System configuration

System configuration allows for system administrators to set some preferences using the file system rather than the Tiki database. Some preferences are hidden by default on Tiki. You can un-hide them and create your own set by providing an .ini file. This file can be used in several Tikis allowing administrators to have a central configuration for different Tiki installations.

Introduced in Tiki8

Preferences configured through system files cannot be adjusted via Tiki interface. They will appear disabled.

New in Tiki10: Option to load the modules from a static files (in profile YAML format like Module Handler. Search for "Module file" in the admin panel.

New in Tiki11: A file db/preconfiguration.php can contain database credentials which are suggested at first Tiki installation. This allows providers of Virtual Servers to preconfigure mysql with different random passwords and usernames before the customers access the Tiki installer.

New in Tiki15: Due to changes in Zend libraries, it is no longer possible to have hierarchical setting of configurations as before. In addition, configuration names can no longer contain dots. For example, client1.example.com will not work. Use client1 instead. Old configurations may not be usable in Tiki 15 and have to be rewritten.

New in Tiki20: Hide classes of preferences that are disabled

New in Tiki20: Permit to hide preferences from admin panel

New in Tiki22: Risky preferences are now hidden by default (System Administrator can un-hide)

Possible uses for Tiki Farms and hosting companies

- Connect to other services, such as BigBlueButton, without intervention from the users
- Configure advanced settings such as Memcache
- Disable features that are not part of the support plan
- Respond to security advisories by globally disabling a feature
- provide tiki_p_admin but make some features inaccessible
- Hosting company may want to offer all-in-one hosting, but restrict usage of
  - experimental features
  - feature that can be used to gain server access (which is OK for most setups, but not cool for a WikiFarm)
- Consulting company wants to make simpler admin panels for a customer
- Using Tiki as an application builder (framework)

Activate the system configuration

The system configuration must be enabled from the db/local.php file. Installations created from Tiki8 and beyond contain samples in the file created by default.

Sample configuration in db/local.php to an absolute path

```
$system_configuration_file = '/etc/tiki.ini'; $system_configuration_identifier = 'client1';
```

Another sample configuration in db/local.php, with a relative path

```
$system_configuration_file = 'db/tiki.ini.php'; $system_configuration_identifier = 'client1';
```

If you do this example, make sure .htaccess is active so the tiki.ini.php can't be read or better yet, ensure it is called tiki.ini.php and refer to section *Protecting the system configuration* below.

Another sample configuration in db/local.php, with a relative path pointing outside the web accessible directory

```
$system_configuration_file = 'db/../../tiki.ini.php'; $system_configuration_identifier = 'client1';
```

If you do this example, put tiki.ini.php above the web directory. Ex.: if your db/local is here:
/home/tiki/domains/dev.tiki.org/public_html/db/local.php
Your tiki.ini.php goes here:
/home/tiki/domains/dev.tiki.org/tiki.ini.php

When enabled, Tiki will read additional directives from the configuration file based on the selected identifier. The identifier is arbitrary, but can be used either to specify the site or the plan. In the configuration file, multiple configurations can be defined and inheritance can be used. (see sample of .ini file below for better understanding)

System configuration files contents

Configuration files are INI files. The following example file defines the browser title to be "Test".

INI file example #1

```
preference.browsertitle = Test
```

Identifier

Contents are grouped per block; the first line of the block must be its identifier wrapped in square brackets `[]`.

identifier example

```
[my-client] ...The rest of the content
```

An identifier can inherit contents from another, enabling the creation of reusable sets of configurations across multiple identifiers or Tiki instances in our case.

Inheritence example

```
[basic] ...Re-usable contents. [client1: basic] ...Contents for client1
```

In the given example, the `client1` block will encompass all contents of the `basic` block, in addition to its own extra contents.

Values

Values can be quoted using double quotes. Quoted values can span several lines. Backslashes in values escape double quotes and backslashes.

Sample INI file

[global] preference.feature_wysiwyg = "n" preference.feature_sefurl = "y" preference.helpurl = "http://support.example.com/" ; ... more settings ... [basic : global] ; ... this hierarchical block no longer works from Tiki 15 onwards preference.feature_wiki = "y" preference.feature_forums = "n" preference.feature_trackers = "n" ; ... more settings ... [pro : global] ; ... this hierarchical block no longer works from Tiki 15 onwards preference.feature_wiki = "y" ; BBB configured, but user can still toggle on/off preference.bigbluebutton_server_location = "bbb.example.com" preference.bigbluebutton_server_salt = "1234abcd1234abcd" ; ... more settings ... [client1 : pro] ; ... this hierarchical block no longer works from Tiki 15 onwards, but [client1] will preference.browsertitle = "Client #1 Intranet" preference.sender_email = client1@example.com

In the example above, the following preferences would be set for using the identifier *client1*:

- feature_wysiwyg = n
- feature_sefurl = y
- helpurl = http://support.example.com/
- feature_wiki = y
- bigbluebutton_server_location = bbb.example.com
- bigbluebutton_server_salt = 1234abcd1234abcd
- browsertitle = Client #1 Intranet
- sender_email = client1@example.com

Lines starting with semi-colons (";") are comments.

