

GitLab Development Kit

Configure and manage a [GitLab](#) development environment.

Read on for installation instructions or skip to [doc/howto](#) for usage documentation.

Overview

GitLab Development Kit (GDK) provides a collection of scripts and other resources to install and manage a GitLab installation for development purposes. The source code of GitLab is spread over multiple repositories and it requires Ruby, Go, Postgres/MySQL, Redis and more to run. GDK helps you install and configure all these different components, and start/stop them when you work on GitLab.

Contributing to GitLab Development Kit

Contributions are welcome, see [CONTRIBUTING.md](#) for more details.

Getting started

The preferred way to use GitLab Development Kit is to install Ruby and dependencies on your 'native' OS. We strongly recommend the native install since it is much faster than a virtualized one. Due to heavy IO operations a virtualized installation will be much slower running the app and the tests.

To do a native install:

1. [Prepare your computer](#)
2. [Set-up GDK](#)

Or if you want to use [Vagrant] instead (e.g. need to do development from Windows), see [the instructions for our Vagrant with Virtualbox setup](#). If you want to use [Vagrant] with [Docker][docker engine] on Linux, see [the instructions for our Vagrant with Docker setup](#).

After installation [learn how to use GDK](#)

If you have an old installation [update your existing GDK installation](#)

Design goals

- Get the user started, do not try to take care of everything
- Run everything as your 'desktop' user on your development machine
- GitLab Development Kit itself does not run `sudo` commands
- It is OK to leave some things to the user (e.g. installing Ruby)

Differences with production

- gitlab-workhorse does not serve static files
- C compiler needed to run `bundle install` (not needed with Omnibus)
- GitLab can rewrite its program code and configuration data (read-only with Omnibus)
- 'Assets' (Javascript/CSS files) are generated on the fly (pre-compiled at build time with Omnibus)
- Gems (libraries) for development and functional testing get installed and loaded
- No unified configuration management for GitLab and gitlab-shell (handled by Omnibus)
- No privilege separation between Ruby, Postgres and Redis
- No easy upgrades
- Need to download and compile new gems ('bundle install') on each upgrade

License

The GitLab Development Kit is distributed under the MIT license, see the LICENSE file.