

Continuous Integration

Tiki [releases every 6 months](#). However, the next release should ideally be "releasable at any time". So we:

- We use Continuous Integration http://en.wikipedia.org/wiki/Continuous_integration [↗](#), in our case using GitLab CI
- Also, the actual process of [Releasing](#) should be very easy (goal: 30 minutes for the technical part).

GitLab CI

Most of the tests are now done via Gitlab CI , which is run on every commit.

See the pipeline status at <https://gitlab.com/tikiwiki/tiki/pipelines> [↗](#)

What do I need to know as a normal developer

The CI runs on every commit. This can take 10 minutes, so you want to run the main tests locally before you commit so you don't constantly break the build, and have to rework your merge request. See [TikiDevNewbie](#)

Running locally and improving the GitLab CI Pipeline as a developer

Why would you want to?

1. To fix a pipeline failure when the code is correct
2. To make the pipeline run faster/be less costly to run (external service quota, etc.)
3. To make parts of the pipeline easier for other developers to work with, or to adapt the pipeline to changes in the underlying tiki tools and scripts.

The pipeline is run from the [.gitlab-ci.yml](#) [↗](#) committed in the repository. This is the [syntax reference](#) [↗](#).

Traditionally, gitlab-ci.yml files are meant to be edited with the gitlab's [pipeline editor](#) [↗](#). While a very nice tool to test and validate some aspects of the logic of that file, iterating on the pipeline and running the actual tests requires a commit. Even on a branch or clone, it is extremely time consuming.

So to work on it your should learn to:

Use the gitlab-ci-local tool

[gitlab-ci-local](#) [↗](#), which allows you to run a specific job locally, mostly as it would on gitlab.

```
gitlab-ci-local check-bom-encoding
```

Keep the actual tests executable locally

The gitlab-ci.yml file should only be concerned with setting up reproducible test environments, not setup tests that depend on one-another, etc. So the actual tests should be in separate scripts called from gitlab-ci.yml. Also, while a developer should not be expected to have every possible dependency installed, the tests should run in their local environment if they are working on this part of the code. For example, manticore tests should be runnable from however manticore is locally installed, not just from docker.

TODO: Clean up everything below. Most of it is still useful (2023-04-03, but hasn't been updated since we moved to gitlab-ci and docker)

Quality at Speed” is the new norm in software development

To get closer to these goals, the ideal would be community Continuous Integration server.

There are [daily builds](#) but if something breaks, there is no alert system to report the issue.

Wishlist for Tiki community

Pre-Dogfood Server

For humans to be able to see yesterday's data with tomorrow's code

- [Pre-Dogfood Server](#)

Profiles demo server

- Having a test server with main [profiles](#) applied regularly, for testing/demo.

Continuous build

For the daily pre-release zip file to be as close as possible to the final one

- Run the existing [scripts we use at release time](#)
- Automatic commits: we could register a "tikiwiki" user at Sourceforge.
 - update of changelog.txt (maybe once a week?)

Continuous Testing Server

Machines testing code, according to a series of tests

- Run all tests
 - Check PHP & Smarty Syntax, etc.
 - Check that all JavaScript can be safely minified with [JSLint](#) [↗](#)
 - Detect closing ?> tags from [DevTips](#)
- Run the security tests regularly (monthly?) and report to Security Team about new potentially risky files
- Ideas from [How to improve the release process](#)
- `php doc/devtools/translate.php englishupdate lag=10 audit --email=me@example.org` to report if anyone breaks strings over the last 10 days
- etc

Notes






Database Compare

In order to run the script `doc/devtools/check_schema_upgrade.php` which will compare your current database with what it should be you need to install DBDiff with this command:

```
. php temp/composer.phar require "dbdiff/dbdiff:@dev"
```

Related links

- <https://trunkbaseddevelopment.com/> [↗](#)
- <https://continuousdelivery.com/> [↗](#)
- [PHPCI is a free and open source continuous integration tool specifically designed for PHP.](#) [↗](#)
 - <https://www.ohloh.net/p/phpci> [↗](#)

- <https://www.phptesting.org/hosted-phpci> 
- <http://erichogue.ca/2011/05/php/continuous-integration-in-php/> 
- <http://docs.codehaus.org/display/SONAR/PHP+Plugin> 
- https://github.com/changi67/Tiki_CodeSniffer 
- <https://www.ohloh.net/p/xinc-continuous-integration-for-php> 
- <http://confluence.public.thoughtworks.org/display/CC/CI+Feature+Matrix> 
- <https://github.com/wimg/PHPCompatibility> 
- <http://sourceforge.net/blog/how-to-use-webhooks-for-git-mercurial-and-svn-repositories/> 
- <https://medium.com/linagora-engineering/dockerized-ci-of-linagoras-james-team-b01bd34a2641> 
- <https://github.com/phpstan/phpstan> 
- https://github.com/squizlabs/PHP_CodeSniffer 

Alias

- Continuous Integration and Continuous Delivery
- CI/CD