Table of contents

Tiki Mods

Modular install for Tikiwiki components

The tikimods feature is in 1.9 tikiwiki release the new tool for distribution, install, upgrade and removal of tikiwiki components, also called *mods*. A Mod can be a feature, a plugin, a module, a theme, a language file, a replacement for an existing feature, or a combination of all the previous items.

Basic principle

The tikimods behaves like many package management software that is known in the unix world, and could be compared to the debian apt, the redhat yum; mandrake urpmi; or gentoo emerge, but adapted to the tikiwiki environment. It's a web-based tool for update operations on some tikiwiki extensions. It fulfills the following functions:

- listing of a remote mods server, by default http://mods.tiki.org/
- maintenance of locally stored mods : comparison with remote, downloading of new versions
- installation of local mods, upgrade and removal

Requirements

For being able to move files around in tikiwiki file tree, apache user needs to be granted the right to do so, which shouldn't be the normal setting. You can enable that permission setting by

- ./setup.sh www-data all
  or nobody, apache or anything instead of www-data, as far as it matches with the apache user
- ./fixperms.sh open

After mods operation, the perms have to be setup back to the previous state, which depends your environment.

- ./fixperms.sh fix
  and when requested give the name of the files owner, which depends if you use CVS.

Note that Windows users don't have to bother with perms.

Morphology

Each Mod is defined by a type and a package name. the types are arbitrary strings to group mods in mods families. The initial set is composed of avatars, icons, menus, themes, languages, sqlupgrade, wikiplugins, features, services ... more types can be added with time, but please do not add wild types. New types should be only added after proper discussion with knowledgeable people (whatever that means for you).

A mod is managed by a control file with following characteristics :

- follows name convention type-name.info.txt
- stored in mods/Packages/ dir typically
- the syntax of this file is intended to be human-readable, composed by parameters and values
- each parameter block is separated by 2 line breaks
- the first line of parameter block is the parameter name, possibly ended by a colomn (:) which is stripped if present
- the following lines, until a blank one appear, is the value of the parameter
- the initial set of parameters is composed of :
  - **contributor** : the login or name of the one that commits, can use the $Author$ cvs macro
  - **revision** : the incremental number of the version of the mod. it can follow the revision number

$Revision$ from cvs but not necessarly
    - **requires** : the optional list of required mods by this mods. It must be one line by depend, each line must be of the form: type-name < | > | <= | >= | = revision (and you can have multiple revision tests in one line)
    - **suggests** : the optional list of suggesteds mods. Same syntax than **requires**.
    - **conflicts**: the optional list of mods that conflict with this mod. Same syntax than **requires**.
    - **lastmodif** : the date of last version of the mod, using the macro $Date$
    - **files** : a list of all the files that compose the mod. One line per file, composed of the origin and the destination separated by a space. The origin is relative to mods/ and the destination relative too the tiki documentroot. Files in mods can be prepended with sample: prefix in which case they will be modified by the configuration at install time.
    - **Description** : an arbitrarily verbose (but not much) description of the functionalities of the mod
    - **Docurl** : the list of relevant urls (one per line) related to documentation of the mod
    - **Devurl** : list of urls used for the development of the mod (usually, but not mandatorily, in http://mods.tiki.org)
    - **Licence** : the legal form under wish the mod is distributed. If not specified, follows the tikiwiki licence scheme
    - **Author** : the name, login or identifier of the author(s) of the mod. If it's a coolactive work, one name per line is accpetable.
    - **Version** : the list of restrictive version for which the mod is designed. If that field is absent, any version should be able to use the mod. *(not functional yet)*
    - **Changelog** : the list of changes, descending, usually one line per change
    - **Configuration** : enables install-time setup of some values. One line per variable, composed of Label, name of the variable and default value from $_SERVER array (for example $HTTP_HOST will propose $_SERVER['HTTP_HOST'] as a default value in configuration screen). The configuration variables suubstituation will be processed only on files begining with "sample:" prefix in mods repository.
    - **Configuration help** : the text displayed on configuration screen at install time (remember to avoid empty lines as they are parameters separators).
    - **help** : a freeform text instruction about the mod installation and first use
    - **sql-install** : the list of sql command issued at install time, one command per line.
    - **sql-remove** : same as sql-install, but executed at removal time.
    - **sql-upgrade** : that's a special sql statement list. It's splited by version numbers to which the upgrade sql statements apply. The version number has to be on a line prepended with a column. The lines below the version value will be associated to that version upgrade, so multiple upgrade can be run at once if needed.

