How to release

Before this step: How to release Preparation

1.1. Requirements

1. Unix shell environment, Linux or OSX suitable.
2. Git
3. PHP 7.x or 8.x CLI with MySQL, iconv, DOM and SimpleXML extensions. **Note about php versions:** Use PHP 7.x for Tiki 25.x releases and lower, and PHP 8.x for Tiki 26.x releases and upper.
4. PHP configuration must allow running external programs (like shell scripts)
5. Starting from Tiki 27.x and later versions release, Node.js >=18.0.0 <22 and NPM >=9.0.0 <11 are required. Node and NPM are required for the JavasScript build system(for js and css compilation).
6. 7za or 7z needed for creating 7zip archives. (may be installed in OSX through homebrew by typing brew **install p7zip**)
7. Plenty of disk space and memory - at least 64M of memory for php (in php.ini for php-cli)
8. The main packaging script is at doc/devtools/release.php but releasing a Tiki version is a lot more than just running the packaging script

1.2. Release Steps

1.2.1. Checkout Tiki

For this step we have 2 options, so you can use one of your choice but the second is recommended.
-1st option: checkout the correct branch, i.e the branch you want to create the release for. For example, if you want to create the 26.0 release, do

```
git checkout 26.x
```

- 2nd option: Clone the branch you need to create the release for in a separate folder to separate it with your development instance by doing the following. Not that the bellow command is an example, so replace the branch value with that you want to work on, the reference repository by the path of your development instance to gain time when cloning and the target folder by that you want. Also remember to ensure that the reference repository is up to date and contains the required objects for successful cloning.

```
git clone --branch=26.x --shared --reference=./tiki https://gitlab.com/tikiwiki/tiki.git tiki26release
```

1.2.2. Setup.sh

When releasing version greater than 9.x there is the Composer to handle external libraries dependencies.

Run:

```
sh ./setup.sh
```

And choose 'b' option to install all necessary setup and exit (composer, npm, etc) instead of 'f', which is not needed for release.

1.2.3. Update lib/setup/twversion.class.php before the release

Look for section with "Release Managers should update this array before release." of twversion.class.php (of your relevant branch)

- increment the version number and the star name in the constructor
- temporarily, remove the vcs from the branch number. E.g.

  ```
  function TWVersion() { // Set the development branch. Valid are: // stable : Represents stable releases. // unstable : Represents candidate and test/development releases. // trunk : Represents next generation development version. $this->branch = 'stable'; // Set everything else, including defaults. $this->version = '22.1vcs'; // needs to have no spaces for releases $this->star = 'Corona Borealis';
  ```

  Since we where to release 22.1, we need to remove the "vcs" from "22.1vcs", so this part will stay as:

  ```
  // Set everything else, including defaults. $this->version = '22.1'; // needs to have no spaces for releases
  ```

- update list of valid releases in `tikiVersions()`
  - Make sure you add all Tiki versions (not just the one you are doing now). Ex.: when 5.0 is released, 4.2 will probably exist, as this was added to branches/4.x but not merged by script.
  - change the version branch to "unstable", "stable", or "trunk" as explained in that file
- Update the star name in the list in the function `tikiStars()`
- Commit your changes with this commit message (change \$VERSION by the version of the release):

  ```
  [REL] Preparing \$VERSION release
  ```

- Then push your commit to https://github.com/tikiwiki/tiki.git in the corresponding branch (the branch you are doing the release for).

1.2.3.1. # ~~Remove the previous secdb file~~

- The release script will do this since Tiki 18

1.2.4. Run script

Roberto Kirschbaum wrote:

> *Big thanks to Xorti for the --use-git option in doc/devtools/release.php. For the first time in History we release Tiki using Git! Thanks also to Jonny for testing a tarball in the process of release.*

The main packaging script is at doc/devtools/release.php

Go to the newly created directory and run

Display Basic Help

```
php doc/devtools/release.php --help
```

to get the command help and see different options that can be used.

