hub: USE GITHUB FROM THE COMMAND-LINE

hub is an extension to command-line git that helps you do everyday GitHub tasks without ever leaving the terminal.

Read the full documentation: man hub, or visit this project on GitHub.

```
# install with Homebrew (macOS, Linux)
# or see other installation options
$ brew install hub

$ hub version
git version 2.25.0
hub version 2.14.1    ← it works!

# indicate that you prefer HTTPS to SSH git clone URLs
$ git config --global hub.protocol https
```

Staying productive on the command-line

hub makes it easy to clone or create repositories, browse project pages, list known issues, ensure your local branches stay up to date, and share logs or code snippets via Gist.

```
# clone your own project
$ hub clone dotfiles
  → git clone git://github.com/YOUR_USER/dotfiles.git

# clone another project
$ hub clone github/hub
  → git clone git://github.com/github/hub.git

# fast-forward all local branches to match the latest state on the remote
$ cd myproject
$ hub sync

# list latest open issues in the current repository
$ hub issue --limit 10

# open the current project's issues page
$ hub browse -- issues
  → open https://github.com/github/hub/issues

# open another project's wiki
$ hub browse rbenv/ruby-build wiki
  → open https://github.com/rbenv/ruby-build/wiki

# share log output via Gist
$ hub gist create --copy build.log
```

```
  → (the URL of the new private gist copied to clipboard)
```

Starting a new project has never been easier:

```
# create a repo to host a new project on GitHub
$ git init
$ git add .
$ git commit -m "And so, it begins."
$ hub create
  → (creates a new GitHub repository with the name of the current directory)
$ git push -u origin HEAD
```

Lowering the barrier to contributing to open-source

Whether you are beginner or an experienced contributor to open-source, hub makes it easier to fork repositories, check the CI status of a branch, and even submit pull requests from the same environment where you write & commit your code.

```
$ hub clone octocat/Spoon-Knife
$ cd Spoon-Knife
# create a topic branch
$ git checkout -b feature
# make some changes...
$ git commit -am "done with feature"

# It's time to fork the repo!
$ hub fork --remote-name origin
  → (forking repo on GitHub...)
  → git remote add origin git@github.com:YOUR_USER/Spoon-Knife.git

# push the changes to your new remote
$ git push origin feature

# check the CI status for this branch
$ hub ci-status --verbose

# open a pull request for the branch you've just pushed
$ hub pull-request
  → (opens a text editor for your pull request message)
```

Automating tasks for fun and profit

Scripting is much easier now that you can list or create issues, pull requests, and GitHub Releases in the format of your choice.

```
# List issues assigned to you that are labeled "urgent"
$ hub issue --assignee YOUR_USER --labels urgent

# List the URLs of at most 20 pull requests based on the "develop" branch:
$ hub pr list --limit 20 --base develop --format='%t [%H] | %U%n'

# Create a GitHub Release from master using release notes from a file
```

```
$ hub release create --copy -F release-notes.txt v2.3.0
  → (the URL of the new release copied to clipboard)
```

Drop down to the API level

Even if hub doesn't support the exact feature you need, you can use `hub api` to manually make requests against any GitHub API—even GraphQL—and have hub handle authentication, JSON encoding/decoding, and pagination for you.

```
# use contents of a file to post a comment on issue #123 of the current
repo
$ hub api repos/{owner}/{repo}/issues/123/comments --field body=@mycomment.txt

# find a pull request that introduced a specific commit SHA into a repo
$ REPO="github/hub"
$ SHA="b0db79db"
$ hub api graphql --flat -f q="repo:$REPO type:pr $SHA" -f query='
  query($q: String!) {
    search(query: $q, type: ISSUE, first: 3) {
      nodes {
        ... on PullRequest {
          url
        }
      }
    }
  }
' | awk '/\.url/ { print $2 }'
```

See hub-api-utils for more examples.

Designed for open-source maintainers

Maintaining a project is easier when you can easily fetch from other forks, check out pull requests, close issues, and even cherry-pick commits by URL.

```
# fetch from multiple trusted forks, even if they don't yet exist as
remotes
$ hub fetch mislav,cehoffman
  → git remote add mislav git://github.com/mislav/hub.git
  → git remote add cehoffman git://github.com/cehoffman/hub.git
  → git fetch --multiple mislav cehoffman

# check out a pull request for review
$ hub pr checkout 134
  → (creates a new branch with the contents of the pull request)
# make new commits, then update the pull request
$ git push

# close an issue
$ hub issue update 134 --state closed

# directly apply all commits from a pull request to the current branch
$ hub am -3 https://github.com/github/hub/pull/134
```

```
# cherry-pick a GitHub URL
$ hub cherry-pick https://github.com/xoebus/hub/commit/177eeb8

# open the GitHub compare view between two releases
$ hub compare v0.9..v1.0

# put the compare URL for a topic branch to your clipboard
$ hub compare --url feature | pbcopy
```

Using GitHub for work

Save time at work by opening pull requests for code reviews and pushing to multiple remotes at once. Even GitHub Enterprise is supported.

```
# have hub recognize your GitHub Enterprise hostname
$ git config --global --add hub.host my.example.org

# transfer an issue to another repo
$ hub issue transfer 123 NEWREPO

# open a pull request with title & body from a file
$ git push origin feature
$ hub pull-request --copy -F prepared-message.md
  → (the URL of the new pull request copied to clipboard)

# push to multiple remotes
$ hub push production,staging
```

See the full reference documentation to learn more.