Configuration

Jump to bottom

Nick Sweeting edited this page on May 9 · 160 revisions

Configuration

Configuration of ArchiveBox is done by using the `archivebox config` command, modifying the `ArchiveBox.conf` file in the data folder, or by using environment variables. All three methods work equivalently when using Docker as well.

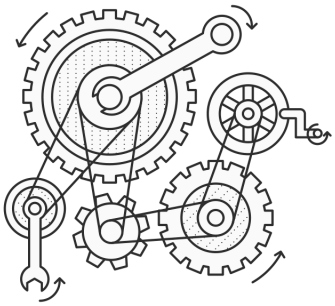
Some equivalent examples of setting some configuration options:

```
archivebox config --set CHROME_BINARY=google-chrome-stable
# OR
echo "CHROME_BINARY=google-chrome-stable" >> ArchiveBox.conf
# OR
env CHROME_BINARY=google-chrome-stable archivebox add ~/Downloads/bookmarks_export.ht
```

Environment variables take precedence over the config file, which is useful if you only want to use a certain option temporarily during a single run. For more examples see Usage: Configuration...

Available Configuration Options:

- General Settings: Archiving process, output format, and timing.
- Archive Method Toggles: On/off switches for methods.
- Archive Method Options: Method tunables and parameters.
- Shell Options: Format & behavior of CLI output.
- Dependency Options: Specify exact paths to dependencies.



********** Configuration Options **********

In case this document is ever out of date, it's recommended to read the code that loads the config directly: `archivebox/config.py` ➡️

General Settings

General options around the archiving process, output format, and timing.

`OUTPUT_PERMISSIONS`

Possible Values: [`755`]/ `644` /...
Permissions to set the output directory and file contents to.

This is useful when running ArchiveBox inside Docker as root and you need to explicitly set the permissions to something that the users on the host can access.

Related options:
`PUID` / `PGID`

`PUID` / `PGID`

Possible Values: [`911`]/ `1000` /...

User and Group ownership to set the output directory and file contents to. **Only settable as environment variables** when using ArchiveBox in Docker.

This is useful on some Docker setups when you want the data dir to be owned by the same UID/GID on the host and inside the container.

`PUID=0` is not allowed (do not run as root), and `PGID=0` is allowed but not recommended. `PUID`s and `PGID`s below `100` cause many issues because they're often already in use by an existing linux user in docker, if the files must be owned by a low value ID e.g. `33` (`www-data`), you may need to use `bindfs` to remap the permissions.

Make sure if using NFS/SMB/FUSE that the volume allows setting ownership on files (e.g. don't set `root_squash` or `all_squash` on NFS shares).

Learn more:

- https://docs.linuxserver.io/general/understanding-puid-and-pgid/

- https://github.com/ArchiveBox/ArchiveBox/wiki/Troubleshooting#docker-permissions-issues
- https://github.com/ArchiveBox/ArchiveBox/issues/1304
- https://github.com/ArchiveBox/ArchiveBox/discussions/1366
- https://github.com/ArchiveBox/ArchiveBox/blob/main/bin/docker_entrypoint.sh

Related options:
`OUTPUT_PERMISSIONS`

`ONLY_NEW`

Possible Values: [`True`]/ `False`
Toggle whether or not to attempt rechecking old links when adding new ones, or leave old incomplete links alone and only archive the new links.

By default, ArchiveBox will only archive new links on each import. If you want it to go back through all links in the index and download any missing files on every run, set this to `False`.

Note: Regardless of how this is set, ArchiveBox will never re-download sites that have already succeeded previously. When this is `False` it only attempts to fix previous pages have missing archive extractor outputs, it does not re-archive pages that have already been successfully archived.

`TIMEOUT`

Possible Values: [`60`]/ `120` /...
Maximum allowed download time per archive method for each link in seconds. If you have a slow network connection or are seeing frequent timeout errors, you can raise this value.

Note: Do not set this to anything less than `15` seconds as it will cause Chrome to hang indefinitely and many sites to fail completely.

`MEDIA_TIMEOUT`

Possible Values: [`3600`]/ `120` /...
Maximum allowed download time for fetching media when `SAVE_MEDIA=True` in seconds. This timeout is separate and usually much longer than `TIMEOUT` because media downloaded with `youtube-dl` can often be quite large and take many minutes/hours to download. Tweak this setting based on your network speed and maximum media file size you plan on downloading.