Run the release command:

Display Basic Help

```
php doc/devtools/release.php --use-git 26.1
```

(replace 26.1 with the corresponding release version)

If the previous command fails at the secdb file creation step, you may need to run the release command providing your MySQL credentials for the script to successfully create the secdb file for you. So as you are releasing from your own computer instead of the release server, you may need to use this command which provides the MySQL credentials:

```
php -d mysqli.default_host=HOST -d mysqli.default_user=USER -d mysqli.default_pw=PASS
doc/devtools/release.php --use-git 26.1
```

(replace 12.1 beta with the corresponding release version)

1.2.5. ~~Create and test pre-release packages~~

You can skip this step and directly go to 1.2.6

- This is done by executing the release script with the release version as argument, using the format major.minor preXYZ (XYZ can be RC1, RC2, alpha, beta1, beta2, etc.)
- Use the --help option of the release script for advanced help on options (like using a web proxy)

```
php7.1 doc/devtools/release.php 19.0 preRC1
```

Note: Use the command for the version of PHP which is the minimum spec for that branch. In this case we need PHP 7.1 for Tiki 19.x, 5.6 for 18.x and 15.x and 5.3 for 12.x, for instance

The release script has an interactive mode, enabled by default, that will ask you if you really want to do each step. It also asks for a confirmation before committing any changes. You can have a look at them with another shell by using this command:

Check for changes made by the release script before validating commits

```
svn --ignore-externals status svn --ignore-externals diff
```

1.2.5.1. Check the generated README file

- Check links/content in the generated README file

1.2.6. Launch the release script again

After testing, launch the release script again, now without "pre":

```
php7.1 doc/devtools/release.php 19.0 RC1
```

1.2.6.1. Troubleshooting

1.2.6.1.1. Mysql credentials

In step 11 (or 8 ?) of the release script ("� **Generate SecDB file 'db/tiki-secdb_12.1_mysql.sql'?**"), you might see this type of error:

```
SecDB step failed because some filenames need escaping but no MySQL connection has been found. Try this command line instead (replace HOST, USER and PASS by a valid MySQL host, user and password) : /usr/bin/php -d mysql.default_host=HOST -d mysql.default_user=USER -d mysql.default_password=PASS doc/devtools/release.php 12.1 beta
```

If so, provide your own HOST, USER and PASS for your MySQL from the machine where you are running the release script, and you should be fine, then.

1.2.7. Test the produced "tarballs" and share the testing

Test on your server. Ideally, you have 1-2 other people trying on different servers. In case of a major version (x.0), you need at least 3 installations from 3 different people

- Make sure to do a file integrity check with `tiki-admin_security.php?check_files`
- There are 4 featured profiles in the installer and these are great tests.

1.2.8. Upload tarballs to Sourceforge

When the "tarballs" are tested, the next step is to upload them to sourceforge. For that we have 3 options:
1 . Using sourceforge web interface
Follow the steps to upload on SourceForge:

- http://sourceforge.net/p/forge/documentation/Files/

2. Using scp
To upload the 'tarballs', copy-paste and execute the following line (and change '$SF_LOGIN' by your SF.net login and $VERSION with the version, which may look like "7.0.rc1"):

```
cd ~/tikipack/$VERSION scp *bz2 *gz *zip *7z $SF_LOGIN@frs.sourceforge.net:/home/pfs/project/t/ti/tikiwiki/$RELEASEFOLDER$
```

Real example

```
#since 15.0 : scp *bz2 *gz *zip *7z username,tikiwiki@frs.sourceforge.net:/home/pfs/project/t/ti/tikiwiki/Tiki_XX.x_Starname/XX.Y
```

