zinit-install.zsh(1)

NAME

zinit-install.zsh - a shell script

SYNOPSIS

Documentation automatically generated with `zshelldoc'

FUNCTIONS

```
ziextract
.zinit-at-eval
.zinit-compile-plugin
.zinit-compinit
.zinit-download-file-stdout
.zinit-download-snippet
.zinit-extract
.zinit-forget-completion
.zinit-get-latest-gh-r-url-part
.zinit-get-package
.zinit-get-url-mtime
.zinit-install-completions
.zinit-mirror-using-svn
.zinit-parse-json
.zinit-setup-plugin-dir
.zinit-update-snippet
zpextract
AUTOLOAD compinit
```

DETAILS

Script Body

Has 3 line(s). No functions are called (may set up e.g. a hook, a Zle widget bound to a key, etc.).

Uses feature(s): *source*

ziextract

```
]]]
FUNCTION: ziextract [[[
If the file is an archive, it is extracted by this function.
Next stage is scanning of files with the common utility `file`,
to detect executables. They are given +x mode. There are also
messages to the user on performed actions.
```

```
$1 - url
$2 - file
```

Has 279 line(s). Calls functions:

```
ziextract
```

Uses feature(s): *setopt, unfunction, zparseopts*

Called by:

```
.zinit-extract
.zinit-get-package
.zinit-setup-plugin-dir
zpextract
```

zinit-at-eval

~~~~~~

```
]]]
FUNCTION: .zinit-at-eval [[[
```

Has 5 line(s). Calls functions:

```
.zinit-at-eval
`-- zinit.zsh/@zinit-substitute
```

Uses feature(s): *eval*

Called by:

```
.zinit-download-snippet
```

zinit-compile-plugin

~~~~~~

```
FUNCTION: .zinit-compile-plugin [[[
Compiles given plugin (its main source file, and also an
additional "....zsh" file if it exists).
```

```
$1 - plugin spec (4 formats: user---plugin, user/plugin, user, plugin)
$2 - plugin (only when $1 - i.e. user - given)
```

Has 60 line(s). Calls functions:

```
.zinit-compile-plugin
|-- zinit-side.zsh/.zinit-compute-ice
`-- zinit-side.zsh/.zinit-first
```

Uses feature(s): *eval, zcompile*

Called by:

```
.zinit-setup-plugin-dir
zinit-autoload.zsh/.zinit-compile-uncompile-all
zinit.zsh/zinit
```

zinit-compinit

~~~~

```
FUNCTION: .zinit-compinit [[[
User-exposed `compinit' frontend which first ensures that all
completions managed by Zinit are forgotten by Zshell. After
that it runs normal `compinit', which should more easily detect
Zinit's completions.
```

```
No arguments.
```

Has 22 line(s). Calls functions:

```
.zinit-compinit
`-- compinit
```

Uses feature(s): *autoload, compinit, setopt, unfunction*

Called by:

```
.zinit-install-completions
zinit-autoload.zsh/.zinit-uninstall-completions
zinit-autoload.zsh/.zinit-update-all-parallel
zinit.zsh/.zinit-prepare-home
zinit.zsh/zinit
```

zinit-download-file-stdout

~~~~~~~~

```
FUNCTION: .zinit-download-file-stdout [[[
Downloads file to stdout. Supports following backend commands:
curl, wget, lftp, lynx. Used by snippet loading.
```

Has 36 line(s). Calls functions:

```
.zinit-download-file-stdout
```

Uses feature(s): *setopt, trap, type*

Called by:

```
.zinit-download-snippet
.zinit-get-package
.zinit-setup-plugin-dir
```

zinit-download-snippet

~~~~~~~~~~

```
FUNCTION: .zinit-download-snippet [[[
Downloads snippet  either a file  with curl, wget, lftp or lynx,
or a directory, with Subversion  when svn-ICE is active. Github
supports Subversion protocol and allows to clone subdirectories.
This is used to provide a layer of support for Oh-My-Zsh and Prezto.
```

Has 360 line(s). Calls functions:

```
.zinit-download-snippet
|-- zinit-side.zsh/.zinit-countdown
|-- zinit-side.zsh/.zinit-store-ices
`-- zinit.zsh/@zinit-substitute
```

Uses feature(s): *eval, setopt, zcompile*

Called by:

```
.zinit-update-snippet
zinit.zsh/.zinit-load-snippet
```

zinit-extract

~~~~~~

```
]]]
FUNCTION: .zinit-extract() [[[
```

Has 28 line(s). Calls functions:

```
.zinit-extract
`-- ziextract
```

Uses feature(s): *setopt*

Called by:

```
.zinit-download-snippet
.zinit-setup-plugin-dir
zinit-autoload.zsh/.zinit-update-or-status
```

zinit-forget-completion

~~~~~~~~

```
]]]
FUNCTION: .zinit-forget-completion [[[
Implements alternation of Zsh state so that already initialized
completion stops being visible to Zsh.
```

```
$1 - completion function name, e.g. "_cp"; can also be "cp"
```

Has 20 line(s). Doesn't call other functions.

Uses feature(s): *setopt, unfunction*

Called by:

```
.zinit-compinit
.zinit-install-completions
zinit-autoload.zsh/.zinit-uninstall-completions
zinit.zsh/zinit
```

zinit-get-latest-gh-r-url-part

~~~~~~~~~~

```
]]]
FUNCTION: .zinit-get-latest-gh-r-url-part [[[
Gets version string of latest release of given Github
package. Connects to Github releases page.
```

Has 75 line(s). Doesn't call other functions.