Note: Do not set this to anything less than `10` seconds as it can often take 5-10 seconds for `youtube-dl` just to parse the page before it starts downloading media files.

Related options:
`SAVE_MEDIA`

`ADMIN_USERNAME` / `ADMIN_PASSWORD`

Possible Values: [`None`]/ `"admin"` /...

Only used on first run / initial setup in Docker. ArchiveBox will create an admin user with the specified username and password when these options are found in the environment. Useful for setting up a Docker instance of ArchiveBox without needing to run a shell command to create the admin user.

Equivalent to:

```
$ archivebox manage createsuperuser
Username: <ADMIN_USERNAME>
Password: <ADMIN_PASSWORD>
Password (again): <ADMIN_PASSWORD>
```

More info:

- https://github.com/ArchiveBox/ArchiveBox/wiki/Setting-up-Authentication
- https://github.com/ArchiveBox/ArchiveBox/wiki/Docker#configuration

Related options:
`PUBLIC_INDEX / PUBLIC_SNAPSHOTS / PUBLIC_ADD_VIEW`

`PUBLIC_INDEX` / `PUBLIC_SNAPSHOTS` / `PUBLIC_ADD_VIEW`

Possible Values: [`True`]/ `False` Configure whether or not login is required to use each area of ArchiveBox.

```
archivebox manage createsuperuser  # set a password before disabling public acce

# these are the default values
archivebox config --set PUBLIC_INDEX=True        # True = allow users to view main sn
archivebox config --set PUBLIC_SNAPSHOTS=True    # True = allow users to view snapsho
archivebox config --set PUBLIC_ADD_VIEW=False    # True = allow users to submit new U
```

More info:

- https://github.com/ArchiveBox/ArchiveBox/wiki/Setting-up-Authentication
- https://github.com/ArchiveBox/ArchiveBox/wiki/Usage#ui-usage

---

`CUSTOM_TEMPLATES_DIR`

Possible Values: [`None`]/ `/path/to/custom_templates` /...

Path to a directory containing custom html/css/images for overriding the default UI styling. Files found in the folder at the specified path can override any of the defaults in the `TEMPLATES_DIR` directory (copy files from that default dir into your custom dir to get started making a custom theme).

If you've used `django` before, this works exactly the same way that `django` template overrides work (because it uses `django` under the hood).

```
pip show -f archivebox | grep Location: | awk '{print $2}'
# /opt/homebrew/lib/python3.11/site-packages

pip show -f archivebox | grep archivebox/templates
# archivebox/templates/admin/app_index.html
# archivebox/templates/admin/base.html
# archivebox/templates/admin/login.html
# ...

# copy default templates into a directory somewhere, edit as needed, then point archi
cp -r /opt/homebrew/lib/python3.11/site-packages/archivebox/templates ~/archivebox/cu
archivebox config --set CUSTOM_TEMPLATES_DIR=~/archivebox/data/custom_templates
```

Related options:
`FOOTER_INFO`

`REVERSE_PROXY_USER_HEADER`

Possible Values: [`Remote-User`]/ `X-Remote-User` /...

HTTP header containing user name from authenticated proxy.

More info:

- https://github.com/ArchiveBox/ArchiveBox/wiki/Setting-up-Authentication
- https://github.com/ArchiveBox/ArchiveBox/pull/866

Related options: `REVERSE_PROXY_WHITELIST`, `LOGOUT_REDIRECT_URL`

`REVERSE_PROXY_WHITELIST`

Possible Values: [`<empty string>`], `172.16.0.0/16`, `2001:d80::/26`/...

Comma separated list of IP CIDRs which are allowed to use reverse proxy authentication. Both IPv4 and IPv6 IPs can be used next to each other. Empty string means "deny all".

More info:

- https://github.com/ArchiveBox/ArchiveBox/wiki/Setting-up-Authentication
- https://github.com/ArchiveBox/ArchiveBox/pull/866

Related options: `REVERSE_PROXY_USER_HEADER`, `LOGOUT_REDIRECT_URL`

`LOGOUT_REDIRECT_URL`

Possible Values: [`/`]/`https://example.com/some/other/app`/...

URL to redirect users back to on logout when using reverse proxy authentication.

