

zinit.zsh(1)

NAME

zinit.zsh - a shell script

SYNOPSIS

Documentation automatically generated with `zshelldoc`

FUNCTIONS

```
pmodload
zicdclear
zicdreplay
zicompdef
zicompinit
zinit
.zinit-add-fpath
.zinit-add-report
.zinit-any-to-user-plugin
.zinit-clear-debug-report
.zinit-compdef-clear
.zinit-compdef-replay
.zinit-debug-start
.zinit-debug-stop
.zinit-debug-unload
+zinit-deploy-message
.zinit-diff
.zinit-diff-env
.zinit-diff-functions
.zinit-diff-options
.zinit-diff-parameter
.zinit-find-other-matches
.zinit-get-mtime-into
.zinit-get-object-path
.zinit-ice
.zinit-load
.zinit-load-ices
.zinit-load-plugin
.zinit-load-snippet
+zinit-message
.zinit-pack-ice
.zinit-prepare-home
@zinit-register-annex
.zinit-register-plugin
```

```
:zinit-reload-and-run
.zinit-run
.zinit-run-task
-zinit_scheduler_add_sh
.zinit-service
.zinit-setup-params
:zinit-shade-alias
:zinit-shade-autoload
:zinit-shade-bindkey
:zinit-shade-compdef
.zinit-shade-off
.zinit-shade-on
:zinit-shade-zle
:zinit-shade-zstyle
.zinit-submit-turbo
@zinit-substitute
.zinit-wrap-track-functions
zpcdclear
zpcdreplay
zpcmpdef
zpcmpinit
zplugin
@zsh-plugin-run-on-unload
@zsh-plugin-run-on-update
AUTOLOAD add-zsh-hook
AUTOLOAD compinit
AUTOLOAD is-at-least
PRECMD-HOOK @zinit-scheduler
```

DETAILS

Script Body

Has 163 line(s). Calls functions:

```
Script-Body
|-- add-zsh-hook
|-- is-at-least
|-- zinit-autoload.zsh/.zinit-module
`-- +zinit-message
```

Uses feature(s): *add-zsh-hook*, *alias*, *autoload*, *export*, *is-at-least*, *setopt*, *source*, *zmodload*, *zstyle*

Exports (environment): *PMSPEC // ZPFX // ZSH_CACHE_DIR*

pmodload

```
FUNCTION: pmodload [[[
Compatibility with Prezto. Calls can be recursive.
```

Has 9 line(s). Calls functions:

```
pmodload
```

Uses feature(s): *zstyle*

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

zicdclear

```
]]]
FUNCTION: zicdclear [[[
A wrapper for 'zinit cdclear -q' which can be called from hook
ices like the atinit', atload'', etc. ices.
```

Has 1 line(s). Calls functions:

```
zicdclear
```

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

zicdreplay

```
FUNCTION: zicdreplay [[[
A function that can be invoked from within 'atinit', 'atload', etc.
ice-mod. It works like 'zinit cdreplay', which cannot be invoked
from such hook ices.
```

Has 1 line(s). Calls functions:

```
zicdreplay
```

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

zicompdef

```
]]]
FUNCTION: zicompdef [[[
Stores compdef for a replay with 'zicdreplay' (turbo mode) or
with 'zinit cdreplay' (normal mode). An utility functton of
an undefined use case.
```

Has 1 line(s). Doesn't call other functions.

