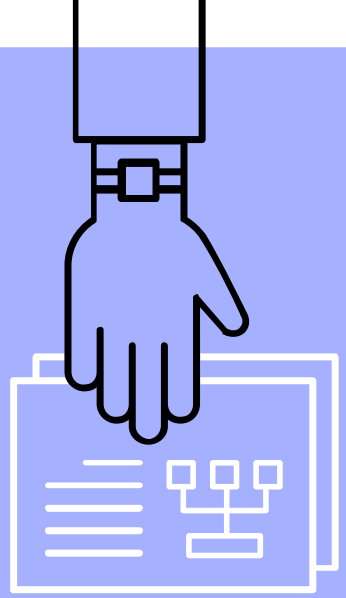
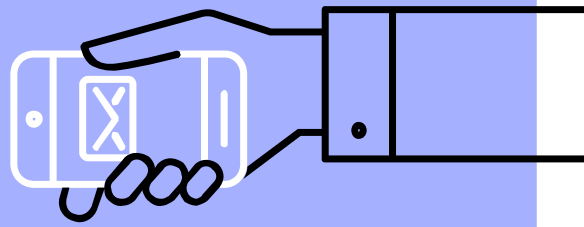
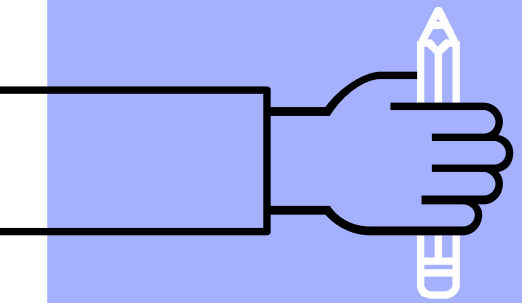
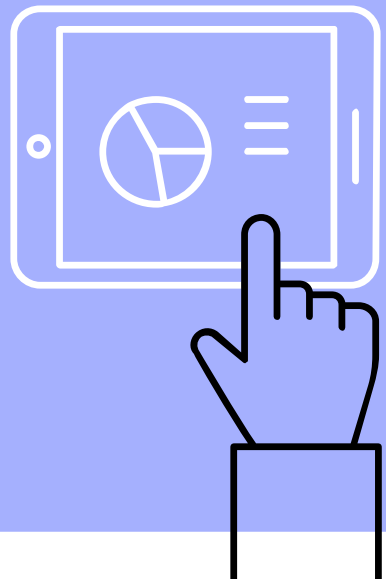
Common Tools for Team Collaboration





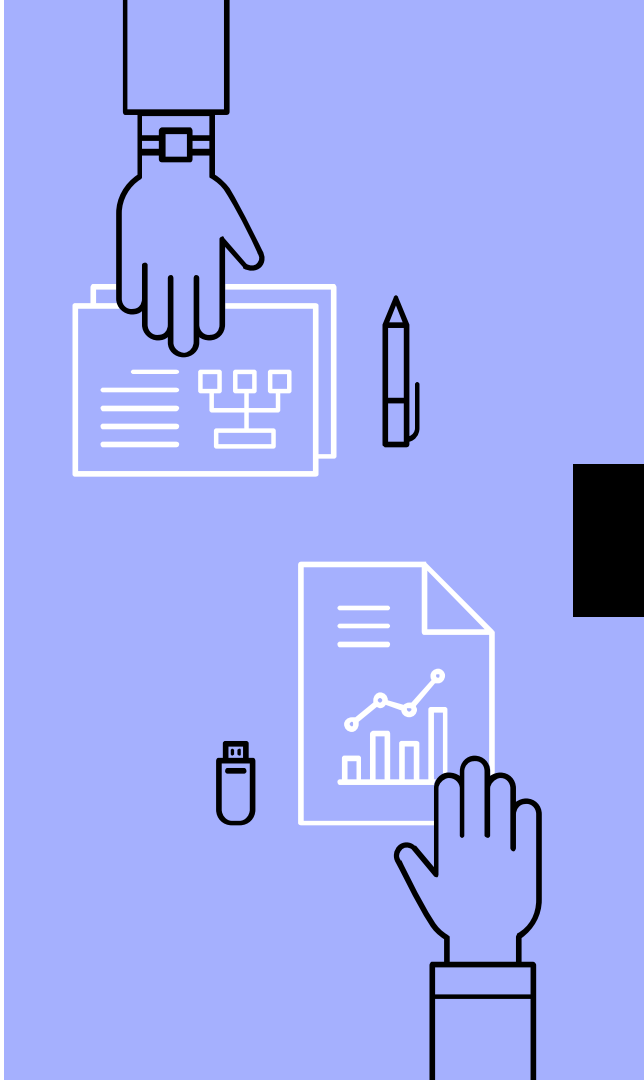




Problem:

Working with a team (especially remotely) can be difficult.

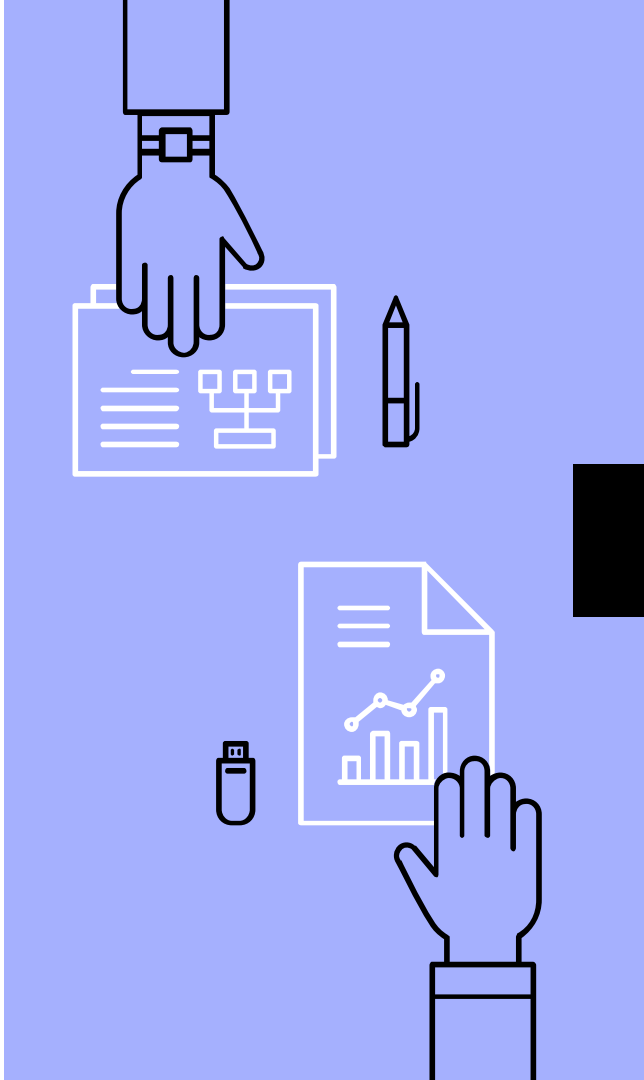
- Team members might have a different idea for the project
- Two or more team members could end up doing the same work
- Or a few team members have nothing to do



Solutions:

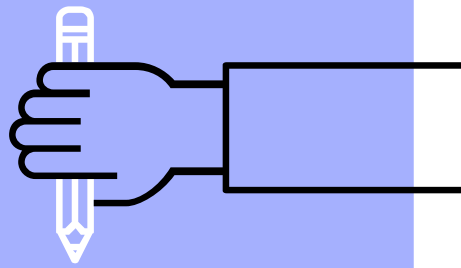
A combination of few tools.

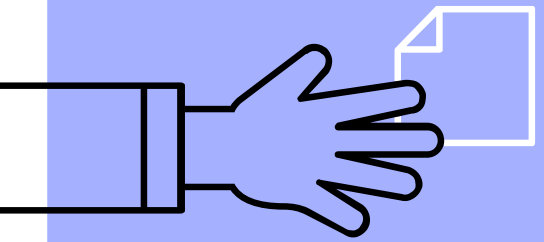
- Communication channels
- Wikis
- Task manager
- Version Control
  - We'll be going in depth with this one!



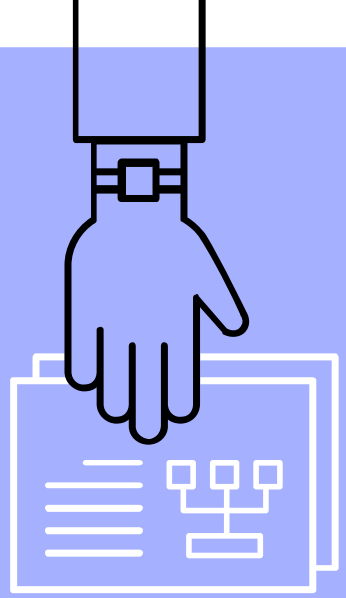
Important!

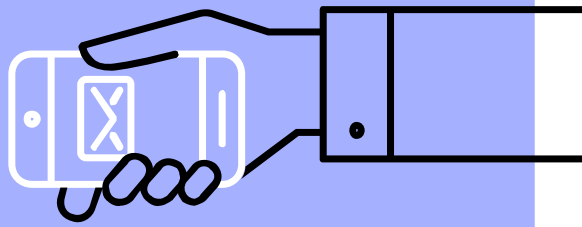
The tools are only as good as your team uses them. Make sure all of your team members agree on what tools to use, and train them thoroughly!

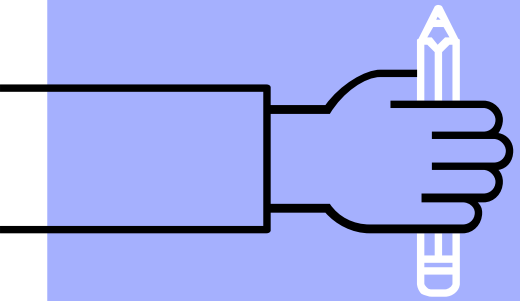


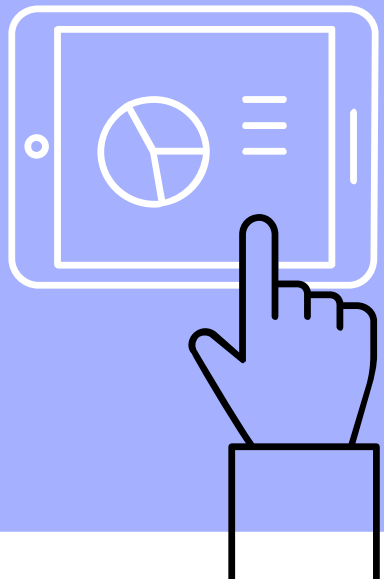


Communication Channels





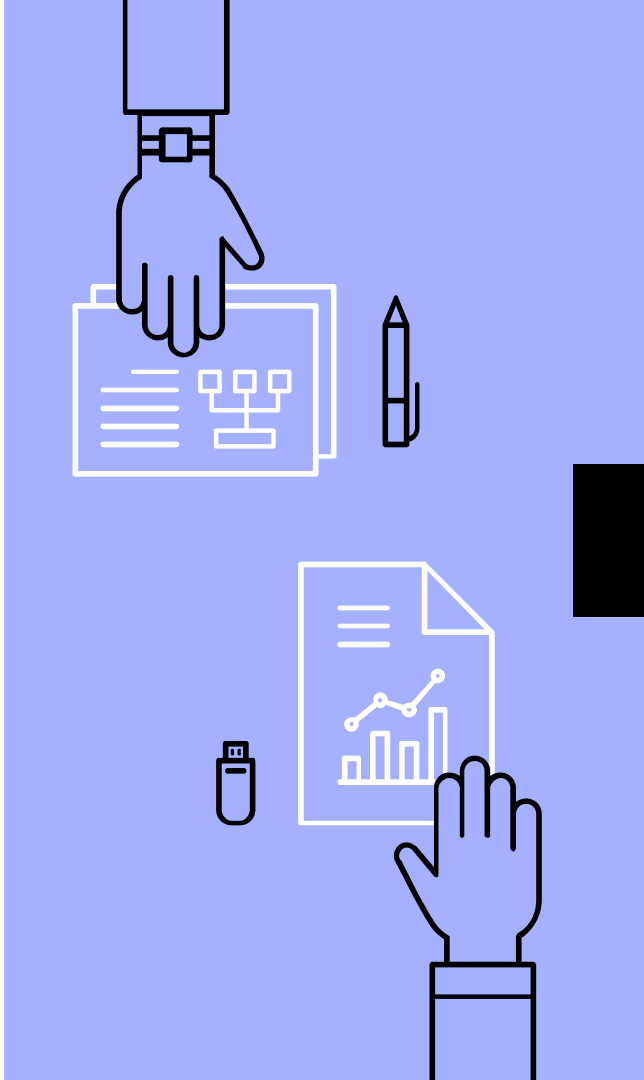




Purpose:

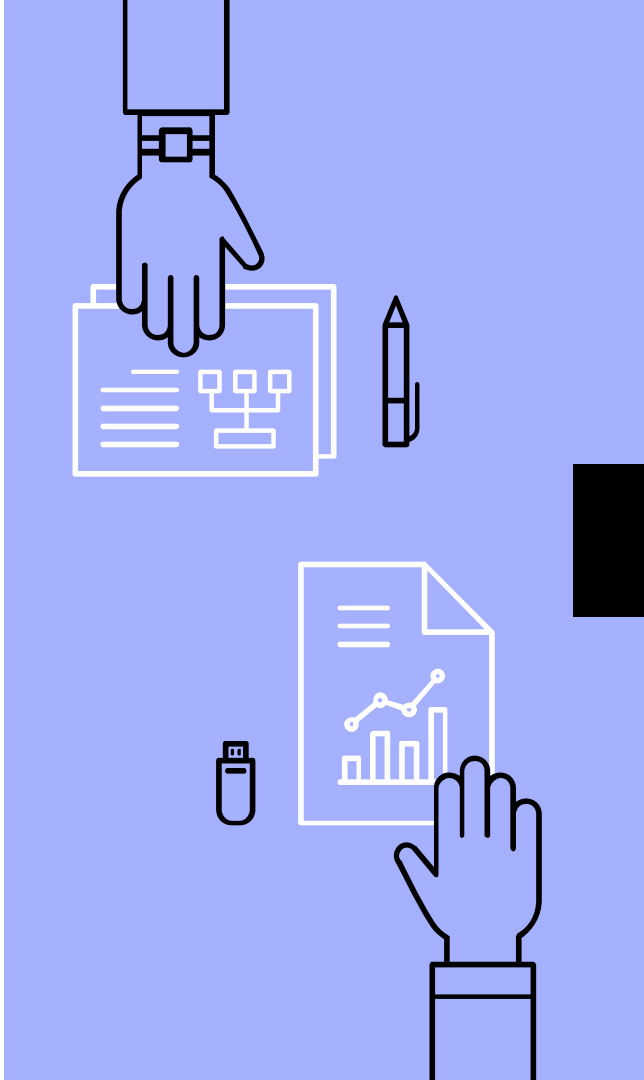
Communication channels provide a way to have team members remotely communicate with one another.

Ideally, the channel will attempt to emulate, as closely as possible, what communication would be like if all of your team members were in the same office.



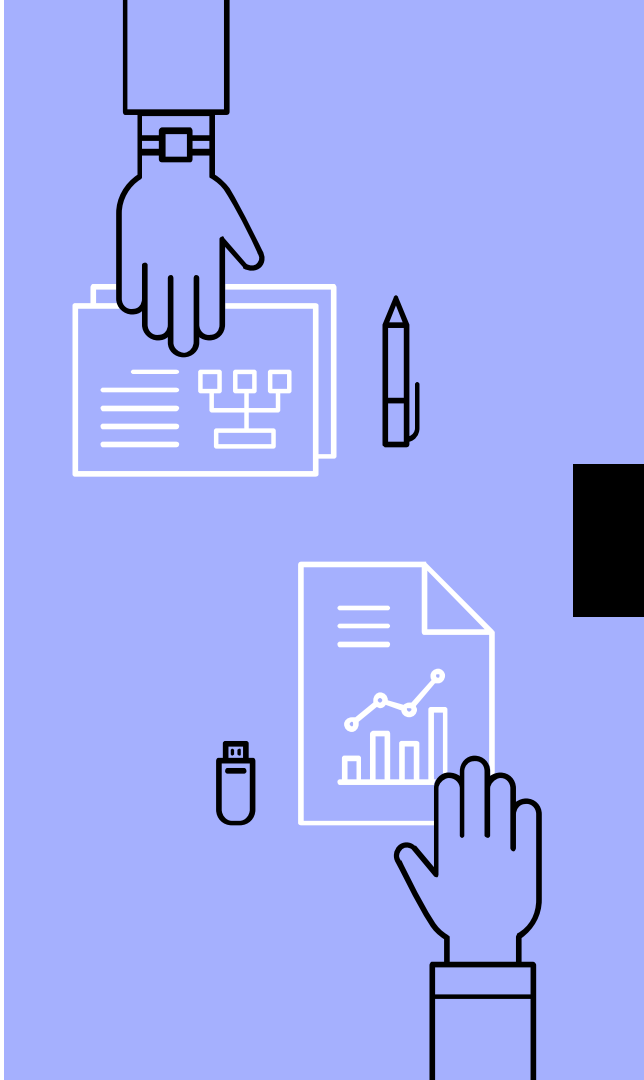
Wait, why not email?