Protecting the system configuration file(s)

Some times, the system configuration files need to reside in a folder that potentially can be accessible from the web. While not desired, it is critical to protect your configuration file(s) is the file(s) are stored in a location accessible from the web. Tiki assure protection and supports '.ini.php' files, that will work in the same way as the normal '.ini' files, but avoid the content to be downloaded from the internet, by using the right header in the '.ini.php' file.

Note that will work only if your files filename contain `.ini` (Eg. tiki.ini.php) else you will end with a White Screen of Death if you add the code below.

Sample INI.PHP file. Eg. tiki.ini.php

<?php // This script may only be included - so it is better to die if called directly. // Keep this block to avoid the content to be read from the internet. if (strpos($_SERVER['SCRIPT_NAME'], basename(__FILE__)) !== false) { header('location: index.php'); exit; } ?> [global] preference.feature_wysiwyg = "n" preference.feature_sefurl = "y" preference.helpurl = "http://support.example.com/" ; ... more settings ...

Define rules

Rules can be defined for various reasons; they may involve hiding or disabling specific preferences, among other purposes.

Hide preferences

rules.0 = hide preference.feature_wysiwyg rules.1 = hide preference.feature_sefurl ... rules[n]

The hidden preferences are completely removed from the interface.

Disable classes of preferences

Preferences can be disabled by setting a value to each of them individually. However, this process may be long and will require maintenance with the evolutions of Tiki.

Tiki allows to set multiple rule priorities, each verification priority can allow or deny a preference. Rules can be specified at various levels, just like the preference overrides above.

Various rules

```
[global] rules.0 = deny experimental new [pro : global] ; ... this hierarchical block no longer works from Tiki 15 onwards rules.5 = allow new rules.10 = allow feature_wysiwyg
```

In the previous example, sites using the pro identifier would be allowed to use new features and wysiwyg (which is tagged as experimental), including new experimental features, but still be denied old experimental features. Higher priorities are evaluated first and the lookup stops once there is a match.

A draconian provider could use something like this:

Draconian provider configuration

```
[global] rules.0 = deny all rules.1 = allow basic
```

The preferences need to be tagged in the definitions using the 'tags' key containing an array. Unless specified otherwise, all preferences are considered 'advanced'.

Hide classes of preferences that are disabled

If additionally you want to hide from the admin interface preferences that you disabled, you can use the keyword 'hide' instead of 'deny' this act as if it was the normal 'deny' but additionally will avoid displaying the value in the admin interface.

Hide preferences rules

```
[global] rules.0 = hide experimental new
```

Array syntax

Instead of...

```
preference.flaggedrev_approval_categories = "40;41;97;231"
```

...the syntax to define preference.flaggedrev_approval_categories as an array of 4 elements would be...

How to set an array

```
preference.flaggedrev_approval_categories.0 = "40" preference.flaggedrev_approval_categories.1 = "41" preference.flaggedrev_approval_categories.2 = "97" preference.flaggedrev_approval_categories.3 = "231"
```

...or...

How to set an array

preference.flaggedrev_approval_categories[] = "40" preference.flaggedrev_approval_categories[] = "41" preference.flaggedrev_approval_categories[] = "97" preference.flaggedrev_approval_categories[] = "231"

Sample of glitchtip configuration

GlitchTip is an Open Source error tracking system, see: GlitchTip

db/tiki.ini

[glitchtip] preference.error_tracking_enabled_php = y preference.error_tracking_enabled_js = y preference.error_tracking_dsn = https://4bd1dec539c9496b9d3f3ef5bd284f70@glitchtip.example.com/4

Of course, replace the preference.error_tracking_dsn link with the DSN generated by GlitchTip

- Now add the following lines to your existing local.php file in the same /db folder

db/local.php

$system_configuration_file = 'db/tiki.ini'; $system_configuration_identifier = 'glitchtip';

Preconfiguration file

When providing on-demand Virtual Machines with Tiki pre-loaded, hosting companies can provide mysql with a database and connection credentials which are different for each customer.
When accessing stage **Set the Database Connection** of the installer, these values will be prefilled for the customer. This is more efficient than providing them by mail or other means.
These information can obviously not be contained in db/local.php since local.php does not yet exist.

