Preference, module, package and PDF operators

The kernel operators ezpreference, ezmodule, module_params and ezpackage and the library operator pdf - what each reads, its parameters, what it returns, which of them have side effects - with examples rendered on a...

Source file: doc/eng-us/help/template-operators/kernel-misc-operators.md Last changed 2026-11-04
https://doc.exponential.earth/exphelp/viewer/help/template-operators/kernel-misc-operators.md

Four operators of `kernel/common/` and one of `lib/ezpdf/`, registered in `kernel/common/eztemplateautoload.php` and `lib/ezpdf/classes/eztemplateautoload.php` (a path of `[TemplateSettings] AutoloadPathList[]` in `site.ini`).

Operator	Class and file
ezpreference	eZKernelOperator, kernel/common/ezkerneloperator.php
ezmodule	eZModuleOperator, kernel/common/ezmoduleoperator.php
module_params	eZModuleParamsOperator, kernel/common/ezmoduleparamsoperator.php
ezpackage	eZPackageOperator, kernel/common/ezpackageoperator.php
pdf	eZPDF, lib/ezpdf/classes/ezpdf.php

The examples ran on the command line as the anonymous user.

Contents

1. The operators: ezpreference, ezmodule, module_params, ezpackage, pdf

The operators

ezpreference

Generated page: ezpreference.

Parameter	Type	Required	Default
name	string	yes	false

- **Input:** none. **Returns:** the current user's stored preference of that name (a string), or `false`.

It calls `eZPreferences::value( $name )` (`kernel/classes/ezpreferences.php`): the value comes from the session when it was stored there for the logged-in user, otherwise from the `ezpreferences` table. For the anonymous user it is always `false`, without a query. Preferences are set by the `user/preferences` view (the admin's "remember this" switches), not by templates. The compiler turns the call into a direct `eZPreferences::value()` call (`preferencesTransformation()`).

Template

```
1 [{ezpreference( 'admin_navigation_content' )}]
2 {ezpreference( 'admin_list_limit' )|get_type}
```

Output:

```
1 []
2 boolean[false]
```

ezmodule

Generated page: ezmodule.

Parameter	Type	Required	Default
uri	string	no	false

- **Input:** none. **Returns:** `true` when the module is in `module.ini` `[ModuleSettings]` `ModuleList[]` and the siteaccess allows the view, `false` otherwise.

It does **not** check the user's policies. `eZModule::accessAllowed()` (`lib/ezutils/classes/ezmodule.php`) applies the siteaccess rules of `site.ini` `[SiteAccessRules]` `Rules[]` and, with `[ConfigurationSettings]` `ManagedByDeployment=enabled`, the views locked for deployment. That is why `setup/info` is allowed below for the anonymous user: whether the user may run it is decided by the view's own policy check. Use it to hide links to views a siteaccess switches off; use `fetch( 'user', 'has_access_to', ... )` for permissions.

module_params

Generated page: module_params.

- **Input:** none. **Parameters:** none.
- **Returns:** `$GLOBALS['eZRequestedModuleParams']`, a hash with `module_name`, `function_name` (the view) and `parameters` (the named view parameters) that `eZModule` stores when it runs a view, or `null` when no view ran (as on the command line, line 5 below).

Template

```
1 content/view: {ezmodule( 'content/view' )|choose( 'no', 'yes' )}
2 setup/info: {ezmodule( 'setup/info' )|choose( 'no', 'yes' )}
3 user/login: {ezmodule( 'user/login' )|choose( 'no', 'yes' )}
4 nosuchmodule/view: {ezmodule( 'nosuchmodule/view' )|choose( 'no', 'yes' )}
5 module_params: {module_params()|get_type}
```

Output:

```
1 content/view: yes
2 setup/info: yes
3 user/login: yes
4 nosuchmodule/view: no
5 module_params: null
```

ezpackage

Generated page: ezpackage.

Parameter	Type	Required	Default
class	string	yes	false
data	string	no	false

- **Input:** an `eZPackage` object (from `fetch( 'package', 'list' )` or `fetch( 'package', 'item' )`).
- **Returns:** a path, depending on `class`:

class	Returns
thumbnail	the path of the package's first file of type `thumbnail`, or `false`
filepath	the path of the file whose variable name is `data`; error "No filepath found" when there is none
fileitempath	the path of the file item given in `data`
documentpath	the path of the package document named `data`; error "No documentpath found" when there is none
dirpath	the package folder in its repository

Any other `class` is the error "Unknown package operator name". The paths are relative to the installation root.

Template

```
{def $packages = fetch( 'package', 'list', hash( 'offset', 0, 'limit', 3 ) )}
{foreach $packages as $p}
{$p.name}: thumbnail={$p|ezpackage( 'thumbnail' )|wash} dir={$p|ezpackage( 'dirpath' )|wash}

{/foreach}
```

Output:

```
slash_quotes: thumbnail= dir=var/storage/packages/local/slash_quotes
ezgmaplocation_extension: thumbnail= dir=var/storage/packages/ez-systems/ezgmaplocation_extension
ezstarrating_extension: thumbnail= dir=var/storage/packages/ez-systems/ezstarrating_extension
```

pdf

Generated page: pdf.

Parameter	Type	Required	Default
operation	string	yes	empty

Further positional parameters depend on the operation, mostly one hash of options.

`eZPDF` drives the PDF export (`content/pdf`, the `pdf` view templates such as `design/standard/templates/node/view/pdf.tpl`). Most operations only **return a markup command** for the PDF engine, which the template collects as text; a few act on the PDF object:

Operations	What they do
`toc`, `set_font`, `table`, `header`, `new_line/newline`, `new_page/newpage`, `image`, `anchor`, `link`, `strike`, `page_number/pageNumber`, `line`, `footer_block`, `header_block`, `footer`, `frame_header`, `frontpage`, `set_margin`, `keyword`, `create_index`, `ul`, `filled_circle`, `rectangle`, `filled_rectangle`, `text`, `text_box`, `text_frame`	return a command string such as `<C:callNewLine>`; defaults come from `pdf.ini`
`create`	creates the PDF object
`execute`	creates the PDF object (size and orientation from the options or `pdf.ini [PDFGeneral]`) and renders the collected text into it
`stream`	sends the PDF to the browser
`close`	writes the PDF to the file name given as the second parameter, through the cluster file handler

Not run on a development installation: `execute`, `stream` and `close` (they render a document, send output or write a file). The operations below only build command strings:

Template

```
1 {pdf( 'new_line' )}
2 {pdf( 'new_page' )}
3 {pdf( 'strike', 'old price' )}
4 {pdf( 'anchor', hash( 'name', 'top' ) )}
5 {pdf( 'link', hash( 'url', 'https://exponential.earth/', 'text', 'Exponential' ) )}
6 {pdf( 'toc', hash( 'size', array( 12, 10 ), 'dots', 'true' ) )}
```

Output:

```
1 <C:callNewLine>
2 <C:callNewPage><C:callNewLine>
3 <c:strike>old price</c:strike>
4 <C:callAnchor:top:FitH:>
5 <c:alink:https%3A%2F%2Fexponential.earth%2F>Exponential</c:alink>
6 <C:callTOC:size:12,10:dots:true>
```

Related pages

- PDF exports
- Role and policy operators
- All template operators (generated)
- Free documentation: this page is free to copy, translate and improve under the GNU FDL 1.3.

> **Become an Exponential Partner**
>
> Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and land new features faster.

Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.