- No voice support
  - Text alone is *not* a sufficient form of communication
- Too slow, no obvious support for notifications
- Lack of flexibility in grouping people



Tools:

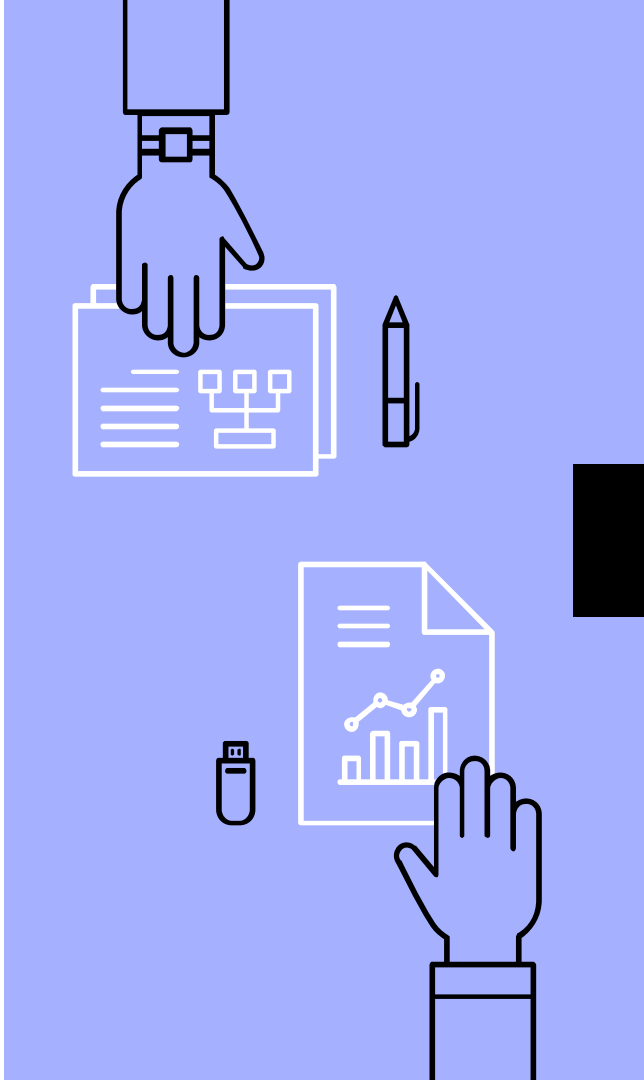
- Discord
  - discordapp.com
- Slack
  - slack.com
- Riot.im
  - about.riot.im



Discord:

Originally used for voice-chat for gaming, Discord provides:

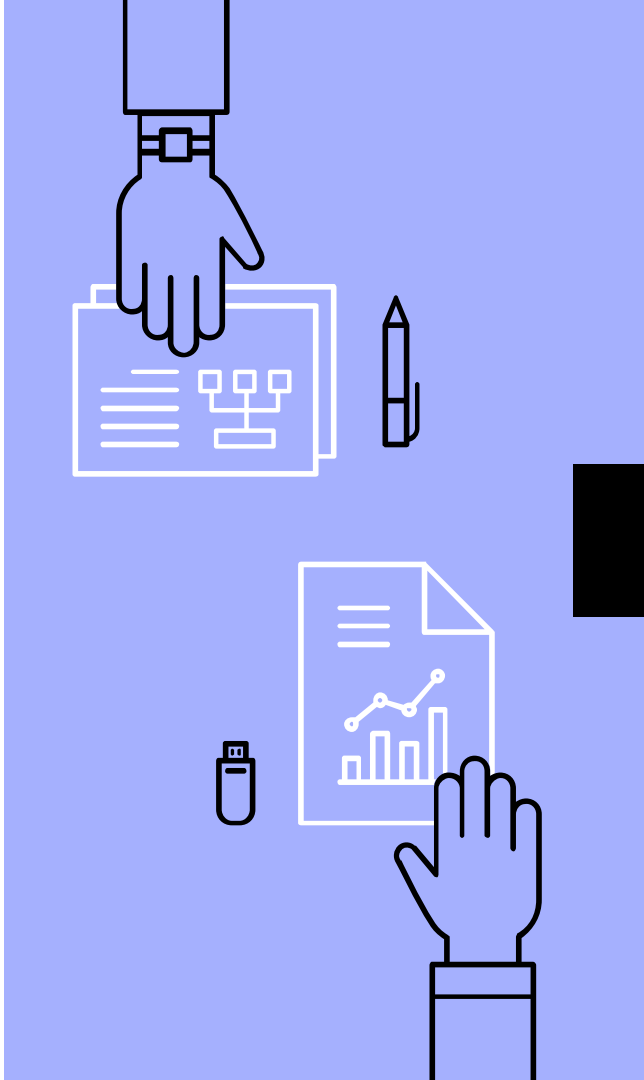
- Voice & video conferencing
- Text communication, separated by channels
- File-sharing
- Private communications
- A mobile, web, and desktop app



Slack:

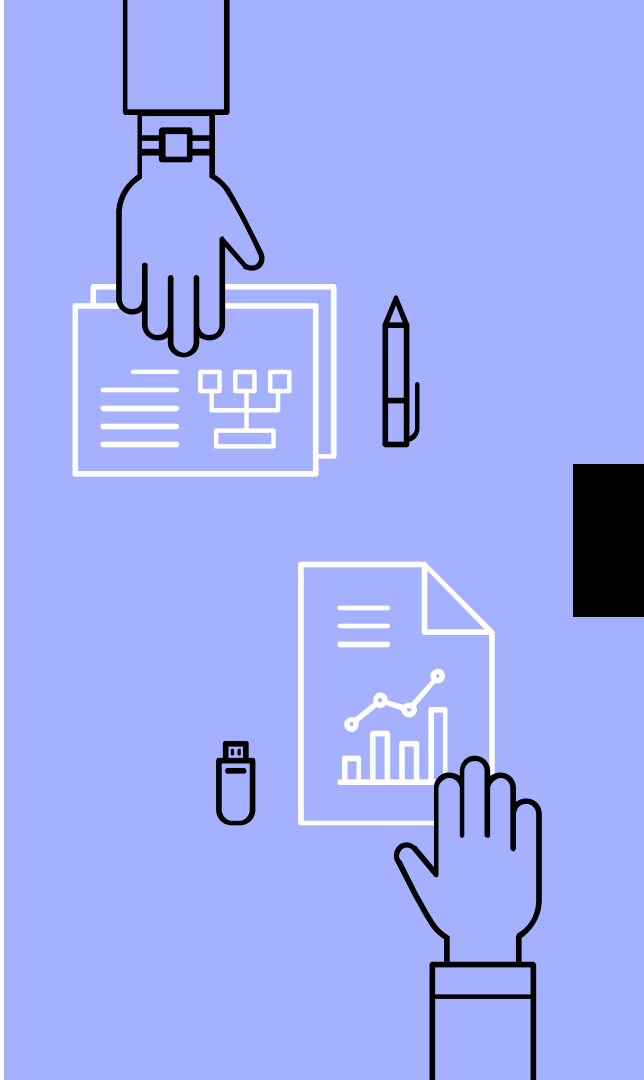
A business-oriented text communication that also supports:

- Everything Discord does, plus...
- Threaded conversations

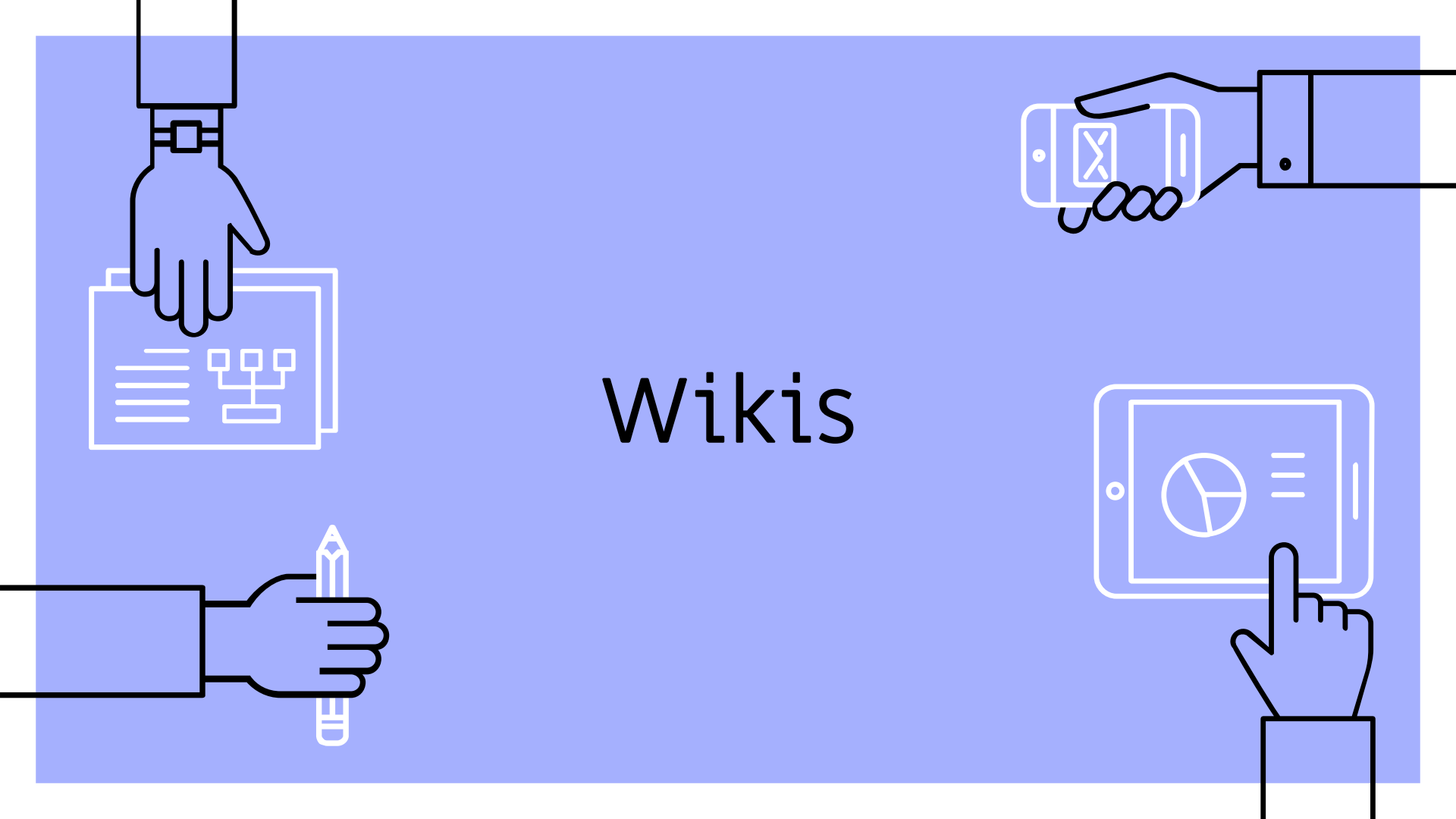


Riot.im:

A self-hosted, open-source alternative to Slack

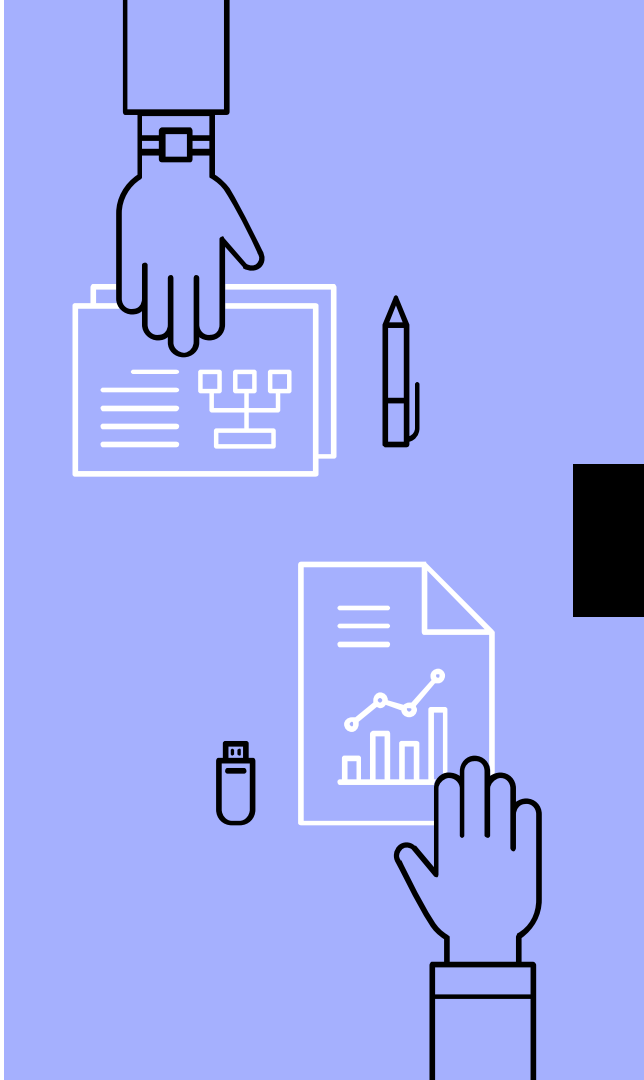


Wikis



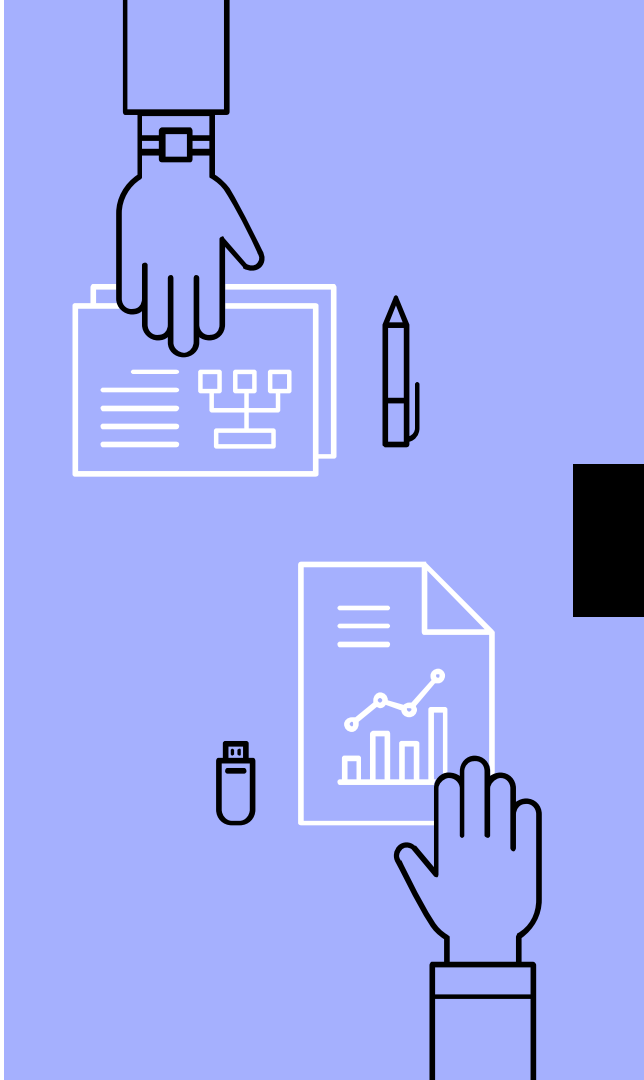
Purpose:

Professionally used as a collaborative game design document, a wiki is a synchronized documentation tool that retains a thorough history of changes that occured on each page. Most wikis provide easy editing, linking, and commenting.



