Composer

Please also see:
A high level overview of how Tiki uses Composer. See also How to pick a software library. In 2021, we moved to Composer v2, in time for Tiki23 so it will be nice and robust for Tiki24LTS. Ref: https://blog.packagist.com/composer-2-0-is-now-available/

Introduction

If you are getting an error about Composer not being set up, run the setup script:

1. Set permissions and run Composer
    - `sh setup.sh`

If you are having issues:

1. Delete the `vendor_bundled/vendor` directory in your Tiki root. Composer will recreate it.
2. Run setup.sh again
    - `sh setup.sh`

If one of the packages is not working for you (ex.: won't download, or some other error) and you don't need it for your Tiki, just:

1. Delete the `vendor_bundled/vendor` directory in your Tiki root (Composer will just recreate)
2. Remove the line for that package from `vendor_bundled/composer.json`
3. Update the auto-generated dependency list.
    - For developers

      ```
      ./temp/composer.phar update --prefer-dist --working-dir="vendor_bundled"
      ```

    - For non-developers:

      ```
      ./temp/composer.phar update --prefer-dist --working-dir="vendor_bundled" --no-dev
      ```

4. Run setup.sh again
    - `sh setup.sh`

Composer is a dependency management tool for PHP.

Composer maintains a list of dependencies using the **composer.json** file and stores downloaded content into the `vendor_bundled/vendor` folder. Additionally, an autoload script is being generated to avoid handling the various paths generated.

Available packages can be found on Packagist, the companion website.

1.2. Description of various vendor folders

Tiki17+

- `vendor`: for your libs managed in composer.json
- `vendor_bundled`: libs bundled in Tiki. List is at `vendor_bunded/composer.json`
- `vendor_custom`: for custom libs, usually manually uploaded (if you need it)
- `vendor_extra`: for libs not yet managed by Composer

Until Tiki16

- `vendor`: libs bundled in Tiki. List is at composer.json
- `vendor_custom`: for custom libs, usually manually uploaded (if you need it)
- `vendor_extra`: for libs not yet managed by Composer

1.3. Production vs Development environments

1.3.1. Production Environment

The release process has been updated. Installation from built packages should not be affected.

Installations from Subversion need to be considered as development environments.

1.3.2. Development Environment

The composer install process has been added to the `setup.sh` script. Running it as usual should ensure composer is downloaded and executed.

Keep in mind this is a fairly new addition to the process. Some environments may run into issues until it is stabilized.

If your environment does not provide all of the tools required to automatically set-up the dependencies, you will need to perform a few steps manually:

1. Download composer and store composer.phar in the `temp/` folder
    - ```
      wget https://getcomposer.org/composer.phar -P temp
      ```
2. Run composer:
    - ```
      php temp/composer.phar install
      ```

Note for Windows users

[+]

1.4. Managing dependencies

1.4.1. How to change composer.json packages and dependencies

When you need to use a certain composer package not available yet in composer.json file or update the tiki dependencies on existing packages:

1. Update composer.json
2. Generate a new composer.lock file
    - `./temp/composer.phar update --prefer-dist --working-dir="vendor_bundled"`

    - You can review the differences with the old one via `svn diff`.
    - If you notice a `dist.url` in packages array that is not pointing to composer.tiki.org, this means that satis configuration of composer.tiki.org should be updated. Please open `doc/devtools/satis.json` and add/change the package requirement in `require` section at the bottom of the file. Please see composer-tiki-org for more info.
    - Make sure every package composer.lock file tries to load from an external url is properly required in satis.json.

1. Then, you should be able to commit satis.json and let composer.tiki.org update which should happen within 10 minutes.
2. After composer.tiki.org is updated, you should run again:
    - `./temp/composer.phar update --prefer-dist --working-dir="vendor_bundled"`

1. Compare the differences of composer.lock file with svn. You should now not see any external urls in `dist.url` sections of packages array. Then, you can safely commit `vendor_bundled/composer.json` and `vendor_bundled/composer.lock` files.