More info:

- https://github.com/ArchiveBox/ArchiveBox/wiki/Setting-up-Authentication
- https://github.com/ArchiveBox/ArchiveBox/pull/866

Related options: `REVERSE_PROXY_USER_HEADER`, `REVERSE_PROXY_WHITELIST`

`LDAP`

Possible Values: [`False`]/`True`

Whether to use an external LDAP server for authentication (e.g. OpenLDAP, MS Active Directory, OpenDJ, etc.).

```
# first, install optional ldap addon to use this feature
pip install archivebox[ldap]
```

Then set these configuration values to finish configuring LDAP:

```
LDAP: True
LDAP_SERVER_URI: "ldap://ldap.example.com:3389"
LDAP_BIND_DN: "ou=archivebox,ou=services,dc=ldap.example.com"
LDAP_BIND_PASSWORD: "secret-bind-user-password"
LDAP_USER_BASE: "ou=users,ou=archivebox,ou=services,dc=ldap.example.com"
LDAP_USER_FILTER: "(objectClass=user)"

LDAP_USERNAME_ATTR: "uid"
LDAP_FIRSTNAME_ATTR: "givenName"
LDAP_LASTNAME_ATTR: "sn"
LDAP_EMAIL_ATTR: "mail"
```

More info:

- https://github.com/ArchiveBox/ArchiveBox/wiki/Setting-up-Authentication
- https://github.com/ArchiveBox/ArchiveBox/pull/1214
- https://github.com/django-auth-ldap/django-auth-ldap#example-configuration
- https://jumpcloud.com/blog/what-is-ldap-authentication

`SNAPSHOTS_PER_PAGE`

Possible Values: [`40`]/ `100` /...

Maximum number of Snapshots to show per page on Snapshot list pages. Lower this value on slower machines to make the UI faster.

Related options:
`SEARCH_BACKEND_TIMEOUT`

`FOOTER_INFO`

Possible Values: [`Content is hosted for personal archiving purposes only.  Contact server owner for any takedown requests.`]/ `Operated by ACME Co.` /...
Some text to display in the footer of the archive index. Useful for providing server admin contact info to respond to takedown requests.

Related options:
`TEMPLATES_DIR`

`URL_DENYLIST`

Possible Values: [`\.(css|js|otf|ttf|woff|woff2|gstatic\.com|googleapis\.com/css)(\?.*)?$`]/ `.+\.exe$` / `http(s)?:\/\/(.+)?example.com\/.*` /...

A regex expression used to exclude certain URLs from archiving. You can use if there are certain domains, extensions, or other URL patterns that you want to ignore whenever they get imported. Blacklisted URLs wont be included in the index, and their page content wont be archived.

When building your exclusion list, you can check whether a given URL matches your regex expression in `python` like so:

```
>>> import re
>>> URL_DENYLIST = r'^http(s)?:\/\/(.+\.)?(youtube\.com)|(amazon\.com)\/.*$'  # repla
>>> URL_DENYLIST_PTN = re.compile(URL_DENYLIST, re.IGNORECASE | re.UNICODE | re.MULTI

>>> bool(URL_DENYLIST_PTN.search('https://test.youtube.com/example.php?abc=123'))  #
True   # this URL would not be archived because it matches the exclusion pattern
```

Note: all assets required to render each page are still archived, `URL_DENYLIST` / `URL_ALLOWLIST` do not apply to images, css, video, etc. visible inline within the page.

Note 2: These options used to be called `URL_WHITELIST` & `URL_BLACKLIST` before `v0.7.1`.

Related options:
`URL_ALLOWLIST`, `SAVE_MEDIA`, `SAVE_GIT`, `GIT_DOMAINS`

`URL_ALLOWLIST`

Possible Values: [`None`]/ `^http(s)?:\/\/(.+)?example\.com\/?.*$` /...

A regex expression used to exclude all URLs that don't match the given pattern from archiving. You can use if there are certain domains, extensions, or other URL patterns that you want to restrict the scope of archiving to (e.g. to only archive a single domain, subdirectory, or filetype, etc..

When building your whitelist, you can check whether a given URL matches your regex expression in `python` like so:

```
>>> import re
>>> URL_ALLOWLIST = r'^http(s)?:\/\/(.+)?example\.com\/?.*$'  # replace this with you
>>> URL_ALLOWLIST_PTN = re.compile(URL_ALLOWLIST, re.IGNORECASE | re.UNICODE | re.MUL

>>> bool(URL_ALLOWLIST_PTN.search('https://test.example.com/example.php?abc=123'))
True      # this URL would be archived
```

```
>>> bool(URL_ALLOWLIST_PTN.search('https://test.youtube.com/example.php?abc=123'))
False     # this URL would be excluded from archiving
```