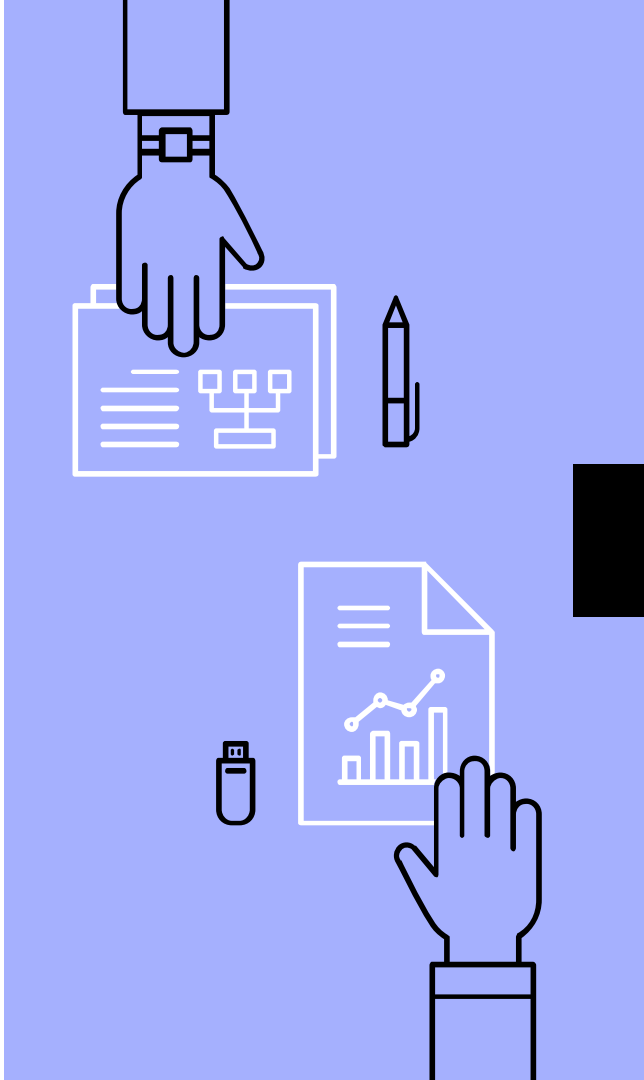
Wait, why not Microsoft Word?

- Can’t be synced with other users (except the online!)
- No way of providing real-time feedback
- Flexible linking between pages/documents
- Provide file attachments
- Not free



Tools:

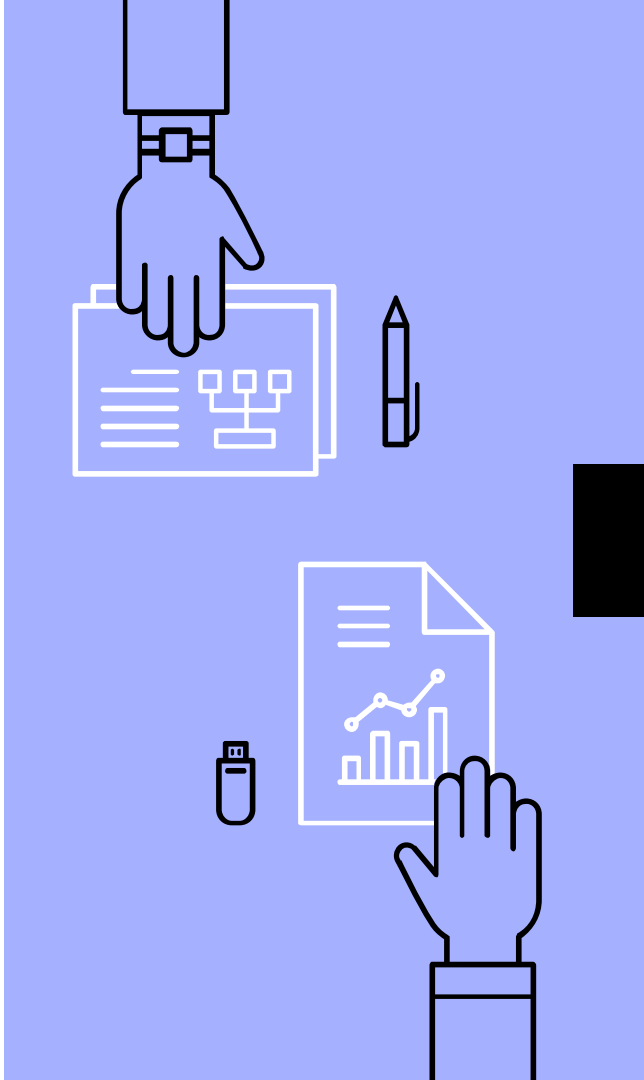
- Notion
  - notion.so
- Google Docs
  - docs.google.com
- TiddlyWiki
  - tiddlywiki.com
- MediaWiki
  - mediawiki.org



Notion:

An all-in-one note-taking solution.
Features include:

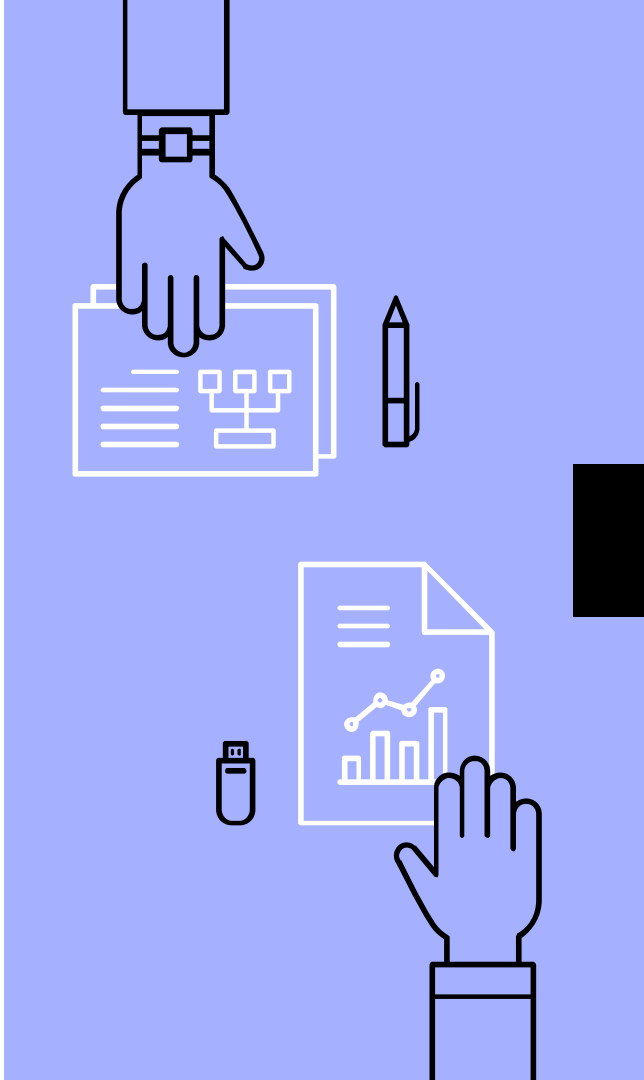
- Lists and checklists
- Task management modules
- Calendar
- Attaching and sharing files
- Commenting
- Team member assignment



Google Docs:

Technically not a wiki, but it does support real-time collaborative editing:

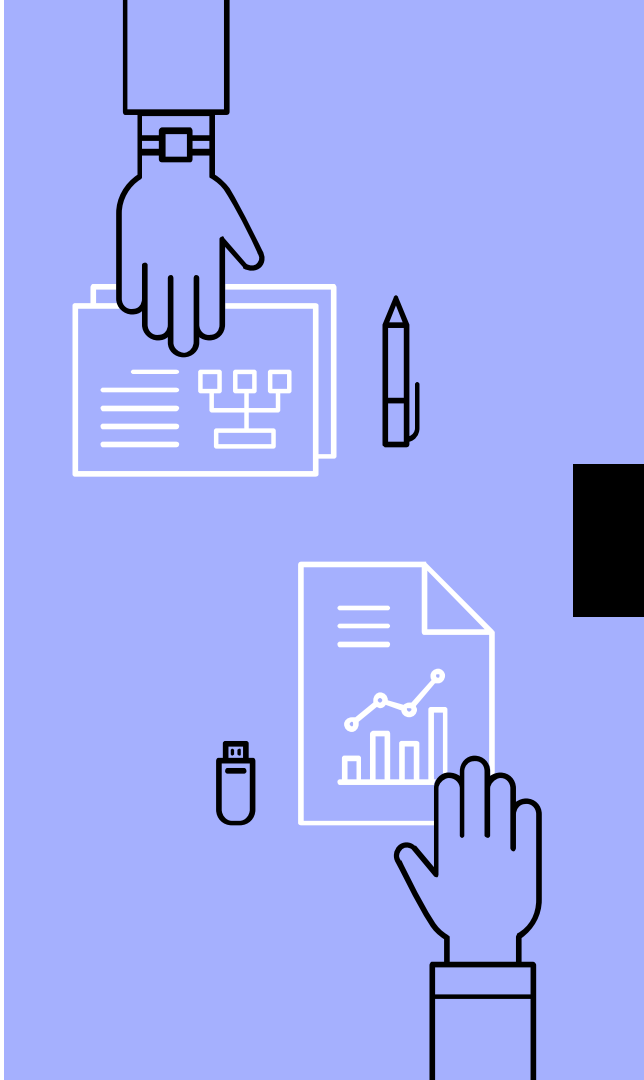
- Commenting
- History tracking and reverting
- Auto-generated table of Content
- Embedding other Google services, including Sheets



TiddlyWiki:

An offline-supporting wiki, this single-HTML file uses a note-like interface. It supports:

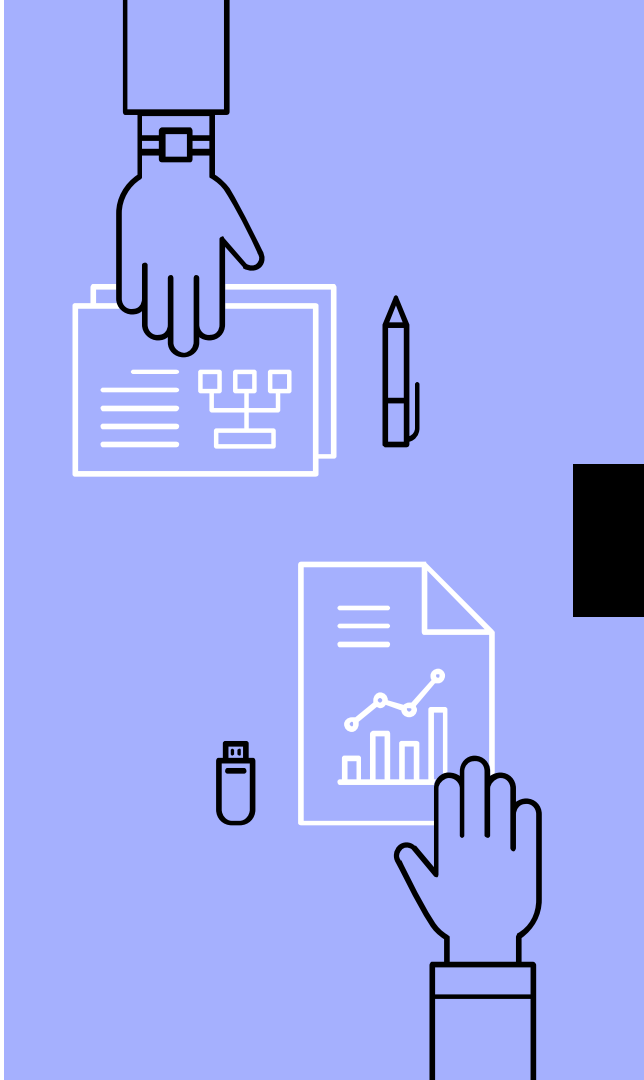
- Creating links that automatically creates a new page.
- Aggregation and site mapping of all notes.
- Online collaboration support.



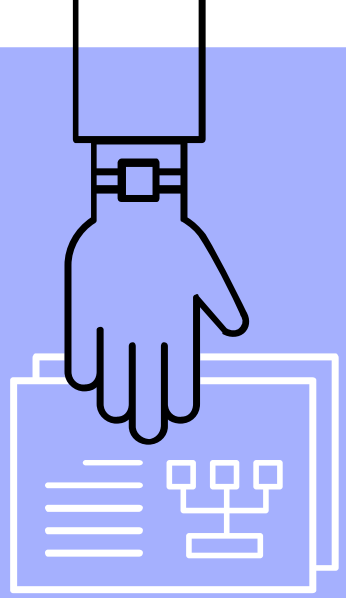
MediaWiki:

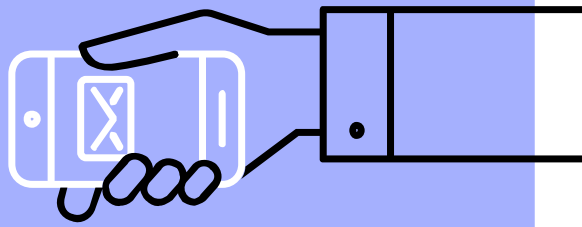
Primarily an online wiki, MediaWiki powers Wikipedia:

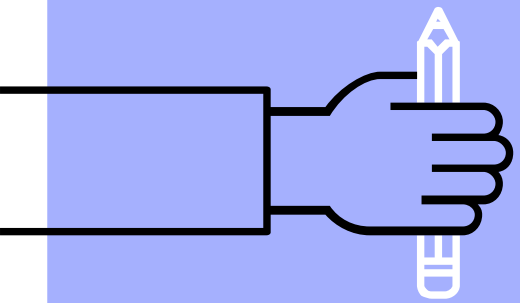
- File sharing
- Extensions
- Forums
- History tracking

