DevTips

Table of contents

These tips are here to help new contributors getting a feel for the environment. Some people are inclined to follow these as "Rules" or "Guidelines"; they are suggestions rather than strict requirements. Consider each point below as information about what is in Tiki. As a contributing coder, you should respect that environment. You may wish to read everything below in detail.

If you only read one line

(aka TL;DR)

- **3 Rules** : 1/ **Respect Environment** 2/ **Commit Early, Commit Often** 3/ **Make it Optional**

Commits

- How to get commit access
- How to commit
- Where to commit

File names conventions and generalities

- All **file names are lowercase**
  - Only exceptions are 3rd party libs like Zend framework and other libs requiring "PSR-0 standard for PHP class naming allowing for autoload"
- For a PHP file name, the pattern is: *tiki-xxx_yyy.php*:
  If possible, *tiki-mainobject_action.php*
  Some exceptions are noticeable in files that are included in others (it seems that they don't follow the tiki-xxx_yyy.php convention)
- A lib file name follows the pattern *lib/object/objectlib.php*
- Give features descriptive names (and use the same keyword in templates, css, libs for easy finding), as there is no feature duplication (we don't have several forums systems in Tiki, we have *one* and **we all collaborate** on it)
- Avoid names like feature_new, feature_newer, feature_newest...

Files content integrity

- **Never change indentation on code you don't change**. It could provoke diff and merge inconsistencies.
- Prefer to use only **tabs for indentation** for a new file.
- Be careful with your text editor. Make sure that it uses **LF** Unix line endings and not CRLF Windows line endings. **Only exception** perhaps are Smarty templates in `templates/mail/` used for sending mails which should use CRLF because some Windows servers seem to have a problem with sending mails then (see this thread). If in doubt, use the end of line character from the file that you changed.
- Use only **UTF-8** encoding for language files. Hence, use a UTF-8-enabled text editor for those files.

Strings integrity

- Follow the format convention: Strings Format Convention
- Reuse the same terminology
- Descriptions (like preferences descriptions) should end with a "." In general, any sentence should end with the complete punctuation.

SVN operations

- Only **commit to one branch**, stable branch or Trunk, depending the current Where to commit

Commit messages

Take in mind that your commit message should be clear and describe all operations that this commit is for. Feel free to give the tracker ticket that this commit is closing (if it is the case).

Your commit message may start with one or more of the tags listed at the top of changelog.txt to distinguish changes.

If you are doing a Backport to an earlier branch of a commit in trunk, see also this page: Merge a commit from trunk

Database conventions

- All table names begin with tiki_ except users_ for historical reasons
- **Table name has to be less than 26 characters**, all lowercase, using chars and underscores
- Primary keys in tables usually use the following convention : objectId with a capital 'I'. It's a rare case where a capital is used in any name
- When you modify the schema (create / modify tables), you need to care for both future installs and existing installs (upgrades). For future installs, you need to modify *db/tiki.sql*. See Database Schema Upgrade for more information
- Respect abstraction scheme and naming conventions for queries as detailed on DbAbstractionDev
- **Do not use reserved words** for column name (for instance **do not use: user, status, order, show for column name**)
    - http://dev.mysql.com/doc/refman/5.1/en/reserved-words.html
    - http://www.postgresql.org/docs/8.4/static/sql-keywords-appendix.html
- Key/index are also limited to 1000 bytes (so 250 chars? - to be checked)

PHP coding habits

- Coding standards: please use Zend Framework Coding Standard for your code.
    - Exceptions:
        - **WE USE TABS** - Indentation using tabs rather than spaces
    - The Zend coding standard was adopted after Tiki had accumulated a significant PHP code base which did not follow this standard. Tiki's current PHP code is still far from adhering to this standard everywhere, in particular when it comes to function names, which were previously written completely lowercase, with underscores between words. For example, as of 2017-01-22, we have clean_logs() , which should be named "cleanLogs" according to our standards. Some old sets of functions called dynamically even require to be named fully lowercase. For example, the FOO plugin requires a function named "wikiplugin_foo" as of 2017-01-22.
- All the strings are written to be easily translated using the *tr()* function. E.g. `tr('some string')`

PHP Security Guidelines

All commits will be reviewed for security issues. It's a good idea to familiarize yourself with the expectations of Tiki commits outlined on the Secure Coding Practice Guidelines page.