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

zicompinit

```
]]]
FUNCTION: zicompinit [[[
A function that can be invoked from within 'atinit', 'atload', etc.
ice-mod. It runs 'autoload compinit; compinit' and respects
ZINIT[ZCOMPDUMP_PATH] and ZINIT[COMPINIT_OPTS].
```

Has 1 line(s). Calls functions:

```
zicompinit
'-- compinit
```

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

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

zinit

```
FUNCTION: zinit [[[
Main function directly exposed to user, obtains subcommand and its
arguments, has completion.
```

Has 432 line(s). Calls functions:

```
zinit
|-- compinit
|-- zinit-autoload.zsh/.zinit-cdisable
|-- zinit-autoload.zsh/.zinit-cenable
|-- zinit-autoload.zsh/.zinit-clear-completions
|-- zinit-autoload.zsh/.zinit-compiled
|-- zinit-autoload.zsh/.zinit-compile-uncompile-all
|-- zinit-autoload.zsh/.zinit-help
|-- zinit-autoload.zsh/.zinit-list-bindkeys
|-- zinit-autoload.zsh/.zinit-list-compdef-replay
|-- zinit-autoload.zsh/.zinit-ls
|-- zinit-autoload.zsh/.zinit-module
|-- zinit-autoload.zsh/.zinit-recently
|-- zinit-autoload.zsh/.zinit-search-completions
|-- zinit-autoload.zsh/.zinit-self-update
|-- zinit-autoload.zsh/.zinit-show-all-reports
|-- zinit-autoload.zsh/.zinit-show-completions
|-- zinit-autoload.zsh/.zinit-show-debug-report
|-- zinit-autoload.zsh/.zinit-show-registered-plugins
|-- zinit-autoload.zsh/.zinit-show-report
|-- zinit-autoload.zsh/.zinit-show-times
|-- zinit-autoload.zsh/.zinit-show-zstatus
|-- zinit-autoload.zsh/.zinit-uncompile-plugin
|-- zinit-autoload.zsh/.zinit-uninstall-completions
|-- zinit-autoload.zsh/.zinit-unload
|-- zinit-autoload.zsh/.zinit-update-or-status
|-- zinit-autoload.zsh/.zinit-update-or-status-all
|-- zinit-install.zsh/.zinit-compile-plugin
|-- zinit-install.zsh/.zinit-compinit
|-- zinit-install.zsh/.zinit-forget-completion
|-- zinit-install.zsh/.zinit-install-completions
`-- +zinit-message
```

Uses feature(s): *autoload, compinit, eval, setopt, source*

Called by:

```
zplugin
```

zinit-add-fpath

~~~~~

```
FUNCTION: .zinit-add-fpath [[]
```

Has 8 line(s). Calls functions:

```
.zinit-add-fpath
```

Called by:

```
zinit
```

*zinit-add-report*

~~~~~

```
FUNCTION: .zinit-add-report [[[  
Adds a report line for given plugin.
```

```
$1 - uspl2, i.e. user/plugin  
$2, ... - the text
```

Has 3 line(s). Doesn't call other functions.

Called by:

```
.zinit-load-plugin  
.zinit-load-snippet  
:zinit-shade-alias  
:zinit-shade-autoload  
:zinit-shade-bindkey  
:zinit-shade-compdef  
:zinit-shade-zle  
:zinit-shade-zstyle
```

zinit-any-to-user-plugin

~~~~~

```
FUNCTION: .zinit-any-to-user-plugin [[[  
Allows elastic plugin-spec across the code.
```

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

```
Returns user and plugin in $reply
```

Has 24 line(s). Doesn't call other functions.

Called by:

```
.zinit-add-fpath
.zinit-load
.zinit-run
zinit-autoload.zsh/.zinit-any-to-uspl2
zinit-autoload.zsh/.zinit-changes
zinit-autoload.zsh/.zinit-compiled
zinit-autoload.zsh/.zinit-compile-uncompile-all
zinit-autoload.zsh/.zinit-create
zinit-autoload.zsh/.zinit-delete
zinit-autoload.zsh/.zinit-find-completions-of-plugin
zinit-autoload.zsh/.zinit-get-path
zinit-autoload.zsh/.zinit-glance
zinit-autoload.zsh/.zinit-show-report
zinit-autoload.zsh/.zinit-stress
zinit-autoload.zsh/.zinit-uncompile-plugin
zinit-autoload.zsh/.zinit-unload
zinit-autoload.zsh/.zinit-unregister-plugin
zinit-autoload.zsh/.zinit-update-all-parallel
zinit-autoload.zsh/.zinit-update-or-status-all
zinit-autoload.zsh/.zinit-update-or-status
zinit-install.zsh/.zinit-install-completions
zinit-side.zsh/.zinit-any-colorify-as-uspl2
zinit-side.zsh/.zinit-compute-ice
zinit-side.zsh/.zinit-exists-physically
zinit-side.zsh/.zinit-first
```

*zinit-clear-debug-report*