3. Using sftp
Alternatively, you can do it with sftp

```
User jsmith seeks to put tiki-12.1.* .7z .zip .tgz .bz2 to the 12.1 directory of his project, tikiwiki: $ sftp jsmith@frs.sourceforge.net Connecting to frs.sourceforge.net... jsmith,fooproject@frs.sourceforge.net's password: sftp> cd /home/frs/project/tikiwiki/Tiki_12.x_Altair/ sftp> mkdir 12.1 sftp> cd 12.1 sftp> mput tiki*.* Uploading tiki-12.1.7z to /home/pfs/project/tikiwiki/Tiki_12.x_Altair/12.1/tiki-12.1.7z tiki-12.1.7z 100% 27MB 2.3MB/s 00:12 Uploading tiki-12.1.tar.bz2 to /home/pfs/project/tikiwiki/Tiki_12.x_Altair/12.1/tiki-12.1.tar.bz2 tiki-12.1.tar.bz2 100% 38MB 2.9MB/s 00:13 Uploading tiki-12.1.tar.gz to /home/pfs/project/tikiwiki/Tiki_12.x_Altair/12.1/tiki-12.1.tar.gz tiki-12.1.tar.gz 100% 43MB 2.7MB/s 00:16 Uploading tiki-12.1.zip to /home/pfs/project/tikiwiki/Tiki_12.x_Altair/12.1/tiki-12.1.zip tiki-12.1.zip 100% 52MB 2.9MB/s 00:18 sftp>
```

You need to have "release technician" status on SourceForge. Ask an Admin if you don't have it.

1.2.8.1. Update default download version on Sourceforge:

1. Go to the Files page while logged in to access the File Manager
2. From the File Manager navigate to the directory that contains the file
3. Select the "i" icon for the file you wish to set as the default download
4. On the bottom right check the boxes for the operating systems which you wish the file to be the default download for.
5. Select save

1.2.9. Update lib/setup/twversion.class.php after the release

- change $this->version = '**23.1**'; to '**23.2vcs**';

Needs to have no spaces and use the format X.YabcZ see tikiVersions fn below for examples

You can add a commit message like this one:

```
[REL]Closing release 12.1beta
```

1.2.10. Update show2.t.o info

Please coordinate with Marc to achieve this step, you need access on dev.t.o and show.t.o server.

There are still a few extra steps needed, like updating the dev.t.o tracker5 & show2.t.o server to allow users to reproduce bugs in tiki instances using the new version, etc.

1.2.10.1. checkout that version in the local cache folder of tiki in show2.t.o

In order to have a new tiki version available in show*.t.o, a new folder and vcs checkout need to be added manually at show2.tiki.org under:

`/usr/local/src/tiki`

in show2.tiki.org

It's using subversion.

1.2.10.2. tell tim to manage also that branch to update tiki from that version also

And some script need to have this new version added as a name of branches to work with. Editing this file:

`/usr/local/sbin/tim-common`

and add the new version (23.x in this example) at this line starting with "BRANCHES=":

```
# SVN uppable stuff BRANCHES="trunk 23.x 22.x 21.x 20.x 19.x 18.x"
```

1.2.10.3. make a tar.gz file from the new branch

This way, making new instances will not need to get the whole tiki file tree from subversion, but they will start by copying and unpacking a pre-fetched tiki version for that branch, and just updating that file tree through subversion. If you don't make tar.gz file, new show instance creation will fail.

And you need to create that tar.gz without the prefix of the parent folder where the new branch is. You can do so with thee type of commands:

```
cd /usr/local/src/tiki/ cd 23.x ln -s _htaccess .htaccess tar -czvf ../23.x.tar.gz ./ -R
```

1.2.10.4. update the field in tracker5 at dev.t.o to allow users to choose that version

Edit this field "Demonstrate Bug (Tiki 19+)":
https://dev.tiki.org/tiki-tracker-edit_field?trackerId=5&fieldId=236&modal=1

Add the new version (23.x in this example) in this field of the form:

Options for show.tiki.org > Supported Versions : `trunk,23.x,22.x,21.x,20.x,19.x,18.x`

Related

- How to release: Branching
- How to release: Follow up
- How to Use the Release Server