Important: when changing satis.json, be sure to not update (or remove) existing package requirements in a way that old packages are excluded. Older tiki versions require older package versions and we should be careful when updating package requirement to keep the old package still accessible. This implies the usage of ~ and >= specifiers instead of simply increasing the version number.

1.4.1.1. Weekly updates of composer.lock

- https://sourceforge.net/p/tikiwiki/code/70743
- https://sourceforge.net/p/tikiwiki/code/73846

1.4.2. Adding dependencies

Adding dependencies is easy:

1. Lookup the package name on Packagist first.
    1. If you can find the package on Packagist go to point 6 to add it to satis.json
2. If that doesn't exist lookup for an alternative for simple code and take advantage of having a package manager by getting it from here: https://asset-packagist.org/
    1. If you can find the package on https://asset-packagist.org/ go to point 6 to add it to satis.json
3. If that doesn't exist (or if those you find there are not the canonical version), try to see if you can work with the author to add it. Fabio has a lot of experience and can guide you if you have questions. Here is an example for Plotly
4. If that is not working, use https://asset-packagist.org with these two commits as guides
    - https://sourceforge.net/p/tikiwiki/code/65977
    - https://sourceforge.net/p/tikiwiki/code/65978
5. If that doesn't exist, look for a zip or svn/github repo.
6. Add to doc/devtools/satis.json in Tiki trunk.
    - If the package exist on Packagist you just need to add a simple entry (with url) like:
      ```
      "drmonty/smartmenus": ">=1.1.0",
      ```
    - If the package exist on asset-packagist you just need to add a simple entry (with url) like:
      ```
      "bower-asset/fontawesome": ">=5.3.1",
      ```
    - Learn more about Satis here.
    - Look at existing examples for the format.
    - See also http://getcomposer.org/doc/04-schema.md.
    - Note: Before moving onto the next step, ensure that https://composer.tiki.org has successfully added your package. composer.tiki.org is updated every X hours from trunk
7. Add the entry to the composer.json file for your branch by specifying the **full release version**.
    - Composer allows for development branches and unstable packages. However, ideally, Tiki should only use released versions of software.
    - Non-exact version numbers could cause conflicts. We don't want that.
8. Clear the Composer cache as follows:
    - ```
      rm -R ~/.composer/cache/
      ```
9. Update the auto-generated composer files by running
    - ```
      php temp/composer.phar update --prefer-dist --working-dir="vendor_bundled"
      ```
10. Test and develop
11. Commit your changes in `vendor_bundled/composer.json` and `vendor_bundled/composer.lock`

Here's a handy tool to test which versions you will be including in the json files

https://semver.madewithlove.com

1.4.3. Updating dependencies

Similar to adding dependencies.

1. Add to doc/devtools/satis.json in Tiki trunk (but don't remove the old one if it's still needed for a previous Tiki version)
    - composer.tiki.org is updated every X hours from trunk
2. Update the version number in composer.json
3. Update composer.lock
    - `./temp/composer.phar update --prefer-dist --working-dir="vendor_bundled"`
    - Note: if you are going to commit the lock file back to Tiki then run update **without** the `--no-dev` option so the require-dev packages are included

http://versioneye.wordpress.com/2013/03/25/minimum-stability/

1.4.3.1. Suppressing the Ambiguous Class Warning

You can add a pipe command to the normal composer.phar commands to hide all the annoying "Ambiguous class resolution" warnings that some of the old third party libraries generate during the autoload files generation, using:

```
php temp/composer.phar --no-dev install 2>&1 | sed '/Warning: Ambiguous class resolution/d'
```

or after changing composer.json use:

```
php temp/composer.phar update -d vendor_bundled --prefer-dist 2>&1 | sed '/Warning: Ambiguous class resolution/d'
```

- Note: the addition of the `-d vendor_bundled` parameter for Tiki 17+ only

1.4.4. Patching dependencies

cweagans/composer-patches was added in Tiki17 (and later enhanced). Although we always want to solve the issue upstream, sometimes, a patch needs to be maintained until things are sorted out.