~~~~~

```
FUNCTION: .zinit-clear-debug-report [[]
Forgets dtrace report gathered up to this moment.
```

Has 1 line(s). Calls functions:

```
.zinit-clear-debug-report
`-- zinit-autoload.zsh/.zinit-clear-report-for
```

Called by:

```
zinit
zinit-autoload.zsh/.zinit-unload
```

zinit-compdef-clear

~~~~~

```
FUNCTION: .zinit-compdef-clear []  
Implements user-exposed functionality to clear gathered compdefs.
```

Has 3 line(s). Calls functions:

```
.zinit-compdef-clear  
'-- +zinit-message
```

Called by:

```
zicdclear  
zinit  
zpcdclear
```

*zinit-compdef-replay*

~~~~~

```
FUNCTION: .zinit-compdef-replay []  
Runs gathered compdef calls. This allows to run 'compinit'  
after loading plugins.
```

Has 16 line(s). Calls functions:

```
.zinit-compdef-replay  
'-- +zinit-message
```

Uses feature(s): *compdef*

Called by:

```
zicdreplay  
zinit  
zpcdreplay
```

zinit-debug-start

~~~~~

```
FUNCTION: .zinit-debug-start []  
Starts Dtrace, i.e. session tracking for changes in Zsh state.
```



Has 9 line(s). Calls functions:

```
.zinit-debug-start  
'-- +zinit-message
```

Called by:

```
zinit
```

*zinit-debug-stop*

~~~~~

```
FUNCTION: .zinit-debug-stop [[]  
Stops Dtrace, i.e. session tracking for changes in Zsh state.
```

Has 3 line(s). Calls functions:

```
.zinit-debug-stop
```

Called by:

```
zinit
```

zinit-debug-unload

~~~~~

```
FUNCTION: .zinit-debug-unload [[]  
Reverts changes detected by dtrace run.
```

Has 6 line(s). Calls functions:

```
.zinit-debug-unload  
|-- zinit-autoload.zsh/.zinit-unload  
'-- +zinit-message
```

Uses feature(s): *source*

Called by:

```
zinit
```

## +zinit-deploy-message

```
]]  
FUNCTION: +zinit-deploy-message [[[  
Deploys a sub-prompt message to be displayed OR a `zle  
.reset-prompt' call to be invoked
```

Has 13 line(s). Doesn't call other functions.

Uses feature(s): *read, zle*

Called by:

```
.zinit-load-snippet  
.zinit-load  
zinit-autoload.zsh/.zinit-recall
```

## *zinit-diff*

~~~~

```
FUNCTION: .zinit-diff [[[  
Performs diff actions of all types
```

Has 4 line(s). Calls functions:

```
.zinit-diff
```

Called by:

```
.zinit-debug-start  
.zinit-debug-stop  
.zinit-load-plugin
```

zinit-diff-env

~~~~

```
FUNCTION: .zinit-diff-env [[[  
Implements detection of change in PATH and FPATH.
```

```
$1 - user/plugin (i.e. uspl2 format)  
$2 - command, can be "begin" or "end"
```

Has 18 line(s). Doesn't call other functions.

Called by:

```
.zinit-diff  
.zinit-load-plugin
```

### *zinit-diff-functions*

~~~~~

```
FUNCTION: .zinit-diff-functions [[]  
Implements detection of newly created functions. Performs  
data gathering, computation is done in *-compute().
```

```
$1 - user/plugin (i.e. uspl2 format)  
$2 - command, can be "begin" or "end"
```

Has 8 line(s). Doesn't call other functions.

Called by:

```
.zinit-diff
```

zinit-diff-options

~~~~~

```
FUNCTION: .zinit-diff-options [[]  
Implements detection of change in option state. Performs  
data gathering, computation is done in *-compute().
```

```
$1 - user/plugin (i.e. uspl2 format)  
$2 - command, can be "begin" or "end"
```

Has 7 line(s). Doesn't call other functions.

Called by:

```
.zinit-diff
```

### *zinit-diff-parameter*

~~~~~

```
FUNCTION: .zinit-diff-parameter [[[
Implements detection of change in any parameter's existence and type.
Performs data gathering, computation is done in *-compute().
```

```
$1 - user/plugin (i.e. uspl2 format)
$2 - command, can be "begin" or "end"
```

Has 9 line(s). Doesn't call other functions.

Called by:

```
.zinit-diff
```

zinit-find-other-matches

~~~~~

```
FUNCTION: .zinit-find-other-matches [[[
Plugin's main source file is in general `name.plugin.zsh'. However,
there can be different conventions, if that file is not found, then
this functions examines other conventions in the most sane order.
```

Has 17 line(s). Doesn't call other functions.

Called by:

```
.zinit-load-plugin
.zinit-load-snippet
zinit-side.zsh/.zinit-first
```

*zinit-get-mtime-into*

~~~~~

```
FUNCTION: .zinit-get-mtime-into [[[
```

Has 7 line(s). Doesn't call other functions.

Called by:

```
Script-Body
zinit-autoload.zsh/.zinit-self-update
zinit-autoload.zsh/.zinit-update-or-status-all
```

zinit-get-object-path

~~~~~

```
FUNCTION: .zinit-get-object-path [[]
```

Has 23 line(s). Doesn't call other functions.

Called by:

```
.zinit-load-ices  
.zinit-load-snippet  
.zinit-run  
zinit  
zinit-install.zsh/.zinit-setup-plugin-dir  
zinit-install.zsh/.zinit-update-snippet  
zinit-side.zsh/.zinit-two-paths
```

## *zinit-ice*

~~~~

```
FUNCTION: .zinit-ice [[]  
Parses ICE specification, puts the result into ZINIT_ICE global hash.  
The ice-spec is valid for next command only (i.e. it "melts"), but  
it can then stick to plugin and activate e.g. at update.
```

Has 13 line(s). Doesn't call other functions.

Uses feature(s): *setopt*

Called by:

```
zinit
```

Environment variables used: ZPFX

zinit-load

~~~~

```
FUNCTION: .zinit-load []  
Implements the exposed-to-user action of loading a plugin.
```

```
$1 - plugin spec (4 formats: user---plugin, user/plugin, user, plugin)  
$2 - plugin name, if the third format is used
```

Has 73 line(s). Calls functions:

```
.zinit-load  
|-- +zinit-deploy-message  
|-- zinit-install.zsh/.zinit-get-package  
`-- zinit-install.zsh/.zinit-setup-plugin-dir
```

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

Called by:

```
.zinit-run-task  
.zinit-service  
zinit
```

*zinit-load-ices*

~~~~~

```
FUNCTION: .zinit-load-ices []
```

Has 22 line(s). Calls functions:

```
.zinit-load-ices
```

Called by:

```
zinit
```

Environment variables used: ZPFEX

zinit-load-plugin

~~~~~

```
FUNCTION: .zinit-load-plugin [[[
Lower-level function for loading a plugin.
```

```
$1 - user
$2 - plugin
$3 - mode (light or load)
```

Has 117 line(s). Calls functions:

```
.zinit-load-plugin
```

Uses feature(s): *eval, setopt, source, unfunction, zle*

Called by:

```
.zinit-load
```

*zinit-load-snippet*

~~~~~

```
FUNCTION: .zinit-load-snippet [[[
Implements the exposed-to-user action of loading a snippet.
```

```
$1 - url (can be local, absolute path)
```

Has 192 line(s). Calls functions:

```
.zinit-load-snippet
|-- +zinit-deploy-message
|-- zinit-install.zsh/.zinit-download-snippet
`-- +zinit-message
```

Uses feature(s): *autoload, eval, setopt, source, unfunction, zparseopts, zstyle*

Called by:

```
pmodload
.zinit-load
.zinit-run-task
.zinit-service
zinit
```

+zinit-message

```
]]]
FUNCTION: +zinit-message [[[
```

Has 4 line(s). Doesn't call other functions.

Called by:

```
Script-Body
.zinit-compdef-clear
.zinit-compdef-replay
.zinit-debug-start
.zinit-debug-unload
.zinit-load-snippet
.zinit-run
zinit
zinit-autoload.zsh/.zinit-build-module
```

zinit-pack-ice

~~~~~

```
FUNCTION: .zinit-pack-ice [[[
Remembers all ice-mods, assigns them to concrete plugin. Ice spec
is in general forgotten for second-next command (that's why it's
called "ice" - it melts), however they glue to the object (plugin
or snippet) mentioned in the next command □ for later use with e.g.
`zinit update ...'
```

Has 3 line(s). Doesn't call other functions.

Called by:



```
.zinit-load-snippet
.zinit-load
@zsh-plugin-run-on-unload
@zsh-plugin-run-on-update
zinit-install.zsh/.zinit-update-snippet
zinit-side.zsh/.zinit-compute-ice
```

*zinit-prepare-home*

~~~~~

```
FUNCTION: .zinit-prepare-home [[]
Creates all directories needed by Zinit, first checks if they
already exist.
```

Has 37 line(s). Calls functions:

```
.zinit-prepare-home
|-- zinit-autoload.zsh/.zinit-clear-completions
`-- zinit-install.zsh/.zinit-compinit
```

Uses feature(s): *source*

Called by:

```
Script-Body
```

Environment variables used: ZPFX

@zinit-register-annex

```
]]]
FUNCTION: @zinit-register-z-annex [[]
Registers the z-annex inside Zinit □ i.e. an Zinit extension
```

Has 4 line(s). Doesn't call other functions.

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

zinit-register-plugin

~~~~~

```
FUNCTION: .zinit-register-plugin [[[
Adds the plugin to ZINIT_REGISTERED_PLUGINS array and to the
zsh_loaded_plugins array (managed according to the plugin standard:
http://zdharma.org/Zsh-100-Commits-Club/Zsh-Plugin-Standard.html)
```

Has 23 line(s). Doesn't call other functions.

Called by:

```
.zinit-load
```

## :zinit-reload-and-run

```
FUNCTION: :zinit-reload-and-run [[[
Marks given function ($3) for autoloading, and executes it triggering the
load. $1 is the fpath dedicated to the function, $2 are autoload options.
This function replaces "autoload -X", because using that on older Zsh
versions causes problems with traps.
```

So basically one creates function stub that calls :zinit-reload-and-run() instead of "autoload -X".

