Version control systems

Version control system (VCS) is a system that records changes to a file or set of files over time so that you can recall specific versions later.

It allows you to revert files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more.

Local VCS, Distributed VCS

Local VCS

- Simple commands
- Checkout by parts
- You need more frequent connection to central server

Distributed VCS

- View history on local machine
- You can work locally more
- You can not create empty folders

Local version control systems

RCS, which is still distributed with many computers today. Even the popular Mac OS X operating system includes the rcs command when you install the Developer Tools. RCS works by keeping patch sets (that is, the differences between files) in a special format on disk; it can then re-create what any file looked like at any point in time by adding up all the patches.

Centralized Version Control Systems (CVCS) - these systems, such as CVS, Subversion, and Perforce, have a single server that contains all the files versions, and a number of clients that check out files from that central place.

Distributed Version Control Systems

In a **DVCS** (such as **Git**, **Mercurial**, **Bazaar** or **Darcs**), clients don’t just check out the latest snapshot of the files: they fully mirror the repository. Thus if any server dies, and these systems were collaborating via it, any of the client repositories can be copied back up to the server to restore it. Every clone is really a full backup of all the data.

GIT frequent commands

Create repositories

To create a new local repository with the specified name

```
$ git init [project-name]
```

To download a project and its entire version history

```
$ git clone [url]
```

https://github.github.com/training-kit/downloads/github-git-cheat-sheet.pdf

GIT frequent commands

Make changes

Lists all new or modified files to be commited

```
$ git status
```

Shows file differences not yet staged

```
$ git diff
```

Snapshots the file in preparation for versioning

```
$ git add [file]
```

Records file snapshots permanently in version history

```
$ git commit -m "[descriptive message]"
```

https://github.github.com/training-kit/downloads/github-git-cheat-sheet.pdf

SVN frequent commands

Create repositories

To create a new local repository with the specified name

```
$ svnadmin create /[svnrepos]
```

Import your project

```
$ svn import /projects/myrailsproject
file:///svnrepos/myrailsproject
```

https://backlog.com/git-tutorial/reference/commands/

Open VCS services

GitLab

“In GitLab, you can create projects for hosting your codebase, use it as an issue tracker, collaborate on code, and continuously build, test, and deploy your app with built-in GitLab CI/CD.

Your projects can be available publicly, internally, or privately, at your choice. GitLab does not limit the number of private projects you create.”

https://docs.gitlab.com/ee/user/project/wiki/