Here are examples of usage:
https://gitlab.com/tikiwiki/tiki/-/tree/master/installer/composer-patches/composer-patches.json

This requires the patch utility and we are coordinating with upstream to do everything in PHP.

The patches should be versioned as binary files, to avoid problems with line breaks

1.4.5. Cleaning the vendor files

To remove unneeded 3rd party vendor files. See clean vendor files here:
https://gitlab.com/tikiwiki/tiki/-/tree/master/lib/core/Tiki/Composer

1.4.6. Being aware of outdated dependencies

- https://security.sensiolabs.org

1.4.7. System dependencies that can block SVN installs

If a system dependency is blocking svn installs, you can do like in this commit:
https://sourceforge.net/p/tikiwiki/code/66977

1.5. Troubleshooting Tips

1.5.1. Determining Whether the Failure is Related to Tiki

Composer is used for many PHP projects. If you have a general problem with Composer on your setup, first make sure it works outside of Tiki. Then once that works, try with Tiki to determine if the issue is specific to Tiki's implementation. If you discover dependencies to use Composer, please add to the Composer section of tiki-check.php

1.5.2. Within Tiki

1. Run Composer in Tiki
   ```
   sh setup.sh
   ```
2. Test within Tiki
   ```
   php temp/composer.phar diagnose
   ```

1.5.3. Test outside Tiki

1. Get Composer
   http://getcomposer.org/download/
2. Run diagnose
   ```
   php composer.phar diagnose
   ```
3. Try to install Symfony:
   http://symfony.com/doc/current/book/installation.html#option-1-composer

1.5.4. Runtime Exception

If you still get a RuntimeException, on downloading Zend, try:

```
php temp/composer.phar --no-dev update bombayworks/zendframework1
```

or

```
composer update --prefer-dist bombayworks/zendframework1
```

1.5.5. Uncaught exception

If you're getting an error message that starts with *Fatal error: Uncaught exception 'ErrorException'* try using `php-cli temp/composer.phar update` instead of `php temp/composer.phar update`.

Try also to assign permission recursive manually (-+chown+-) of the entire Tiki directory to web user:web group so the Composer can download files and proceed.

1.5.6. Corrupted Download

If you got this error message:

```
The download is corrupted (phar "/var/www/sites/yourwebsitepath/html/temp/composer-temp.phar"
has a broken signature).
Please re-run the self-update command to try again.
```

Doing an "update" won't solve the issue, however re-installing will:

```
php temp/composer.phar install
```

(thanks to Jonny)

1.5.7. Download Failures

System admins may try to run "setup.sh" as "root". This is a bad idea. Downloading composer files under root have great chance to fail due to permission error. If downloading keeps failing, try these steps:

1. Delete `temp/composer.phar`
2. Clear the Composer cache as follows:
   ```
   rm -R ~/.composer/cache/
   ```
3. Retry using another user than root or try:
   ```
   sudo sh setup.sh
   ```

1.5.8. PHP version used on OSX (may be applicable for WAMP on win)

The php version installed with Mac OSX may be different from the one you use with your dev environment (Brew, MacPorts, MAMP or other?). Changes are it is not the one you use on your local. The script setup.sh will try to determine one or several path to check if version and extensions installed are up to date with composer before running it. It may fell to find the path of the php you are using with MAMP (or MAMP Pro). You can then edit or create a file t ~/.bash_profile and add the follwing:

```
export PATH=/Applications/MAMP/bin/php/PHP_VERSION_YOU_USE/bin:$PATH
```

Save and restart your terminal (refreshing it may be enough).

refresh terminal settings

```
source ~/.bash_profile
```

In some other cases (MAMP) it is preferable to declare directly the exact path of the php (version) you want to use.

```
sh setup.sh -p /Applications/MAMP/bin/php/php7.4.12/bin/php
```

1.5.9. Error after switching from a previous branch (Ie switching from 16x to 17x)

When switching from a branch to a newer one some error caused by changes in the way Tiki set or handle Composer may occurs.
You may have to delete caches or some files from previous install.

