

TASKWARRIOR: GIT INTEGRATION

Eric Bailey

July 27, 2019¹

¹ Last updated Saturday 18th January, 2020

Taskwarrior's hook system² provides the means to run other programs at certain points in its execution, potentially affecting processing. What follows is an **on-exit** hook, which is triggered after all processing, but before output is displayed.

² <https://taskwarrior.org/docs/hooks.html>

SHEBANG

First, use a fairly portable Bash shebang, and be safe.³

```
< * 1a>≡
#! /usr/bin/env bash
```

```
set -eufo pipefail
```

This definition is continued in chunks 1 and 2.
Root chunk (not used in this document).

Echo commands if running *<in debug mode 1b>*.

```
< * 1a>+≡
if <in debug mode 1b>; then
    set -x
fi
```

³ **-exit** immediately upon failure, treat **-unset** variables as an error, disable **-file** globbing, and fail if any part of a pipeline fails (**-o pipefail**).

Run *<in debug mode 1b>* if the environment variable **DEBUG** is nonempty.

```
<in debug mode 1b>≡
[ -n "${DEBUG:-}" ]
```

This code is used in chunks 1 and 2.

ARGUMENT PARSING

Ignore the **api** version, **rc** file, or Taskwarrior **version**.

```
<extraneous information 1d>≡
# api="${1#api:}"
# rc="${4#rc:}"
# version="${6#version:}"
```

Root chunk (not used in this document).

N.B. The positional arguments are formatted as **foo:bar**, where **foo** is the name of the argument, and **bar** is the value.

Parse and store the values of the arguments, **args**, **command**, and **data** directory.

```
< * 1a>+≡
args="${2#args:}"
command="${3#command:}"
data="${5#data:}"
```

GIT INTEGRATION

Throughout this script, run *<git 1f>* as if it were started in the Taskwarrior **data** directory.

```
<git 1f>≡
git -C "$data"
```

This code is used in chunk 2.

If *<there are no changes 2a>*, **exit** successfully, printing an informative message if running in debug mode.

```
<* 1a>+≡
if <there are no changes 2a>; then
    if <in debug mode 1b>; then
        echo 'No changes to commit'
    fi
    exit 0
```

Like **diff**, the following **exits** with status code 0 if there are no differences between the working tree and the index.

```
<there are no changes 2a>≡
<git 1f> diff -quiet
```

This code is used in chunk 2b.

If present, stage the changes or die trying.

```
<* 1a>+≡
elif ! <git 1f> add -A; then
    echo 'Failed to add files to the index'
    exit 100
```

Try to *<commit the changes 2e>*, or else.

```
<* 1a>+≡
elif ! <commit the changes 2e>; then
    echo 'Failed to record changes to the repository'
    exit 101
```

Quietly store the current contents of the index in a new commit along with a log message describing the changes.

```
<commit the changes 2e>≡
<git 1f> commit -qm "$command: ${args##task $command}"
```

This code is used in chunk 2d.

If running *<in debug mode 1b>*, print a brief summary of the commit.

```
<* 1a>+≡
elif <in debug mode 1b>; then
    <git 1f> log -oneline -1
fi
```

Chunks

```
<* 1a>
<commit the changes 2e>
<extraneous information 1d>
<git 1f>
<in debug mode 1b>
<there are no changes 2a>
```

Index

```
api: 1d
args: 1e
command: 1e, 2e
data: 1e, 1f
rc: 1d
version: 1d
```