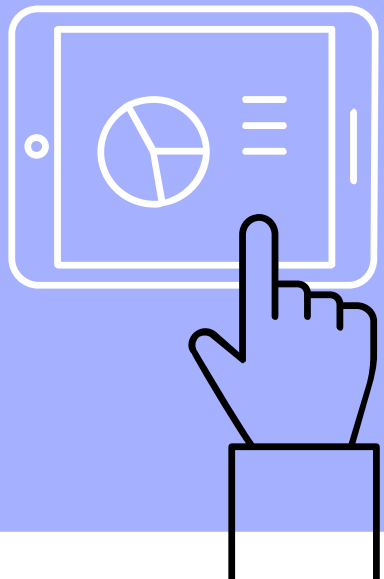


Task Manager



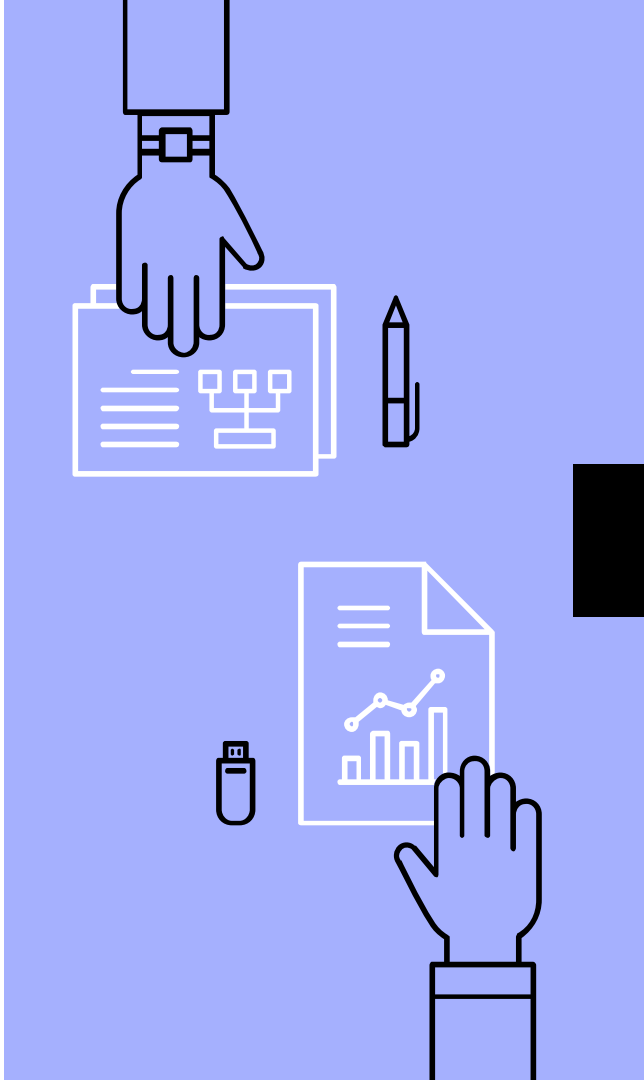






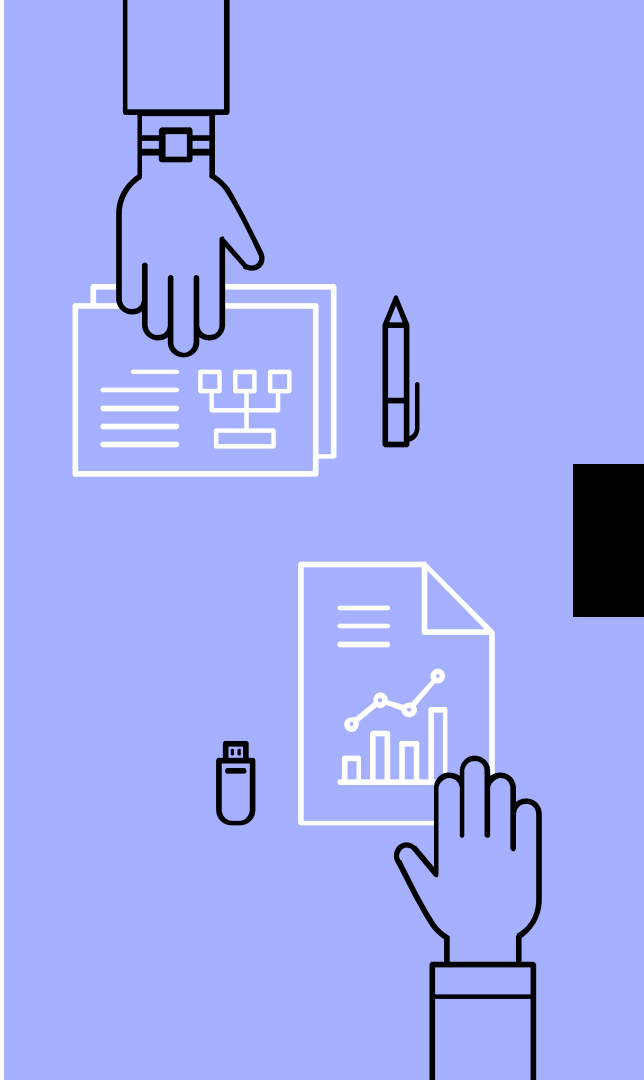
Purpose

Lists out the tasks left in a project, who is responsible for handling the task, and what state the task is in. Also provides grouping tasks, marking a task as dependent/blocking another, provide subtasks, etc.



Wait, why not a good-old checklist?

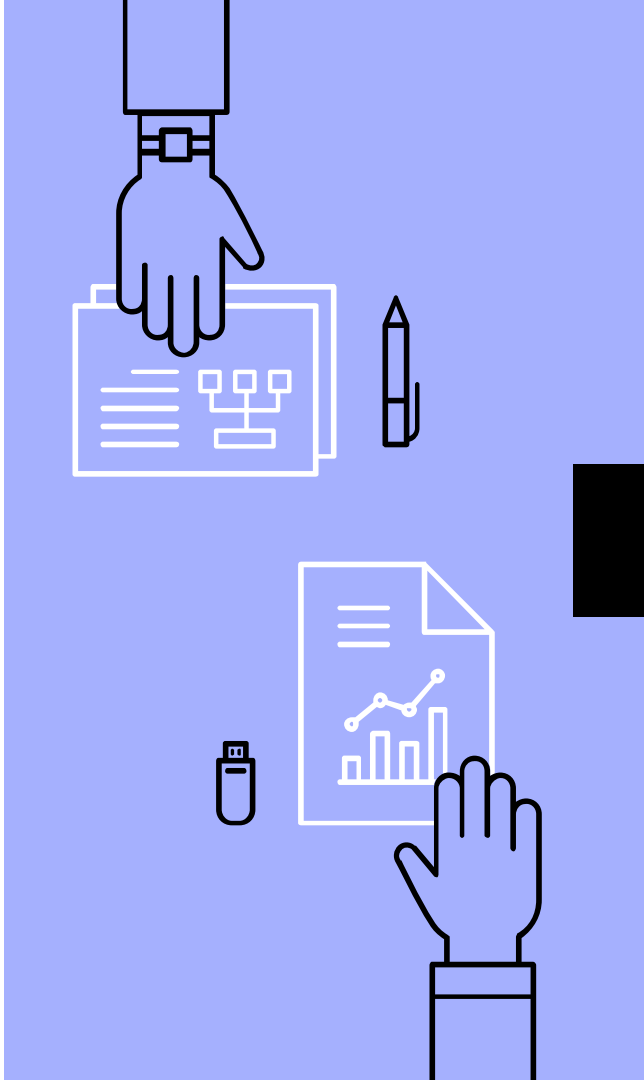
- Can’t be synced with other users
- No way of providing real-time feedback
- Hard to categorize tasks
- Hard to indicate estimates on how long a task takes
- Difficult to group tasks



Some Notes About the Following Tools:

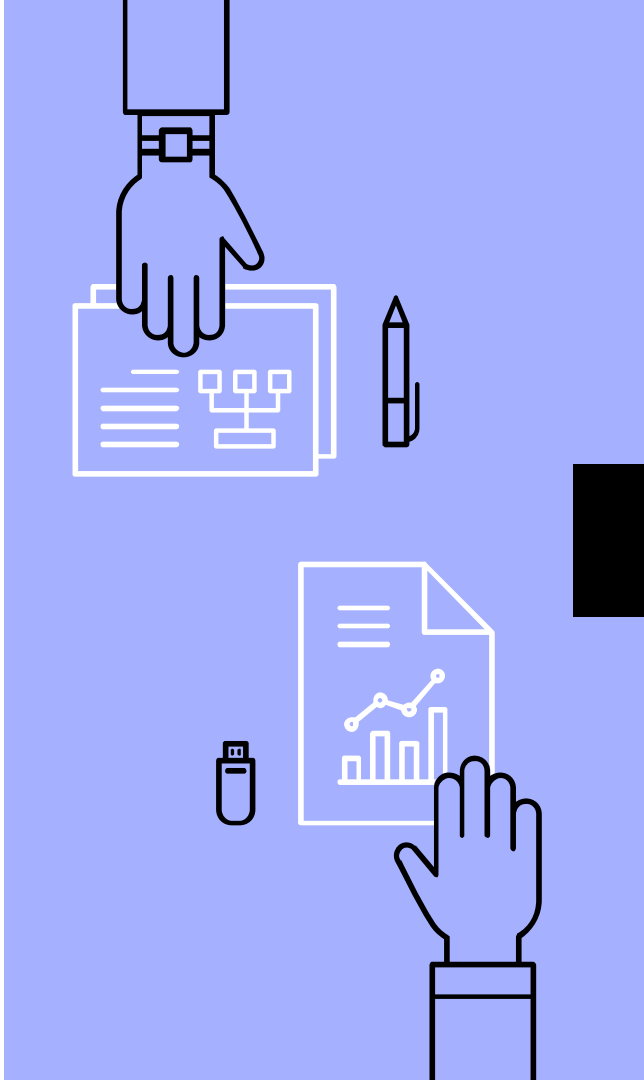
Most modern task management tools are either built for Agile development process.

- Agile focuses on always having a working project after every “sprint”
- Sprints are typically two weeks long.



Some Notes About the Following Tools:

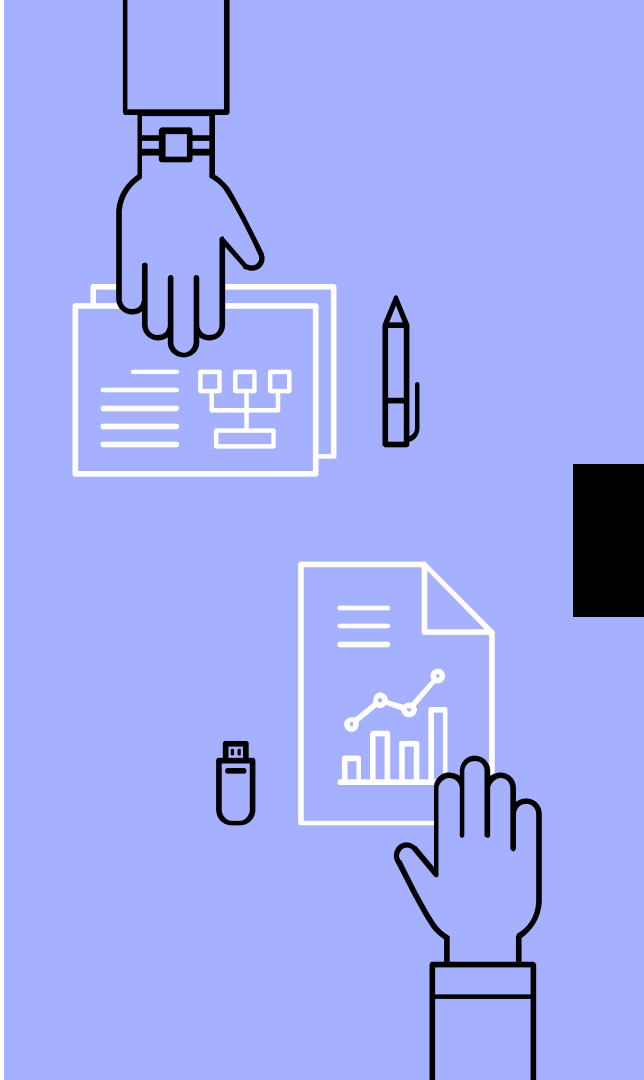
- Sprints builds on top of each other to reach a “milestone,” a longer-term goal.
- To support these goals, tasks are often categorized to a specific sprint, which in turn is categorized to a milestone.



Some Notes About the Following Tools:

Finally, most task managers follow the kanban board design.

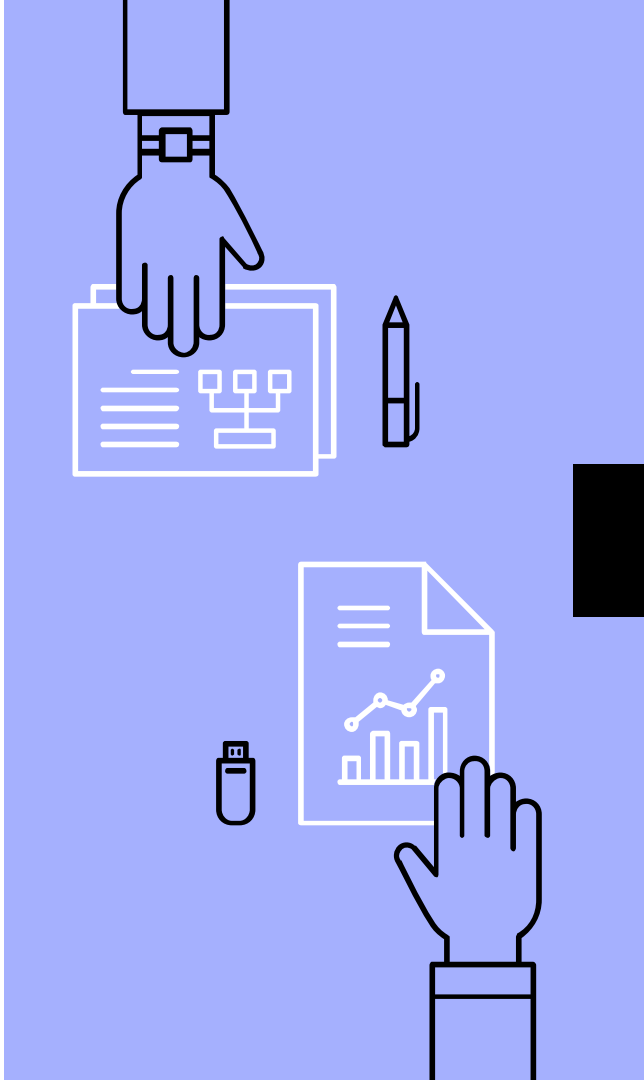
- Checklist statuses are binary, either checked or unchecked.
- Kanban's equivalent of a "status" are lists, i.e. the list a task is in marks the status of that task.



Some Notes About the Following Tools:

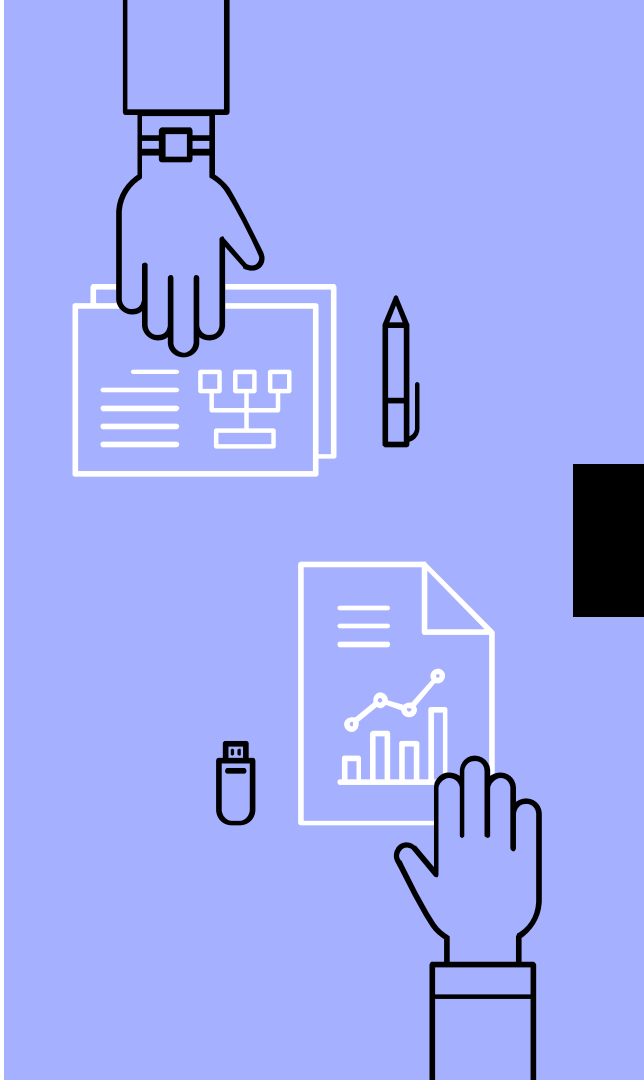
Common list names include:

- Suggestion
- In Sprint
- In Progress
- Needs Review
- Completed



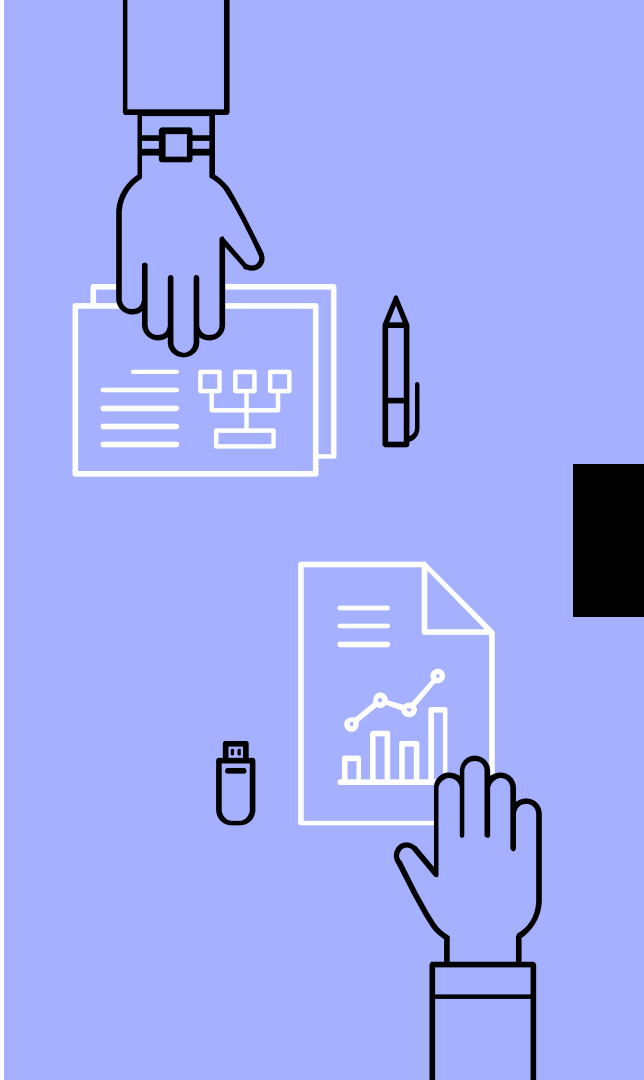
Online Tools:

- Workflowy
  - workflowy.com
- Trello
  - trello.com
- Asana
  - asana.com



Self-hosted Tools:

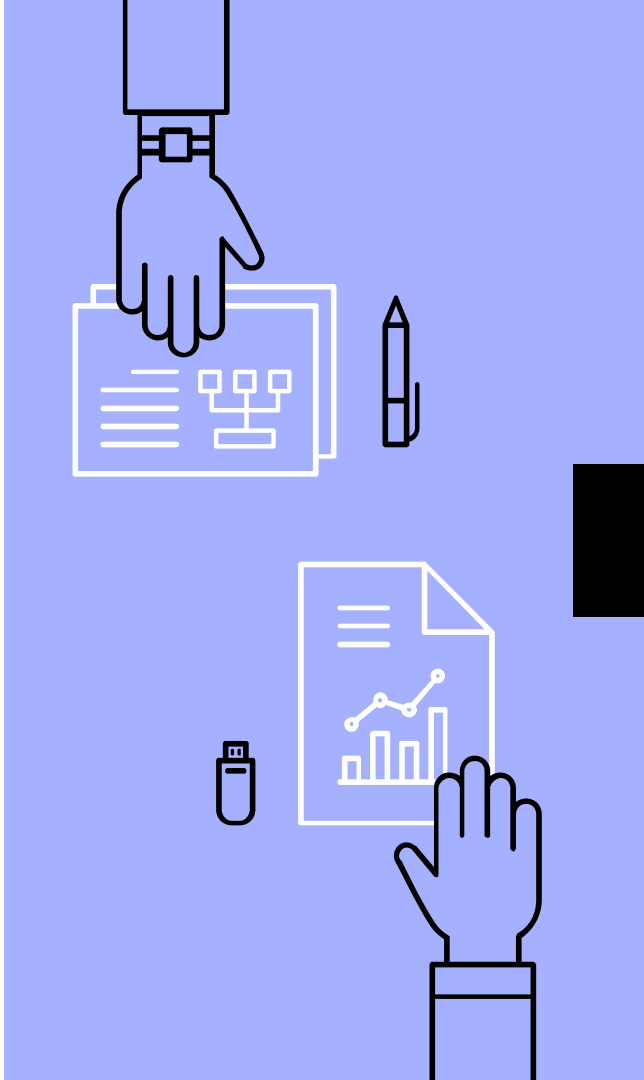
- Taigo.io
  - Taiga.io
- Collabtive
  - collabtive.o-dyn.de



Workflowy:

A good old online checklist. This simple tool supports:

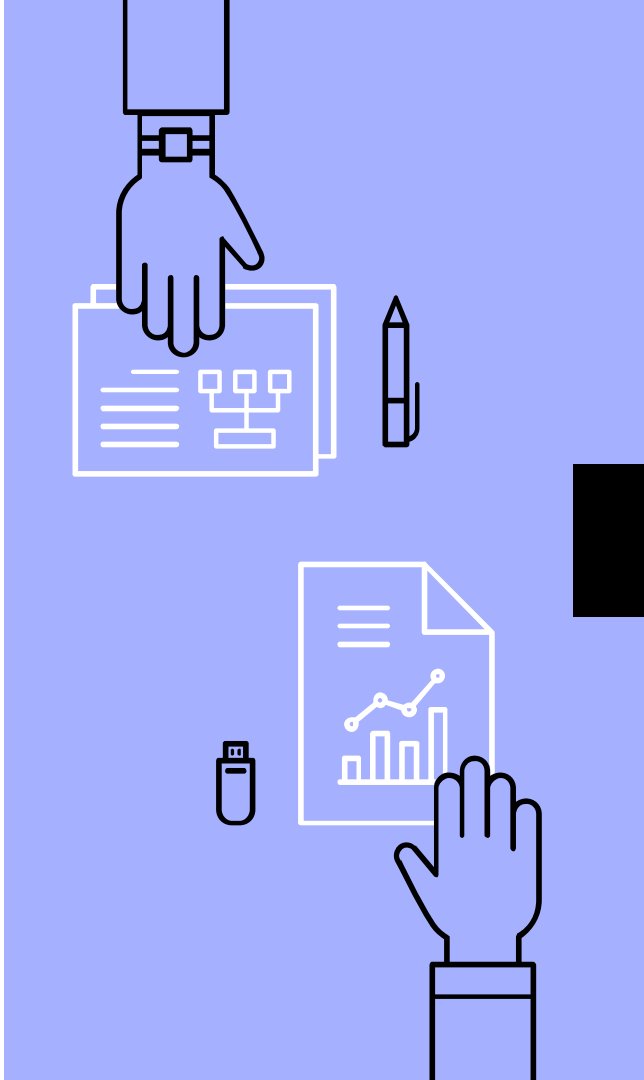
- Assigning tasks to a team member
- Nested subtasks
- Hashtags for easier search



Trello:

One of the most popular online kanban board. This simple tool supports:

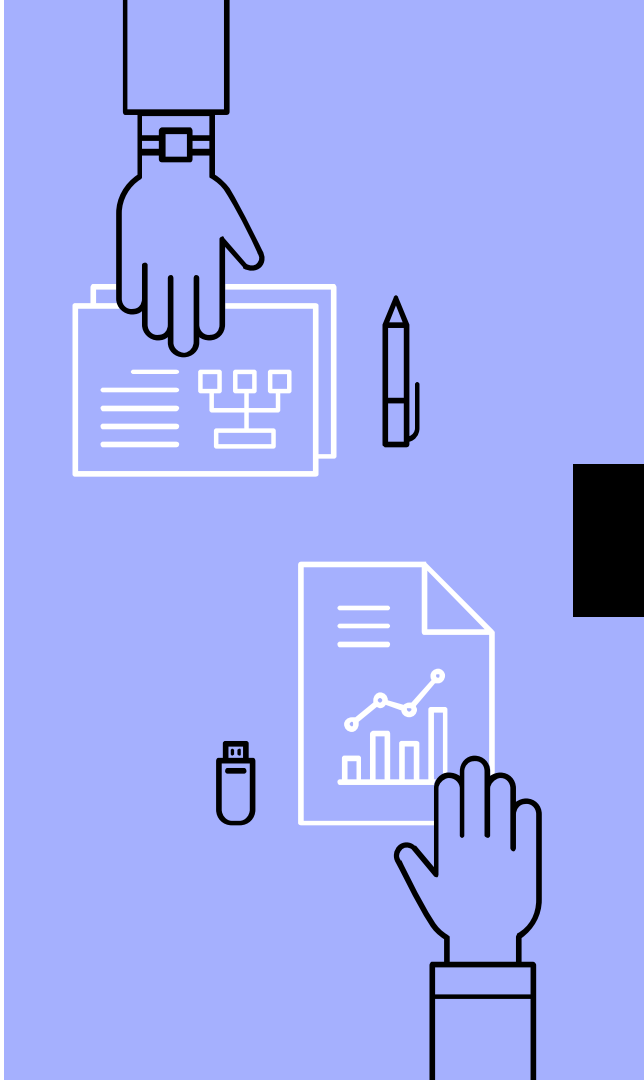
- Assigning tasks to a team member
- Custom lists
- Custom tags
- File and link attachments
- Deadline and calendar support



Asana:

Comparably more complex, Asana supports multiple workflows:

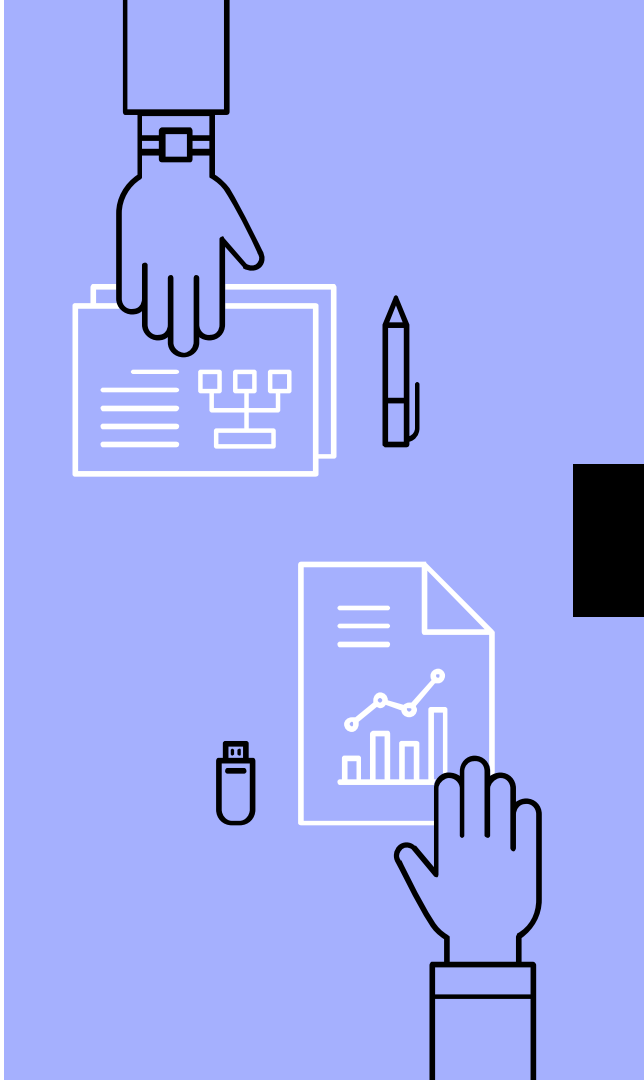
- Regular checklist
- Filtered task list
- Kanban board
- Gnatt board for graphing
- Weekly summary report



Taigo.io:

A self-hosted task management software not unlike Asana:

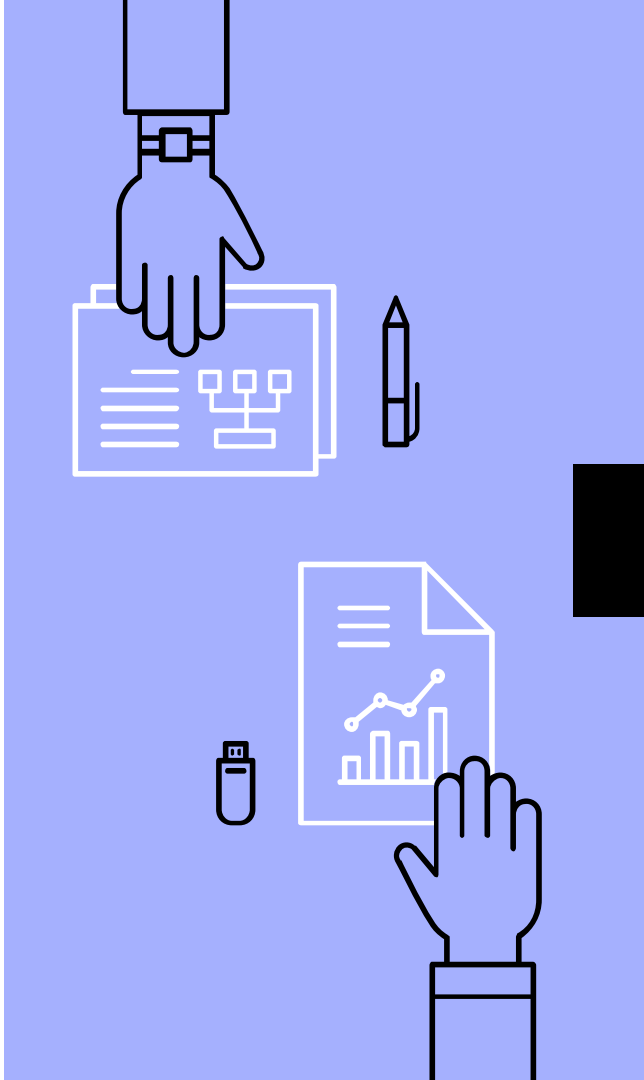
- Regular checklist
- Filtered task list
- Kanban board
- Gnatt board for graphing



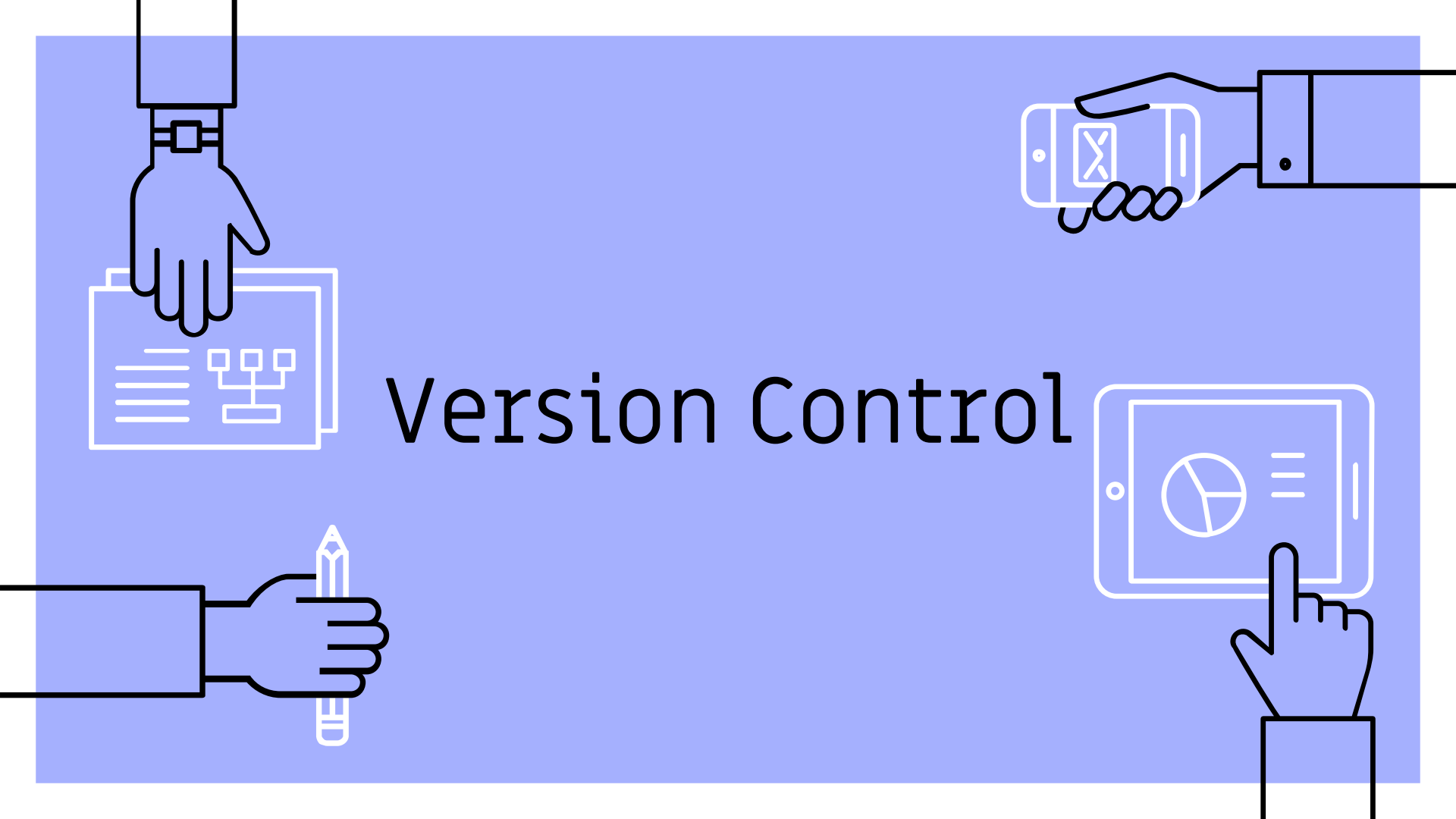
Collabtive:

An (unfortunately not very active) self-hosted task manager, using good old checklist system. Supports:

- File Sharing
- Instant messaging
- Contact sharing

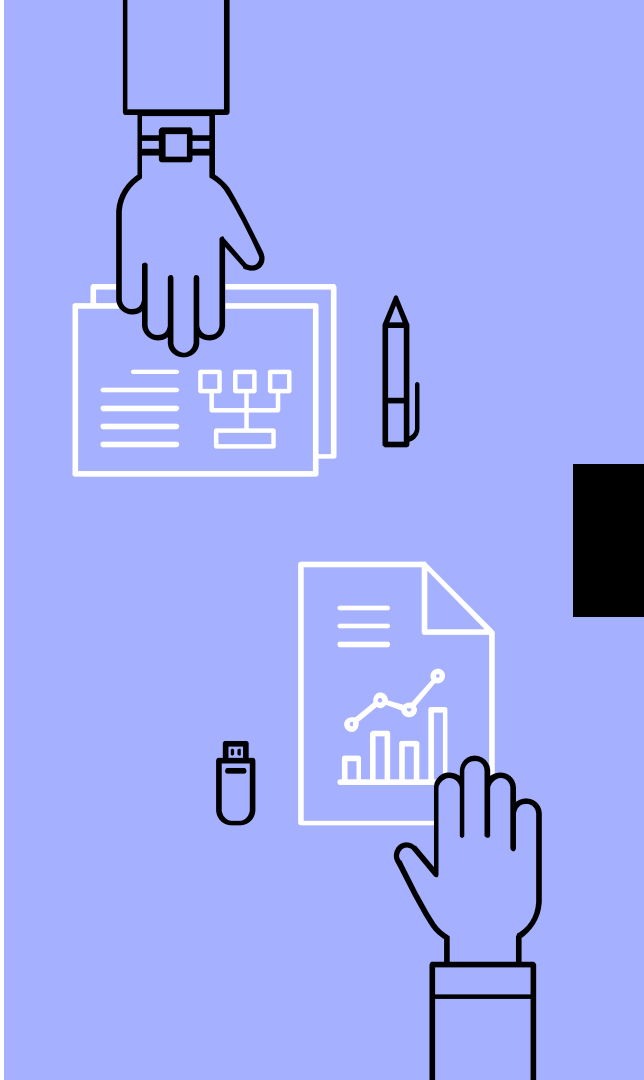


Version Control



Purpose

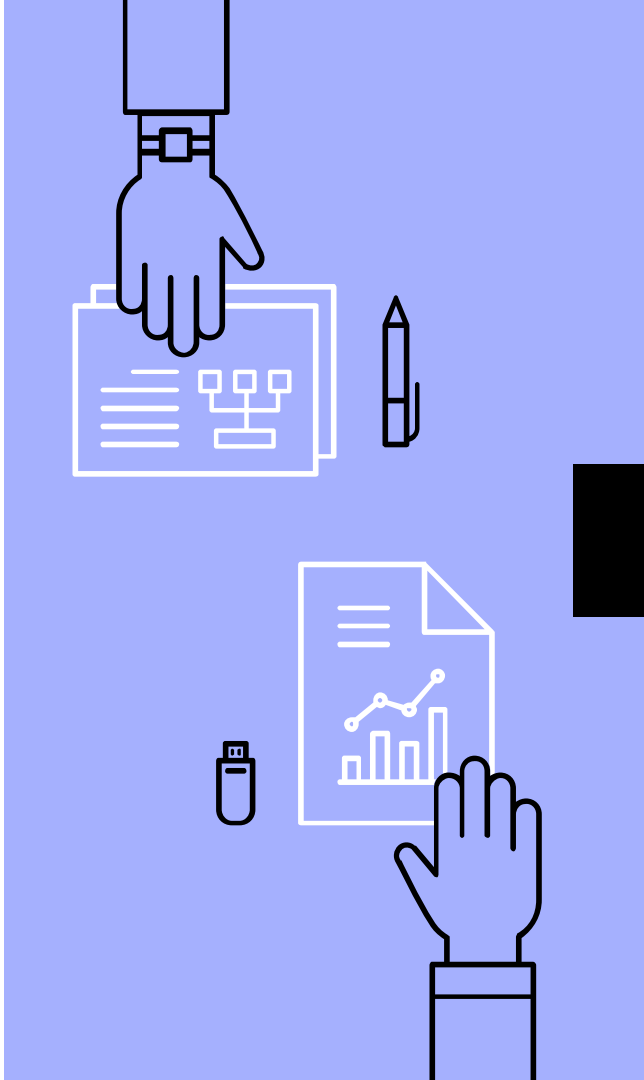
Version controls (also known as source control management, or SCM) is a commonly command line client-&-server tool that retains a history of how the project's assets have changed. Each tool have various capabilities of merging files, especially text and code.



More About Version Control

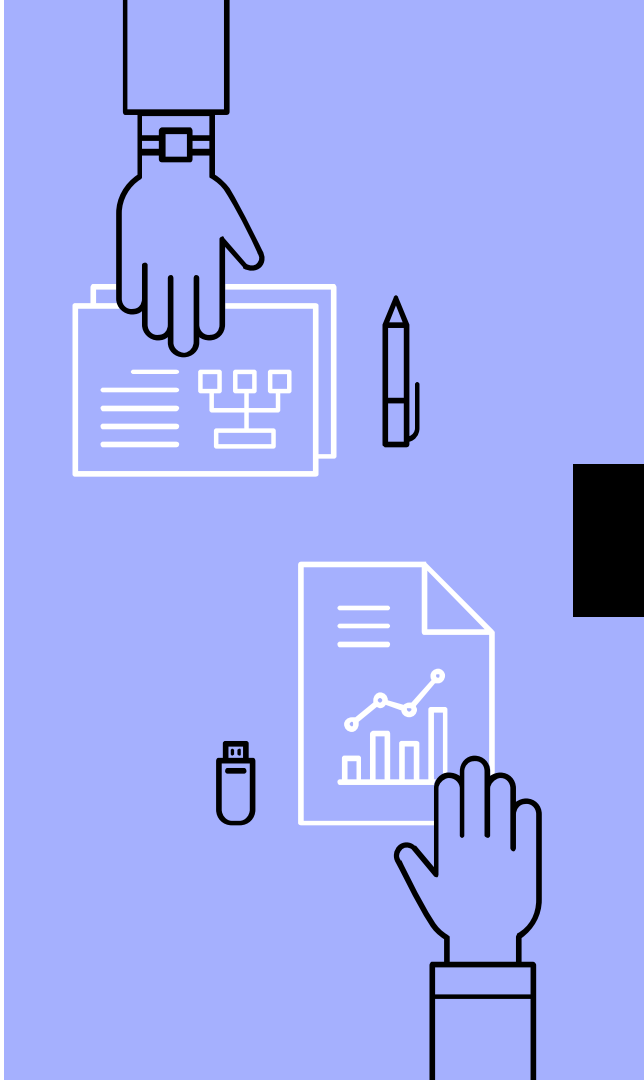
Version controls are powerful, but also a bit complex. They can:

- Reverse changes (including modifications and deletion) in files, folders, or the entire project to a previous state.
- Detail who made what changes to the project, and how.



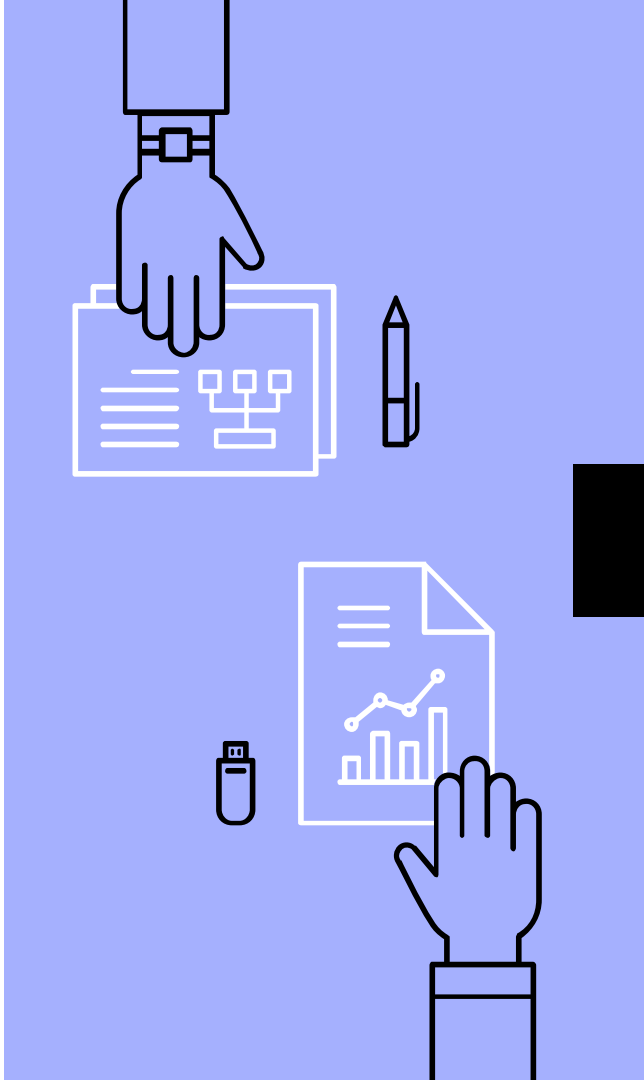
More About Version Control, cont.

- Bookmark specific points in history, great for marking releases.
- Provide sandboxes to experiment with new features and enhancements.



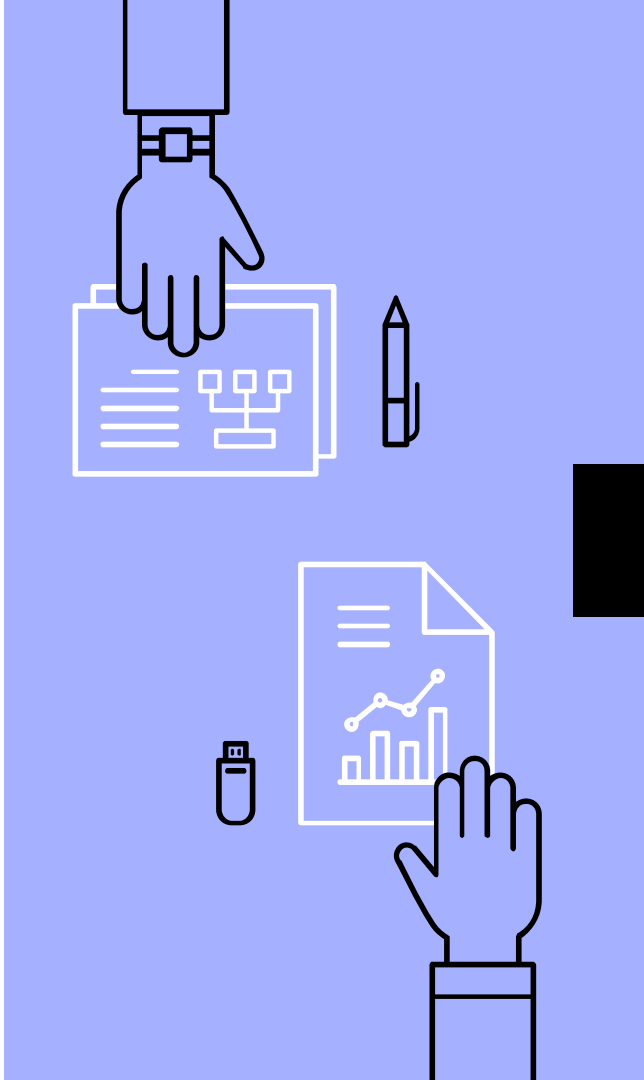
Wait, why not use multiple folders?

- Can’t be synced with other users
- Uses a *lot* less storage space
- Provides extra features, like branch and tags
- GUI programs provide a list of files that actually changed



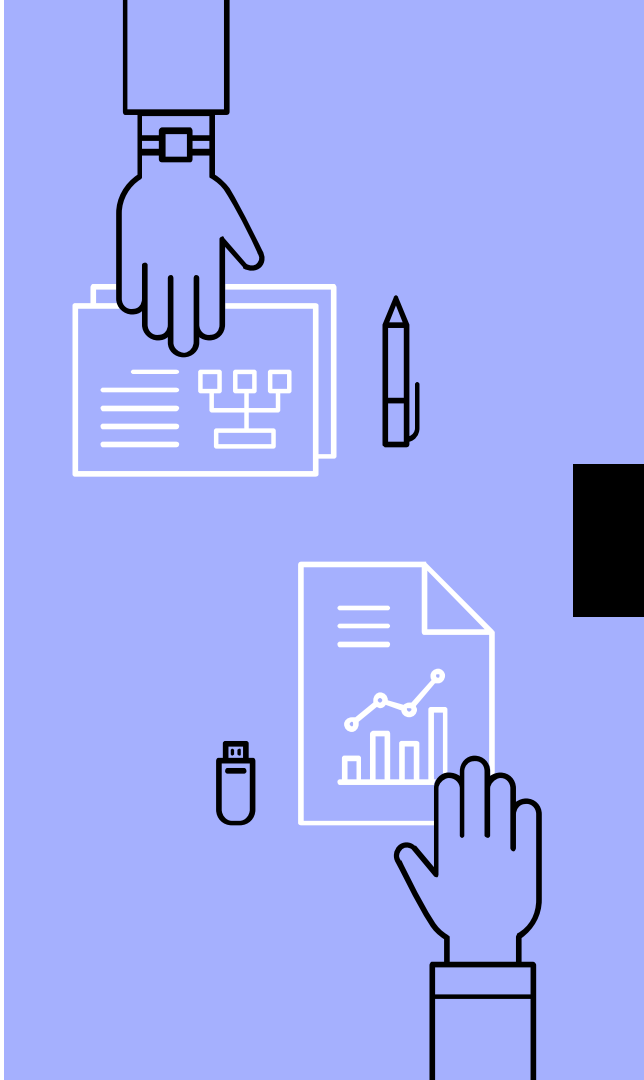
Terms:

- Repository
  - The server; retains the history of the project
- Client
  - Your computer
- GUI
  - Short for Graphical user interface; a proper graphical menu and application



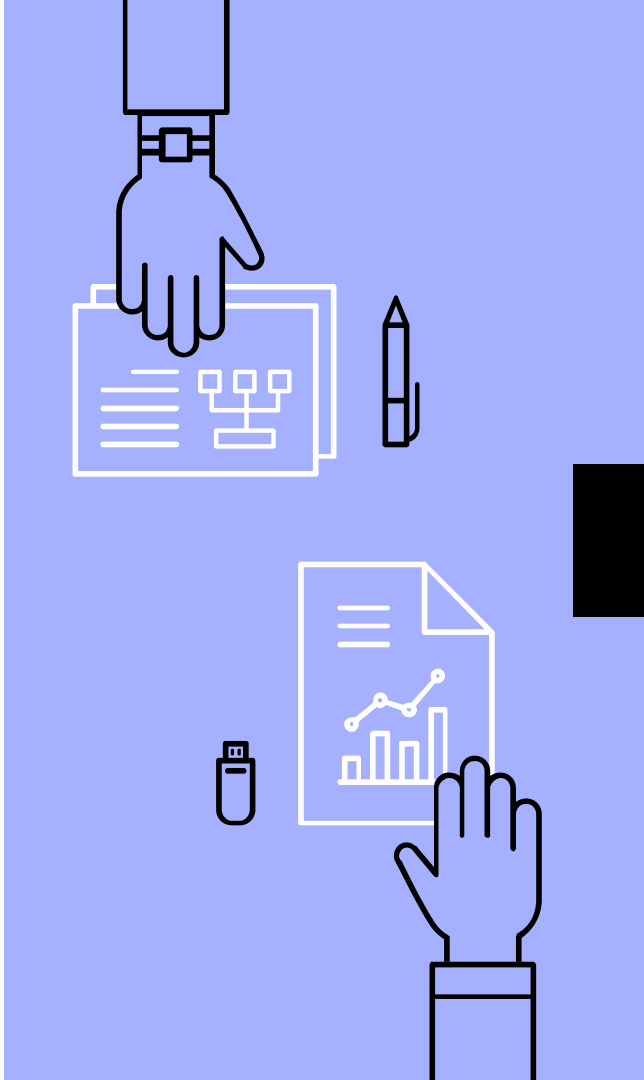
Tools:

- Git
  - git-scm.com
- Mercurial
  - mercurial-scm.org
- Subversion
  - subversion.apache.org