The maintenance of the mods repository is managed by other text files that have to be present in the mods/Packages/ dir as well. Those files are indexes in CSV format, so they can be used by other tools. The indexes are following the conventions :

- name convention is that all indexes files begin with 00_list. and end with .txt so they are at the top of the list when Packages content is sorted alphabetically
- each line in indexes is composed by strings enclosed with simple quotes and separated with commas
- the fields are 'type','mod name','revision','description','licence' (probably more will be added, but the order shouldn't change).
- of course the content of fields has to be addslashed (simple quotes preceeded by a backslash).
- there is differnt index files for different uses
    - 00_list.txt : is the index of locally available mods. This index can be rebuidl locally by using the 'rebuild local list' link in tiki-mods.php
    - 00_list.public.txt : is the index that is generated when you setup your mods to be able to be

mods provider. It lists the mods that are available for other tikiwikis
    - 00_list.xxxxxx.txt : is the index of remotely available from the sevrer configured in the tiki-mods_admin.php panel where xxxxxx is the urlencoded url of the server it comes from

Mods panels

tiki-mods.php

It is the main mods management panel. It displays a list of both local and remote mods listed in index (00_list) files.
The top link row proposes 3 links :

- **Mods configuration** leads to the configuration panel
- **Update remote index** rebuilds the list of **remote** mods from the mods provider configured in the admin panel
- **Rebuild local list** rebuild the list of **local** mods, by reading all the control files

If the mods provider option is enabled, there are 3 more links :

- **Republish all** rebuilds and upgrades all available packages from local ones
- **Publish all** publishes all packages
- **Unpublish all** unpublishes everything

The table displayed below the links row is basically listing the index informations :

- the rows are displaying all mods, separated by types
- if the mods provider checkbox is checked, the first column displays the publication status of the mod, with a link to publish or unpublish it. That operation mainly rebuilds the 00_list.public.txt file by adding or removing the mods of that row. The version number of the packaged tarball is displayed, and if local verison is more recent, and upgrade link (republish) is visible.
- the next column is the version available remotely, with a link to download it. That is the first column is you are not providing mods to others
- then the name of the mods. If the mod is present locally, a link leads to the detailed view of the mods.
- next columns are revision number, licence and description
- the last column is only displayed if the mod is present locally (either downloaded from remote provider either built locally and specific to your use)
    - if the mod is not installed, an install link is the only thing displayed
    - otherwise, the columns holds 3 parts :

    - the status of the revision : up to date, or not
    - the status of installation (basically installed but more can come there)
    - a link to remove the mod from tikiwiki (but it will still remain in the local mods repository
- when a local mod is clicked, the mods table is displayed with the specific mod information displayed just below eth row of the selected mod, reading the control file and displaying author, lastmod, changelog, as well as list of files contained in that mod (and ultimately any information available about that selected mod).

tiki_mods_admin.php

That is the mods cobnfiguration panel, that for now handles 3 parameters

- a checkbox for being a mods provider. It will add the 'publish' link in each row of the mods main panel
- the local mods directory
- the url of the remote mods provider

Mods Provider

The mods features make possible for anyone to become a mods provider, for local or specific releases. The tikiwiki.org repository should be the collective one, and other repositories can either provide the same repository (which is the cvs module named **mods** on sourceforge) for local areas as mirrors for faster access, or be dedicated to personal development, as it's rather easy to build a new mod following the above guidelines.

- the publish link will add the mod to 00_list.public.txt for remote reading
- that link also build the tarball in mods/Dist/ directory, with name syntax type-name-revision.tgz, using the tarlib (no exec needed).

for developers : mods.sh

In 1.9cvs you can find that script in doc/devtools/mods.sh. This script is recent and has only a few features, but the goal is to make possible to manage the mods only with it. Before use **edit this script** to adapt the configuration to your environment, and when done you can launch it from your tiki root directory.

> Usage: ./doc/devtools/mods.sh [copy|diff|install] copy : copy tiki files to mods diff : diff tiki files with mods install : copy mods files to tiki without 2nd arg, returns the list of files from package in tiki tree !!! note that this script only works with installed packages !!! !!! (this is work in progress, use is quite limited for now) !!!

Mods evolution

The first version of mods feature is very basic, and will need to handle some more things :

- md5 signature of the tarball present in the control file
- management of scripted operation at install and remove time, for patching or anything. But anything being able to be anything by nature, it requires a good security attention and a non-risky context.
- adding more information in the control file
- possibility for having multiple author displayed in the mod details
- new type of mods for maintenance operations like upgrade, janitoring, cleaning, archiving, or anything that can be handled by sql commands, those mods are never installed and can be run at will
- languages modularisation, that would involve a new convention about translating mods-specific strings as well as gathering all strings in a common place for easy translation.
- styles modularization, by cutting the css in part, so mod can embed, as for language, the specific part needed for its use.
- Link to the installer to garantee a non-regressive upgrade from 1.8 or 1.9rc.
- development of a shell-based script for mods management, for easy administration and avoiding the perms change.
- maybe even development of a VB-based clickable tool for binboze users that reproduces the functionalities of the shell script cited above.
- find a way to manage the perms change for ftp-only users. Maybe the best way would be to use a ftp script to install mods on a remote server, using lftp on *nix or the vb-script for binboze victims.
- manage a dependancies system for linking installs
- add a way to update everything in a click
- manage a checking of existing tiki version to take in account the required minimal version for each mod
- remove the mod-application_menu.tpl so we can refer to the sql-based application menu (thhat can be changed easier by the mods installer
- change the features admin panel for it takes the features from the db rather than hardcoded, so mods can interact and add items in features admin panel.
- setup a new page for displaying the focus. The disply inside the list won't last long (as the list will grow).

Most of those evolutions should be ready before 1.9 final release.

How to add your own mods

Let's work a simple mods, a wikiplugin

Get _mods

First get a CVS version of _mods from HEAD. From now on, all paths will be relative to the directory _mods.

Create a wikiplugin

Create your plugin: