Tiki Manager

Tiki Manager permits you to manage various instances of Tiki. You can install, upgrade, backup, clone, check the file integrity and do various other things. This can be done on the same server or a remote server. Most actions can be run unattended on a cron job. It can be used via the command line or a web interface.

Tiki Manager supercedes TRIM.

Coming to Tiki25: Tiki Manager Package

Requirements

- SQLite3 for data storage
- PHP 7.4+ with Command-line access (CLI). (Before 2023-07-04, it required PHP 7.1

It uses the Symfony Console Component.

    - It uses Composer for dependency management.
- Git
- Standard server tools like rsync, unzip, etc.
- It's been developed on GNU/Linux, and designed to work anywhere PHP, SQLite and Git are available (Windows, MacOS, *BSD, etc.)

You can use Check to verify that your server can run Tiki or Tiki Manager

Installation

These are general instructions for Linux. You may need to adapt for your operating system. See also Additional configuration below.

At a high level:

1. Use Server Check to review and get all the server dependencies
2. Get code from Git in a non-web accessible directory: https://gitlab.com/tikiwiki/tiki-manager.git
3. Run Tiki Manager with `php tiki-manager.php` to complete set up. It will instruct you how to do so (ex.: run Composer to get PHP dependencies)

Example commands (that worked for me on Debian 9)

Install Tiki Manager

Navigate to your working directory (__Do not install in a web accessible directory__) and run the following git command $ git clone --depth 1 --branch master https://gitlab.com/tikiwiki/tiki-manager.git # Navigate to the tiki-manager directory with "cd tiki-manager" # Then run the following. On first run, it will finish the setup (Fetch Composer and get PHP dependencies, Create the SQLite database, etc.) $ php tiki-manager.php manager:info

Documentation

Here is a brief explanation of the basic commands that can be used. All commands follow a wizard pattern.

For more help, you can add --help to any command as in this example

php tiki-manager.php instance:create --help

instance

INSTANCE:CREATE

Adds an instance to be managed by Tiki Manager, which can mean either:

- create a fresh Tiki instance (including a database) fetching files from Git (and optionally SVN)
- or detect an existing Tiki installed via Git or Subversion (SVN) and "adopt" it.
- You may need to convert your Tiki instance from FTP to Git.

Params

Params	Type	Description
blank	option	Blank Instance
type	option	Instance connection type

php tiki-manager.php instance:create

Non interactive command:

php tiki-manager.php instance:create --type=local --url=http://manager.tiki.org --name=manager.tiki.org --email=manager@example.org --webroot=/www/manager --tempdir=/tmp/trim_temp --branch=branches/19.x --backup-user=www-data --backup-group=www-data --backup-permission=750 --db-host=localhost --db-user=root --db-pass=secret --db-prefix=manager

INSTANCE:LIST

php tiki-manager.php instance:list

Params	Type	Description

json	option	Instance list in a parsable format

INSTANCE:ACCESS

Opens a shell to the remote host.
Since Tiki Manager manages all your connections, using this command simply avoids needing to remember passwords.

Params

Params	Type	Description
instances	option	List of instance IDs or Names to be checked, separated by comma (,)

php tiki-manager.php instance:access

INSTANCE:BACKUP

Tiki Manager performs a complete backup of the Tiki instance.
The backup includes all files (using rsync for efficient bandwidth management) on remote host (including user files and files stored out of the web root) and a dump of the database. Each backup is archived in the backup/archive folder as a .tar.bz2 file, where Tiki Manager is installed.
The backup file contains:

- manifest.txt: Indicates where folders were located on the remote host
- database_dump.sql: Self-explanatory
- [a-f0-9]{32}: Folders named using a hash. Content of the folders on remote host. The manifest.txt file lists these hashes.

Params

Params	Type	Description

php tiki-manager.php instance:backup

INSTANCE:BLANK

Like instance:create but it doesn't actually add a Tiki. It just creates an instance that can be used to restore or clone another instance.

php tiki-manager.php instance:blank

php tiki-manager.php instance:create blank

INSTANCE:CHECKOUT

CheckoutCommand: allow checking out a specific Git branch and revision for main Tiki codebase or local checkouts of themes, for example. Original commit.

Params

Params	Type	Description
instance	accepts one instance ID	
folder	either specify 'tiki' to update main Tiki codebase or a subfolder (e.g. themes/XYZ) - folder might exist or not exist. If it exists - it must be a Git checkout. In this case, the specified branch/revision will be checked out. If it doesn't exist, you should also specify a URL and it will be cloned into that folder.	

php tiki-manager.php instance:checkout

This useful for A theme managed in Git

INSTANCE:CLONE

Makes another identical copy of Tiki. This is basically a combination of make backup and make restore in one operation. The destination instance can be blank or another configured and managed Tiki Instance. Be very careful, when using existing another Tiki Instance, as the data here will be wiped and replaced by the data from the source instance.

As two Tiki instances with the same settings (ex.: both point to the same Elasticsearch index can cause issues), recommended reading: Divergent Preferences in Staging Development Production

The following are excluded from the cloning process:

- db/local.php (the database access info needs to point to another database)
- the .ini.php file (if any) used for System Configuration

Params

Params	Type	Description

php tiki-manager.php instance:clone

INSTANCE:CLONEANDREDACT

This command allows you to clone an instance and redact the clone, make a clone of an instance and redact it.

Params

Params	Type	Description
instances	option	List of instance IDs to be redacted, separated by comma (,).

php tiki-manager.php instance:cloneandredact

INSTANCE:CLONEANDUPGRADE

Like instance:clone but with an extra upgrade operation.

As two Tiki instances with the same settings (ex.: both point to the same Elasticsearch index can cause issues), recommended reading: Divergent Preferences in Staging Development Production

The following are excluded from the cloning process:

- db/local.php (the database access info needs to point to another database)
- the .ini.php file (if any) used for System Configuration

Params

Params	Type
mode	parameter
check	option
skip-reindex	option

Description

Check if is a clone or upgrade.

Check files checksum. Only used in mode upgrade.

Skip rebuilding index step.

skip-cache-warmup	option
live-reindex	option
direct	option
keep-backup	option
use-last-backup	option
source	option
target	option

Skip generating cache step.

Set instance maintenance off and after perform index rebuild.

Use rsync to copy files between local instances.

Source instance backup is not deleted before the process finished.

Use source instance last created backup.db-host

op tio n

Target instance database host

Use a certain source instance, ID or name.

Use a certain target instance, ID or name.

branch	option
db-host	option
db-user	option
db-pass	option
db-prefix	option
db-name	option
stash	option

The branch (version) used for the upgrade i.e the branch you want to upgrade to.

Target instance database host.

Target instance database user.

Target instance database password.

Target instance database prefix (creates an user and database).

Target instance database name (when db-prefix is not used).

Saves your local modifications, and try to apply after update/upgrade.

timeout	option

Modify the default command execution timeout from 3600 seconds to a custom value

```
php tiki-manager.php instance:cloneandupgrade
```

INSTANCE:CONSOLE

Allow to run any Console command from Tiki.

```
php tiki-manager.php instance:console
```

Example to clear cache (it will request the instance ID):

```
php tiki-manager.php instance:console --command="cache:clear all"
```

Example with chosen instance:

php tiki-manager.php instance:console --instances=5 --command="cache:clear all"

INSTANCE:COPYSSHKEY

Copy the SSH key to the remote instance. This is used as part of other commands but can be used as standalone as well - copies the Tiki Manager SSH key to the remote machine by asking you for the password to connect to, so Tiki Manager can authenticate via its public/private key pair and skip asking for passwords on all subsequent operations.

php tiki-manager.php instance:copysshkey

INSTANCE:DELETE

Delete the instance via the command line (you could also do via the web interface). This does NOT delete your Tiki. It just deletes the instance connection to it.

php tiki-manager.php instance:delete

INSTANCE:DETECT

Detect Tiki branch or tag, and PHP version. For debugging purposes. Also useful if you manually proceeded to svn switch and Tiki Manager needs to update its internal database about a Tiki instance.

php tiki-manager.php instance:detect

INSTANCE:EDIT

Permits to modify an instance.

---

php tiki-manager.php instance:edit

INSTANCE:FIXPERMISSIONS

Run setup.sh on the remote host using automated parameters. It should work in most cases. If the command proposed my setup.sh without parameters or super user rights are required, you should connect to the remote host manually using `instance:access`.

