

# Bashmatic Usage Docs (v1.9.2)

# Table of Contents

|  |    |
|--|----|
| 1. File <b>lib/array.sh</b> .....        | 2  |
| 1.1. array.has-element() .....           | 2  |
| 1.1.1. Example .....                     | 2  |
| 1.2. array.includes() .....              | 2  |
| 1.3. array.join() .....                  | 2  |
| 1.3.1. Example .....                     | 2  |
| 1.4. array.sort() .....                  | 2  |
| 1.4.1. Example .....                     | 3  |
| 1.5. array.sort-numeric() .....          | 3  |
| 1.5.1. Example .....                     | 3  |
| 1.6. array.min() .....                   | 3  |
| 1.6.1. Example .....                     | 3  |
| 1.7. array.max() .....                   | 3  |
| 1.7.1. Example .....                     | 3  |
| 1.8. array.uniq() .....                  | 3  |
| 1.8.1. Example .....                     | 4  |
| 2. File <b>lib/asciidoc.sh</b> .....     | 5  |
| asciidoc .....                           | 6  |
| 3. Overview .....                        | 7  |
| 3.1. asciidoc.rouge-themes() .....       | 7  |
| 4. File <b>lib/output-utils.sh</b> ..... | 8  |
| 4.1. is-dbg() .....                      | 8  |
| 4.2. dbg() .....                         | 8  |
| 5. File <b>lib/output.sh</b> .....       | 9  |
| 5.1. section() .....                     | 9  |
| 5.1.1. Arguments .....                   | 9  |
| 6. File <b>lib/path.sh</b> .....         | 10 |
| 6.1. path.add() .....                    | 10 |
| 6.2. path.append() .....                 | 10 |
| 6.3. PATH_add() .....                    | 10 |
| 7. File <b>lib/osx.sh</b> .....          | 11 |
| osx.sh .....                             | 12 |
| 8. Overview .....                        | 13 |
| 8.1. osx.app.is-installed() .....        | 13 |
| 8.1.1. Example .....                     | 13 |
| 8.1.2. Arguments .....                   | 13 |
| 8.1.3. Exit codes .....                  | 13 |
| 9. File <b>lib/db.sh</b> .....           | 14 |
| 9.1. db.config.parse() .....             | 14 |
| 9.1.1. Example .....                     | 14 |

|   |    |
|---|----|
| 9.2. db.psql.connect()                    | 14 |
| 9.2.1. Example                            | 14 |
| 9.3. db.psql.connect.just-data()          | 14 |
| 9.3.1. Example                            | 14 |
| 9.4. db.psql.connect.table-settings-set() | 15 |
| 9.4.1. Example                            | 15 |
| 9.5. db.psql.db-settings()                | 15 |
| 9.5.1. Example                            | 15 |
| 9.6. db.psql.connect.db-settings-pretty() | 15 |
| 9.6.1. Example                            | 15 |
| 9.6.2. Arguments                          | 15 |
| 9.7. db.psql.connect.db-settings-toml()   | 15 |
| 9.7.1. Example                            | 15 |
| 9.7.2. Arguments                          | 15 |
| 10. File <b>lib/shdoc.sh</b>              | 16 |
| lib/shdoc.sh                              | 17 |
| 11. Overview                              | 18 |
| 11.1. gawk.install()                      | 18 |
| 12. File <b>lib/git.sh</b>                | 19 |
| 12.1. git.cfgu()                          | 19 |
| 12.1.1. Example                           | 19 |
| 12.2. git.open()                          | 19 |
| 12.2.1. Example                           | 19 |
| 12.2.2. Arguments                         | 19 |
| 13. File <b>lib/shasum.sh</b>             | 20 |
| shasum.sh                                 | 21 |
| 14. Overview                              | 22 |
| 14.1. shasum.set-command()                | 22 |
| 14.2. shasum.set-algo()                   | 22 |
| 14.2.1. Example                           | 22 |
| 14.3. shasum.sha()                        | 22 |
| 14.4. shasum.sha-only()                   | 22 |
| 14.5. shasum.sha-only-stdin()             | 22 |
| 14.6. shasum.to-hash()                    | 23 |
| 14.6.1. Example                           | 23 |
| 14.7. shasum.all-files()                  | 23 |
| 14.7.1. Example                           | 23 |
| 14.8. shasum.all-files-in-dir()           | 23 |
| 14.8.1. Example                           | 23 |
| 15. File <b>lib/pg.sh</b>                 | 24 |
| 15.1. pg.is-running()                     | 24 |
| 15.2. pg.running.server-binaries()        | 24 |
| 15.3. pg.running.data-dirs()              | 24 |

