Dotfiles

These are all my config files. There are two branches. Master and gentoo. Master is my arch config and gentoo is well, gentoo. They're mostly the same, but if you're not a gentoo user then I recommend you pull from the master branch instead. Its cleaner.

Getting started

If youre wondering why my directories look like that its because I use stow. You'll need it if you want to pull from the repo. Stow is a game changer when orgnazing your dotfiles. You never have to worry about updating a file when pushing it to a repo. The way stow works makes that process automatic.

Already a pro? Just edit this README.md and make it your own. Want to make it easy? Use the template at the bottom!

Add your files

- Create or upload files
- Add files using the command line or push an existing Git repository with the following command:

```
cd existing_repo
git remote add origin https://gitlab.com/chris-m/dotfiles.git
git branch -M main
git push -uf origin main
```

Integrate with your tools

- Set up project integrations

Collaborate with your team

- Invite team members and collaborators
- Create a new merge request
- Automatically close issues from merge requests
- Enable merge request approvals
- Automatically merge when pipeline succeeds

Test and Deploy

Use the built-in continuous integration in GitLab.

- Get started with GitLab CI/CD
- Analyze your code for known vulnerabilities with Static Application Security Testing(SAST)
- Deploy to Kubernetes, Amazon EC2, or Amazon ECS using Auto Deploy
- Use pull-based deployments for improved Kubernetes management
- Set up protected environments

Editing this README

When you're ready to make this README your own, just edit this file and use the handy template below (or feel free to structure it however you want - this is just a starting point!). Thank you to makeareadme.com for this template.

Suggestions for a good README

Every project is different, so consider which of these sections apply to yours. The sections used in the template are suggestions for most open source projects. Also keep in mind that while a README can be too long and detailed, too long is better than too short. If you think your README is too long, consider utilizing another form of documentation rather than cutting out information.

Name

Choose a self-explaining name for your project.

Description

Let people know what your project can do specifically. Provide context and add a link to any reference visitors might be unfamiliar with. A list of Features or a Background subsection can also be added here. If there are alternatives to your project, this is a good place to list differentiating factors.

Badges

On some READMEs, you may see small images that convey metadata, such as whether or not all the tests are passing for the project. You can use Shields to add some to your README. Many services also have instructions for adding a badge.

Visuals

Depending on what you are making, it can be a good idea to include screenshots or even a video (you'll frequently see GIFs rather than actual videos). Tools like ttygif can help, but check out Asciinema for a more sophisticated method.

Installation

Within a particular ecosystem, there may be a common way of installing things, such as using Yarn, NuGet, or Homebrew. However, consider the possibility that whoever is reading your README is a novice and would like more guidance. Listing specific steps helps remove ambiguity and gets people to using your project as quickly as possible. If it only runs in a specific context like a particular programming language version or operating system or has dependencies that have to be installed manually, also add a Requirements subsection.

Usage

Use examples liberally, and show the expected output if you can. It's helpful to have inline the smallest example of usage that you can demonstrate, while providing links to more sophisticated examples if they are too long to reasonably include in the README.

Support

Tell people where they can go to for help. It can be any combination of an issue tracker, a chat room, an email address, etc.

Roadmap

If you have ideas for releases in the future, it is a good idea to list them in the README.

Contributing

State if you are open to contributions and what your requirements are for accepting them.

For people who want to make changes to your project, it's helpful to have some documentation on how to get started. Perhaps there is a script that they should run or some environment variables that they need to set. Make these steps explicit. These instructions could also be useful to your future self.

You can also document commands to lint the code or run tests. These steps help to ensure high code quality and reduce the likelihood that the changes inadvertently break something. Having instructions for running tests is especially helpful if it requires external setup, such as starting a Selenium server for testing in a browser.

Authors and acknowledgment

Show your appreciation to those who have contributed to the project.

License

For open source projects, say how it is licensed.

Project status

If you have run out of energy or time for your project, put a note at the top of the README saying that development has slowed down or stopped completely. Someone may choose to fork your project or volunteer to step in as a maintainer or owner, allowing your project to keep going. You can also make an explicit request for maintainers.