GUI Tools:

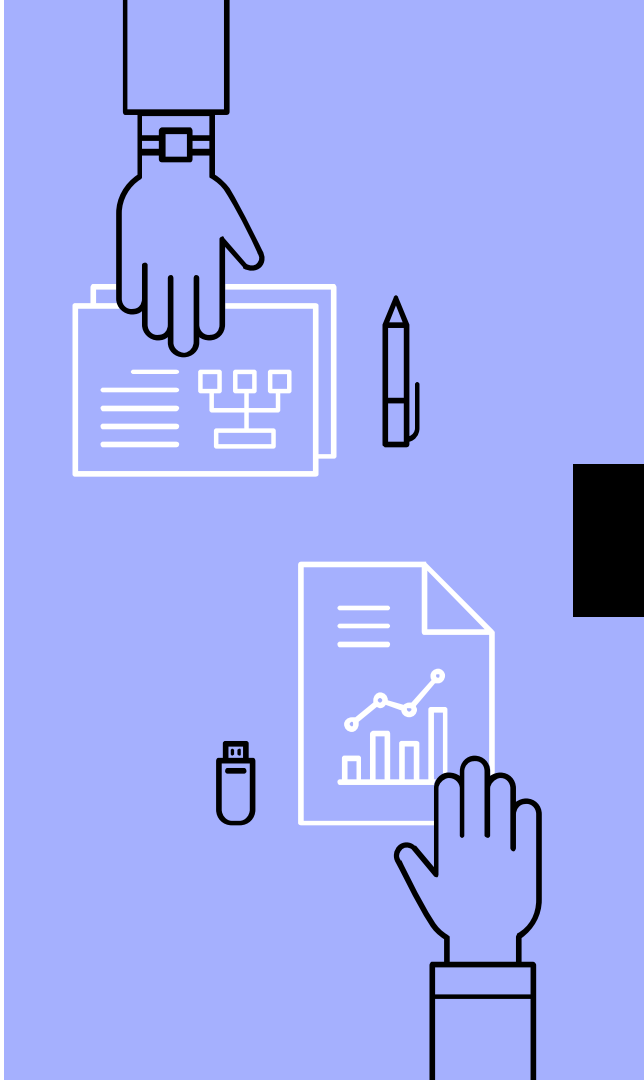
- TortoiseGit, TortoiseHg, & TortoiseSVN
  - tortoisegit.org
  - tortoisehg.bitbucket.io
  - tortoisesvn.net
- GitHub Desktop
  - desktop.github.com
- Sourcetree
  - sourcetreeapp.com



How client-server SCM works (e.g. Subversion):

Commit: pushes changes from the client to the repository, creating a new revision in the repository's history.

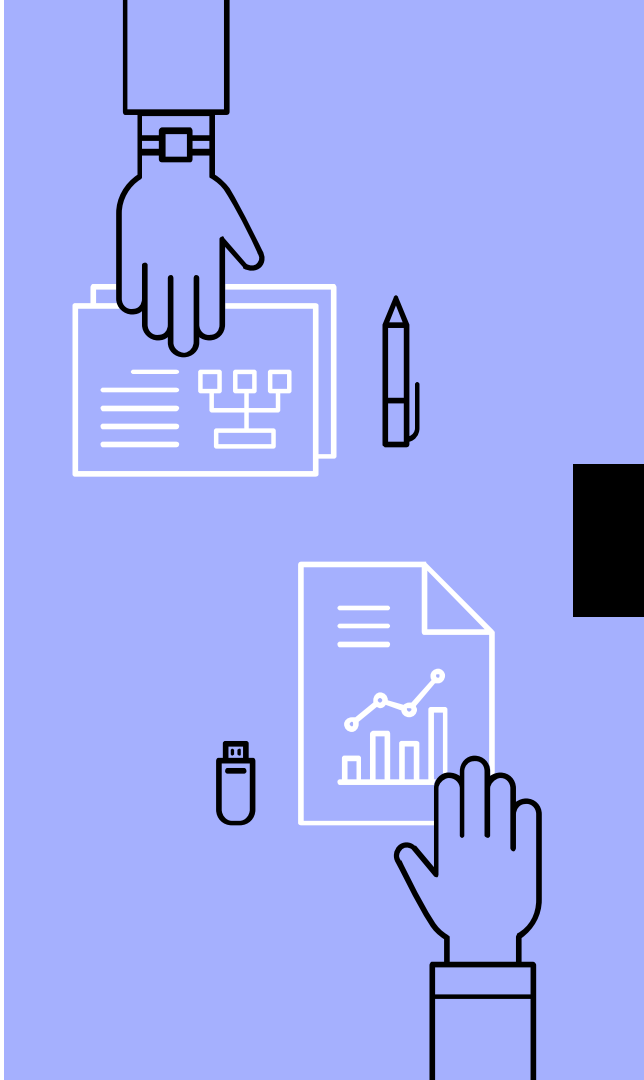
Update: pulls the relevant history from the repository, and applies it to the client. The client is then updated to the latest revision.



How distributed SCM works (e.g. Git, Mercurial):

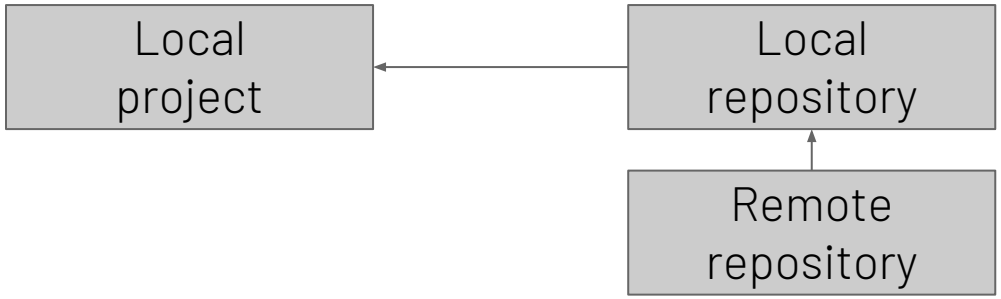
Commit: pushes changes from the local project to the local repository, creating a new revision. Server is untouched.

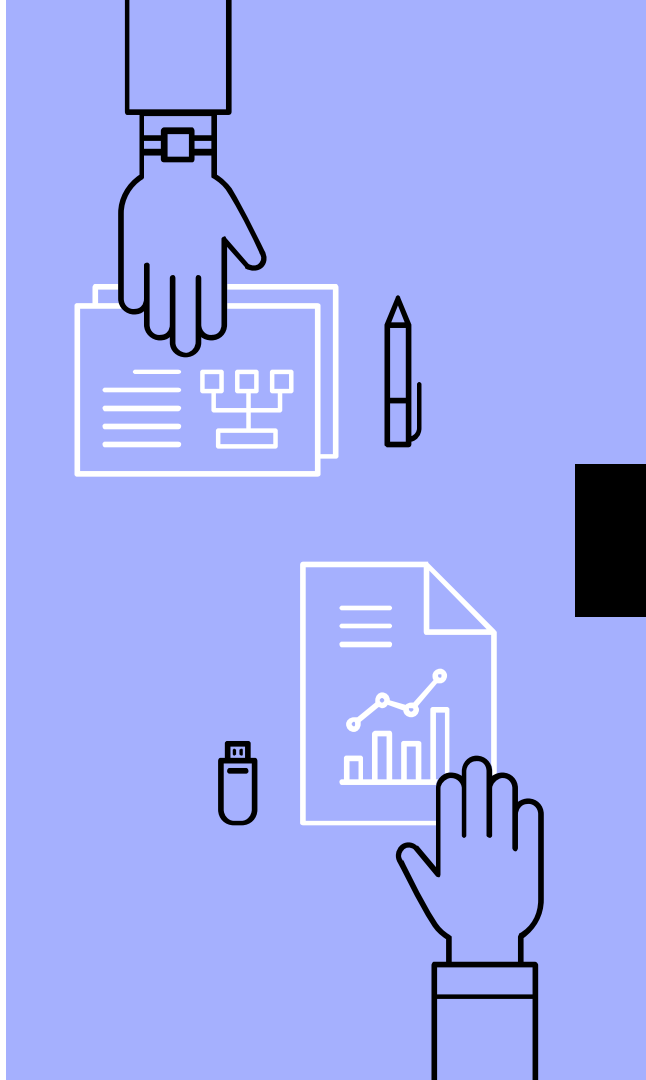
Push: pushes changes from the local repository to the server, merging the two histories.



How distributed SCM works (e.g. Git, Mercurial):

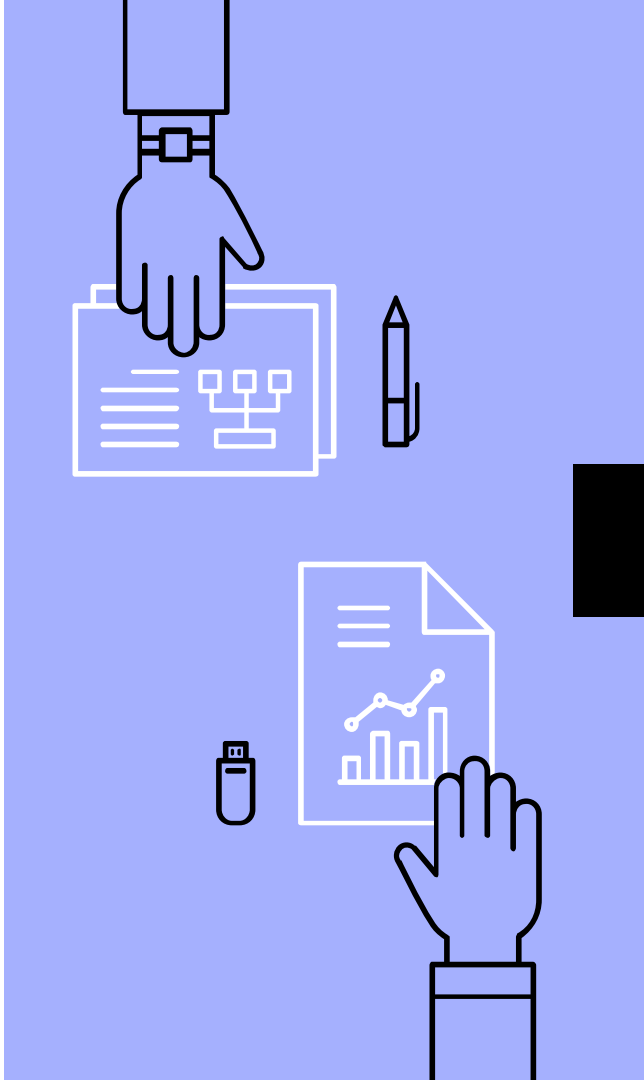
Pull: downloads the remote history onto the local repository, merging their histories. Then it applies the latests changes to the local project.





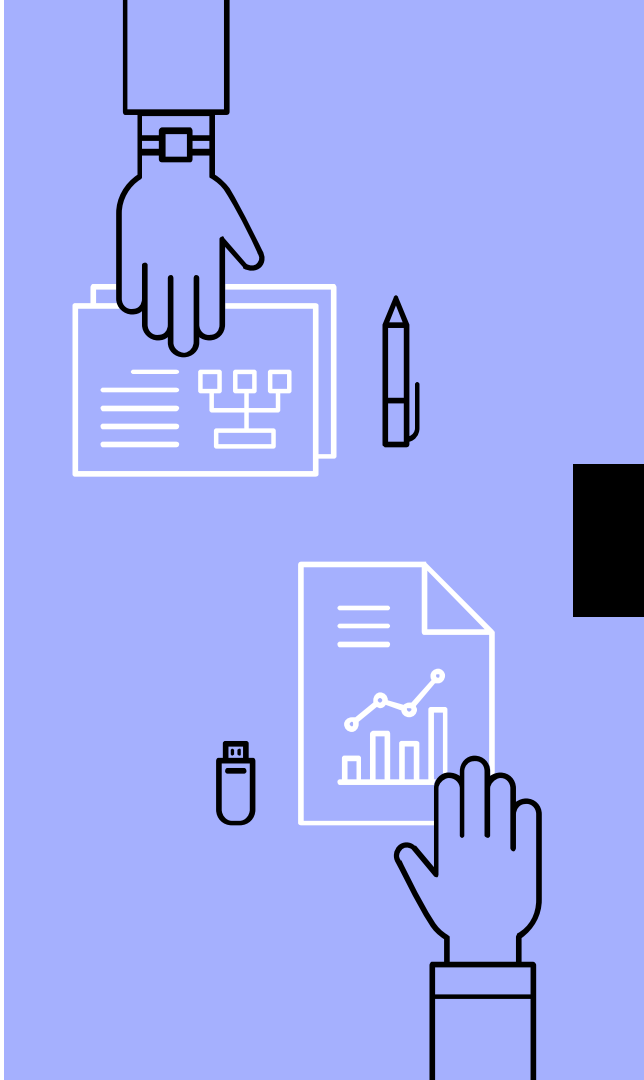
Client-Server SCM:

- Pros
  - Easier to use
  - Has a “lock” feature: always keeps only one team member editing a file
  - Per-folder revision handling
- Cons
  - Poor merge algorithms



Distributed SCM:

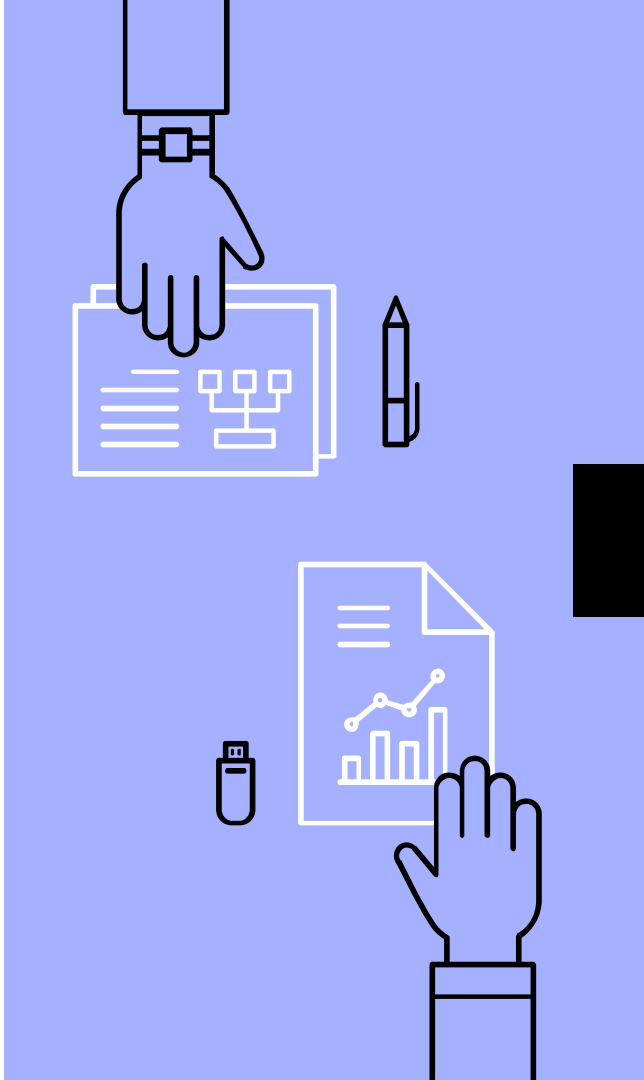
- Pros
  - Much more efficient
  - Per project revision
  - Better merging, handling of moved and renamed files
  - Better branching, tags
- Cons
  - Harder to use