```
$1 - FPATH dedicated to function
$2 - autoload options
$3 - function name (one that needs autoloading)
```

Author: Bart Schaefer

Has 11 line(s). Doesn't call other functions.

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

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

*zinit-run*

~~~

```
]]]
FUNCTION: .zinit-run [[[
Run code inside plugin's folder
It uses the 'correct' parameter from upper's scope zinit()
```

Has 23 line(s). Calls functions:

```
.zinit-run  
'-- +zinit-message
```

Uses feature(s): *eval, setopt*

Called by:

```
zinit
```

zinit-run-task

~~~~~

```
]]]  
FUNCTION: .zinit-run-task [[[  
A backend, worker function of .zinit-scheduler. It obtains the tasks  
index and a few of its properties (like the type: plugin, snippet,  
service plugin, service snippet) and executes it first checking for  
additional conditions (like non-numeric wait'' ice).
```

```
$1 - the pass number, either 1st or 2nd pass  
$2 - the time assigned to the task  
$3 - type: plugin, snippet, service plugin, service snippet  
$4 - task's index in the ZINIT[WAIT_ICE...] fields  
$5 - mode: load or light  
$6 - the plugin-spec or snippet URL or alias name (from id-as'')
```

Has 44 line(s). Calls functions:

```
.zinit-run-task  
'-- zinit-autoload.zsh/.zinit-unload
```

Uses feature(s): *eval, source, zle, zpty*

Called by:

```
@zinit-scheduler
```

## @zinit-scheduler

```
]]
```

```
FUNCTION: @zinit-scheduler [[[
```

Searches for timeout tasks, executes them. There's an array of tasks waiting for execution, this scheduler manages them, detects which ones should be run at current moment, decides to remove (or not) them from the array after execution.

\$1 - if "following", then it is non-first (second and more) invocation of the scheduler; this results in chain of 'sched' invocations that results in repetitive @zinit-scheduler activity

if "burst", then all tasks are marked timeout and executed one by one; this is handy if e.g. a docker image starts up and needs to install all turbo-mode plugins without any hesitation (delay), i.e. "burst" allows to run package installations from script, not from prompt

Has 75 line(s). **Is a precmd hook.** Calls functions:

```
@zinit-scheduler  
'-- add-zsh-hook
```

Uses feature(s): *add-zsh-hook*, *sched*, *setopt*, *zle*

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

## **-zinit\_scheduler\_add\_sh**

```
]]
```

```
FUNCTION: -zinit_scheduler_add_sh [[[
```

Copies task into ZINIT\_RUN array, called when a task timeouts. A small function ran from pattern in /-substitution as a math function.

Has 7 line(s). Doesn't call other functions.

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

*zinit-service*

~~~~~

```
FUNCTION: .zinit-service [[[
Handles given service, i.e. obtains lock, runs it, or waits if no lock
```

```
$1 - type "p" or "s" (plugin or snippet)
$2 - mode - for plugin (light or load)
$3 - id - URL or plugin ID or alias name (from id-as''')
```

Has 33 line(s). Calls functions:

```
.zinit-service
```

Uses feature(s): *kill, read, setopt*

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

zinit-setup-params

~~~~~

```
]]]
FUNCTION: .zinit-setup-params [[[
```

Has 3 line(s). Doesn't call other functions.

Called by:

```
.zinit-load-snippet
.zinit-load
```

## **:zinit-shade-alias**

```
FUNCTION: :zinit-shade-alias [[[
Function defined to hijack plugin's calls to 'alias' builtin.
```

```
The hijacking is to gather report data (which is used in unload).
```

Has 36 line(s). Calls functions:

```
:zinit-shade-alias
```

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

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

## :zinit-shade-autoload

```
FUNCTION: :zinit-shade-autoload [[]  
Function defined to hijack plugin's calls to 'autoload' builtin.
```

The hijacking is not only to gather report data, but also to run custom 'autoload' function, that doesn't need FPATH.

Has 58 line(s). Calls functions:

```
:zinit-shade-autoload
```

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

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

## :zinit-shade-bindkey

```
FUNCTION: :zinit-shade-bindkey [[]  
Function defined to hijack plugin's calls to 'bindkey' builtin.
```

The hijacking is to gather report data (which is used in unload).

Has 120 line(s). Calls functions:

```
:zinit-shade-bindkey  
'-- is-at-least
```

Uses feature(s): *bindkey*, *is-at-least*, *setopt*, *zparseopts*

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

## :zinit-shade-compdef