|   |    |
|---|----|
| 15.4. pg.server-in-path.version()         | 24 |
| 16. File <b>lib/dir.sh</b>                | 25 |
| 16.1. dir.short-home()                    | 25 |
| 17. File <b>lib/is.sh</b>                 | 26 |
| is.sh                                     | 27 |
| 18. Overview                              | 28 |
| 18.1. __is.validation.error()             | 28 |
| 18.1.1. Arguments                         | 28 |
| 18.1.2. Exit codes                        | 28 |
| 18.2. is-validations()                    | 28 |
| 18.3. __is.validation.ignore-error()      | 28 |
| 18.4. __is.validation.report-error()      | 28 |
| 18.5. whenever()                          | 28 |
| 18.5.1. Example                           | 28 |
| 19. File <b>lib/util.sh</b>               | 30 |
| util.sh                                   | 31 |
| 20. Overview                              | 32 |
| 20.1. util.rot13-stdin()                  | 32 |
| 20.1.1. Example                           | 32 |
| 21. File <b>lib/pdf.sh</b>                | 33 |
| Bashmatic Utilities for PDF file handling | 34 |
| 22. Overview                              | 35 |
| 22.1. pdf.combine()                       | 35 |
| 22.1.1. Example                           | 35 |
| 22.1.2. Arguments                         | 35 |
| 23. File <b>bin/install-direnv</b>        | 36 |
| install-direnv                            | 37 |
| 24. Overview                              | 38 |
| 24.1. direnv.register()                   | 38 |
| 25. File <b>bin/regen-usage-docs</b>      | 39 |
| regen-usage-docs                          | 40 |
| 26. Overview                              | 41 |
| 27. File <b>bin/specs</b>                 | 42 |
| 27.1. specs.init()                        | 42 |
| 27.2. specs.determine-test-filename()     | 42 |
| 28. File <b>bin/pdf-reduce</b>            | 43 |
| 28.1. pdf.do.shrink()                     | 43 |
| 29. Copyright & License                   | 44 |

NOTICE: [shdoc](#) documentation is auto-extracted from the Bashmatic Sources.

# Chapter 1. File `lib/array.sh`

- `array.has-element()`
- `array.includes()`
- `array.join()`
- `array.sort()`
- `array.sort-numeric()`
- `array.min()`
- `array.max()`
- `array.uniq()`

## 1.1. `array.has-element()`

Returns "true" if the first argument is a member of the array passed as the second argument:

### 1.1.1. Example

```
$ declare -a array=("a string" test2000 moo)
if [[ $(array.has-element "a string" "${array[@]}") == "true" ]]; then
    ...
fi
```

## 1.2. `array.includes()`

Similar to `array.has-elements`, but does not print anything, just returns 0 if includes, 1 if not.

## 1.3. `array.join()`

Joins a given array with a custom character

### 1.3.1. Example

```
$ declare -a array=(one two three)
$ array.join "," "${array[@]}"
one,two,three
```

## 1.4. `array.sort()`

Sorts the array alphanumerically and prints it to STDOUT

## 14.1 Example

```
declare -a unsorted=(hello begin again again)
local sorted="$(array.sort "${unsorted[@]}")"
```

## 15. array.sort-numeric()

Sorts the array numerically and prints it to STDOUT

### 15.1 Example

```
declare -a unsorted=(1 2 34 45 6)
local sorted="$(array.sort-numeric "${unsorted[@]}")"
```

## 16. array.min()

Returns a minimum integer from an array. Non-numeric elements are ignored and skipped over. Negative numbers are supported, but non-integers are not.