All-in-One Solutions

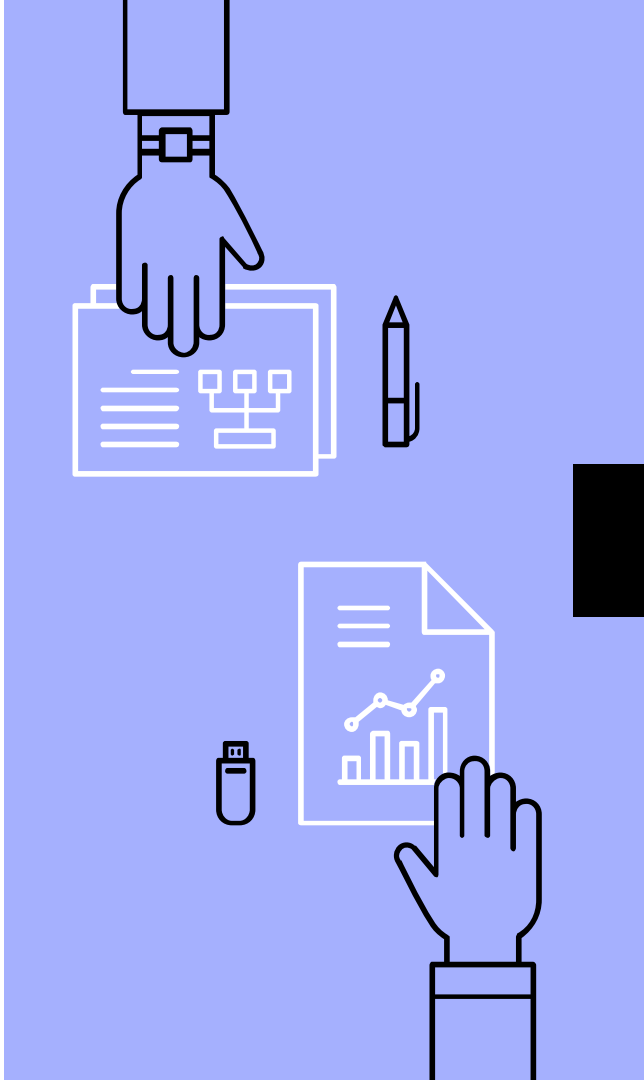
Online Tools:

- GitHub
  - github.com
- BitBucket
  - bitbucket.org
- Microsoft Teams
  - products.office.com/en-us/microsoft-teams/free

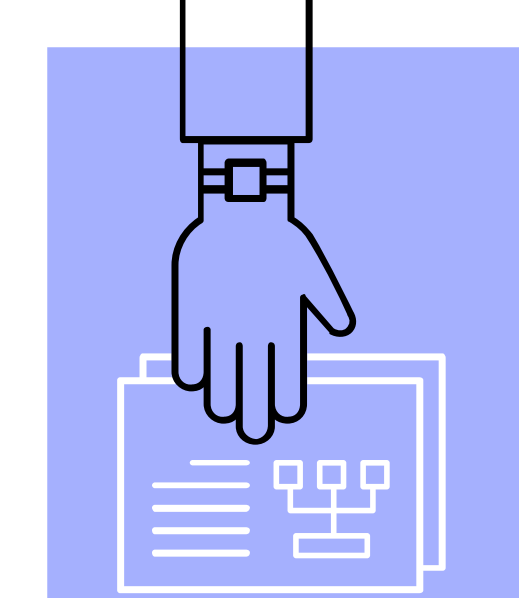


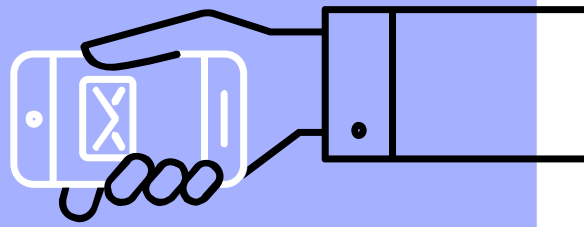
Self-hosted Tools:

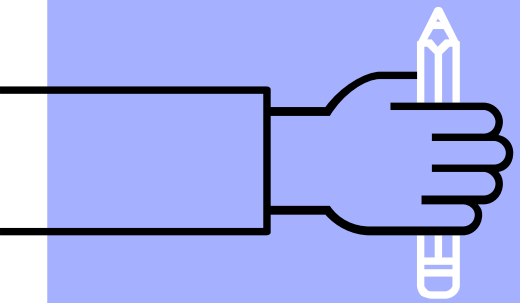
- Redmine
  - Redmine.org
- Phabricator
  - phacility.com/phabricator
- Fossil
  - fossil-scm.org

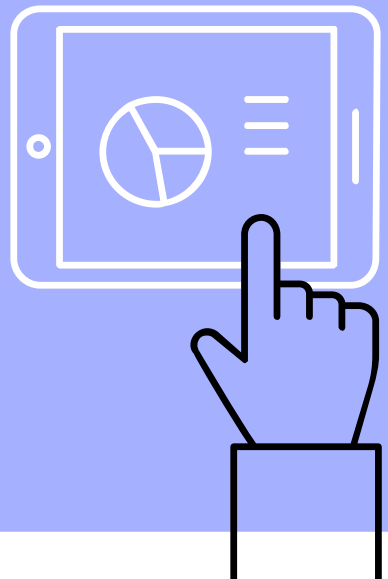


Other Tools to Consider





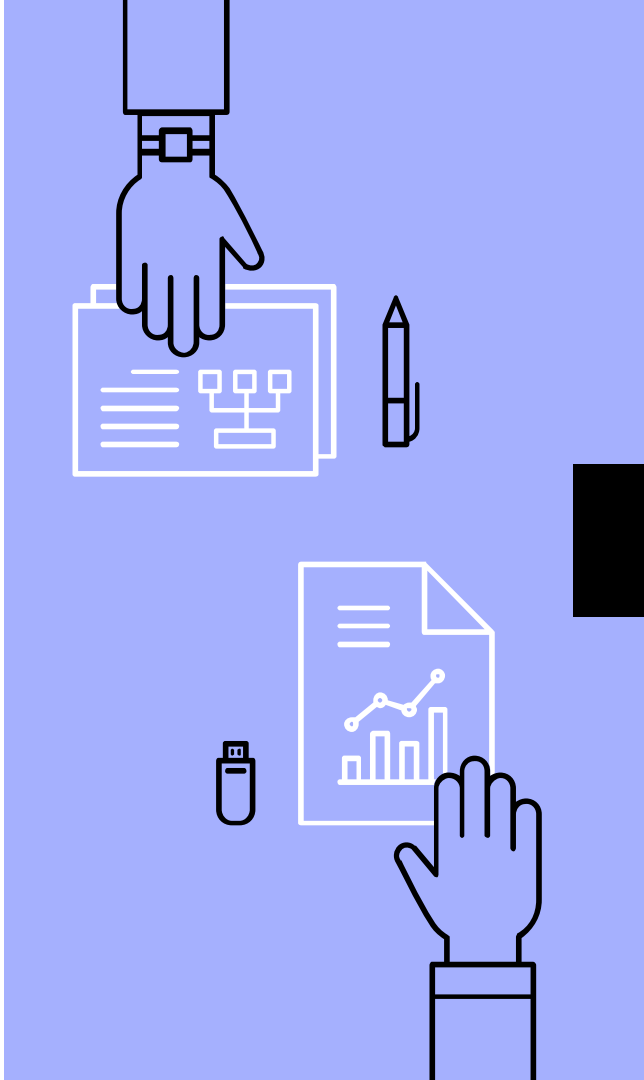




Content Management Systems:

A blog + extra features. Examples:

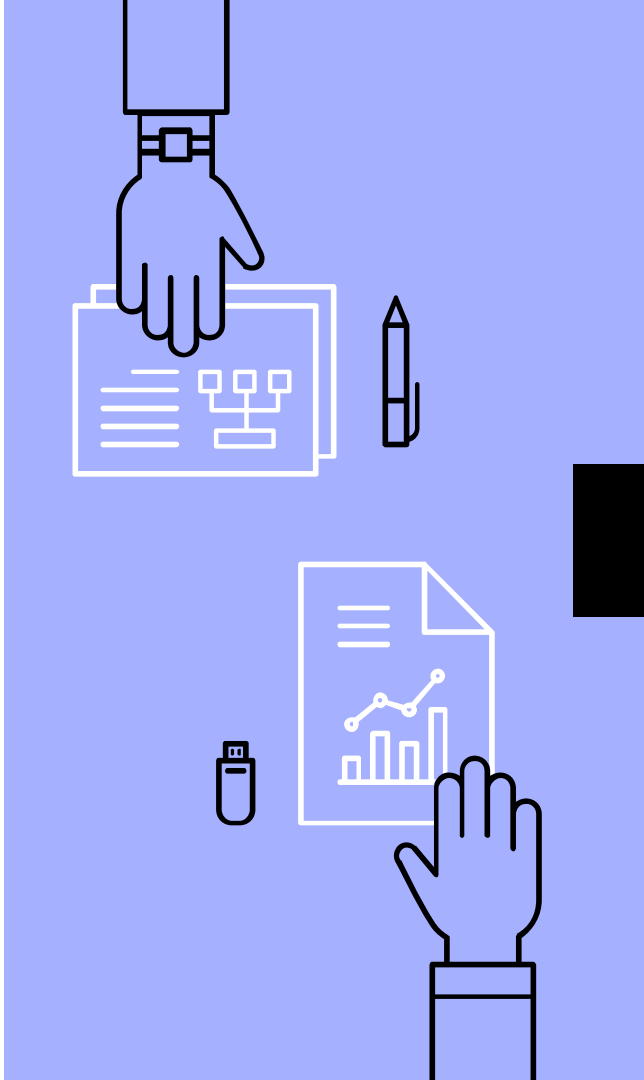
- Wordpress
  - wordpress.org
- Drupal
  - drupal.org
- ghost
  - ghost.org
- Grav
  - getgrav.org



Forums:

A threaded message board for fans to interact with your team. Examples:

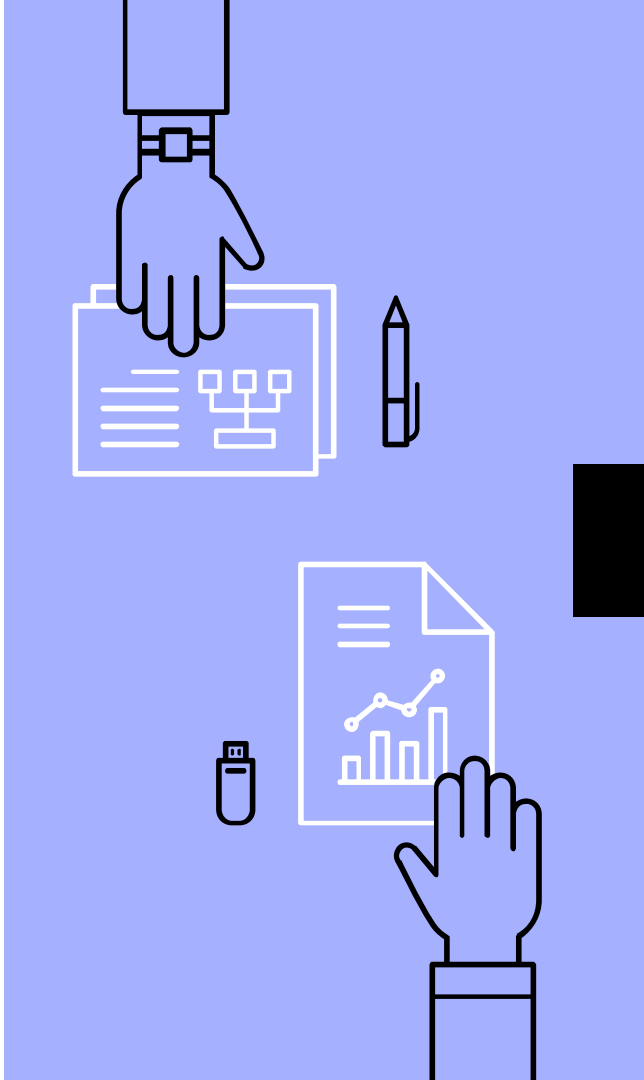
- bbPress
  - bbpress.org
- FluxBB
  - fluxbb.org
- phpBB
  - phpbb.com



Continuous Integration:

A server that monitors if there are any changes to a repository. If so, compiles the project to a new executable. Examples:

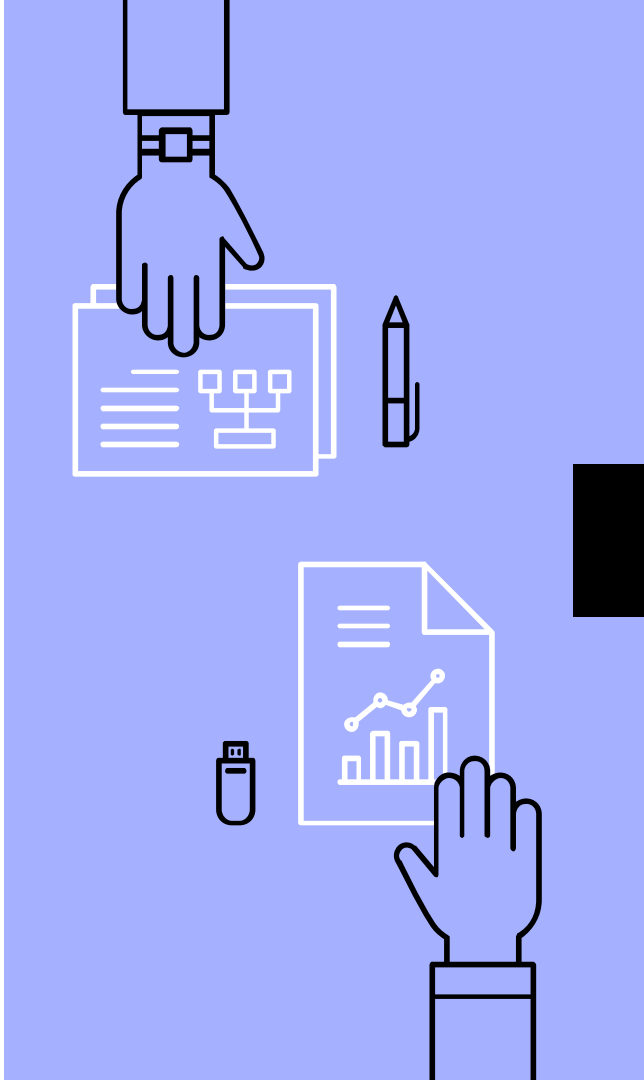
- Unity Cloud Build
  - build.cloud.unity3d.com
- Jenkins
  - jenkins.io



Automated Testing:

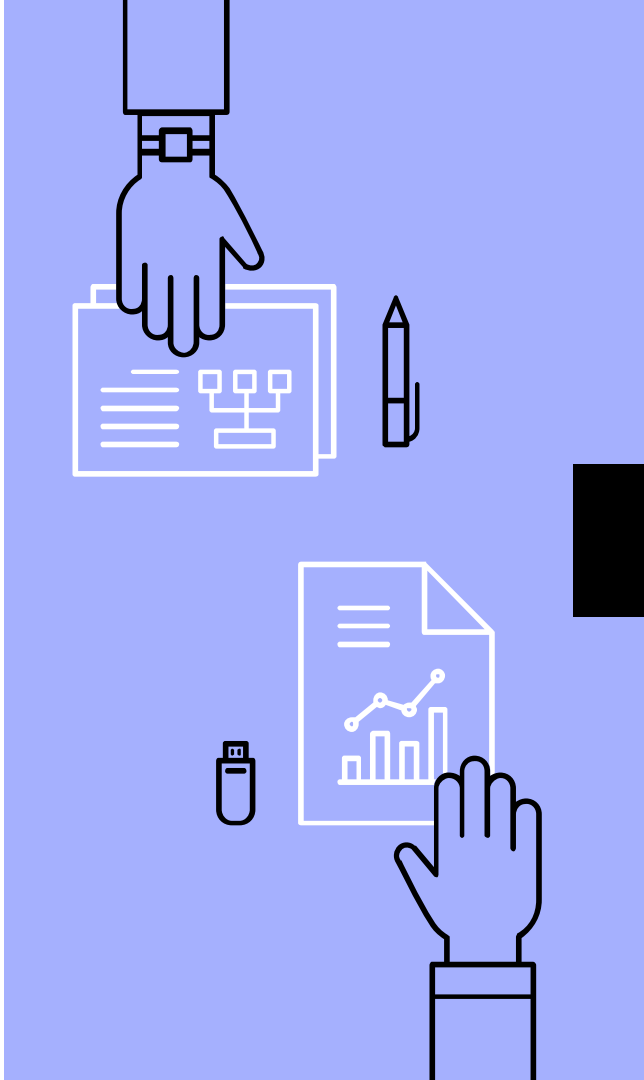
An app or scripting framework to simulate the user interaction. Often integrates with Continuous Integration. Examples:

- Appium
  - appium.io
- SikuliX
  - ikulix.com
- Selenium



Resources:

- AlternativeTo
  - alternativeto.net
- Awesome-Selfhosted
  - github.com/Kickball/awesome-selfhosted



Any Questions?