---

php tiki-manager.php instance:fixpermissions

INSTANCE:IMPORT

Import an instance to the instances list, if detects a Tiki instance, and itâ€™s not yet managed by Tiki Manager.

Params

Params	Type	Description
type	option	Instance connection type
host	option	Remote host name
port	option	Remote port number
user	option	Remote User
pass	option	Remote password
url	option	Instance web URL
name	option	Instance name
email	option	Instance contact email
webroot	option	Instance web root

tempdir option Instance temporary directory

---

php tiki-manager.php instance:import

Non interactive command:

---

php tiki-manager.php instance:import --type=local --url=http://manager.tiki.org --name=manager.tiki.org --email=manager@example.org --webroot=/www/manager --tempdir=/tmp/trim_temp

INSTANCE:MAINTENANCE

Put instances under maintenance or live mode.

---

php tiki-manager.php instance:maintenance

instance:patch

https://gitlab.com/tikiwiki/tiki-manager/-/merge_requests/271

INSTANCE:PATCH:LIST

php tiki-manager.php instance:patch:list

INSTANCE:PATCH:DELETE

php tiki-manager.php instance:patch:delete

INSTANCE:PATCH:APPLY

php tiki-manager.php instance:patch:apply

--instances=INSTANCES	List of instance IDs to apply the patch on, separated by comma (,)
--package=PACKAGE	Composer package name or 'tiki' if it is a MR to the Tiki codebase
--url=URL	Url of the patch, e.g. https://gitlab.com/tikiwiki/tiki/-/merge_requests/1374.patch

Both GitLab and GitHub support patch and diff outputs of Merge/Pull requests:

- https://patch-diff.githubusercontent.com/raw/jasonmunro/cypht/pull/548.patch
- https://patch-diff.githubusercontent.com/raw/fintech-systems/virtualmin-api/pull/19.diff
- https://gitlab.com/tikiwiki/tiki/-/merge_requests/1245.patch

Example:

```
php tiki-manager.php instance:patch:apply https://gitlab.com/tikiwiki/tiki/-/merge_requests/1245.patch
```

INSTANCE:PROFILE:APPLY

Apply a profile to an instance.

```
php tiki-manager.php instance:profile:apply
```

INSTANCE:RESTORE

Restore on a blank installation. Ref: instance:blank If you have data files that are not stored in the database, you should use the ideal scenario for data file storage and relative paths.

```
php tiki-manager.php instance:restore
```

INSTANCE:REVERT

Revert a particular instance working dir to its original branch state (aka git hard reset). Thus, it removes any applied patches

```
php tiki-manager.php instance:revert
```

INSTANCE:SETUP-SCHEDULER-CRON

This command allows you to enable the cron to run the schedulers

```
php tiki-manager.php instance:setup-scheduler-cron
```

Itcan be used interactively or without interaction. Ex.:

```
php tiki-manager.php instance:setup-scheduler-cron -i 1 --time="*/10 * * * *"
```

Possible options:

--update // Update existing cronjob --enable // Uncomment the # from the cronjob line --disable // Comment with # the cronjob line --check // Just check if there is cronjob configured

The command will not work for instances running Windows.

INSTANCE:STATS

Extract stats (KPIs) from selected instances as csv. Optionally that information can be pushed to another Tiki instance.

Params

Params	Type	Description

```
# Display all instance stats php tiki-manager.php instance:stats
```

```
# Save stats to a csv file php tiki-manager.php instance:stats --instances=all --file=instance_stats.csv
```

```
# Upload instances stats to another tiki instance # Using TIKI_ROOT will match instance webroot php tiki-manager.php instance:stats --instances=all --exclude=1 --file=TIKI_ROOT/temp/instance_stats.csv --push-to=1
```

INSTANCE:UPDATE

- Does a dry-run first, and aborts on any conflicts
- Updates to latest code in that branch (or trunk) using svn up and thus merging any changes
- And all operations should be done after updating the code
    - Updates the file hashes accordingly. The hash verification/update may prompt with some files containing conflicts if modifications were made on the instance.
    - Performs the database update.
    - Runs setup and Composer,
    - Clears cache

- Rebuilds search index

During the update process, the instance is disabled using a .htaccess file (previous one is preserved), making the site unavailable until the update is completed.