### 16.1 Example

```
$ declare -a array=(10 20 30 -5 5)
$ array.min "," "${array[@]}"
-5
```

## 17. array.max()

Returns a maximum integer from an array. Non-numeric elements are ignored and skipped over. Negative numbers are supported, but non-integers are not.

### 17.1 Example

```
$ declare -a array=(10 20 30 -5 5)
$ array.min "," "${array[@]}"
30
```

## 18. array.uniq()

Sorts and uniqs the array and prints it to STDOUT

## 18.1 Example

```
declare -a unsorted=(hello hello hello goodbye)
local uniqued="$(array.sort-numeric "${unsorted[@]}")"
```

---



## Chapter 2. File `lib/asciidoc.sh`

# asciidoc

# Chapter 3. Overview

Provides helper functions for dealing with asciidoc format.

- [asciidoc.rouge-themes\(\)](#)

## 3.1. `asciidoc.rouge-themes()`

Installs gem "rouge" and prints all available themes

---

## Chapter 4. File `lib/output-utils.sh`

- `is-dbg()`
- `dbg()`

### 4.1 `is-dbg()`

Checks if we have debug mode enabled

### 4.2 `dbg()`

Local debugging helper, activate it with `DEBUG=1`

---

# Chapter 5. File `lib/output.sh`

- `section()`

## 5.1 `section()`

Prints a "arrow-like" line using powerline characters

### 5.1.1 Arguments

- `@arg1` Width (optional) ~~2-2~~ only interpreted as width if the first argument is a number.
  - `@args` Text to print
-

## Chapter 6. File `lib/path.sh`

- `path.add()`
- `path.append()`
- `PATH_add()`

### 6.1 `path.add()`

Adds valid directories to those in the `PATH` and prints to the output. DOES NOT MODIFY `$PATH`

### 6.2 `path.append()`

Appends valid directories to those in the `PATH`, and exports the new value of the `PATH`

### 6.3 `PATH_add()`

This function exists within `direnv`, but since we are sourcing in `.envrc` we need to have this defined to avoid errors.

---

## Chapter 7. File `lib/osx.sh`

**osx.sh**



# Chapter 8. Overview

OSX Specific Helpers and Utilities

- `osx.app.is-installed()`

## 8.1. `osx.app.is-installed()`

Checks if a given parameter matches any of the installed applications under `/Applications` and `~/Applications`

By the default prints the matched application. Pass `-q` as a second argument to disable output.

### 8.1.1 Example

```
> osx.app.is-installed safari
Safari.app
> osx.app.is-installed safari -q && echo installed
installed
> osx.app.is-installed microsoft -c
6
```

### 8.1.2. Arguments

- `$1(a)`: string value to match (case insentively) for an app name
- `$2..`: additional arguments to the last invocation of `grep`

### 8.1.3. Exit codes

- `0`: if match was found
- `1`: if not

# Chapter 9. File `lib/db.sh`

- `db.config.parse()`
- `db.psql.connect()`
- `db.psql.connect.just-data()`
- `db.psql.connect.table-settings-set()`
- `db.psql.db-settings()`
- `db.psql.connect.db-settings-pretty()`
- `db.psql.connect.db-settings-toml()`

## 9.1. `db.config.parse()`

Returns a space-separated values of db host, db name, username and password

### 9.1.1 Example

```
db.config.set-file ~/.db/database.yml
db.config.parse development
#=> hostname dbname dbuser dbpass
declare -a params=($(db.config.parse development))
echo ${params[0]} # host
```

## 9.2. `db.psql.connect()`

Connect to one of the databases named in the YAML file, and optionally pass additional arguments to psql. Informational messages are sent to STDERR.

### 9.2.1 Example

```
db.psql.connect production
db.psql.connect production -c 'show all'
```

## 9.3. `db.psql.connect.just-data()`

Similar to the `db.psql.connect`, but outputs just the raw data with no headers.