IE:

```
Fatal error: Uncaught exception
'Symfony\Component\DependencyInjection\Exception\InvalidArgumentException' with message 'You
cannot set service "service_container".'
```

Is solved by deleting : temp/cache/container.php

1.5.10. Cannot allocate memory error

Happened to me installing Tiki21 alpha

Your php may run out of memory on running and in the case given as sample installing a dependency.

Example of memory error

```
Gathering patches for dependencies. This might take a minute.
  - Installing plotly/plotly.js (v1.52.2): Downloading (65%)
mmap() failed: [12] Cannot allocate memory

mmap() failed: [12] Cannot allocate memory
PHP Fatal error:  Out of memory (allocated 146796544) (tried to allocate 132120608 bytes) in
phar:///var/www/virtual/dev.example.fr/html/temp/composer.phar/src/Composer/Util/RemoteFilesys
tem.php on line 598
```

To check the memory limit value in your php.ini file you can run:

```
php --info | grep memory_limit
```

You need to find what php is running the specific process (the process may use a different one than the one you set for the all setup.sh script) and adjust the "memory_limit" setting to "-1" (no limit) or any limit you want.

In my case the following command from within the Tiki directory didn’t return the Configuration File .ini to change but the website php (php7.2)

```
php --ini
```

Troubleshooting steps on clearOs 7 while running composer on Tiki21 alpha

Check php memory limit

```
php --info | grep memory_limit
memory_limit => 128M => 128M
```

Check which php the core is running

```
php -v
PHP 5.4.16 (cli) (built: Nov  1 2019 16:04:20)
Copyright (c) 1997-2013 The PHP Group
Zend Engine v2.4.0, Copyright (c) 1998-2013 Zend Technologies
```

Locate the different php.ini files

```
locate php.ini
/etc/php.ini
/etc/opt/rh/rh-php70/php.ini
/etc/opt/rh/rh-php71/php.ini
/etc/opt/rh/rh-php72/php.ini
.../... there are more results but they aren’t relevant.
```

I edited /etc/php.ini, change the memory_limit from "128M" to "-1" and restarted services. (in doubt I restarted both)

restart services

```
restart mysql.service httpd.service
systemctl restart rh-php72-php-fpm.service httpd.service
```

I re run sh setup.sh and the installation completed.
I reset back the value to 128M as it is a protection to stop a broken process to exhaust your server memory.

1.6. Bluehost

If you are on Bluehost, please see: Bluehost

1.7. Shared Hosts and website control panel (Virtualmin, etc)

Shared web hosting services that offer a choice of different PHP versions may require additional configuration in order to apply the chosen PHP version when running Composer through shell. This happens when shell uses a default PHP version even when phpinfo within Tiki shows the chosen version (the version used in shell can be

checked by typing `php -v`). This can cause Composer to not run if the default version is older than the minimum required version. The way to correct it is to set the path to the right PHP version in a `~/.bashrc` file in your home directory.

Website control panel (Virtualmin, Plesk, CPanel, etc) comes with install scripts that install tools and configuration them for the user. They allow using several version of PHP therefore you may have to adapt and give the setup.sh script details on your configuration.

Below are a couple of examples:

1.7.1.1. Set php to be used with setup.sh (PHP CLI version)

While setup.sh look for "php72", "php7.2" or "php7.2-cli" your hosting company or server admin may use a slightly different name for the php path you need to run Composer.

Different path for PHP 7.2 in a shared host environment

```
sh setup.sh -p php-7.2
```

1.7.1.2. Set PHP PATH to be used with setup.sh (this worked with Plesk using Centos 7)

Tested with Centos 7 and Tiki23

Path for PHP 7.4 in a shared host environment

```
sh setup.sh -p /opt/plesk/php/7.4/bin/php
```

1.7.1.3. Set PHP PATH to be used with setup.sh (this worked with SiteGround)

It is possible with Tiki20 (may be with previous version too) to set the PATH setup.sh will use to point to the right PHP version.