Uses feature(s): *setopt*

Called by:

```
.zinit-setup-plugin-dir
zinit-autoload.zsh/.zinit-update-or-status
```

zinit-get-package

~~~~~

```
]]]
FUNCTION: .zinit-get-package [[[
```

Has 158 line(s). Calls functions:

```
.zinit-get-package
|-- ziextract
`-- zinit.zsh/@zinit-substitute
```

Uses feature(s): *eval, setopt*

Called by:

```
zinit.zsh/.zinit-load
```

Environment variables used: zinit.zsh → ZPFX

zinit-get-url-mtime

~~~~~~~~~~

```
FUNCTION: .zinit-get-url-mtime [[[
For the given URL returns the date in the Last-Modified
header as a time stamp
```

Has 35 line(s). Doesn't call other functions.

Uses feature(s): *read, setopt, trap, type*

Called by:

```
.zinit-download-snippet
```

zinit-install-completions

~~~~~~~~~~

```
FUNCTION: .zinit-install-completions [[[
Installs all completions of given plugin. After that they are
visible to `compinit'. Visible completions can be selectively
disabled and enabled. User can access completion data with
`clist' or `completions' subcommand.
```

```
$1 - plugin spec (4 formats: user---plugin, user/plugin, user, plugin)
$2 - plugin (only when $1 - i.e. user - given)
$3 - if 1, then reinstall, otherwise only install completions that aren't there
```

Has 55 line(s). Calls functions:

```
.zinit-install-completions
|-- zinit-side.zsh/.zinit-any-colorify-as-uspl2
|-- zinit-side.zsh/.zinit-exists-physically-message
`-- zinit.zsh/.zinit-any-to-user-plugin
```

Uses feature(s): *setopt*

Called by:

```
.zinit-download-snippet
.zinit-setup-plugin-dir
zinit.zsh/zinit
```

zinit-mirror-using-svn

~~~~~~~~~~

```
FUNCTION: .zinit-mirror-using-svn [[[
Used to clone subdirectories from Github. If in update mode
(see $2), then invokes `svn update', in normal mode invokes
`svn checkout --non-interactive -q <URL>'. In test mode only
compares remote and local revision and outputs true if update
is needed.
```

```
$1 - URL
$2 - mode, "" - normal, "-u" - update, "-t" - test
$3 - subdirectory (not path) with working copy, needed for -t and -u
```

Has 29 line(s). Doesn't call other functions.

Uses feature(s): *setopt*

Called by:

```
.zinit-download-snippet
```

zinit-parse-json

~~~~~~~~

```
FUNCTION: .zinit-parse-json [[[
Retrievies the ice-list from given profile from
the JSON of the package.json.
```

Has 107 line(s). Calls functions:

```
.zinit-parse-json
```

Uses feature(s): *setopt*

Called by:

```
.zinit-get-package
```

zinit-setup-plugin-dir

~~~~~~~~~~

```
]]]
FUNCTION: .zinit-setup-plugin-dir [[[
Clones given plugin into PLUGIN_DIR. Supports multiple
sites (respecting `from' and `proto' ice modifiers).
Invokes compilation of plugin's main file.
```

```
$1 - user
$2 - plugin
```

Has 229 line(s). Calls functions:

```
.zinit-setup-plugin-dir
|-- ziextract
|-- zinit-side.zsh/.zinit-any-colorify-as-uspl2
|-- zinit-side.zsh/.zinit-countdown
|-- zinit-side.zsh/.zinit-store-ices
|-- zinit.zsh/.zinit-get-object-path
`-- zinit.zsh/@zinit-substitute
```

Uses feature(s): *eval, setopt, trap*

Called by:

```
zinit-autoload.zsh/.zinit-update-or-status
zinit.zsh/.zinit-load
```

zinit-update-snippet

~~~~~~~

```
]]]
FUNCTION: .zinit-update-snippet [[[
```

Has 67 line(s). Calls functions:

```
.zinit-update-snippet
|-- zinit.zsh/.zinit-get-object-path
`-- zinit.zsh/.zinit-pack-ice
```

Uses feature(s): *eval, setopt*

Called by:

```
zinit-autoload.zsh/.zinit-update-or-status-snippet
```

zpextract

```
]]]
FUNCTION: zpextract [[[
```

Has 1 line(s). Calls functions:

```
zpextract
`-- ziextract
```

Not called by script or any function (may be e.g. a hook, a Zle widget, etc.).

compinit

```
Initialisation for new style completion. This mainly contains some helper
functions and setup. Everything else is split into different files that
will automatically be made autoloaded (see the end of this file).  The
names of the files that will be considered for autoloading are those that
begin with an underscores (like `_condition).
```

```
The first line of each of these files is read and must indicate what
should be done with its contents:
```

```
`#compdef <names ...>'
```

Has 549 line(s). Doesn't call other functions.

Uses feature(s): *autoload, bindkey, compdef, compdump, eval, read, setopt, unfunction, zle, zstyle*

Called by:

```
.zinit-compinit
```