### 9.3.1 Example

```
db.psql.connect.just-data production -c 'select datname from pg_database;'
```

## 9.4. db.psql.connect.table-settings-set()

Set per-table settings, such as autovacuum, eg:

### 9.4.1 Example

```
db.psql.connect.table-settings-set prod users autovacuum_analyze_threshold 1000000
db.psql.connect.table-settings-set prod users autovacuum_analyze_scale_factor 0
```

## 9.5. db.psql.db-settings()

Print out PostgreSQL settings for a connection specified by args

### 9.5.1 Example

```
db.psql.db-settings -h localhost -U postgres appdb
```

## 9.6. db.psql.connect.db-settings-pretty()

Print out PostgreSQL settings for a named connection

### 9.6.1 Example

```
db.psql.connect.db-settings-pretty primary
```

### 9.6.2 Arguments

- @arg1 dbname database entry name in ~/.db/database.yml

## 9.7. db.psql.connect.db-settings-toml()

Print out PostgreSQL settings for a named connection using TOML/ini format.

### 9.7.1 Example

```
db.psql.connect.db-settings-toml primary > primary.ini
```

### 9.7.2 Arguments

- @arg1 dbname database entry name in ~/.db/database.yml

## Chapter 10. File `lib/shdoc.sh`

# lib/shdoc.sh

Helpers to install gawk and shdoc properly.0

# Chapter 11. Overview

see `${BASHMATIC_HOME}/lib/shdoc.md` for an example of how to use SHDOC. and also [project's github page](#).

- `gawk.install()`

## 11.1 `gawk.install()`

Installs gawk into `/usr/local/bin/gawk`

---

## Chapter 12. File `lib/git.sh`

- `git.cfgu()`
- `git.open()`

### 12.1 `git.cfgu()`

Sets or gets user values from global gitconfig.

#### 12.1.1 Example

```
git.cfgu email
git.cfgu email kigster@gmail.com
git.cfgu
```

### 12.2 `git.open()`

Reads the remote of a repo by name provided as an argument (or defaults to "origin") and opens it in the browser.

#### 12.2.1 Example

```
git clone git@github.com:kigster/bashmatic.git
cd bashmatic
source init.sh
git.open
git.open origin # same thing
```

#### 12.2.2 Arguments

- `$1` (optional): name of the remote to open, defaults to "origin"

## Chapter 13. File `lib/shasum.sh`



# shasum.sh

SHA Functions

# Chapter 14. Overview

SHASUM related functions, that compute SHA for a single file, collection of files, or entire directories.

- `shasum.set-command()`
- `shasum.set-algo()`
- `shasum.sha()`
- `shasum.sha-only()`
- `shasum.sha-only-stdin()`
- `shasum.to-hash()`
- `shasum.all-files()`
- `shasum.all-files-in-dir()`

## 14.1. `shasum.set-command()`

Override the default SHA command and algorithm Default is `shasum -a 256`

## 14.2. `shasum.set-algo()`

Override the default SHA algorithm

### 14.2.1 Example

```
$ shasum.set-algo 256
```

## 14.3. `shasum.sha()`

Compute SHA for all given files, ignore STDERR NOTE: first few arguments will be passed to the `shasum` command, or whatever you set via `shasum.set-command`.

## 14.4. `shasum.sha-only()`

Print SHA ONLY removing the file components

## 14.5. `shasum.sha-only-stdin()`

Print SHA ONLY removing the file components

## 14.6. shasum.to-hash()

This function populates a pre-declare associative array with filenames mapped to their SHAs, but only in the current directory. Call **dbg-on** to enable additional debugging info.

### 14.6.1 Example

```
$ declare -A file_shas
$ shasum.to-hash file_shas $(find . -type f -maxdepth 2)
$ echo "Total of ${#file_shas[@]} files in the hash"
```

## 14.7. shasum.all-files()

For a given array of files, sort them, take a SHA of each file, and return a single SHA finger-printing this set of files. # NOTE: the files are sorted prior to hashing, so the return SHA should ONLY change when files are either changed, or added/removed. Only computes SHA of the files provided, does not recurse into folders

### 14.7.1 Example

```
$ shasum.all-files *.cpp
```

## 14.8. shasum.all-files-in-dir()

For a given directory and an optional file pattern, use **find** to grab every single file (that matches optional pattern) and return a single SHA