Path for PHP 7.2 in a shared host environment

```
sh setup.sh -p /usr/local/php72/bin/php-cli
```

1.7.1.4. Set PHP PATH to be used with setup.sh after php8 update on virtualmin

You can check the path of the CLI you want to use by using `ls -l /usr/bin/`.
You will be able to see all the php version installed. To use PHP7.4:

```
sh setup.sh -p /usr/bin/php7.4
```

1.7.1.5. Set PATH variable (this worked with HostGator)

1. Modify (or create) your `~/.bashrc` file to prepend the correct path to the PHP file you want to use. For example, if the PHP executable file is at `/opt/php55/bin/php`, you would add the following command:

`export PATH=/opt/php55/bin:$PATH;`

2. Then type `source ~/.bashrc` in shell for the change to take effect before running composer or `sh setup.sh`

1.7.1.6. Use Full PHP Path to Run Composer Directly

This is a little hacky, but if nothing else is working, instead of running composer through `sh setup.sh`, composer can be run directly using the specific PHP path you desire, for example: `/usr/local/php56/bin/php temp/composer.phar --no-dev update`

1.8. Composer.tiki.org

- composer.tiki.org fixed by running Satis
- ~~Find a solution for versions~~ Done
- ~~use https when possible~~ done
- ~~As part of the release process, have a way to check Composer packages vs original code~~ Added to Releasing

1.8.1. To flush out an incorrect package on composer.tiki.org

If you happen (as I did) to commit a mistake in satis.json, i.e. link to the wrong zip file for instance as corrected in **r54048**, composer.tiki.org will continue to serve up the wrong file until that package is updated again, so here's how to fix it.

- First get a friendly member of the Infrastructure Team with some spare time to help
- They will need to remove the offending package file and it's cached copy, then run the satis rebuild script, here is an example with the chosen zip above:

```
cd /var/local/composer.tiki.org/bin/satis/
rm /var/local/composer.tiki.org/www/dist/jquery-plugins-chosen-1.3.0.zip
rm  ~/.composer/cache/files/jquery/plugins/chosen/1.3.0.0-1.3.0.zip -- if it exists
mv satis.json.prev satis.json.prev.bak -- reset for the satis build script
/etc/cron.hourly/qualityupdate -- contains the satis/composer build commands as well as the
code.tiki.org script
```

1.8.2. To make sure all packages are linked directly from composer.tiki.org

Needed for environments where only composer.tiki.org is allowed as a repository

- Search through `vendor_bundled/composer.lock` for the regular expression `dist^\}*\.com` and locate packages not on composer.t.o
- Check if the package is available on https://packagist.org (easiest if it is)
- Search again in the composer.lock file for where the package is required (often as a dependency of another package), and copy the name and version requirement
- Add that to `doc/devtools/satis.json` in trunk, commit and wait for it to (hopefully) appear on https://composer.tiki.org
- Delete your local `vendor_bundled/composer.lock` and `vendor_bundled/vendor` directory
- Run `php temp/composer.phar clearcache`

- Run `php71 temp/composer.phar update -d vendor_bundled --prefer-dist` (whatch out for using the correct php version for the branch you are updating)

1.9. Other notes

2013-07-11 topics to discuss with Nils

LP & Marc will see Nils at http://www.phpquebec.org/content/composer-project-dependency-management-php and we'll discuss with him:

- The checksum verification of the file failed
    - This is reported to be fixed
- Continue on error (if package is not available)
    - Now, retries on failure
- zip/unzip perms for FullCalendar
- Satis override composer.json
- See if tiki-admin.php could report if composer is up to date
- Sbox_aufs specific issues