This option is useful for **recursive archiving** of all the pages under a given domain or subfolder (aka crawling/spidering), without following links to external domains / parent folders.

```
# temporarily enforce a whitelist by setting the option as an environment variab
export URL_ALLOWLIST='^http(s)?:\/\/(.+)?example\.com\/?.*$'

# then run your archivebox commands in the same shell
archivebox add --depth=1 'https://example.com'
archivebox list https://example.com | archivebox add --depth=1
archivebox list https://example.com | archivebox add --depth=1
archivebox list https://example.com | archivebox add --depth=1   # repeat up to desir
...
# all URLs that don't match *.example.com will be excluded, e.g. a link to youtube.co
```

Note: all assets required to render each page are still archived, `URL_DENYLIST` / `URL_ALLOWLIST` do not apply to images, css, video, etc. visible inline within the page.

Related options:
`URL_DENYLIST`, `SAVE_MEDIA`, `SAVE_GIT`, `GIT_DOMAINS`

Archive Method Toggles

High-level on/off switches for all the various methods used to archive URLs.

`SAVE_TITLE`

Possible Values: [`True`]/`False`
By default ArchiveBox uses the title provided by the import file, but not all types of imports provide titles (e.g. Plain texts lists of URLs). When this is True, ArchiveBox downloads the page (and follows all redirects), then it attempts to parse the link's title from the first `<title></title>` tag found in the response. It may be buggy or not work for certain sites that use JS to set the title, disabling it will lead to links imported without a title showing up with their URL as the title in the UI.

Related options:
`ONLY_NEW`, `CHECK_SSL_VALIDITY`

`SAVE_FAVICON`

Possible Values: [`True`]/`False`
Fetch and save favicon for the URL from Google's public favicon service: `https://www.google.com/s2/favicons?domain={domain}`. Set this to `FALSE` if you don't need favicons.

Related options:
`TEMPLATES_DIR`, `CHECK_SSL_VALIDITY`, `CURL_BINARY`

`SAVE_WGET`

Possible Values: [`True`]/`False`
Fetch page with wget, and save responses into folders for each domain, e.g. `example.com/index.html`, with `.html` appended if not present. For a full list of options used during the `wget` download process, see the `archivebox/archive_methods.py:save_wget(...)` function.

Related options:
`TIMEOUT`, `SAVE_WGET_REQUISITES`, `CHECK_SSL_VALIDITY`, `COOKIES_FILE`, `WGET_USER_AGENT`, `SAVE_WARC`, `WGET_BINARY`

`SAVE_WARC`

Possible Values: [`True`]/`False`
Save a timestamped WARC archive of all the page requests and responses during the wget archive process.

Related options:
`TIMEOUT`, `SAVE_WGET_REQUISITES`, `CHECK_SSL_VALIDITY`, `COOKIES_FILE`, `WGET_USER_AGENT`, `SAVE_WGET`, `WGET_BINARY`

`SAVE_PDF`

Possible Values: [`True`]/`False`
Print page as PDF.

Related options:
`TIMEOUT`, `CHECK_SSL_VALIDITY`, `CHROME_USER_DATA_DIR`, `CHROME_BINARY`

`SAVE_SCREENSHOT`

Possible Values: [`True`]/`False`
Fetch a screenshot of the page.

Related options:
`RESOLUTION`, `TIMEOUT`, `CHECK_SSL_VALIDITY`, `CHROME_USER_DATA_DIR`, `CHROME_BINARY`

`SAVE_DOM`

Possible Values: [`True`]/`False`
Fetch a DOM dump of the page.

Related options:
`TIMEOUT`, `CHECK_SSL_VALIDITY`, `CHROME_USER_DATA_DIR`, `CHROME_BINARY`

`SAVE_SINGLEFILE`

Possible Values: [`True`]/`False`
Fetch an HTML file with all assets embedded using Single File.

Related options:
`TIMEOUT`, `CHECK_SSL_VALIDITY`, `CHROME_USER_DATA_DIR`, `CHROME_BINARY`, `SINGLEFILE_BINARY`

`SAVE_READABILITY`

Possible Values: [`True`]/`False`
Extract article text, summary, and byline using Mozilla's Readability library. Unlike the other methods, this does not download any additional files, so it's practically free from a disk usage perspective. It works by using any existing downloaded HTML version (e.g. wget, DOM dump, singlefile) and piping it into readability.