All there is to do is uncomment the appropriate lines in file db/preconfiguration.php and fill the appropriate values:

db/preconfiguration.php

<?php // This is Tiki's preconfiguration file. // This holds values which can provide preconfiguration values in the installer, when local.php does not yet exist // Uncomment and change values as appropriate $host_tiki_preconfig='localhost'; $user_tiki_preconfig='tiki'; $pass_tiki_preconfig='password'; $dbs_tiki_preconfig='tiki_db';

Environment variable configuration

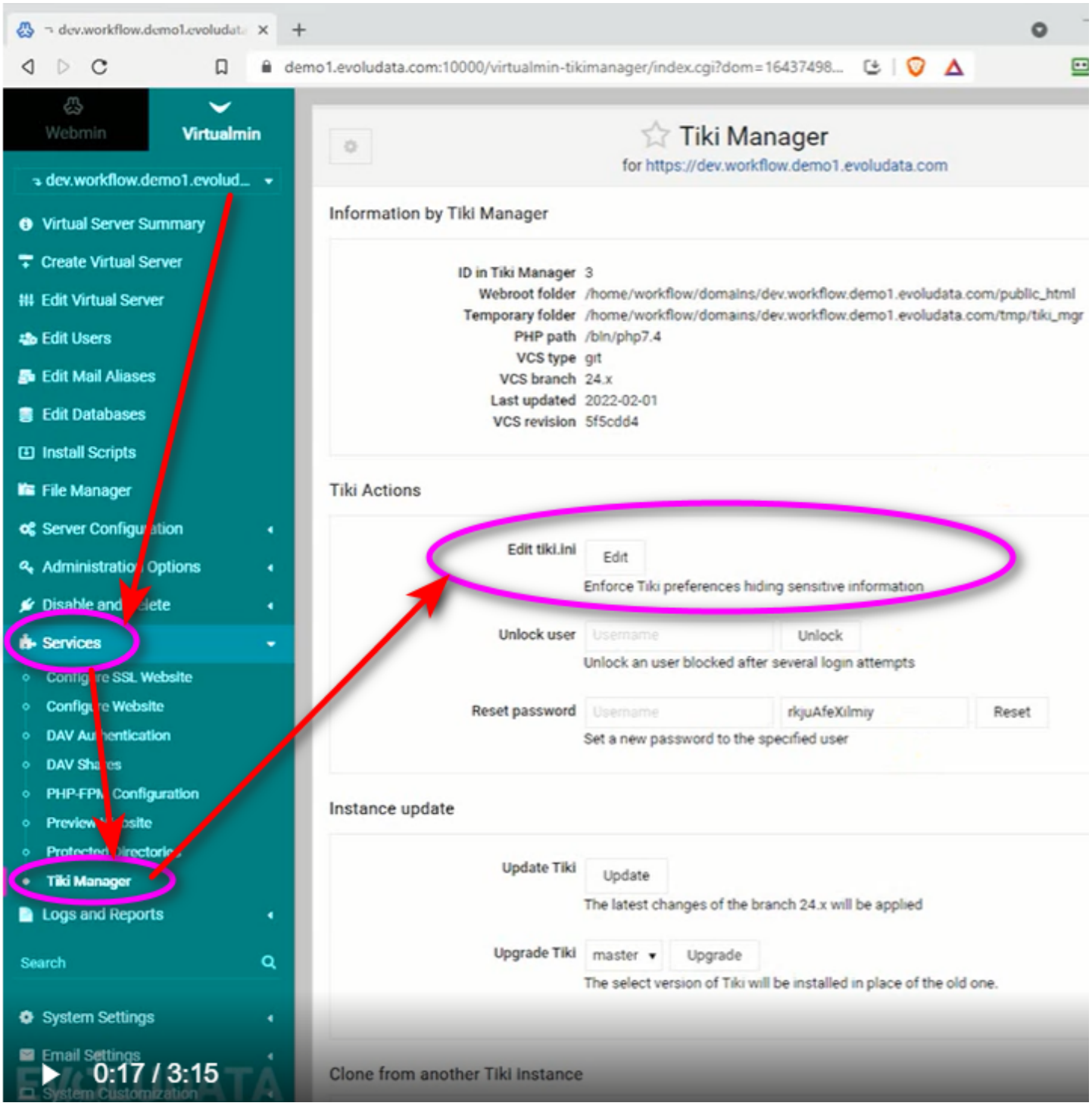
Tiki will also read information about the database connection and system configuration through environment variables. These can be set through the web server's configuration.

Sample nginx configuration

fastcgi_pass unix:/var/run/php5-fpm.sock; fastcgi_index route.php; include fastcgi_params; fastcgi_param MYSQLCONNSTR_Tiki "Database=client1;Data Source=localhost;User Id=client1;Password=secret"; fastcgi_param TIKI_INI_FILE "/etc/tiki.ini"; fastcgi_param TIKI_INI_IDENTIFIER "client1";

In WikiSuite

If you are using WikiSuite, you have a GUI to edit:



Related

- TRIM
- Multitiki

Tiki.ini File | Tiki.ini | INI