Params

Params	Type	Description
mode	parameter	Check if is an auto or switch mode
instances	option	List of instance IDs or Names to be checked, separated by comma (,)

php tiki-manager.php instance:update

INSTANCE:UPGRADE

Similar to update. Requests for the branch to switch to over the update. You will have several choices of branches. Please note that you should **NOT** downgrade as Tiki doesn't support a downgrade database script. An upgrade is a one-way street! You should make a backup before you upgrade so you can return to this version if issues arise.

If you choose to do a manual upgrade (with svn switch).

Params

Params	Type	Description

php tiki-manager.php instance:upgrade

INSTANCE:VERIFY

This is equivalent to a secdb check, except that it's safer because it's made from a trusted host and will take your custom modifications on host into consideration. Upon first run, the check will ask where it should fetch the hashes from.

Params

Params	Type	Description
instances	option	List of instance IDs or Names to be checked, separated by comma (,)

php tiki-manager.php instance:verify

INSTANCE:WATCH

This command perform the Hash check (instance:verify). The script will prompt for a contact email address to notify with the log in the case of a suspicious file change (which could be an intrusion or someone who legitimately changed a file on the server).

Params

Params	Type	Description
email	option	Email address to contact.
exclude	option	List of instance IDs to be excluded, separated by comma (,)

php tiki-manager.php instance:watch

backups

BACKUPS:SETUP

Same as manager:setup-backups.

php tiki-manager.php backups:setup

BACKUPS:DELETE

Delete Tiki Manager backups folder and contents. This folder contains the backups of instances managed by Tiki Manager.

php tiki-manager.php backups:delete

cache

CACHE:CLEAR

Delete Tiki Manager cache folder. Useful for development.

---

php tiki-manager.php cache:clear

database

DATABASE:DELETE

- Delete Tiki Manager database. Useful for development.

```
php tiki-manager.php database:delete
```

DATABASE:VIEW

- View Tiki Manager database. For debug purposes and useful for development.

```
php tiki-manager.php database:view
```

logs

LOGS:CLEAR

Clear Tiki Manager logs folder.

---

php tiki-manager.php logs:clear

manager

MANAGER:INFO

- Display running OS, PHP version and binary used by Tiki Manager

php tiki-manager.php manager:info

MANAGER:CHECK

- Check OS requirements to execute Tiki Manager

php tiki-manager.php manager:check

MANAGER:REPORT

- Reports, and send reports to a Tiki instance using Data Channels.

php tiki-manager.php manager:report

MANAGER:SETUP-CLONE

- Setup a cronjob to perform instance clone.

```
php tiki-manager.php manager:setup-clone
```

MANAGER:TEST-SEND-EMAIL

- Test send email.

```
php tiki-manager.php manager:test-send-email
```

TIKI:VERSIONS

- Shows all versions of Tiki.

```
php tiki-manager.php tiki:versions
```

MANAGER:RESET

- Delete Tiki Manager backup, cache, and log files. Useful for development.

php tiki-manager.php manager:reset

MANAGER:SETUP-BACKUPS

Set-up a cronjob to perform automatic instance(s) backups (instance:backup) every day at a specific time. The script will prompt for the time which the cron should run at and the instances that will be ignored by these automatic backups (all instances are selected by default).

- Automatic backups cronjob should not run at the same time as the "manager:setup-update" cron job command. Make sure you pick different run times.

Params

Params	ParamsType	Description
time	option	Time to trigger the instance(s) backup using the format :

php tiki-manager.php manager:setup-backups

MANAGER:SETUP-UPDATE

Set-up a cron job to perform automatic instance(s) update (instance:update) every day at a specific time. The script will prompt for the time which the cron should run at and the instances that will be affected by this automatic update.

Params

Params	ParamsType	Description
time	option	Time to trigger the instance(s) update using the format :

php tiki-manager.php manager:setup-update

MANAGER:SETUP-WATCH

Set-up a cron job on the Tiki Manager master to perform the Hash check (instance:verify) automatically every day. The script will prompt for a contact email address to notify with the log in the case of a suspicious file change (which could be an intrusion or someone who legitimately changed a file on the server) and the time at which the script should run.

Params

Params	Type	Description
email	option	Email address to contact.
time	option	The time update should run.
exclude	option	List of instance IDs to be excluded, separated by comma (,)

php tiki-manager.php manager:setup-watch

MANAGER:UPDATE

Update Tiki Manager to the latest version available. This updates the PHP code from Git, and updates the Composer dependencies.

Params

Params	Short	Type	Description
--check	-c	option	Only checks if there is new version available for update.
--yes	-y	option	Say yes to update

php tiki-manager.php manager:update --check

---

php tiki-manager.php manager:update -y

For installations (without Phar or Git), to enable update, add the following file `.version` to the Tiki-Manager root.

{"version":"bf63ffa", "date":"2020-05-03T23:32:58+01:00"}

Additional configuration

To easily configure the Tiki Manager application, copy `.env.dist` file to `.env` and insert your configurations for the uncommented (#) entries.

VERSION CONTROLÂ SYSTEM

Tiki Manager by default uses git and public repository. If you want to use SVN (but you shouldn't as SVN is no longer supported) as your default vcs or another repository please use the following lines in your .env file.

DEFAULT_VCS=svn GIT_TIKIWIKI_URI= SVN_TIKIWIKI_URI=

BEHIND PROXY OR WITHOUT INTERNETÂ CONNECTION

Tiki Manager is able to use Tiki's distributed version packages as an alternative when there is no connection to external servers like GitLab or SourceForge.

Setting the default VCS to src, Tiki Manager will use existing packages in the data/tiki_src folder (default).

```
DEFAULT_VCS=src
```

Download the distributed Tiki packages, from https://sourceforge.net/projects/tikiwiki/files/, and save them into data/tiki_src folder.

EMAIL SETTINGS

To configure Tiki Manager email sender address add the following line to your .env file.

```
FROM_EMAIL_ADDRESS=
```

CONFIGURE SMTPÂ SERVER

By default Tiki Manager uses sendmail to send email notifications. If you intend to use SMTP instead add the following lines to your .env file.

```
```

SMTP_HOST= SMTP_PORT= SMTP_USER=(optional if authentication is required) SMTP_PASS=(optional if authentication is required)

WEB MANAGERÂ SETTINGS

If you want to setup a default folder to install your web manager or apache user:group are different than apache:apache you can add the following settings to your .env file.

```
WWW_PATH= WWW_USER= WWW_GROUP=
```

To change the maximum number of failed login attempts on Web Manager, add the following setting to your .env file.

```
MAX_FAILED_LOGIN_ATTEMPTS=
```

Timeouts during long runningÂ operations

During long operations (like clone or clone and upgrade) you may receive an HTTP error code 503 with a message "Service Unavailable", you can increase Apache's proxy timeout to a more suitable value.

To do that at the Virtual Host level (so it's only enabled for tiki manager) you need to do the following:

Add the Apache directive `ProxyTimeout` to Tiki Managers VirtualHost configuration file.
Example: `ProxyTimeout 300` will set the proxy timeout to 5 minutes

Hooks

Tiki Manager since 2024-02-08 supports hook system, to execute scripts (shell scripts) before or after the command.
This can be useful to execute a backup encryption or move backups to different folders or even trigger notifications.

Every command can have a pre and post scripts.

See more context in the feature request: https://gitlab.com/tikiwiki/tiki-manager/-/issues/22

Adding aÂ hook

To add a script to be executed before or after the command, you need to add the script file into the folder following this structure:
`/hooks///.sh`

Command is the name of the command being executed, replace `:` with `-`

In this example lets create a post execution hook for instance:create command
The path will be `/hooks/instance-create/post/notify.sh`

Hook variables

Each command can register different variables. Commands that do interact with instances will add the instances properties.

INSTANCE_IDS # A list with the instances ids, that are available in the environment variables (comma separated)
INSTANCE_TYPE_ INSTANCE_VCS_TYPE_ INSTANCE_NAME_ INSTANCE_WEBROOT_ INSTANCE_WEBURL_ INSTANCE_TEMPDIR_
INSTANCE_PHPEXEC_ INSTANCE_PHPVERSION_ INSTANCE_BACKUP_USER_ INSTANCE_BACKUP_GROUP_
INSTANCE_BACKUP_PERM_ INSTANCE_BRANCH_ INSTANCE_LAST_ACTION_ INSTANCE_LAST_ACTION_DATE_

ADDITIONAL VARS PERÂ COMMAND

instance:patch:apply

INSTANCE_BACKUP_FILE_ PATCH_PACKAGE PATCH_URL

instance:backup

INSTANCE_BACKUP_FILE_

instance:clone

SOURCE_INSTANCE_ID SOURCE_INSTANCE_BACKUP

instance:maintenance

INSTANCE_MAINTENANCE_STATUS_

instance:profile:apply

INSTANCE_PROFILE

instance:setup-scheduler-cron

INSTANCE_JOB_ENABLED_ INSTANCE_JOB_TIME_ INSTANCE_JOB_COMMAND_

instance:stats

INSTANCE_STATS_

instance:upgrade

INSTANCE_PREVIOUS_BRANCH_

instance:watch

INSTANCE_REVISION_ INSTANCE_REVISION_ERROR_

Other Notes

Tiki Manager vsÂ MultiTiki

Using the `instance:console` command in Tiki Manager you can access the multitiki commands in the remote instance, like this. Assuming we are using instance #42 and the virtual domain is example.com, and each branch is in the same directory (i.e. `tiki/branches/20.x` and `tiki/branches/21.x` in this instance.

DETAILS

[+]

To list the domains in the 20.x tiki use:

$ php tiki-manager.php instance:console -i42 -c'multitiki:list ../20.x' Calling command in tiki-dev/21.x Result: example.com staging.example.com other.example.com

And then to preview moving one from there to the 21.x instance use:

$ php tiki-manager.php instance:console -i42 -c'multitiki:move example.com ../20.x' Calling command in tiki-dev/21.x Result: Will move: /home/auser/tiki/branches/20.x/db/example.com to /home/auser/tiki/branches/21.x/db/example.com Will move: /home/auser/tiki/branches/20.x/dump/example.com to /home/auser/tiki/branches/21.x/dump/example.com Will move: /home/auser/tiki/branches/20.x/img/wiki/example.com to /home/auser/tiki/branches/21.x/img/wiki/example.com Will move: /home/auser/tiki/branches/20.x/img/wiki_up/example.com to /home/auser/tiki/branches/21.x/img/wiki_up/example.com Will move: /home/auser/tiki/branches/20.x/img/trackers/example.com to /home/auser/tiki/branches/21.x/img/trackers/example.com

Will move: /home/auser/tiki/branches/20.x/modules/cache/example.com to /home/auser/tiki/branches/21.x/modules/cache/example.com Will move: /home/auser/tiki/branches/20.x/temp/example.com to /home/auser/tiki/branches/21.x/temp/example.com Will move: /home/auser/tiki/branches/20.x/temp/cache/example.com to /home/auser/tiki/branches/21.x/temp/cache/example.com Will move: /home/auser/tiki/branches/20.x/temp/public/example.com to /home/auser/tiki/branches/21.x/temp/public/example.com Will move: /home/auser/tiki/branches/20.x/templates/example.com to /home/auser/tiki/branches/21.x/templates/example.com Will move: /home/auser/tiki/branches/20.x/themes/example.com to /home/auser/tiki/branches/21.x/themes/example.com Will move: /home/auser/tiki/branches/20.x/whelp/example.com to /home/auser/tiki/branches/21.x/whelp/example.com Will move: /home/auser/tiki/branches/20.x/mods/example.com to /home/auser/tiki/branches/21.x/mods/example.com Will move: /home/auser/tiki/branches/20.x/files/example.com to /home/auser/tiki/branches/21.x/files/example.com Will move: /home/auser/tiki/branches/20.x/tiki_tests/tests/example.com to /home/auser/tiki/branches/21.x/tiki_tests/tests/example.com Use --confirm to perform moves

More info here: MultiTiki

Troubleshooting

If you have weird errors and/or an inability to update, it could be

- that Tiki Manager dependencies are messed up. Just delete the vendor/ directory and try again (Tiki Manager will re-download them)
- you have modified code. You can check with `git status` and `git diff`

Source code

The source code is managed here:
https://gitlab.com/tikiwiki/tiki-manager/

Roadmap

- Manager

Related

- https://wikisuite.org/Virtualmin-Tiki-Manager

alias

- Tiki Manager