Files should start with...

1st line of Smarty templates (.tpl)

```
{* $Id$ *}
```

Start of PHP files (.php)

```
<?php
// (c) Copyright by authors of the Tiki Wiki/CMS/Groupware Project
//
// All Rights Reserved. See copyright.txt for details and a complete list of authors.
// Licensed under the GNU LESSER GENERAL PUBLIC LICENSE. See license.txt for details.
// $Id$
```

Id

To expand the Id

```
svn propset svn:keywords Id filename.php
```

To commit

```
svn commit filename.php
```

Front-end

Tiki serves **HTML** (version 5) documents. The W3C Markup Validator can help checking the validity of your design, but is experimental.

Since Tiki versions prior to 9 used XHTML, several void elements are written with needless trailing slashes (for example, "<img />", instead of "<img>") . For the following tags you can (or should ?) use the shorter notation:

- br, img, link, param, meta, input, area, base, br, col, command, and embed (following "void elements")

CSS stylesheets are widely used to **separate design from content** and to create Themes.

Smarty templates

- Strings displayed are enclosed by *{tr}{/tr}* blocks for translation.
- There are no standards about Smarty template formatting (either from Tiki or Smarty). However, templates shipped by Smarty and Smarty documentation appear to use template indentation rather than output indentation (in other words, the indentation facilitates reading the template rather than reading the generated HTML), as can be seen in Smarty's "Calling a recursive menu example".
- Comments in Smarty code can be useful. The Smarty comment delimiter is *{* *}*.

Introduction to Tiki Code

Introduction to Tiki Code Layout

Best Practices

- Bootstrap
- Templates Best Practices
- Filtering Best Practices

Icons

- See icons if you want to use icons in the templates

Think of ReleaseProcess

- Mention your meaningful changes on a TikiXY page (where XY is main Tiki version number, e.g. Tiki17) on https://doc.tiki.org. It is useful for users to know what is new or changed in the upcoming Tiki version (we do not output the commit messages to changelog.txt file anymore since Tiki 16.2).

About license

- Code needs to have **LGPL**, MIT or BSD-like licences if bundled in Tiki (not GPL). Please see LibLicense

~~External GPL code, like Wollabot, is an exception.~~

Setting up a debugger/development environment

The following links may be a useful reference:
Xdebug etc

Discussion/Participation

[-]
How about using a style-fixing program on commit, to help enforce some of these rules? — mdavey

---

See also, on t.o, in progress of migration here :

- 3 Rules
  some basic principles about contribution in Tiki development community
- GuidelinesDev
  old page before this one, still holds good information
- DbAbstractionDev
  a little messy but very helpful page to know how to write your sql queries in libraries.
- TikiDevelopment
  Useful links to development tools and material.
- TikiDevNewbie
  advises from a tiki contributor to new developers
- HowToDev
  rather old and badly maintained useful page
- Strings Format Convention
  String format convention
- a tutorial for developer beginners
- Permission Revamp explains how the permissions work

Code Maps and Howtos

Because Tiki is a large system, it can sometimes be hard to find your way around the code, or how to accomplish a specific task like "adding a new icon". The Code Maps and Howtos page contains links to:

- Code "maps" that can help you orient yourself and navigate a particular part of the code (ex: the multilingual functionalities of Tiki).
- Code "howtos" that can help you figure out how to carry out a specific programming task, for example, "adding a new icon" to the UI.

All coders are encouraged to write such maps to help others. If you are a newbie to a particular part of the code or particular kind of task and find that they are not covered, you might want to create a map or how to keep track of your findings and document what you find so others will benefit from it in the future.