

# Developer Information

## Table of contents

OneRoof Development Environment .....	1
Pixi .....	1
Just .....	2
Pre-commit .....	2
Docker .....	2
Quarto .....	2
Dorado .....	2
Nextflow organization .....	2
Nextflow configuration .....	2
Python development .....	2

## OneRoof Development Environment

Most oneroof development happens in the dev branch of this repo. To get started with doing so yourself, clone and switch to the dev branch like so:

```
git clone https://github.com/nrminor/onerroof.git && \  
cd oneroof && \  
git checkout dev
```

From there, you'll have a few tools to considering installing if you haven't already:

1. The Pixi package manager, available here: <https://pixi.sh/latest/>
2. The command runner Just, available here: [https://just.systems/man/en/chapter\\_4.html](https://just.systems/man/en/chapter_4.html)
3. Pre-commit, available here: <https://pre-commit.com/>
4. Docker to use the repo Dockerfile as a dev container, available here: <https://docs.docker.com/engine/install/>
5. Quarto to modify documentation, available here: <https://quarto.org/docs/get-started/>
6. An editor with language support for Nextflow (or Groovy as a fallback), Python, and Quarto (or Markdown as a fallback).

Most of this project was written in VSCode, as it's currently the only editor with plugins for Nextflow, Quarto, and Python linters.

### Pixi

Of these tools, Pixi is the only one that is essential, as it handles installing the pipeline's dependencies from various conda registries as well as the Python Package Index. That said, you could even get away with not using pixi by manually installing the packages in `pyproject.toml` in a conda environment, using `pip` when needed.

If you are sticking with Pixi, run `pixi shell` in the project root before you get started. This will create a terminal environment with everything you need to work on and run oneroof.

## Just

While Just isn't as important as Pixi, I would still recommend installing it because of the conveniences it offers. With Just, the repo `justfile` provides a switchboard of command shorthands, including:

- `just docs`, which runs a series of Quarto commands to render and bundle the repo docs (including this file) as well as construct an updated readme
- `just py`, which lints and formats all the repo's Python files.
- `just docker`, which builds and pushes a new version of the repo Docker Image.
- `just env`, which instantiates the Pixi environment.
- `just all`, which does everything.

Run `just` in the same directory as the repo `justfile` to list all available recipes, and check out the Just Programmer's Manual for more about Just.

## Pre-commit

If you run `pre-commit install` in the repo root directory, it will install the pre-commit hooks in our `.pre-commit-config.yaml`. These hooks make sure that formatting throughout the repo files are consistent before they can be committed. Again, not essential, but nice!

## Docker

If you don't want to futz with all of the above or with setting up Dorado locally, you can also use the Docker Hub image `nrminor/dorado-and-friends:v0.1.0` as a dev container. It should have everything the pipeline and its dev environment needs, though its use as a dev container has not yet been tested.

## Quarto

Quarto can be thought of as a renderer or compiler for documents written in supercharged Markdown. It can run code blocks, render to HTML, PDF, `reveal.js` presentations, websites, books, and dozens of other formats. As such, I use Quarto documents as the sort of "ur-format," and render it out to other formats as desired. To render the project readme or other documents, you will need Quarto installed. The Quarto config file, `_quarto.yml`, controls project level settings, including a post-render section telling it to regenerate the Github-markdown-formatted readme from `assets/index.qmd` (via the project's `just readme` recipe).

## Dorado

## Nextflow organization

## Nextflow configuration

## Python development