### 14.8.1 Example

```
$ shasum.all-files-in-dir . '*.pdf'
cc35aad389e61942c75e111f1eddb634d74b4b1
```

## Chapter 15. File `lib/pg.sh`

- `pg.is-running()`
- `pg.running.server-binaries()`
- `pg.running.data-dirs()`
- `pg.server-in-path.version()`

### 15.1. `pg.is-running()`

Returns true if PostgreSQL is running locally

### 15.2. `pg.running.server-binaries()`

if one or more PostgreSQL instances is running locally, prints each server's binary postgres file path

### 15.3. `pg.running.data-dirs()`

For each running server prints the data directory

### 15.4. `pg.server-in-path.version()`

Grab the version from `postgres` binary in the PATH and remove fractional sub-version

---

# Chapter 16. File `lib/dir.sh`

- `dir.short-home()`

## 16.1. `dir.short-home()`

Replaces the first part of the directory that matches `${HOME}` with `~/`

---

## Chapter 17. File `lib/is.sh`

is.sh

# Chapter 18. Overview

Various validations and asserts that can be chained and be explicit in a DSL-like way.

- `<<isvalidationerror,is.validation.error(>>`
- `is-validations()`
- `<<isvalidationignore-error,is.validation.ignore-error(>>`
- `<<isvalidationreport-error,is.validation.report-error(>>`
- `whenever()`

## 18.1. `__is.validation.error()`

Invoke a validation on the value, and process the invalid case using a customizable error handler.

### 18.1.1. Arguments

- `@arg1 func` Validation function name to invoke
- `@arg2 var` Value under the test
- `@arg4 error_func` Error function to call when validation fails

### 18.1.2. Exit codes

- `0`: if validation passes

## 18.2. `is-validations()`

Returns the list of validation functions available

## 18.3. `__is.validation.ignore-error()`

Private function that ignores errors

## 18.4. `__is.validation.report-error()`

Private function that ignores errors

## 18.5. `whenever()`

a convenient DSL for validating things

### 18.5.1. Example



```
whenever /var/log/postgresql.log is.an-empty-file && {  
    touch /var/log/postgresql.log  
}
```

---

## Chapter 19. File `lib/util.sh`

# util.sh

# Chapter 20. Overview

Miscellaneous utilities.

- [util.rot13-stdin\(\)](#)

## 20.1. util.rot13-stdin()

Convert STDIN using rot13

### 20.1.1. Example

```
echo "test" | util.rot13-stdin
```

## Chapter 21. File `lib/pdf.sh`

# Bashmatic Utilities for PDF file handling

# Chapter 22. Overview

Install and uses GhostScript to manipulate PDFs.

- [pdf.combine\(\)](#)

## 22.1. pdf.combine()

Combine multiple PDFs into a single one using ghostscript.

### 22.11. Example

```
pdf.combine ~/merged.pdf 'my-book-chapter*'
```

### 22.12. Arguments

- `$1` (pathname): to the merged file
  - ... (the): rest of the PDF files to combine
-

## Chapter 23. File `bin/install-direnv`



# install-direnv

# Chapter 24. Overview

Add direnv hook to shell RC files

- [direnv.register\(\)](#)

## 24.1. `direnv.register()`

Add direnv hook to shell RC files

---

## Chapter 25. File `bin/regen-usage-docs`

# regen-usage-docs

# Chapter 26. Overview

Regenerates USAGE.adoc && USAGE.pdf

---

## Chapter 27. File `bin/specs`

- `specs.init()`
- `specs.determine-test-filename()`

### 27.1 `specs.init()`

Initialize specs

### 27.2 `specs.determine-test-filename()`

Based on a shortname attempt to determine the actual test file names

---

## Chapter 28. File `bin/pdf-reduce`

- `pdf.do.shrink()`

### 28.1 `pdf.do.shrink()`

shrinks PDF

---

# Chapter 29. Copyright & License

- Copyright © 2017-2021 Konstantin Gredeskoul, All rights reserved.
- Distributed under the MIT License.