Some handy aliases for your ~/.bashrc (or ~/.profile depending on platform)

```
alias composerupdate='php temp/composer.phar update -d vendor_bundled --prefer-dist 2>&1 | sed '\''/Warning: Ambiguous class resolution/d'\'''

alias composerinstall='php temp/composer.phar install -d vendor_bundled --prefer-dist 2>&1 | sed '\''/Warning: Ambiguous class resolution/d'\'''
```

1.10. Web-based management

Something like proc_open should be used to run it

- https://github.com/CurosMJ/NoConsoleComposer
- http://stackoverflow.com/questions/15969948/use-composer-without-ssh-access-to-server
- http://stackoverflow.com/questions/20894518/how-do-i-install-composer-on-a-shared-hosting
- http://stackoverflow.com/questions/17219436/run-composer-with-a-php-script-in-browser
- https://github.com/composer/composer/issues/2545
- https://github.com/composer/composer/issues/2684

1.11. Optional libraries

See Install optional libraries in Tiki via Composer

Composer was originally to permit the phpBB project to manage their extensions

- Optional components should part of the plan for File and directory structure revamp
- https://mediawiki.org/wiki/Composer/For_extensions
- https://github.com/wikimedia/composer-merge-plugin (Merge one or more additional composer.json files

at Composer runtime)

- https://github.com/contao-community-alliance/composer-client
- https://www.drupal.org/project/composer_manager

1.11.1. Options to implement optional libraries

The should be an option via the command line, and via FTP to add additional optional libraries

1.11.1.1. Option via command line require

How to add a lib

```
php temp/composer.phar require mpdf/mpdf "*"
php temp/composer.phar update --prefer-dist
```

Pros

- Works

Cons

- Will break on FTP upgrades
- Can cause merge issues even on svn
- Requires command line

Notes

- For this to work, we should store the config in the database, so it can be re-ran at each upgrade
- Will this work with installs without shell access?

Split composer.json information into two files

Similar to https://github.com/symfony/symfony-standard/blob/master/composer.json and https://github.com/symfony/symfony/blob/master/composer.json, we could split in two files.

tiki-base/composer.json would have most of the information we have today.
tiki/composer.json would call tiki-base/composer.json and permit end users to add their own dependencies, either manually or via the require command. It reduces the odds of merge conflicts

1.11.1.2. Option Mediawiki Composer plugin

- https://github.com/wikimedia/composer-merge-plugin/blob/master/composer.json

Pros

- Used by others so must work
- Possible to have multiple composer.json files and have them merged

Cons

An extra lib to maintain

Notes

- Will this work with installs without shell access?

1.11.1.3. plugins.roundcube.net

Another to investigate: https://plugins.roundcube.net/

1.11.1.4. Composer/Installer/PluginInstaller.php

https://github.com/composer/composer/blob/master/src/Composer/Installer/PluginInstaller.php

1.11.1.5. Locations of optional libraries

- /vendor/ directory should be deletable so it can be re-created by Composer (and thus libs listed in composer.json) So for manual downloads, perhaps vendor_custom or vendor_local or something else?
- See also: File and directory structure revamp
- Files will be detected by security check on files

1.11.1.6. Other options

- https://github.com/francoispluchino/composer-asset-plugin

1.12. Related

- https://github.com/vinkla/climb/blob/master/README.md
- https://github.com/composer/installers
- http://composer.tiki.org
- https://igor.io/2013/09/04/composer-vendor-directory.html
- http://robloach.github.io/component-installer/
- https://github.com/component/component/wiki/F.A.Q
- Composer Cheat Sheet for developers
- https://github.com/clue/graph-composer
- http://12factor.net/dependencies
- VersionEye
- http://depending.in/
- https://vinkla.com/2016/composer-outdated/
- http://www.jsdb.io/
- https://github.com/FitchLearning/satisfy
- http://webtips.krajee.com/using-forked-version-of-packages-with-composer/
- http://webtips.krajee.com/setting-composer-minimum-stability-application/
- Composer package version strategy