Related options:
`TIMEOUT`, `SAVE_WGET`, `SAVE_DOM`, `SAVE_SINGLEFILE`, `SAVE_MERCURY`

`SAVE_MERCURY`

Possible Values: [`True`]/`False`
Extract article text, summary, and byline using the Mercury library. Unlike the other methods, this does not download any additional files, so it's practically free from a disk usage perspective. It works by using any existing downloaded HTML version (e.g. wget, DOM dump, singlefile) and piping it into Mercury.

Related options:
`TIMEOUT`, `SAVE_WGET`, `SAVE_DOM`, `SAVE_SINGLEFILE`, `SAVE_READABILITY`

`SAVE_GIT`

Possible Values: [`True`]/`False`
Fetch any git repositories on the page.

Related options:
`TIMEOUT`, `GIT_DOMAINS`, `CHECK_SSL_VALIDITY`, `GIT_BINARY`

`SAVE_MEDIA`

Possible Values: [`True`]/`False`
Fetch all audio, video, annotations, and media metadata on the page using `youtube-dl`. Warning, this can use up *a lot* of storage very quickly.

Related options:
`MEDIA_TIMEOUT`, `CHECK_SSL_VALIDITY`, `YOUTUBEDL_BINARY`

`SAVE_ARCHIVE_DOT_ORG`

Possible Values: [`True`]/`False`
Submit the page's URL to be archived on Archive.org. (The Internet Archive)

Related options:
`TIMEOUT`, `CHECK_SSL_VALIDITY`, `CURL_BINARY`

Archive Method Options

Specific options for individual archive methods above. Some of these are shared between multiple archive methods, others are specific to a single method.

`CHECK_SSL_VALIDITY`

Possible Values: [`True`]/`False`
Whether to enforce HTTPS certificate and HSTS chain of trust when archiving sites. Set this to `False` if you want to archive pages even if they have expired or invalid certificates. Be aware that when `False` you cannot guarantee that you have not been man-in-the-middle'd while archiving content, so the content cannot be verified to be what's on the original site.

`SAVE_WGET_REQUISITES`

Possible Values: [`True`]/`False`
Fetch images/css/js with wget. (True is highly recommended, otherwise your won't download many critical assets to render the page, like images, js, css, etc.)

Related options:
`TIMEOUT`, `SAVE_WGET`, `SAVE_WARC`, `WGET_BINARY`

`RESOLUTION`

Possible Values: [`1440,2000`]/`1024,768`/...
Screenshot resolution in pixels width,height.

Related options:
`SAVE_SCREENSHOT`

`CURL_USER_AGENT`

Possible Values: [`Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.61 Safari/537.36 ArchiveBox/{VERSION} (+https://github.com/ArchiveBox/ArchiveBox/) curl/{CURL_VERSION}`]/`"Mozilla/5.0 ..."`/...
This is the user agent to use during curl archiving. You can set this to impersonate a more common browser like Chrome or Firefox if you're getting blocked by servers for having an unknown/blacklisted user agent.

Related options:
`USE_CURL`, `SAVE_TITLE`, `CHECK_SSL_VALIDITY`, `CURL_BINARY`, `WGET_USER_AGENT`, `CHROME_USER_AGENT`

`WGET_USER_AGENT`

Possible Values: [`Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.61 Safari/537.36 ArchiveBox/{VERSION} (+https://github.com/ArchiveBox/ArchiveBox/) wget/{WGET_VERSION}`]/ `"Mozilla/5.0 ..."`/...
This is the user agent to use during wget archiving. You can set this to impersonate a more common browser like Chrome or Firefox if you're getting blocked by servers for having an unknown/blacklisted user agent.

Related options:
`SAVE_WGET`, `SAVE_WARC`, `CHECK_SSL_VALIDITY`, `WGET_BINARY`, `CHROME_USER_AGENT`

`CHROME_USER_AGENT`

Possible Values: [`Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.61 Safari/537.36 ArchiveBox/{VERSION} (+https://github.com/ArchiveBox/ArchiveBox/)`]/ `"Mozilla/5.0 ..."`/...

This is the user agent to use during Chrome headless archiving. If you're experiencing being blocked by many sites, you can set this to hide the `Headless` string that reveals to servers that you're using a headless browser.

Related options:
`SAVE_PDF`, `SAVE_SCREENSHOT`, `SAVE_DOM`, `CHECK_SSL_VALIDITY`, `CHROME_USER_DATA_DIR`, `CHROME_HEADLESS`, `CHROME_BINARY`, `WGET_USER_AGENT`

`GIT_DOMAINS`

Possible Values:
[`github.com,bitbucket.org,gitlab.com,gist.github.com,codeberg.org,gitea.com,git.sr.ht`]/ `git.example.com`/...
Domains to attempt download of git repositories on using `git clone`.

