Конфигурация

DokuWiki позволяет создавать очень гибкие плагены и шаблоны делая их легко настраиваемыми..

Default Settings

Чтобы сделать плагин или шаблон настраиваемым, необходимо предоставить `lib/plugins/<plugin>/conf/default.php` который будет содержать настройки по умолчанию, и `lib/plugins/<plugin>/conf/metadata.php` который содержит описывающие метаданные конфигурации используемые диспетчером конфигурации для обработки/отображения параметров [1)].

```
$conf[<setting>] = <value>;
```

Конфигурационные метаданные

Для каждого параметра `lib/plugins/<your plugin>/conf/default.php` должно быть `$meta[<setting>]` значение, определенное в `lib/plugins/<your plugin>/conf/metadata.php`[2)]:

```
$meta[<setting>] = array(<setting class>, <param name> => <param value>);
```

Если для класса настроек не требуется никаких параметров (см. ниже), то это просто:

```
$meta[<setting>] = array(<setting class>);
```

Примеры:

```
$meta['_basic']     = array('fieldset');
$meta['title']      = array('string');
$meta['lang']       = array('dirchoice','_dir' => DOKU_INC.'inc/lang/');
$meta['dmode']      = array('numeric','_pattern' => '/0[0-7]{3,4}/'); //
only accept octal representation
$meta['allowdebug'] = array('onoff');
$meta['passcrypt']  = array('multichoice','_choices' =>
array('smd5','md5','sha1','ssha','crypt','mysql','my411'));
```

Классы

''	DKласс по умолчанию («настройка»), текстовая область, минимальная проверка входных данных, вывод настроек в кавычках.
'string'	Ввод текста в одну строку, минимальная проверка ввода, вывод в кавычках.

'numeric'	Ввод текста, принимает числа и арифметические операторы, установка вывода без кавычек. Если заданы параметры `_min` и `_max` используются для проверки.
'numericopt'	То же, что и выше, но принимает пустые значения.
'onoff'	Входные данные флажка, выходные данные настройки `0` или `1` (обратите внимание, что значения по умолчанию должны быть одним из этих целых чисел, а не логическим значением)..
'multichoice'	Выберите входные данные (единственный выбор), установите выходные данные в кавычках, обязательный `_choices` параметр.
'email'	Ввод текста, ввод должен соответствовать формату адреса электронной почты, вывод должен быть заключен в кавычки.
'password'	Ввод пароля, минимальная проверка ввода, настройка вывода обычного текста в кавычках. После установки пароль больше не отображается в менеджере конфигурации. Может быть запутан в конфигурации через `_code` параметр.
'dirchoice'	Как `multichoice`, варианты выбора основаны на папках, найденных в месте, указанном в `_dir` параметре (обязательно).
'multicheckbox'	Флажок для каждого варианта плюс строковый ввод «другое», настройка файла конфигурации представляет собой список отмеченных вариантов, разделенных запятыми.
'fieldset'	Используется для группировки настроек конфигурации, но сам по себе не является настройкой. Чтобы это было понятно в языковых файлах, ключи для этого типа должны начинаться с `_`.
'authtype'	Создает выборку доступных методов аутентификации на основе имен классов, `inc/auth` соответствующих шаблону `authtype.class.php`.
'array'	простой (одномерный) массив строковых значений, отображаемый в виде списка, разделенного запятыми, в диспетчере конфигурации, но сохраняемый как PHP array(). Значения не могут содержать запятые. `_patter` nподдерживается сопоставление значений массива.
'regex'	Строка регулярного выражения, обычно без разделителей; как и для `string`, дополнительно протестировано, чтобы увидеть, будет ли компилироваться и запускаться как регулярное выражение. В дополнение к `_pattern`, также принимает `_delimiter` (default `/`) и `_pregflags` (default `ui`).

Параметры

'_pattern'	String, a regular expression. Input is tested against this pattern before being accepted. Optional all classes, except `onoff`, `multichoice` and `dirchoice` which ignore it.
'_choices'	Array of choices. Used to populate a selection box. Choice will be replaced by a localised language string, indexed by `<setting name>_o_<choice>`, if one exists. Required by `multichoice` & `multicheckbox` classes, ignored by others.
'_dir'	Location of directory to be used to populate choice list. Required by `dirchoice` class, ignored by other classes.
'_code'	Sets the obfuscation option for password fields. May be `plain`, `base64` or `uuencode`. When using the latter two, you need to use conf_decodeString() to access the plain value.
'_combine'	Complimentary output setting values which can be combined into a single display `checkbox`. Optional for `multicheckbox`, ignored by other classes.

'_code'	Encoding method to use, accepted values: `base64`, `uuencode`, `plain`. Defaults to plain.
'_min'	Minimum numeric value. Optional for `numeric` and `numericopt`, ignored by others.
'_max'	Maximum numeric value. Optional for `numeric` and `numericopt`, ignored by others.
'_delimiter'	String, default `/`, a single character used as a delimiter for testing regex input values.
'_pregflags'	String, default `ui`, valid preg pattern modifiers used when testing regex input values, for more information see http://uk1.php.net/manual/en/reference.pcre.pattern.modifiers.php.
'_multiple'	Bool, allow multiple comma separated email values. Optional for `email`, ignored by others.
'_other'	How to handle other values (not listed in `_choices`). Accepted values: `always`, `exists`, `never`. Default value `always`. `exists` only shows 'other' input field when the setting contains value(s) not listed in choices (e.g. due to manual editing or update changing `_choices`). This is safer than `never` as it will not discard unknown/other values. Optional for `multicheckbox`, ignored by others.

Examples

"multicheckbox" with "_other"

Let's say initially a plugin uses the syntax:

```
$meta['multi'] = array('multicheckbox', '_choices' =>
array('a','b','c','d'));
```

this means the multicheckbox will show four choices plus a string input.

Let's suppose also that the user inserts one or more extra values not in `_choices`.

In the next release of the plugin the author's plugin decides to use the multicheckbox without the extra string, but, in order to be backwardly compatible, adopts:

```
$meta['multi'] = array('multicheckbox', '_other' => 'exists', '_choices' =>
array('a','b','c','d'));
```

which means:

- if the user has inserted comma separated values, they will be printed in the extra string;
- otherwise the extra string won't show.

Note also that if the user inserts comma separated values which already exist (or some of them) in `_choices` but:

- they are already ticked, then the extra string will be removed and nothing else;
- they are not already ticked, then the extra string will be removed and the relative value will be ticked.

Accessing Settings

Core Settings

Inside inc/init.php the configurations settings are read into a global array `$conf[]`. When no settings were set, these are read from the default settings file. You can access the core settings anywhere by using the `$conf[]` array.

```
$startpage = $conf['start'];
```

Plugin settings

You can access settings in plugins by using the `$this->getConf('<setting>')` method. In your plugin class you use:

```
$keyvalue = $this->getConf('key');
```

Template settings

In templates you can use `tpl_getConf('<setting>')`.

```
$nicetoknow = tpl_getConf('special');
```

Labels in Configuration Manager

For every setting in `lib/plugins/<your plugin>/conf/default.php` there can be a `$lang[<setting>]` value defined in `lib/plugins/<your plugin>/lang/en/settings.php`. This value will be displayed as the label of the setting in the configuration manager. If the label file is left out or doesn't contain a value for the setting, the configuration manager will display «`plugin <plugin name> <setting>`» as the label instead.

You can also create a `settings.php` file for other languages.

Again, this also applies to templates (see localization for further details).

1)
шаблоны аналогичны lib/tpl/<template>/conf/default.php и т.д.
2)
Для шаблонов все полностью аналогично: lib/tpl/<your template>/conf/default.phpи lib/tpl/<your template>/conf/metadata.php.