

install.zsh(1)

NAME

install.zsh - a shell script

SYNOPSIS

Documentation automatically generated with `zsdoc'

FUNCTIONS

```
.zi-at-eval
.zi-compile-plugin
.zi-compinit
.zi-download-file-stdout
.zi-download-snippet
.zi-extract
.zi-forget-completion
.zi-get-cygwin-package
.zi-get-latest-gh-r-url-part
.zi-get-package
.zi-get-url-mtime
.zi-install-completions
.zi-mirror-using-svn
.zi-parse-json
.zi-setup-plugin-dir
.zi-update-snippet
zicp
ziextract
zimv
zpextract
❑zi-atclone-hook
❑zi-atpull-e-hook
❑zi-atpull-hook
❑zi-compile-plugin-hook
❑zi-cp-hook
❑zi-extract-hook
❑zi-make-e-hook
❑zi-make-ee-hook
❑zi-make-hook
❑zi-mv-hook
❑zi-ps-on-update-hook
❑zi-reset-hook
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*

zi-at-eval

~~~~~

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

Has 7 line(s). Calls functions:

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

Uses feature(s): *eval*

Called by:

```
□zi-atpull-e-hook
□zi-atpull-hook
```

*zi-compile-plugin*

~~~~~

```
FUNCTION: .zi-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 81 line(s). Calls functions:

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

```
`-- zi.zsh/+zi-message
```

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

Called by:

```
zi-compile-plugin-hook  
autoload.zsh/.zi-compile-uncompile-all  
zi.zsh/zi
```

zi-compinit

~~~~~

```
FUNCTION: .zi-compinit []  
User-exposed 'compinit' frontend which first ensures that all completions managed  
by Zi are forgotten by Z-shell.  
After that it runs normal 'compinit', which should more easily detect Zi  
completions.
```

No arguments.

Has 25 line(s). Calls functions:

```
.zi-compinit  
|-- compinit  
`-- zi.zsh/+zi-message
```

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

Called by:

```
.zi-install-completions  
autoload.zsh/.zi-uninstall-completions  
autoload.zsh/.zi-update-or-status-all  
zi.zsh/.zi-prepare-home  
zi.zsh/zi
```

*zi-download-file-stdout*

~~~~~

```
FUNCTION: .zi-download-file-stdout []  
Downloads file to stdout.  
Supports following backend commands: curl, wget, lftp, lynx.
```

Used by snippet loading.

Has 45 line(s). Calls functions:

```
.zi-download-file-stdout  
'-- zi.zsh/+zi-message
```

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

Called by:

```
.zi-download-snippet  
.zi-get-cygwin-package  
.zi-get-package  
.zi-setup-plugin-dir
```

zi-download-snippet

~~~~~

FUNCTION: `.zi-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 344 line(s). Calls functions:

```
.zi-download-snippet  
|-- side.zsh/.zi-store-ices  
'-- zi.zsh/+zi-message
```

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

Called by:

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

*zi-extract*

~~~~~

```
FUNCTION: .zi-extract() [[[
```

Has 22 line(s). Calls functions:

```
.zi-extract
|-- zi.zsh/+zi-message
`-- ziextract
    `-- zi.zsh/+zi-message
```

Uses feature(s): *setopt*

Called by:

```
zi-extract-hook
```

zi-forget-completion

~~~~~

```
]]]
FUNCTION: .zi-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:

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

*zi-get-cygwin-package*

~~~~~

```
FUNCTION: .zi-get-cygwin-package [[[
```

Has 70 line(s). Calls functions:

```
.zi-get-cygwin-package  
'-- zi.zsh/+zi-message
```

Uses feature(s): *setopt*

Called by:

```
.zi-setup-plugin-dir
```

zi-get-latest-gh-r-url-part

~~~~~

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

Has 109 line(s). Calls functions:

```
.zi-get-latest-gh-r-url-part  
'-- zi.zsh/+zi-message
```

Uses feature(s): *setopt*

Called by:

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

*zi-get-package*

~~~~~

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

Has 155 line(s). Calls functions:

```
.zi-get-package  
|-- zi.zsh/+zi-message  
|-- zi.zsh/@zi-substitute  
'-- ziextract
```

```
`-- zi.zsh/+zi-message
```

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

Called by:

```
zi.zsh/.zi-load
```

zi-get-url-mtime

~~~~~

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

Has 34 line(s). Doesn't call other functions.

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

Called by:

```
.zi-download-snippet
```

*zi-install-completions*

~~~~~

```
FUNCTION: .zi-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 59 line(s). Calls functions:

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

Uses feature(s): *setopt*

Called by:

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

zi-mirror-using-svn

~~~~~

```
FUNCTION: .zi-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 27 line(s). Calls functions:

```
.zi-mirror-using-svn  
'-- zi.zsh/+zi-message
```

Uses feature(s): *setopt*

Called by:

```
.zi-download-snippet
```

*zi-parse-json*

~~~~~

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

Has 102 line(s). Calls functions:


```
.zi-parse-json
```

Uses feature(s): *setopt*

Called by:

```
.zi-get-package
```

zi-setup-plugin-dir

~~~~~

```
FUNCTION: .zi-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 208 line(s). Calls functions:

```
.zi-setup-plugin-dir  
|-- side.zsh/.zi-any-colorify-as-uspl2  
|-- side.zsh/.zi-store-ices  
|-- zi.zsh/+zi-message  
|-- zi.zsh/.zi-get-object-path  
'-- ziextract  
    '-- zi.zsh/+zi-message
```

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

Called by:

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

*zi-update-snippet*

~~~~~

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

Has 71 line(s). Calls functions:

```
.zi-update-snippet
|-- zi.zsh/+zi-message
|-- zi.zsh/.zi-get-object-path
`-- zi.zsh/.zi-pack-ice
```

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

Called by:

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

zicp

```
]]]
FUNCTION zicp [[[
```

Has 27 line(s). Doesn't call other functions.

Uses feature(s): *setopt*

Called by:

```
zimv
```

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 265 line(s). Calls functions:

```
ziextract
```

```
`-- zi.zsh/+zi-message
```

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

Called by:

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

zimv

```
]]]  
FUNCTION zimv [[[
```

Has 3 line(s). Calls functions:

```
zimv  
`-- zicp
```

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

zpextract

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

Has 1 line(s). Calls functions:

```
zpextract  
`-- ziextract  
   `-- zi.zsh/+zi-message
```

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

∞zi-atclone-hook

```
FUNCTION: ∞zi-atclone-hook [[[
```

Has 18 line(s). Calls functions:

```
❏zi-atclone-hook  
|-- side.zsh/.zi-countdown  
'-- zi.zsh/@zi-substitute
```

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

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

∞zi-atpull-e-hook

```
FUNCTION: ❏zi-atpull-e-hook [[]
```

Has 17 line(s). Calls functions:

```
❏zi-atpull-e-hook  
'-- side.zsh/.zi-countdown
```

Uses feature(s): *setopt*

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

∞zi-atpull-hook

```
FUNCTION: ❏zi-atpull-hook [[]
```

Has 16 line(s). Calls functions:

```
❏zi-atpull-hook  
'-- side.zsh/.zi-countdown
```

Uses feature(s): *setopt*

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

∞zi-compile-plugin-hook

```
FUNCTION: ❏zi-compile-plugin-hook [[]
```

Has 15 line(s). Calls functions:

```
❏zi-compile-plugin-hook
```

Uses feature(s): *setopt*

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

∞zi-cp-hook

```
FUNCTION: ❏zi-cp-hook [][]
```

Has 25 line(s). Calls functions:

```
❏zi-cp-hook  
'-- zi.zsh/@zi-substitute
```

Uses feature(s): *setopt*

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

∞zi-extract-hook

```
FUNCTION: ❏zi-extract-hook [][]
```

Has 5 line(s). Calls functions:

```
❏zi-extract-hook  
'-- zi.zsh/@zi-substitute
```

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

∞zi-make-e-hook

```
FUNCTION: ❏zi-make-e-hook [][]
```

Has 7 line(s). Calls functions:

```
❏zi-make-e-hook  
|-- side.zsh/.zi-countdown  
'-- zi.zsh/@zi-substitute
```

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

∞zi-make-ee-hook

FUNCTION: `zi-make-ee-hook []`

Has 7 line(s). Calls functions:

```
zi-make-ee-hook
|-- side.zsh/.zi-countdown
`-- zi.zsh/@zi-substitute
```

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

∞zi-make-hook

FUNCTION: `zi-make-hook []`

Has 6 line(s). Calls functions:

```
zi-make-hook
|-- side.zsh/.zi-countdown
`-- zi.zsh/@zi-substitute
```

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

∞zi-mv-hook

FUNCTION: `zi-mv-hook []`

Has 26 line(s). Calls functions:

```
zi-mv-hook
|-- zi.zsh/+zi-message
`-- zi.zsh/@zi-substitute
```

Uses feature(s): *setopt*

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

∞zi-ps-on-update-hook

```
FUNCTION: ∞zi-ps-on-update-hook [][]
```

Has 14 line(s). Calls functions:

```
∞zi-ps-on-update-hook  
'-- zi.zsh/+zi-message
```

Uses feature(s): *eval*

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

∞zi-reset-hook

```
]]]  
FUNCTION: ∞zi-reset-opt-hook [][]
```

Has 73 line(s). Calls functions:

```
∞zi-reset-hook  
'-- zi.zsh/+zi-message
```

Uses feature(s): *eval*

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:

```
.zi-compinit
```