Related options:
`SAVE_GIT`, `CHECK_SSL_VALIDITY`

`COOKIES_FILE`

Possible Values: [`None`]/ `/path/to/cookies.txt`/...

Cookies file to pass to `wget`, `curl`, `yt-dlp` and other extractors that don't use Chrome (with its `CHROME_USER_DATA_DIR`) for authentication. To capture sites that require a user to be logged in, you configure this option to point to a netscape-format `cookies.txt` file containing all the cookies you want to use during archiving.

You can generate this `cookies.txt` file by using a number of different browser extensions that can export your cookies in this format, or by using `wget` on the command line with `--save-cookies` + `--user=... --password=...`.

> ⚠ Warning
>
> **Make sure you use separate burner credentials dedicated to archiving,** e.g. don't re-use your normal daily Facebook/Instagram/Youtube/etc. account cookies as server responses often contain your name/email/PII, session tokens, etc. which then get preserved in your snapshots!
>
> Future viewers of your archive may be able to use any reflected archived session tokens to log in as you, or at the very least, associate the content with your real identity. Even if this tradeoff seems acceptable now or you plan to keep your archive data private, you may want to share a snapshot with others in the future, and snapshots are very hard to sanitize/anonymize after-the-fact!

Related options:
`SAVE_WGET`, `SAVE_WARC`, `CHECK_SSL_VALIDITY`, `WGET_BINARY`

`CHROME_USER_DATA_DIR`

Possible Values: [`~/.config/google-chrome`]/`/tmp/chrome-profile`/...

Path to a Chrome user profile directory. To capture sites that require a user to be logged in, you can specify a path to a Chrome user profile (which loads the cookies needed for the user to be logged in). If you don't have an existing Chrome profile, create one with `chromium-browser --user-data-dir=/tmp/chrome-profile`, and log into the sites you need. Then set `CHROME_USER_DATA_DIR=/tmp/chrome-profile` to make ArchiveBox use that profile.

For a guide on how to set this up, see our Chromium Install: Setting up a profile wiki.

Note: Make sure the path does not have `Default` at the end (it should the the parent folder of `Default`), e.g. set it to `CHROME_USER_DATA_DIR=~/.config/chromium` and not `CHROME_USER_DATA_DIR=~/.config/chromium/Default`.

> ⚠ Warning

> **Make sure you use separate burner credentials dedicated to archiving,** e.g. don't log in with your normal daily Facebook/Instagram/Youtube/etc. accounts as server responses and page content will often contain your name/email/PII, session cookies, private tokens, etc. which then get preserved in your snapshots!
>
> Future viewers of your archive may be able to use any reflected archived session tokens to log in as you, or at the very least, associate the content with your real identity. Even if this tradeoff seems acceptable now or you plan to keep your archive data private, you may want to share a snapshot with others in the future, and snapshots are very hard to sanitize/anonymize after-the-fact!

When set to `None`, ArchiveBox `<v0.7.2` used to try to find any existing profile on your system automatically, but this behavior has been disabled in later versions for security reasons, it must now be set explicitly if you want to use a profile.

Related options:
`SAVE_PDF`, `SAVE_SCREENSHOT`, `SAVE_DOM`, `CHECK_SSL_VALIDITY`, `CHROME_HEADLESS`, `CHROME_BINARY`, `COOKIES_FILE`

`CHROME_HEADLESS`

Possible Values: [`True`]/`False`
Whether or not to use Chrome/Chromium in `--headless` mode (no browser UI displayed). When set to `False`, the full Chrome UI will be launched each time it's used to archive a page, which greatly slows down the process but allows you to watch in real-time as it saves each page.

Related options:
`SAVE_PDF`, `SAVE_SCREENSHOT`, `SAVE_DOM`, `CHROME_USER_DATA_DIR`, `CHROME_BINARY`

`CHROME_SANDBOX`

Possible Values: [`True`]/`False`
Whether or not to use the Chrome sandbox when archiving.

If you see an error message like this, it means you are trying to run ArchiveBox as root:

```
:ERROR:zygote_host_impl_linux.cc(89)] Running as root without --no-sandbox is no p
```

*Note: **Do not run ArchiveBox as root!** The solution to this error is not to override it by setting `CHROME_SANDBOX=False`, it's to use create another user (e.g. `www-data`) and run ArchiveBox under that new, less privileged user. This is a security-critical setting, only set this to `False` if you're running ArchiveBox inside a container or VM where it doesn't have access to the rest of your system!

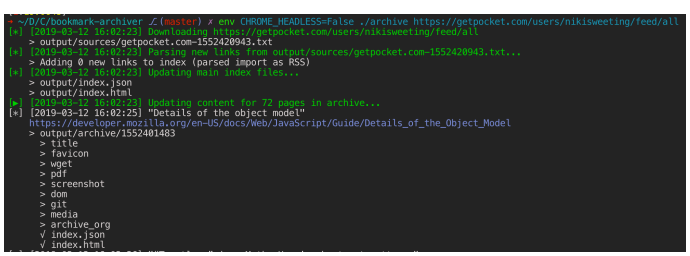
Related options:
`SAVE_PDF`, `SAVE_SCREENSHOT`, `SAVE_DOM`, `CHECK_SSL_VALIDITY`, `CHROME_USER_DATA_DIR`, `CHROME_HEADLESS`, `CHROME_BINARY`

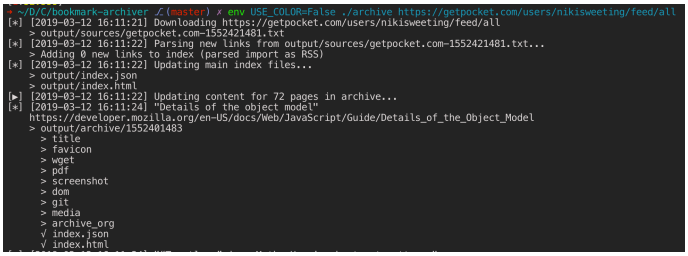
Shell Options

Options around the format of the CLI output.

`USE_COLOR`

Possible Values: [`True`]/`False`
Colorize console output. Defaults to `True` if stdin is a TTY (interactive session), otherwise `False` (e.g. if run in a script or piped into a file).

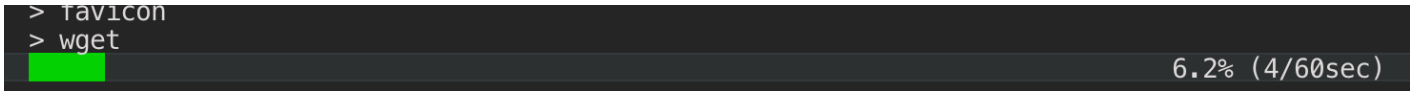




`SHOW_PROGRESS`

Possible Values: [`True`]/`False`
Show real-time progress bar in console output. Defaults to `True` if stdin is a TTY (interactive session), otherwise `False` (e.g. if run in a script or piped into a file).

Note: We use asymptotic progress bars because most tasks complete quickly! ✨



Dependency Options

Options for defining which binaries to use for the various archive method dependencies.

`CHROME_BINARY`

Possible Values: [`chromium-browser`]/`/usr/local/bin/google-chrome`/...
Path or name of the Google Chrome / Chromium binary to use for all the headless browser archive methods.

Without setting this environment variable, ArchiveBox by default look for the following binaries in `$PATH` in this order:

- `chromium-browser`
- `chromium`
- `google-chrome`
- `google-chrome-stable`
- `google-chrome-unstable`
- `google-chrome-beta`
- `google-chrome-canary`
- `google-chrome-dev`

You can override the default behavior to search for any available bin by setting the environment variable to your preferred Chrome binary name or path.

The chrome/chromium dependency is *optional* and only required for screenshots, PDF, and DOM dump output, it can be safely ignored if those three methods are disabled.

Related options:
`SAVE_PDF`, `SAVE_SCREENSHOT`, `SAVE_DOM`, `SAVE_SINGLEFILE`, `CHROME_USER_DATA_DIR`, `CHROME_HEADLESS`, `CHROME_SANDBOX`

`WGET_BINARY`

Possible Values: [`wget`]/`/usr/local/bin/wget`/...
Path or name of the wget binary to use.

Related options:
`SAVE_WGET`, `SAVE_WARC`

`YOUTUBEDL_BINARY`

Possible Values: [`youtube-dl`]/ `/usr/local/bin/youtube-dl` /...
Path or name of the youtube-dl binary to use.

Related options:
`SAVE_MEDIA`

`GIT_BINARY`

Possible Values: [`git`]/ `/usr/local/bin/git` /...
Path or name of the git binary to use.

Related options:
`SAVE_GIT`

`CURL_BINARY`

Possible Values: [`curl`]/ `/usr/local/bin/curl` /...
Path or name of the curl binary to use.

Related options:
`SAVE_FAVICON` , `SAVE_ARCHIVE_DOT_ORG`

`SINGLEFILE_BINARY`

Possible Values: [`single-file`]/ `./node_modules/single-file/cli/single-file` /...
Path or name of the SingleFile binary to use.

This can be installed using `npm install --no-audit --no-fund 'git+https://github.com/gildas-lormeau/SingleFile.git'` .

Related options:
`SAVE_SINGLEFILE` , `CHROME_BINARY` , `CHROME_USER_DATA_DIR` , `CHROME_HEADLESS` , `CHROME_SANDBOX`

`READABILITY_BINARY`

Possible Values: [`readability-extractor`]/ `./node_modules/readability-extractor/readability-extractor` /...
Path or name of the Readability extrator binary to use.

This can be installed using `npm install --no-audit --no-fund 'git+https://github.com/ArchiveBox/readability-extractor.git'`.

Related options:
`SAVE_READABILITY`

`MERCURY_BINARY`

Possible Values: [`mercury-parser`]/ `./node_modules/@postlight/mercury-parser/cli.js` /...
Path or name of the Mercury parser extractor binary to use.

This can be installed using `npm install --no-audit --no-fund '@postlight/mercury-parser'`.

Related options:
`SAVE_MERCURY`

`RIPGREP_BINARY`

Possible Values: [`rg`]/ `rga` /...

Path or name of the ripgrep binary to use for full text search.

This can be installed using your system package manager, e.g. `apt install ripgrep` or `brew install ripgrep`.

Optionally switch this to use `ripgrep-all` for full-text search support across more filetypes (e.g. PDF): https://github.com/phiresky/ripgrep-all.

Related options:
`SEARCH_BACKEND_ENGINE`

`SINGLEFILE_ARGS`

Possible Values: [`["--back-end=playwright-firefox","--load-deferred-images-dispatch-scroll-event=true"]`]/..

Arguments that are passed to the SingleFile binary. The values should be a valid JSON string.

Related options:
`SINGLEFILE_BINARY`

`CURL_ARGS`

Possible Values: [`["--tlsv1.3","--http2"]`]/..

Arguments that are passed to the curl binary. The values should be a valid JSON string.

Related options:
`CURL_BINARY`

`WGET_ARGS`

Possible Values: [`["--https-only"]`]/..

Arguments that are passed to the wget binary. The values should be a valid JSON string.

Related options:
`WGET_BINARY`

`YOUTUBEDL_ARGS`

Possible Values: [`["--limit-rate=10M"]`]/..

Arguments that are passed to the youtube-dl binary. The values should be a valid JSON string.

Related options:
`YOUTUBEDL_BINARY`

`GIT_ARGS`

Possible Values: [`["--depth=1"]`]

Arguments that are passed to the `git clone` subcommand. The values should be a valid JSON string.

Related options:
`GIT_BINARY`

##

▶️ *Looking for more? Sometimes this document is out of date. Check the source code for extra undocumented options:* `archivebox/config.py` .

✏️ Help improve our documentation...

▸ **Pages** 37

Getting Started

- 🔢 Quickstart
- 🖥️ Install
- 🐳 Docker
- ➡️ Supported Sources
- ⬅️ Supported Outputs

Usage

- $ Command Line
- 🌐 Web UI
- 🧩 Browser Extension

- 👾 REST API / Webhooks
- 📜 Python API / REPL / SQL API

Reference

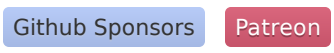
- ⚙️ Configuration
- 📦 Dependencies
- 💿 Disk Layout
- 🔒 Security Overview
- 📝 Developer Documentation

Guides

- Upgrading
- Setting up Storage (NFS/SMB/S3/etc)
- Setting up Authentication (SSO/LDAP/etc)
- Setting up Search (rg/sonic/etc)
- Scheduled Archiving
- Publishing Your Archive
- Chromium Install
- Cookies & Sessions Setup
- Merging Collections
- Troubleshooting

More Info

- ⭐ Web Archiving Community
- Background & Motivation
- Comparison to Other Tools
- Changelog & Roadmap



Clone this wiki locally

```
https://github.com/ArchiveBox/ArchiveBox.wiki.git
```