphical client and you want to upgrade to Subversion 1.7, upgrade both the command-line and graphical clients at the same time. If you don't do this, one or other of the clients might complain about the working copy being in a new format that they don't understand.

In general, it's always safe to upgrade a Subversion client, but a Subversion server requires more attention. You should always ensure you have a repository backup before doing a server upgrade and that you have tested that the backup restores correctly.

Online Resources

All Pragmatic books have an online component. You can find the home page for this book here:

http://pragprog.com/titles/pg_svn/

From here you can download code and the example mBench project used throughout the book, view the book's errata, and chat with the author and other readers in a dedicated online forum.

Subversion is a mature open source system, and there is a lot of community support for it. A web search will usually turn up loads of extra information about a topic, most of which is excellent and high quality. Don't be afraid to look beyond the tasks in this book and explore for yourself.

The Pragmatic Bookshelf

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help you stay on top of your game.

Visit Us Online

Pragmatic Guide to Subversion
http://pragprog.com/titles/pg_svn
Source code from this book, errata, and other resources. Come give us feedback, too!

Register for Updates
http://pragprog.com/updates
Be notified when updates and new books become available.

Join the Community
http://pragprog.com/community
Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

New and Noteworthy
http://pragprog.com/news
Check out the latest pragmatic developments, new titles and other offerings.

Buy the Book

If you liked this eBook, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: pragprog.com/titles/pg_svn.

Contact Us

Online Orders:	www.pragprog.com/catalog
Customer Service:	support@pragprog.com
Non-English Versions:	translations@pragprog.com
Pragmatic Teaching:	academic@pragprog.com
Author Proposals:	proposals@pragprog.com
Contact us:	1-800-699-PROG (+1 919 847 3884)