A Wiki Administrator can make a lot of customizations simply by setting variables in the */local/config.php* and defining cascading style sheets in */pub/css/local.css* files. Any group or page can also have its own configuration file and configuration css file.

From its inception, PmWiki has been designed so that Wiki Administrators can greatly customize the way PmWiki displays pages and the markup sequences used to generate pages. (This is even mentioned explicitly in PmWiki Philosophy #4.) As a result, the core *pmwiki.php* script makes extensive use of PmWiki.Variables to determine how markup sequences will be processed and what each individual page will output.

The bulk of this page describes how customizations work in general, see PmWiki.Documentation Index for specific customizations that are commonly performed at many PmWiki installations, including:

- Skins - Change the look and feel of part or all of PmWiki
- Internationalizations - Language internationalisation of web pages
- Custom Markup - Using the Markup() function for custom wiki syntax
- Custom InterMaps - PmWiki's default InterMap links, and how to can create additional InterMap links

The simplest type of customization is merely setting a variable to 1 (or TRUE). Here's an example that enables ?action=diag and ?action=phpinfo actions:

```
$EnableDiag = 1;
```

You can begin a line with a "#" (an octothorpe, a.k.a. a hash symbol or pound sign) to add a comment. Additionally, some built-in PmWiki variables take values other than 1 or 0 (true or false). Here's another example that customizes the wiki's behavior with respect to search engine web robots (see Cookbook:ControllingWebRobots):

```
# Remove the default "rel='nofollow'" attribute for external links.
$UrlLinkFmt = "<a class='urllink' href='\$LinkUrl'>\$LinkText</a>";
```

The *scripts/* subdirectory (below the directory holding the *pmwiki.php* script) has many customizations. The PmWiki Cookbook contains many example customizations (recipes) that you can download into the *cookbook/* subdirectory, The first few lines of each of these scripts generally contain instructions about how to enable (and use) the feature provided by the script.

These customizations are included in your *config.php* site configuration. For most scripts this is done by simply adding lines like:

```
include_once("cookbook/recipefile.php");
```

and

```
include_once("scripts/scriptfile.php");
```

at the end of the *config.php* file to enable them.

Some of the scripts are automatically enabled for you via the *scripts/stdconfig.php* script unless you disable it by setting `$EnableStdConfig=0;` in *local/config.php*.

Note that you should strongly resist the temptation to directly modify the *pmwiki.php* script or the files in the *scripts/* subdirectory. Any modifications you make to these files will probably be overwritten whenever you perform a PmWiki.Upgrade. Instead, look at some of the sample scripts for examples of customizations that can be performed from *config.php*. You can even create your own script to do a customization and use `include_once(...)` to include it from *config.php*. If you do make your own customization script, you can safely put it in the *cookbook/* subdirectory--it won't get overwritten by an upgrade there. You might also want to submit your customization to the pmwiki-users mailing list or the Cookbook so that others can benefit from your effort and so that it can perhaps be included in future releases of PmWiki.

There's no "config.php"; it's not even clear what a "local customisation file" is!

The "sample-config.php" file in the "docs" folder, is given as an example. Copy it to the "local" folder and rename it to "config.php". You can then remove the "#" symbols or add other commands shown in the documentation. See also Group Customizations.

Can I change the default page something other than Main.HomePage (`$DefaultPage`)?

Yes, just set the `$DefaultPage` variable to the name of the page you want to be the default. You might also look at the `$DefaultGroup` and `$DefaultName` configuration variables.

```
$DefaultPage = 'ABC.StartPage';
```

How do I get the group / page name in a local configuration file (e.g. *local/config.php*)?

Use the following markup in pmwiki-2.1.beta21 or newer:

```
## Get the group and page name
$pagename = ResolvePageName($pagename);
$page = PageVar($pagename, '$FullName');
$group = PageVar($pagename, '$Group');
$name = PageVar($pagename, '$Name');
```

Can I remove items from the wikilib.d folder on my site?

Yes, the files in *wikilib.d/* can be safely removed. They'll reappear again when you upgrade, however.

How do I customize my own 404 error page for non-existent pages?

To change the text of the message, try editing the Site.PageNotFound page.

Is the order of customizations in config.php important? Are there certain things that should come before or after others in that file? [1]

Yes, the following has been recommended on the mailing list:

- define `$ScriptUrl` and `$PubDirUrl`, if needed,
- define any custom PageStore class, like SQLite, CompressedPageStore or PerGroupSubDirectories,
- next include_once scripts/xlpage-utf-8.php,
- next call XLPage() which needs the definitive (rw) `$WikiDir` already set in order to find the wiki page containing the translations,
- next include authuser.php (if needed), because PmWiki caches some group and page authorization levels when a page is accessed,
- next include any other scripts and recipes,
- any direct function call in config.php, like ResolvePageName(), CondAuth(), PageTextVar(), PageVar(), RetrieveAuthPage(), or others, if possible, should be done near the end of config.php.

*Note, each part is **not** required, but if your wiki needs it, this is the recommended order in config.php.*