```
FUNCTION: :zinit-shade-compdef [[]  
Function defined to hijack plugin's calls to 'compdef' function.  
The hijacking is not only for reporting, but also to save compdef  
calls so that 'compinit' can be called after loading plugins.
```

Has 6 line(s). Calls functions:

```
:zinit-shade-compdef
```

Uses feature(s): *setopt*

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

*zinit-shade-off*

~~~~~

```
FUNCTION: .zinit-shade-off [[[
Turn off shadeing completely for a given mode ("load", "light",
"light-b" (i.e. the 'trackbinds' mode) or "compdef").
```

Has 19 line(s). Doesn't call other functions.

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

Called by:

```
.zinit-debug-stop
.zinit-load-plugin
```

zinit-shade-on

~~~~~

```
FUNCTION: .zinit-shade-on [[[
Turn on shadeing of builtins and functions according to passed
mode ("load", "light", "light-b" or "compdef"). The shadeing is
to gather report data, and to hijack 'autoload', 'bindkey' and
'compdef' calls.
```

Has 25 line(s). Doesn't call other functions.

Called by:

```
.zinit-debug-start
.zinit-load-plugin
```

## **:zinit-shade-zle**

```
FUNCTION: :zinit-shade-zle [[[
Function defined to hijack plugin's calls to `zle' builtin.
```

```
The hijacking is to gather report data (which is used in unload).
```

Has 36 line(s). Calls functions:

```
:zinit-shade-zle
```

Uses feature(s): *setopt, zle*

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

## **:zinit-shade-zstyle**

```
FUNCTION: :zinit-shade-zstyle [[[
Function defined to hijack plugin's calls to `zstyle' builtin.
```

```
The hijacking is to gather report data (which is used in unload).
```

Has 23 line(s). Calls functions:

```
:zinit-shade-zstyle
```

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

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

*zinit-submit-turbo*

~~~~~

```
FUNCTION: .zinit-submit-turbo [[[
If `zinit load`, `zinit light` or `zinit snippet` will be
preceded with `wait`, `load`, `unload` or `on-update-of`/`subscribe`
ice-mods then the plugin or snippet is to be loaded in turbo-mode,
and this function adds it to internal data structures, so that
@zinit-scheduler can run (load, unload) this as a task.
```

Has 16 line(s). Doesn't call other functions.

Called by:


```
zinit
```

@zinit-substitute

```
]]]  
FUNCTION: @zinit-substitute [[[
```

Has 40 line(s). Doesn't call other functions.

Uses feature(s): *setopt*

Called by:

```
zinit-autoload.zsh/.zinit-at-eval  
zinit-autoload.zsh/.zinit-update-or-status  
zinit-install.zsh/.zinit-at-eval  
zinit-install.zsh/.zinit-download-snippet  
zinit-install.zsh/.zinit-get-package  
zinit-install.zsh/.zinit-setup-plugin-dir
```

Environment variables used: ZPFX

zinit-wrap-track-functions

~~~~~

```
]]]  
FUNCTION: .zinit-wrap-track-functions [[[
```

Has 19 line(s). Doesn't call other functions.

Uses feature(s): *eval*

Called by:

```
.zinit-load-plugin  
.zinit-load-snippet
```

## zpcdclear

Has 1 line(s). Calls functions:

```
zpcdclear
```

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

## zpcdreplay

Has 1 line(s). Calls functions:

```
zpcdreplay
```

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

## zpcmpdef

Has 1 line(s). Doesn't call other functions.

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

## zpcmpinit

Has 1 line(s). Calls functions:

```
zpcmpinit  
'-- compinit
```

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

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

## zplugin

```
Compatibility functions [[[
```

Has 1 line(s). Calls functions:

```

zplugin
`-- zinit
    |-- compinit
    |-- zinit-autoload.zsh/.zinit-cdisable
    |-- zinit-autoload.zsh/.zinit-cenable
    |-- zinit-autoload.zsh/.zinit-clear-completions
    |-- zinit-autoload.zsh/.zinit-compiled
    |-- zinit-autoload.zsh/.zinit-compile-uncompile-all
    |-- zinit-autoload.zsh/.zinit-help
    |-- zinit-autoload.zsh/.zinit-list-bindkeys
    |-- zinit-autoload.zsh/.zinit-list-compdef-replay
    |-- zinit-autoload.zsh/.zinit-ls
    |-- zinit-autoload.zsh/.zinit-module
    |-- zinit-autoload.zsh/.zinit-recently
    |-- zinit-autoload.zsh/.zinit-search-completions
    |-- zinit-autoload.zsh/.zinit-self-update
    |-- zinit-autoload.zsh/.zinit-show-all-reports
    |-- zinit-autoload.zsh/.zinit-show-completions
    |-- zinit-autoload.zsh/.zinit-show-debug-report
    |-- zinit-autoload.zsh/.zinit-show-registered-plugins
    |-- zinit-autoload.zsh/.zinit-show-report
    |-- zinit-autoload.zsh/.zinit-show-times
    |-- zinit-autoload.zsh/.zinit-show-zstatus
    |-- zinit-autoload.zsh/.zinit-uncompile-plugin
    |-- zinit-autoload.zsh/.zinit-uninstall-completions
    |-- zinit-autoload.zsh/.zinit-unload
    |-- zinit-autoload.zsh/.zinit-update-or-status
    |-- zinit-autoload.zsh/.zinit-update-or-status-all
    |-- zinit-install.zsh/.zinit-compile-plugin
    |-- zinit-install.zsh/.zinit-compinit
    |-- zinit-install.zsh/.zinit-forget-completion
    |-- zinit-install.zsh/.zinit-install-completions
`-- +zinit-message

```

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

## @zsh-plugin-run-on-unload

```

]]]
FUNCTION: @zsh-plugin-run-on-update [[[
The Plugin Standard required mechanism, see:
http://zdharm.org/Zsh-100-Commits-Club/Zsh-Plugin-Standard.html

```

Has 2 line(s). Calls functions:

```
@zsh-plugin-run-on-unload
```

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

## @zsh-plugin-run-on-update

```
]]]
FUNCTION: @zsh-plugin-run-on-update [[[
The Plugin Standard required mechanism
```

Has 2 line(s). Calls functions:

```
@zsh-plugin-run-on-update
```

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

## add-zsh-hook

```
Add to HOOK the given FUNCTION.
HOOK is one of chpwd, precmd, preexec, periodic, zshaddhistory,
zshexit, zsh_directory_name (the _functions subscript is not required).
```

```
With -d, remove the function from the hook instead; delete the hook
variable if it is empty.
```

```
-D behaves like -d, but pattern characters are active in the
function name, so any matching function will be deleted from the hook.
```

Has 93 line(s). Doesn't call other functions.

Uses feature(s): *autoload*, *getopts*

Called by:

```
Script-Body
@zinit-scheduler
```

## 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:

```
zcompinit  
zinit  
zpcompinit
```

## is-at-least

Test whether `$ZSH_VERSION` (or some value of your choice, if a second argument is provided) is greater than or equal to `x.y.z-r` (in argument one). In fact, it'll accept any dot/dash-separated string of numbers as its second argument and compare it to the dot/dash-separated first argument. Leading non-number parts of a segment (such as the "zefram" in 3.1.2-zefram4) are not considered when the comparison is done; only the numbers matter. Any left-out segments in the first argument that are present in the version string compared are considered as zeroes, eg `3 == 3.0 == 3.0.0 == 3.0.0.0` and so on.

Has 56 line(s). Doesn't call other functions.

Called by:

```
Script-Body  
:zinit-shade-bindkey
```